3D Modeling & Animation:

Synthesis and Analysis Techniques for the Human Body

Nikos Sarris
Michael G. Strintzis

3D Modeling and Animation:

Synthesis and Analysis Techniques for the Human Body

Nikos Sarris
Informatics & Telematics Institute, Greece

Michael G. Strintzis
Informatics & Telematics Institute, Greece

IRM Press
Publisher of innovative scholarly and professional
information technology titles in the cyberage
Hershey • London • Melbourne • Singapore

Acquisition Editor:	Mehdi Khosrow-Pour
Senior Managing Editor:	Jan Travers
Managing Editor:	Amanda Appicello
Development Editor:	Michele Rossi
Copy Editor:	Bernard J. Kieklak, Jr.
Typesetter:	Amanda Appicello
Cover Design:	Shane Dillow
Printed at:	Integrated Book Technology

Published in the United States of America by
IRM Press (an imprint of Idea Group Inc.)
701 E. Chocolate Avenue, Suite 200
Hershey PA 17033-1240
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@idea-group.com
Web site: http://www.irm-press.com

and in the United Kingdom by
IRM Press (an imprint of Idea Group Inc.)
3 Henrietta Street
Covent Garden
London WC2E 8LU
Tel: 44 20 7240 0856
Fax: 44 20 7379 3313
Web site: http://www.eurospan.co.uk

Library of Congress Cataloging-in-Publication Data

3d modeling and animation : synthesis and analysis techniques for the
human body / Nikos Sarris, Michael G. Strintzis, editors.
p. cm.
Includes bibliographical references and index.
ISBN 1-931777-98-5 (s/c) -- ISBN 1-931777-99-3 (ebook)
1. Computer animation. 2. Body, Human--Computer simulation. 3.
Computer simulation. 4. Three-dimensional display systems. 5. Computer
graphics. I. Title: Three-D modeling and animation. II. Sarris, Nikos,
1971- III. Strintzis, Michael G.
TR897.7.A117 2005
006.6'93--dc22
2003017709

ISBN 1-59140-299-9 h/c

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

All work contributed to this book is new, previously-unpublished material. The views expressed in this book are those of the authors, but not necessarily of the publisher.

3D Modeling and Animation:

Synthesis and Analysis Techniques for the Human Body

Table of Contents

Preface

The emergence of virtual reality applications and human-like interfaces has given rise to the necessity of producing realistic models of the human body. Building and animating a synthetic, cartoon-like, model of the human body has been practiced for many years in the gaming industry and advances in the game platforms have led to more realistic models, although still cartoon-like. The issue of building a virtual human clone is still a matter of ongoing research and relies on effective algorithms which will determine the 3D structure of an actual human being and duplicate this with a three-dimensional graphical model, fully textured, by correct mapping of 2D images of the human on the 3D model.

Realistic human animation is also a matter of ongoing research and, in the case of human cloning, relies on accurate tracking of the 3D motion of a human, which has to be duplicated by his 3D model. The inherently complex articulation of the human body imposes great difficulties in both the tracking and animation processes, which are being tackled by specific techniques, such as modeling languages, as well as by standards developed for these purposes. Particularly the human face and hands present the greatest difficulties in modeling and animation due to their complex articulation and communicative importance in expressing the human language and emotions.

Within the context of this book, we present the state-of-the-art methods for analyzing the structure and motion of the human body in parallel with the most effective techniques for constructing realistic synthetic models of virtual humans.

The level of detail that follows is such that the book can prove useful to students, researchers and software developers. That is, a level low enough to describe modeling methods and algorithms without getting into image processing and programming principles, which are not considered as prerequisite for the target audience.

The main objective of this book is to provide a reference for the state-of-the-art methods delivered by leading researchers in the area, who contribute to the appropriate chapters according to their expertise. The reader is presented with the latest, research-level, techniques for the analysis and synthesis of still and moving human bodies, with particular emphasis on facial and gesture characteristics.

Attached to this preface, the reader will find an introductory chapter which revises the state-of-the-art on established methods and standards for the analysis and synthesis of images containing humans. The most recent vision-based human body modeling techniques are presented, covering the topics of 3D human body coding standards, motion tracking, recognition and applications. Although this chapter, as well as the whole book, examines the relevant work in the context of computer vision, references to computer graphics techniques are given, as well.

The most relevant international standard established, MPEG-4, is briefly discussed in the introductory chapter, while its latest amendments, offering an appropriate framework for the animation and coding of virtual humans, is described in detail in Chapter 2. In particular, in this chapter *Preda et al.* show how this framework is extended within the new MPEG-4 standardization process by allowing the animation of any kind of articulated models, while addressing advanced modeling and animation concepts, such as “Skeleton, Muscle and Skin”-based approaches.

The issue of camera calibration is of generic importance to any computer vision application and is, therefore, addressed in a separate chapter by *Lei, Hendriks and Katsaggelos*. Thus, Chapter 3 presents a comprehensive overview of passive camera calibration techniques by comparing and evaluating existing approaches. All algorithms are presented in detail so that they can be directly implemented.

The detection of the human body and the recognition of human activities and hand gestures from multiview images are examined by *Ozer, Lv and Wolf* in

Chapter 4. Introducing the subject, the authors provide a review of the main components of three-dimensional and multiview visual processing techniques. The real-time aspects of these techniques are discussed and the ways in which these aspects affect the software and hardware architectures are shown. The authors also present the multiple-camera system developed by their group to investigate the relationship between the activity recognition algorithms and the architectures required to perform these tasks in real-time.

Gesture analysis is also discussed by *Karpouzis et al.* in Chapter 5, along with facial expression analysis within the context of human emotion recognition. A holistic approach to emotion modeling and analysis is presented along with applications in Man-Machine Interaction, aiming towards the next-generation interfaces that will be able to recognize the emotional states of their users.

The face, being the most expressive and complex part of the human body, is the object of discussion in the following five chapters as well. Chapter 6 examines techniques for the analysis of facial motion aiming mainly to the understanding of expressions from monoscopic images or image sequences. In Chapter 7 *Eisert* also addresses the same problem with his methods, paying particular attention to understanding and normalizing the illumination of the scene. *Kalberer, Müller and Van Gool* present their work in Chapter 8, extending the state-of-the-art in creating highly realistic lip and speech-related facial motion.

The deformation of three-dimensional human face models guided by the facial features captured from images or image sequences is examined in Chapters 9 and 10. *Kampmann and Zhang* propose a solution of varying complexity applicable to video-conferencing systems, while *Wen et al.* present a framework, based on machine learning, for the modeling, analysis and synthesis of facial deformation.

The book concludes with Chapter 11, by *Karatzoulis, Davarakis and Tzovaras*, providing a reference to current relevant R&D projects worldwide. This closing chapter presents a number of promising applications and provides an overview of recent developments and techniques in the area of analysis and synthesis techniques for the human body. Technical details are provided for each project and the provided results are also discussed and evaluated.

Chapter I

Advances in Vision-Based Human Body Modeling

Angel Sappa
Computer Vision Center, Spain

Niki Aifanti
Informatics & Telematics Institute, Greece

Nikos Grammalidis
Informatics & Telematics Institute, Greece

Sotiris Malassiotis
Informatics & Telematics Institute, Greece

Abstract

This chapter presents a survey of the most recent vision-based human body modeling techniques. It includes sections covering the topics of 3D human body coding standards, motion tracking, recognition and applications. Short summaries of various techniques, including their advantages and disadvantages, are introduced. Although this work is focused on computer vision, some references from computer graphics are also given. Considering that it is impossible to find a method valid for all applications, this chapter

intends to give an overview of the current techniques in order to help in the selection of the most suitable method for a certain problem.

Introduction

Human body modeling is experiencing a continuous and accelerated growth. This is partly due to the increasing demand from computer graphics and computer vision communities. Computer graphics pursue a realistic modeling of both the human body geometry and its associated motion. This will benefit applications such as games, virtual reality or animations, which demand highly realistic Human Body Models (HBMs). At the present, the cost of generating realistic human models is very high, therefore, their application is currently limited to the movie industry where HBM's movements are predefined and well studied (usually manually produced). The automatic generation of a realistic and fully configurable HBM is still nowadays an open problem. The major constraint involved is the computational complexity required to produce realistic models with natural behaviors. Computer graphics applications are usually based on motion capture devices (e.g., magnetic or optical trackers) as a first step, in order to accurately obtain the human body movements. Then, a second stage involves the manual generation of HBMs by using editing tools (several commercial products are available on the market).

Recently, computer vision technology has been used for the automatic generation of HBMs from a sequence of images by incorporating and exploiting prior knowledge of the human appearance. Computer vision also addresses human body modeling, but in contrast to computer graphics it seeks more for an efficient than an accurate model for applications such as intelligent video surveillance, motion analysis, telepresence or human-machine interface. Computer vision applications rely on vision sensors for reconstructing HBMs. Obviously, the rich information provided by a vision sensor, containing all the necessary data for generating a HBM, needs to be processed. Approaches such as *tracking-segmentation-model fitting* or *motion prediction-segmentation-model fitting* or other combinations have been proposed showing different performances according to the nature of the scene to be processed (e.g.. indoor environments, studio-like environments, outdoor environments, single-person scenes, etc). The challenge is to produce a HBM able to faithfully follow the movements of a real person.

Vision-based human body modeling combines several processing techniques from different research areas which have been developed for a variety of conditions (e.g., tracking, segmentation, model fitting, motion prediction, the study of kinematics, the dynamics of articulated structures, etc). In the current work, topics such as motion tracking and recognition and human body coding standards will be particularly treated due to their direct relation with human body modeling. Despite the fact that this survey will be focused on recent techniques involving HBMs within the computer vision community, some references to works from computer graphics will be given.

Due to widespread interest, there has been an abundance of work on human body modeling during the last years. This survey will cover most of the different techniques proposed in the bibliography, together with their advantages or disadvantages. The outline of this work is as follows. First, geometrical primitives and mathematical formalism, used for 3D model representation, are addressed. Next, standards used for coding HBMs, as well as a survey about human motion tracking and recognition are given. In addition, a summary of some application works is presented. Finally, a section with a conclusion is introduced.

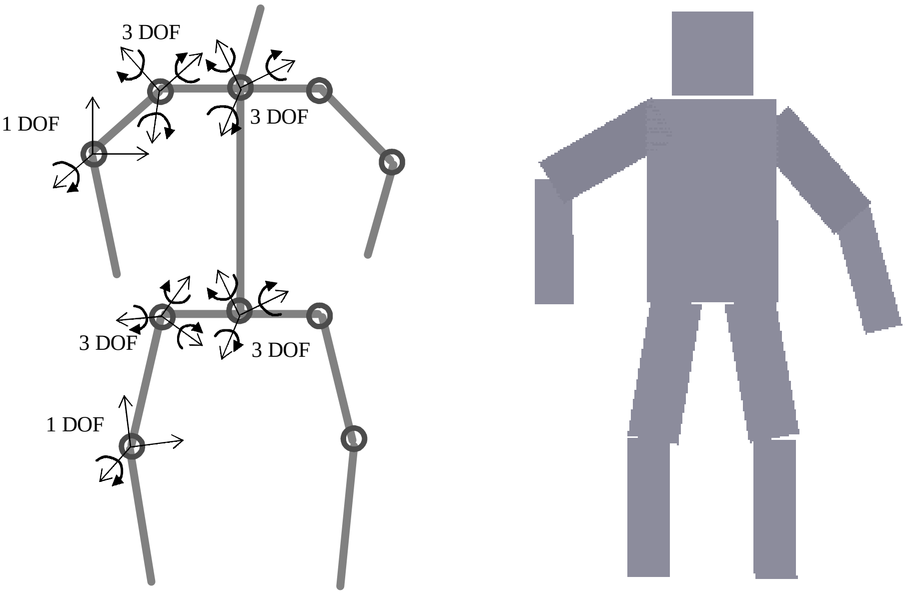
3D Human Body Modeling

Modeling a human body first implies the adaptation of an articulated 3D structure, in order to represent the human body biomechanical features. Secondly, it implies the definition of a mathematical model used to govern the movements of that articulated structure.

Several 3D articulated representations and mathematical formalisms have been proposed in the literature to model both the structure and movements of a human body. An HBM can be represented as a chain of rigid bodies, called *links*, interconnected to one another by *joints*. Links are generally represented by means of sticks (Barron & Kakadiaris, 2000), polyhedrons (Yamamoto et al., 1998), generalized cylinders (Cohen, Medioni & Gu, 2001) or superquadrics (Gavrila & Davis, 1996). A joint interconnects two links by means of rotational motions about the axes. The number of independent rotation parameters will define the *degrees of freedom* (DOF) associated with a given joint. Figure 1 (*left*) presents an illustration of an articulated model defined by 12 links (sticks) and ten joints.

In computer vision, where models with only medium precision are required, articulated structures with less than 30 DOF are generally adequate. For example, Delamarre & Faugeras (2001) use a model of 22 DOF in a multi-view tracking system. Gavrila & Davis (1996) also propose the use of a 22-DOF model without modeling the palm of the hand or the foot and using a rigid head-torso approximation. The model is defined by three DOFs for the positioning of the root of the articulated structure, three DOFs for the torso and four DOFs for

Figure 1. Left: Stick representation of an articulated model defined by 22 DOF. Right: Cardboard person model.

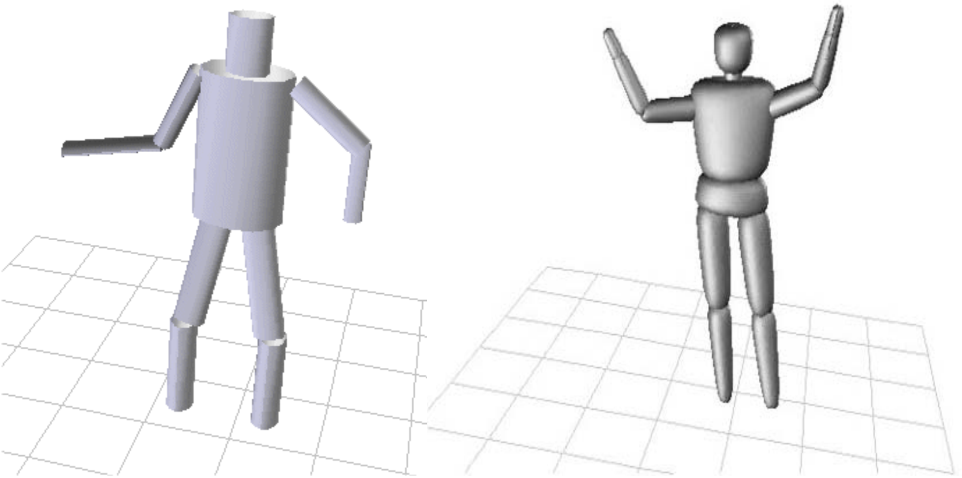


each arm and each leg. The illustration presented in Figure 1 (*left*) corresponds to an articulated model defined by 22 DOF.

On the contrary, in computer graphics, highly accurate representations consisting of more than 50 DOF are generally selected. Aubel, Boulic & Thalmann (2000) propose an articulated structure composed of 68 DOF. They correspond to the real human joints, plus a few global mobility nodes that are used to orient and position the virtual human in the world.

The simplest 3D articulated structure is a stick representation with no associated volume or surface (Figure 1 (*left*)). Planar 2D representations, such as the cardboard model, have also been widely used (Figure 1 (*right*)). However, volumetric representations are preferred in order to generate more realistic models (Figure 2). Different volumetric approaches have been proposed, depending upon whether the application is in the computer vision or the computer graphics field. On one hand, in computer vision, where the model is not the purpose, but the means to recover the 3D world, there is a trade-off between accuracy of representation and complexity. The utilized models should be quite realistic, but they should have a low number of parameters in order to be processed in real-time. Volumetric representations such as parallelepipeds,

Figure 2. Left: Volumetric model defined by 10 cylinders – 22 DOF. Right: Volumetric model built with a set of superquadrics – 22 DOF.



cylinders (Figure 2 (*left*)), or superquadrics (Figure 2 (*right*)) have been largely used. Delamarre & Faugeras (2001) propose to model a person by means of truncated cones (arms and legs), spheres (neck, joints and head) and right parallelepipeds (hands, feet and body). Most of these shapes can be modeled using a compact and accurate representation called superquadrics. Superquadrics are a family of parametric shapes that can model a large set of blob-like objects, such as spheres, cylinders, parallelepipes and shapes in between. Moreover, they can be deformed with tapering, bending and cavities (Solina & Bajcsy, 1990).

On the other hand, in computer graphics, accurate surface models consisting of thousands of polygons are generally used. Plänkers & Fua (2001) and Aubel, Boulic & Thalmann (2000) present a framework that retains an articulated structure represented by sticks, but replace the simple geometric primitives by soft objects. The result of this soft surface representation is a realistic model, where body parts such as chest, abdomen or biceps muscles are well modeled.

By incorporating a mathematical model of human motion in the geometric representation, the HBM comes *alive*, so that an application such as human body tracking may be improved. There are a wide variety of ways to mathematically model articulated systems from a kinematics and dynamics point of view. Much of these materials come directly from the field of robotics (Paul, 1981; Craig

1989). A mathematical model will include the parameters that describe the links, as well as information about the constraints associated with each joint. A model that only includes this information is called a *kinematic model* and describes the possible static states of a system. The state vector of a kinematic model consists of the model state and the model parameters. A system in motion is modeled when the dynamics of the system are modeled as well. A *dynamic model* describes the state evolution of the system over time. In a dynamic model, the state vector includes linear and angular velocities, as well as position (Wren & Pentland, 1998).

After selecting an appropriate model for a particular application, it is necessary to develop a concise mathematical formulation for a general solution to the kinematics and dynamics problem, which are non-linear problems. Different formalism have been proposed in order to assign local reference frames to the links. The simplest approach is to introduce joint hierarchies formed by independent articulation of one DOF, described in terms of Euler angles. Hence, the body posture is synthesized by concatenating the transformation matrices associated with the joints, starting from the root. Despite the fact that this formalism suffers from singularities, Delamarre & Faugeras (2001) propose the use of compositions of translations and rotations defined by Euler angles. They solve the singularity problems by reducing the number of DOFs of the articulation.

3D Human Body Coding Standards

As it was mentioned in the previous section, an HBM consists of a number of segments that are connected to each other by joints. This physical structure can be described in many different ways. However, in order to animate or interchange HBMs, a standard representation is required. This standardization allows compatibility between different HBM processing tools (e.g., HBMs created using one editing tool could be animated using another completely different tool). In the following, the Web3D H-anim standards, the MPEG-4 face and body animation, as well as MPEG-4 AFX extensions for humanoid animation, are briefly introduced.

The Web3D H-Anim Standards

The Web3D H-anim working group (H-anim) was formed so that developers could agree on a standard naming convention for human body parts and joints. The human form has been studied for centuries and most of the parts already have medical (or Latin) names. This group has produced the Humanoid Animation Specification (H-anim) standards, describing a standard way of representing humanoids in VRML. These standards allow humanoids created using authoring tools from one vendor to be animated using tools from another. H-anim humanoids can be animated using keyframing, inverse kinematics, performance animation systems and other techniques. The three main design goals of H-anim standards are:

- Compatibility: Humanoids should be able to display/animate in any VRML compliant browser.
- Flexibility: No assumptions are made about the types of applications that will use humanoids.
- Simplicity: The specification should contain only what is absolutely necessary.

Up to now, three H-anim standards have been produced, following developments in VRML standards, namely the H-anim 1.0, H-anim 2.0 and H-anim 2001 standards.

The H-anim 1.0 standard specified a standard way of representing humanoids in VRML 2.0 format. The VRML Humanoid file contains a set of *Joint* nodes, each defining the rotation center of a joint, which are arranged to form a hierarchy. The most common implementation for a joint is a VRML Transform node, which is used to define the relationship of each body segment to its immediate parent, although more complex implementations can also be supported. Each Joint node can contain other Joint nodes and may also contain a *Segment* node, which contains information about the 3D geometry, color and texture of the body part associated with that joint. Joint nodes may also contain hints for inverse-kinematics systems that wish to control the H-anim figure, such as the upper and lower joint limits, the orientation of the joint limits, and a stiffness/resistance value. The file also contains a single *Humanoid* node, which stores human-readable data about the humanoid, such as author and copyright information. This node also stores references to all the Joint and Segment nodes. Additional nodes can be included in the file, such as *Viewpoints*, which may be used to display the figure from several different perspectives.

The H-anim 1.1 standard has extended the previous version in order to specify humanoids in the VRML97 standard (successor of VRML 2.0). New features include *Site* nodes, which define specific locations relative to the segment, and *Displacer* nodes that specify which vertices within the segment correspond to a particular feature or configuration of vertices. Furthermore, a Displacer node may contain “hints” as to the direction in which each vertex should move, namely a maximum 3-D displacement for each vertex. An application may uniformly scale these displacements before applying them to the corresponding vertices. For example, this field is used to implement Facial Definition and Animation Parameters of the MPEG-4 standard (FDP/FAP).

Finally, the H-anim 2001 standard does not introduce any major changes, e.g., new nodes, but provides better support of deformation engines and animation tools. Additional fields are provided in the Humanoid and the Joint nodes to support continuous mesh avatars and a more general context-free grammar is used to describe the standard (instead of pure VRML97, which is used in the two older H-anim standards). More specifically, a skeletal hierarchy can be defined for each H-anim humanoid figure within a *Skeleton* field of the Humanoid node. Then, an H-anim humanoid figure can be defined as a continuous piece of geometry, within a *Skin* field of the Humanoid node, instead of a set of discrete segments (corresponding to each body part), as in the previous versions. This Skin field contains an indexed face set (coordinates, topology and normals of skin nodes). Each Joint node also contains a *SkinCoordWeight* field, i.e., a list of floating point values, which describes the amount of “weighting” that should be used to affect a particular vertex from a *SkinCoord* field of the Humanoid node. Each item in this list has a corresponding index value in the *SkinCoordIndex* field of the Joint node, which indicates exactly which coordinate is to be influenced.

Face and Body Animation in the MPEG-4 Standard

The MPEG-4 SNHC (Synthetic and Natural Hybrid Coding) group has standardized two types of streams in order to animate avatars:

- The Face/Body Definition Parameters (FDP/BDP) are avatar-specific and based on the H-anim specifications. More precisely the MPEG-4 BDP Node contains the H-anim Humanoid Node.
- The Face/Body Animation Parameters (FAP/BAP) are used to animate face/body models. More specifically, 168 Body Animation Parameters (BAPs) are defined by MPEG-4 SNHC to describe almost any possible body posture. A single set of FAPs/BAPs can be used to describe the face/body posture of different avatars. MPEG-4 has also standardized the compressed form of the resulting animation stream using two techniques: DCT-based or prediction-based. Typical bit-rates for these compressed bit-streams are two kbps for the case of facial animation or 10 to 30 kbps for the case of body animation.

In addition, complex 3D deformations that can result from the movement of specific body parts (e.g., muscle contraction, clothing folds, etc.) can be modeled by using Face/Body Animation Tables (FAT/BATs). These tables specify a set of vertices that undergo non-rigid motion and a function to describe this motion with respect to the values of specific FAPs/BAPs. However, a significant problem with using FAT/BAT Tables is that they are body model-dependent and require a complex modeling stage. On the other hand, BATs can prevent undesired body animation effects, such as broken meshes between two linked segments. In order to solve such problems, MPEG-4 addresses new animation functionalities in the framework of AFX group (a preliminary specification has been released in January 2002) by including also a generic seamless virtual model definition and bone-based animation. Particularly, the AFX specification describes state of the art components for rendering geometry, textures, volumes and animation. A hierarchy of geometry, modeling, physics and biomechanical models are described along with advanced tools for animating these models.

AFX Extensions for Humanoid Animation

The new Humanoid Animation Framework, defined by MPEG-4 SNHC (Preda, 2002; Preda & Prêteux, 2001) is defined as a biomechanical model in AFX and is based on a rigid skeleton. The skeleton consists of bones, which are rigid objects that can be transformed (rotated around specific joints), but not deformed. Attached to the skeleton, a skin model is defined, which smoothly follows any skeleton movement.

More specifically, defining a skinned model involves specifying its static and dynamic (animation) properties. From a geometric point of view, a skinned model consists of a single list of vertices, connected as an indexed face set. All the shapes, which form the skin, share the same list of vertices, thus avoiding seams at the skin level during animation. However, each skin facet can contain its own set of color, texture and material attributes.

The dynamic properties of a skinned model are defined by means of a skeleton and its properties. The skeleton is a hierarchical structure constructed from bones, each having an influence on the skin surface. When bone position or orientation changes, e.g., by applying a set of Body Animation Parameters, specific skin vertices are affected. For each bone, the list of vertices affected by the bone motion and corresponding weight values are provided. The weighting factors can be specified either explicitly for each vertex or more compactly by defining two influence regions (inner and outer) around the bone. The new position of each vertex is calculated by taking into account the influence of each bone, with the corresponding weight factor. BAPs are now applied to bone nodes and the new 3D position of each point in the global seamless mesh is computed as a weighted combination of the related bone motions.

The skinned model definition can also be enriched with *inverse kinematics*-related data. Then, bone positions can be determined by specifying only the position of an end effector, e.g., a 3D point on the skinned model surface. No specific inverse kinematics solver is imposed, but specific constraints at bone level are defined, e.g., related to the rotation or translation of a bone in a certain direction. Also *muscles*, i.e., NURBS curves with an influence region on the model skin, are supported. Finally, interpolation techniques, such as simple linear interpolation or linear interpolation between two quaternions (Preda & Prêteux, 2001), can be exploited for key-value-based animation and animation compression.

Human Motion Tracking and Recognition

Tracking and recognition of human motion has become an important research area in computer vision. Its numerous applications contributed significantly to this development. Human motion tracking and recognition encompasses challenging and ill-posed problems, which are usually tackled by making simplifying assumptions regarding the scene or by imposing constraints on the motion. Constraints, such as making sure that the contrast between the moving people and the background should be high and that everything in the scene should be static except for the target person, are quite often introduced in order to achieve accurate segmentation. Moreover, assumptions such as the lack of occlusions, simple motions and known initial position and posture of the person, are usually imposed on the tracking processes. However, in real-world conditions, human motion tracking constitutes a complicated problem, considering cluttered backgrounds, gross illumination variations, occlusions, self-occlusions, different clothing and multiple moving objects.

The first step towards human tracking is the segmentation of human figures from the background. This problem is addressed either by exploiting the temporal relation between consecutive frames, i.e., by means of background subtraction (Sato & Aggarwal, 2001), optical flow (Okada, Shirai & Miura, 2000) or by modeling the image statistics of human appearance (Wren et al., 1997). The output of the segmentation, which could be edges, silhouettes, blobs etc., comprises the basis for feature extraction. In tracking, feature correspondence is established in order to locate the subject. Tracking through consecutive frames commonly incorporates prediction of movement, which ensures continuity of motion, especially when some body parts are occluded. Some techniques focus on tracking the human body as a whole, while other techniques try to determine the precise movement of each body part, which is more difficult to achieve, but necessary for some applications. Tracking may be classified as 2D or 3D. 2D tracking consists in following the motion in the image plane either by exploiting low-level image features or by using a 2D human model. 3D tracking aims at obtaining the parameters, which describe body motion in three dimensions. The 3D tracking process, which estimates the motion of each body part, is inherently connected to 3D human pose recovery. However, tracking either 2D or 3D may also comprise a prior, but significant, step to recognition of specific movements.

3D pose recovery aims at defining the configuration of the body parts in the 3D space and estimating the orientation of the body with respect to the camera. Pose recovery techniques may be roughly classified as appearance-based and model-based. Our survey will mainly focus on model-based techniques, since they are commonly used for 3D reconstruction. Model-based techniques rely on a mathematical representation of human body structure and motion dynamics. The type of the model used depends upon the requisite accuracy and the permissible complexity of pose reconstruction. Model-based approaches usually exploit the kinematics and dynamics of the human body by imposing constraints on the model's parameters. The 3D pose parameters are commonly estimated by iteratively matching a set of image features extracted from the current frame with the projection of the model on the image plane. Thus, 3D pose parameters are determined by means of an energy minimization process.

Instead of obtaining the exact configuration of the human body, human motion recognition consists of identifying the action performed by a moving person. Most of the proposed techniques focus on identifying actions belonging to the same category. For example, the objective could be to recognize several aerobic exercises or tennis strokes or some everyday actions, such as sitting down, standing up, or walking.

Next, some of the most recent results addressing human motion tracking and 3D human pose recovery in video sequences, using either one or multiple cameras, are presented. In this subsection, mainly 3D model-based tracking approaches are reviewed. The following subsection introduces whole-body human motion recognition techniques. Previous surveys of vision-based human motion analysis have been carried out by Cédras & Shah (1995), Aggarwal & Cai (1999), Gavrila (1999), and Moeslund & Granum (2001).

Human Motion Tracking and 3D Pose Recovery

The majority of model-based human motion tracking techniques may be classified into two main categories. The first one explicitly poses kinematic constraints to the model parameters, for example, by means of Kalman filtering or physics-based modeling. The second one is based on learning the dynamics of low-level features or high-level motion attributes from a set of representative image sequences, which are then used to constrain the model motion, usually within a probabilistic tracking framework. Other subdivisions of the existing techniques may rely on the type of the model or the type of image features (edges, blobs, texture) used for tracking.

Tracking relies either on monocular or multiple camera image sequences. This comprises the classification basis in this subsection. Using **monocular** image sequences is quite challenging, due to occlusions of body parts and ambiguity in recovering their structure and motion from a single perspective view (different configurations have the same projection). On the other hand, single camera views are more easily obtained and processed than multiple camera views.

In one of the most recent approaches (Sminchisescu & Triggs, 2001), 3D human motion tracking from monocular sequences is achieved by fitting a 3D human body model, consisting of tampered superellipsoids, on image features by means of an iterative cost function optimization scheme. The disadvantage of iterative model fitting techniques is the possibility of being trapped in local minima in the multidimensional space of DOF. A multiple-hypothesis approach is proposed with the ability of escaping local minima in the cost function. This consists of observing that local minima are most likely to occur along local valleys in the cost surface. In comparison with other stochastic sampling approaches, improved tracking efficiency is claimed.

In the same context, the algorithm proposed by Cham & Rehg (1999) focuses on 2D image plane human motion using a 2D model with underlying 3D kinematics. A combination of CONDENSATION style sampling with local optimization is proposed. The probability density distribution of the tracker state is represented as a set of modes with piece-wise Gaussians characterizing the neighborhood around these modes. The advantage of this technique is that it does not require the use of discrete features and is suitable for high-dimensional state-spaces.

Probabilistic tracking such as CONDENSATION has been proven resilient to occlusions and successful in avoiding local minima. Unfortunately, these advances come at the expense of computational efficiency. To avoid the cost of learning and running a probabilistic tracker, linear and linearised prediction techniques, such as Kalman or extended Kalman filtering, have been proposed. In this case, a strategy to overcome self-occlusions is required. More details on

CONDENSATION algorithms used in tracking and a comparison with the Kalman filters can be found in Isard & Blake (1998).

In Wachter & Nagel (1999), a 3D model composed of right-elliptical cones is fitted to consecutive frames by means of an iterated extended Kalman filter. A motion model of constant velocity for all DOFs is used for prediction, while the update of the parameters is based on a maximum *a-posteriori* estimation incorporating edge and region information. This approach is able to cope with self-occlusions occurring between the legs of a walking person. Self-occlusions are also tackled in a Bayesian tracking system presented in Howe, Leventon & Freeman (1999). This system tracks human figures in short monocular sequences and reconstructs their motion in 3D. It uses prior information learned from training data. Training data consists of a vector gathered over 11 successive frames representing the 3D coordinates of 20 tracked body points and is used to build a mixture-of-Gaussians probability density model. 3D reconstruction is achieved by establishing correspondence between the training data and the features extracted. Sidenbladh, Black & Sigal (2002) also use a probabilistic approach to address the problem of modeling 3D human motion for synthesis and tracking. They avoid the high dimensionality and non-linearity of body movement modeling by representing the posterior distribution non-parametrically. Learning state transition probabilities is replaced with an efficient probabilistic search in a large training set. An approximate probabilistic tree-search method takes advantage of the coefficients of a low-dimensional model and returns a particular sample human motion.

In contrast to single-view approaches, **multiple camera** techniques are able to overcome occlusions and depth ambiguities of the body parts, since useful motion information missing from one view may be recovered from another view.

A rich set of features is used in Okada, Shirai & Miura (2000) for the estimation of the 3D translation and rotation of the human body. Foreground regions are extracted by combining optical flow, depth (which is calculated from a pair of stereo images) and prediction information. 3D pose estimation is then based on the position and shape of the extracted region and on past states using Kalman filtering. The evident problem of pose singularities is tackled probabilistically.

A framework for person tracking in various indoor scenes is presented in Cai & Aggarwal (1999), using three synchronized cameras. Though there are three cameras, tracking is actually based on one camera view at a time. When the system predicts that the active camera no longer provides a sufficient view of the person, it is deactivated and the camera providing the best view is selected. Feature correspondence between consecutive frames is achieved using Bayesian classification schemes associated with motion analysis in a spatial-temporal domain. However, this method cannot deal with occlusions above a certain level.

Dockstader & Tekalp (2001) introduce a distributed real-time platform for tracking multiple interacting people using multiple cameras. The features extracted from each camera view are independently processed. The resulting state vectors comprise the input to a Bayesian belief network. The observations of each camera are then fused and the most likely 3D position estimates are computed. A Kalman filter performs state propagation in time. Multi-viewpoints and a viewpoint selection strategy are also employed in Utsumi et al. (1998) to cope with self-occlusions and human-human occlusions. In this approach, tracking is based on Kalman filtering estimation as well, but it is decomposed into three sub-tasks (position detection, rotation angle estimation and body-side detection). Each sub-task has its own criterion for selecting viewpoints, while the result of one sub-task can help estimation in another sub-task.

Delamarre & Faugeras (2001) proposed a technique which is able to cope not only with self-occlusions, but also with fast movements and poor quality images, using two or more fixed cameras. This approach incorporates physical forces to each rigid part of a kinematic 3D human body model consisting of truncated cones. These forces guide the 3D model towards a convergence with the body posture in the image. The model's projections are compared with the silhouettes extracted from the image by means of a novel approach, which combines the Maxwell's demons algorithm with the classical ICP algorithm.

Some recently published papers specifically tackle the **pose recovery** problem using multiple sensors. A real-time method for 3D posture estimation using trinocular images is introduced in Iwasawa et al. (2000). In each image the human silhouette is extracted and the upper-body orientation is detected. With a heuristic contour analysis of the silhouette, some representative points, such as the top of the head are located. Two of the three views are finally selected in order to estimate the 3D coordinates of the representative points and joints. It is experimentally shown that the view-selection strategy results in more accurate estimates than the use of all views.

Multiple views in Rosales et al. (2001) are obtained by introducing the concept of "virtual cameras", which is based on the transformation invariance of the Hu moments. One advantage of this approach is that no camera calibration is required. A Specialized Mappings Architecture is proposed, which allows direct mapping of the image features to 2D image locations of body points. Given correspondences of the most likely 2D joint locations in virtual camera views, 3D body pose can be recovered using a generalized probabilistic structure from motion technique.

Human Motion Recognition

Human motion recognition may also be achieved by analyzing the extracted 3D pose parameters. However, because of the extra pre-processing required, recognition of human motion patterns is usually achieved by exploiting low-level features (e.g., silhouettes) obtained during tracking.

Continuous human activity (e.g., walking, sitting down, bending) is separated in Ali & Aggarwal (2001) into individual actions using one camera. In order to detect the commencement and termination of actions, the human skeleton is extracted and the angles subtended by the torso, the upper leg and the lower leg, are estimated. Each action is then recognized based on the characteristic path that these angles traverse. This technique, though, relies on lateral views of the human body.

Park & Aggarwal (2000) propose a method for separating and classifying not one person's actions, but two humans' interactions (shaking hands, pointing at the opposite person, standing hand-in-hand) in indoor monocular grayscale images with limited occlusions. The aim is to interpret interactions by inferring the intentions of the persons. Recognition is independently achieved in each frame by applying the K-nearest-neighbor classifier to a feature vector, which describes the interpersonal configuration. In Sato & Aggarwal (2001), human interaction recognition is also addressed. This technique uses outdoor monocular grayscale images that may cope with low-quality images, but is limited to movements perpendicular to the camera. It can classify nine two-person interactions (e.g., one person leaves another stationary person, two people meet from different directions). Four features are extracted (the absolute velocity of each person, their average size, the relative distance and its derivative) from the trajectory of each person. Identification is based on the feature's similarity to an interaction model using the nearest mean method.

Action and interaction recognition, such as standing, walking, meeting people and carrying objects, is addressed by Haritaoglu, Harwood & Davis (1998, 2000). A real-time tracking system, which is based on outdoor monocular grayscale images taken from a stationary visible or infrared camera, is introduced. Grayscale textural appearance and shape information of a person are combined to a textural temporal template, which is an extension of the temporal templates defined by Bobick & Davis (1996).

Bobick & Davis (1996) introduced a real-time human activity recognition method, which is based on a two-component image representation of motion. The first component (Motion Energy Image, MEI) is a binary image, which displays where motion has occurred during the movement of the person. The second one (Motion History Image, MHI) is a scalar image, which indicates the temporal history of motion (e.g., more recently moving pixels are brighter). MEI and MHI temporal templates are then matched to stored instances of views of known actions.

A technique for human motion recognition in an unconstrained environment, incorporating hypotheses which are probabilistically propagated across space and time, is presented in Bregler (1997). EM clustering, recursive Kalman and Hidden Markov Models are used as well. The feasibility of this method is tested on classifying human gait categories (running, walking and skipping). HMMs are quite often used for classifying and recognizing human dynamics. In Pavlovic & Rehg (2000), HMMs are compared with switching linear dynamic systems (SLDS) towards human motion analysis. It is argued that the SLDS framework demonstrates greater descriptive power and consistently outperforms standard HMMs on classification and continuous state estimation tasks, although the learning-inference mechanism is complicated.

Finally, a novel approach for the identification of human actions in an office (entering the room, using a computer, picking up the phone, etc.) is presented in Ayers & Shah (2001). The novelty of this approach consists in using prior knowledge about the layout of the room. Action identification is modeled by a state machine consisting of various states and the transitions between them. The performance of this system is affected if the skin area of the face is occluded, if two people get too close and if prior knowledge is not sufficient. This approach may be applicable in surveillance systems like those ones described in the next section.

Applications

3D HBMs have been used in a wide spectrum of applications. This section is only focused on the following four major application areas: a) Virtual reality; b) Surveillance systems; c) User interface; and d) Medical or anthropometric applications. A brief summary is given below.

Virtual Reality

The efficient generation of 3D HBMs is one of the most important issues in all virtual reality applications. Models with a high level of detail are capable of conveying emotions through facial animation (Aubel, Boulic & Thalmann, 2000). However, it is still nowadays very hard to strike the right compromise between realism and animation speed. Balcisoy et al. (2000) present a combination of virtual reality with computer vision. This system—*augmented reality system*—allows the interaction of real and virtual humans in an augmented reality context. It can be understood as the link between computer graphics and computer vision communities.

Kanade, Rander & Narayanan (1997) present a technique to automatically generate 3D models of real human bodies, together with a virtual model of their surrounding environment, from images of the real world. These virtual models allow a spatio-temporal view interpolation and the users can select their own viewpoints, independent of the actual camera positions used to capture the event. The authors have coined the expression *virtualized reality* to call their novel approach. In the same direction, Hoshnio (2002) presents a model-based synthesis and analysis of human body images. It is used in virtual reality systems to imitate appearance and behavior of a real-world human from video sequences. Such a human model can be used to generate multiple-views, merge virtual objects and change motion characteristics of human figures in video. Hilton et al. (1999) introduce a new technique for automatically building realistic models of people for use in virtual reality applications. The final goal is the development of an automatic low-cost modeling of people suitable for populating virtual worlds with personalised avatars. For instance, the participants in a multi-user virtual environment could be represented by means of a realistic facsimile of their shape, size and appearance. The proposed technique is based on a set of low-cost color images of a person taken from four orthogonal views. Realistic representation is achieved by mapping color texture onto the 3D model.

Surveillance Systems

Another important application domain is surveillance systems. Smart surveillance systems, capable of more than single-motion detection, can take advantage of the study of 3D human motion analysis by incorporating specific knowledge about human shape and appearance, in order to decrease false alarms. In addition, high-level analysis might even be able to distinguish between simple authorized and non-authorized activities. Wren et al. (1997) present a real-time system for tracking people and interpreting their behavior to be used, for example, in surveillance systems. The proposed system uses a probabilistic approach that segments the subject into a number of blobs and tracks those over time. The disadvantages of the work proposed by Wren et al. (1997) are that the system can only handle a single person with fixed-camera situations.

He & Derunner (2000) propose a different approach based on the study of the periodicity of human actions. Periodic motions, specifically walking and running, can be recognized. This approach is robust over variations in scene background, walking and running speeds and direction of motion. One of the constraints is that the motion must be front-parallel. Gavrila & Philomin (1999) present a shape-based object detection system, which can also be included into the surveillance category. The system detects and distinguishes, in real-time, pedestrians from a moving vehicle. It is based on a template-matching approach. Some of the system's limitations are related to the segmentation algorithm or the position of pedestrians (the system cannot work with pedestrians very close to the camera). Recently Yoo, Nixon & Harris (2002) have presented a new method for extracting human gait signatures by studying kinematics features. Kinematics features include linear and angular position of body articulations, as well as their displacements and time derivatives (linear and angular velocities and accelerations). One of the most distinctive characteristics of the human gait is the fact that it is individualistic. It can be used in vision surveillance systems, allowing the identification of a human by means of its gait motion.

User Interface

User interface is another application domain that takes advantage of 3D human body modeling. Wingbermuehle, Weik & Kopernik (1997) present an approach to generate highly realistic 3D models of participants for distributed 3D videoconferencing systems. Using 3D data obtained by means of stereoscopy, the size and shape of each real person is recovered and represented through a triangular mesh. In addition, texture extracted from the real images is mapped to the 3D models leading to a natural impression. Together with a flexible triangular mesh, a skeleton structure of the human model is build. The latter is used to preserve the anthropomorphic constraint. Cohen, Medioni & Gu (2001) present another real-time 3D human body reconstruction for vision-based perceptual user interface. The proposed system uses multiple silhouettes extracted automatically from a synchronized multi-camera system. Silhouettes of the detected regions are extracted and registered, allowing a 3D reconstruction of the human body using generalized cylinders. An articulated body model (defined by 32 DOF) is fitted to the 3D data and tracked over time using a particle filtering method. Later on, Cohen & Lee (2002) presented an extension of this work that consists of an appearance-based learning formalism for classifying and identifying human postures.

Davis & Bobick (1998a) present a novel approach for extracting the silhouette of a participant within an interactive environment. This technique has been used in Davis & Bobick (1998b) for implementing a virtual Personal Aerobics Trainer (PAT). A computer vision system is responsible for extracting the human body movements and reporting them to a virtual instructor. With this information, the virtual instructor gives comments for pushing or complementing the user in a TV screen interface.

Medical or Antropometric Applications

Medical or anthropometric applications can be roughly divided into three different categories: *human body surface reconstruction*, *internal structure reconstruction* or *motion analysis*. The first two categories mainly rely on range data obtained from a person with a static posture. Therefore, only a static 3D model of the human body is generated. Without motion information, it is difficult to accurately position the corresponding articulated structure inside the surface. Models are represented as single entities by means of smooth surfaces or polygonal meshes (Douros, Dekker & Buxton, 1999). On the contrary, techniques focused on motion analysis for other applications, such as the study of movement disabilities, are based on articulated 3D models. Hence, kinematics and dynamics parameters of the human body need to be determined (Marzani et al. 1997).

Human body surface recovering has an increasing number of applications. For example, Fouchet (1999) presents a 3D body scanner together with a set of algorithms in order to generate a 3D model of the whole human body or part of it. The model includes 3D shapes and the corresponding grey-level information. The main purpose of this system is to provide dermatologists with a new tool able to build a cartography of dermatological lesions of human body skin. The evolution of a dermatological lesion can be followed and the efficiency of different medical treatments can be quantified. In this kind of approach — 3D–scanner-based — the body surface is represented as a single cloud of 3D points. Therefore, if human body parts need to be identified, a segmentation algorithm should be applied in order to cluster those points properly. In this same sense, Werghi & Xiao (2002) present an algorithm for segmenting 3D human body scans. Their work pursues the description of a scanned human body by means of a set of body parts (head, torso, legs, arms and hands). In the same direction, Nurre et al. (2000) propose an algorithm for clustering a cloud of points describing a human body surface.

Internal structure recovering allows 3D reconstruction of anatomical parts for biomedical applications. In addition, it is a powerful way to detect deformities of the human body (e.g., curvature of the spine and axial rotation of individual vertebrae). Medical imaging has become a useful tool for both diagnosing and monitoring such deformities. Durdle et al. (1997) developed a system consisting of computer graphics and imaging tools for the assessment of these kinds of deformities. The proposed system uses stereovision cameras to capture the 3D data. Other techniques for anatomical parts recovering or biomedical applications were presented in Weng, Yang and Pierson (1996) and Tognola et al. (2002). The first one is based on laser spot and two CCD cameras system to recover the 3D data, while the second one is based on an optical flow approach (the object remains stationary while the camera undergoes translational motion). Barron & Kakadiaris (2000) present a four-step technique for estimating a human's anthropometric measurements from a single image. Pose and anthropometric measurements are obtained by minimizing a cost function that computes the difference between a set of user-selected image points and the corresponding projected points of a 3D stick model.

Finally, *motion analysis* systems, which are based on the study of kinematics and dynamics parameters, allow detection of movement disabilities of a given patient. Marzani et al. (1997) and Marzani, Calais & Legrand (2001) present a system for the analysis of movement disabilities of a human leg during gait. The proposed system is based on grey-level image processing without the need of markers. Superquadric surfaces are used to model the legs. This system can be used in human motion analysis for clinical applications, such as physiotherapy.

Conclusions

Human body modeling is a relatively recent research area with a higher complexity than the classical rigid object modeling. It takes advantage of most of the techniques proposed within the rigid object modeling community, together with a prior-knowledge of human body movements based on a kinematics and dynamics study of the human body structure. The huge amount of articles published during the last years involving 3D human body modeling demonstrates the increasing interest in this topic and its wide range of applications. In spite of this, many issues are still open (e.g., unconstrained image segmentation, limitations in tracking, development of models including prior knowledge, modeling of multiple person environments, real-time performance). Each one of these topics represents a stand-alone problem and their solutions are of interest not only to human body modeling research, but also to other research fields.

Unconstrained image segmentation remains a challenge to be overcome. Another limitation of today's systems is that commonly the motion of a person is constrained to simple movements with a few occlusions. Occlusions, which comprise a significant problem yet to be thoroughly solved, may lead to erroneous tracking. Since existence and accumulation of errors is possible, the systems must become robust enough to be able to recover any loss of tracking. Similarly, techniques must be able to automatically self-tune the model's shape parameters, even in unconstrained environments. Moreover, in modeling, dynamics and kinematics should be thoroughly exploited, while in motion recognition, generic human actions should be tackled.

In addition to the aforementioned issues, the reduction of the processing time is still nowadays one of the milestones in human body modeling. It is highly dependent on two factors: on the one hand, computational complexity and, on the other hand, current technology. Taking into account the last years' evolution, we can say that computational complexity will not be significantly reduced during the years ahead. On the contrary, improvements in the current technology have become commonplace (e.g., reduction in acquisition and processing times, increase in the memory size). Therefore, algorithms that nowadays are computationally prohibitive, are expected to have a good performance with the next technologies. The latter gives rise to a promising future for HBM applications and, as an extension, to non-rigid object modeling in general.

The area of human body modeling is growing considerably fast. Therefore, it is expected that most of the current drawbacks will be solved efficiently through the next years. According to the current trend, human body modeling will remain as an application-oriented research field, i.e., the need will dictate the kind of systems that will be developed. Thus, it will be difficult to see general techniques that are valid for all of the cases.

References

Aggarwal, J. K. & Cai, Q. (1999). Human motion analysis: A review. *Computer Vision and Image Understanding, 73*(3), 428-440.

Ali, A. & Aggarwal, J.K. (2001). Segmentation and recognition of continuous human activity. *IEEE Workshop on Detection and Recognition of Events in Video.* Vancouver, Canada.

Aubel, A., Boulic, R. & Thalmann D. (2000). Real-time display of virtual humans: Levels of details and impostors. *IEEE Trans. on Circuits and Systems for Video Technology, Special Issue on 3D Video Technology, 10*(2), 207-217.

Ayers, D. & Shah, M. (2001). Monitoring human behavior from video taken in an office environment. *Image and Vision Computing, 19*(12), 833-846.

Balcisoy, S., Torre, R., Ponedr, M., Fua, P. & Thalmann, D. (2000). Augmented reality for real and virtual humans. *Symposium on Virtual Reality Software Technology*. Geneva, Switzerland.

Barron, C. & Kakadiaris, I. (2000). Estimating anthropometry and pose from a single camera. *IEEE Int. Conf. on Computer Vision and Pattern Recognition*. Hilton Head Island, SC.

Bobick, A. F. & Davis, J. W. (1996). Real-time recognition of activity using temporal templates. *3rd IEEE Workshop on Application of Computer Vision*. Sarasota, FL.

Bregler, C. (1997). Learning and recognizing human dynamics in video sequences. *IEEE Int. Conf. on Computer Vision and Pattern Recognition*. San Juan, PR.

Cai, Q. & Aggarwal, J. K. (1999). Tracking human motion in structured environments using a distributed-camera system. *Trans. on Pattern Analysis and Machine Intelligence*, *21*(12), 1241-1247.

Cédras, C. & Shah, M. (1995). Motion-based recognition: A survey. *Image and Vision Computing*, *13*(2), 129-155.

Cham, T. J. & Rehg, J. M. (1999). A multiple hypothesis approach to figure tracking. *Computer Vision and Pattern Recognition*, *2*, 239-245.

Cohen, I. & Lee, M. (2002). 3D body reconstruction for immersive interaction. *Second International Workshop on Articulated Motion and Deformable Objects*. Palma de Mallorca, Spain.

Cohen, I., Medioni, G. & Gu, H. (2001). Inference of 3D human body posture from multiple cameras for vision-based user interface. *World Multiconference on Systemics, Cybernetics and Informatics*. USA.

Craig, J. (1989). *Introduction to robotics: mechanics and control*. Addison Wesley, 2nd Ed.

Davis, J. & Bobick, A. (1998a). A robust human-silhouette extraction technique for interactive virtual environments. *Lecture Notes in Artificial Intelligence*. N. Magnenat-Thalmann & D. Thalmann (Eds.), Heidelberg: Springer-Verlag, 12-25.

Davis, J. & Bobick, A. (1998b). Virtual PAT: a virtual personal aerobics trainer. *Workshop on Perceptual User Interface*. San Francisco, CA.

Delamarre, Q. & Faugeras, O. (2001). 3D articulated models and multi-view tracking with physical forces. *Special Issue on Modelling People, Computer Vision and Image Understanding*, *81*, 328-357.

Dockstader, S. L. & Tekalp, A. M. (2001). Multiple camera tracking of interacting and occluded human motion. *Proceedings of the IEEE*, *89*(10), 1441-1455.

Douros, I., Dekker, L. & Buxton, B. (1999). An improved algorithm for reconstruction of the surface of the human body from 3D scanner data using local B-spline patches. *IEEE International Workshop on Modeling People*. Corfu, Greece.

Durdle, N., Raso, V., Hill, D. & Peterson, A. (1997). Computer graphics and imaging tools for the assessment and treatment of spinal deformities. *IEEE Canadian Conference on Engineering Innovation: Voyage of Discover*. St. Johns, Nfld., Canada.

Fouchet, X. (1999). *Body modelling for the follow-up of dermatological lesions*. Ph.D. Thesis. Institut National Polytechnique de Toulouse.

Gavrila, D. M. (1999). The visual analysis of human movement: A Survey. *Computer Vision and Image Understanding, 73*(1), 82-98.

Gavrila, D. M. & Davis, L. (1996). 3D model-based tracking of humans in action: A multi-view approach. *IEEE Int. Conf on Computer Vision and Pattern Recognition*. San Francisco, CA.

Gavrila, D. M. & Philomin, V. (1999). Real-time object detection for "smart" vehicles. *IEEE International Conference on Computer Vision*. Kerkyra, Greece.

Haritaoglu, I., Harwood, D. &. Davis, L. S. (1998). W^4: real-time system for detecting and tracking people. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Haritaoglu, I., Harwood, D. & Davis, L. S. (2000). W^4: Real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 22* (8), 809-830.

He, Q. & Debrunner, C. (2000). Individual recognition from periodic activity using hidden Markov models. *IEEE Workshop on Human Motion 2000.* Los Alamitos, CA.

Hilton, A., Beresford, D., Gentils, T., Smith, R. & Sun, W. (1999). Virtual people: Capturing human models to populate virtual worlds. *IEEE Proceedings of Computer Animation*. Geneva, Switzerland.

Hoshnio, J. (2002). Building virtual human body from video. *IEEE Proceedings of Virtual Reality 2002*. Orlando, FL.

Howe, N., Leventon, M. & Freeman, W. (1999). Bayesian reconstruction of 3D human motion from single-camera video. *Advances in Neural Information Processing Systems 12 Conf.*

Humanoid Animation Group (H-anim). Retrieved from the World Wide Web: http://www.h-anim.org.

Isard, M. & Blake, A. (1998). CONDENSATION-conditional density propagation for visual tracking. *International Journal on Computer Vision,* 5-28.

Iwasawa, S., Ohya, J., Takahashi, K., Sakaguchi, T., Ebihara, K. & Morishima, S. (2000). Human body postures from trinocular camera images. *4th IEEE International Conference on Automatic Face and Gesture Recognition*. Grenoble, France.

Kanade, T., Rander, P. & Narayanan, P. (1997). Virtualized reality: Constructing virtual worlds from real scenes. IEEE Multimedia.

Marzani, F., Calais, E. & Legrand, L. (2001). A 3-D marker-free system for the analysis of movement disabilities-an application to the Legs. *IEEE Trans. on Information Technology in Biomedicine, 5*(1), 18-26.

Marzani, F., Maliet, Y., Legrand, L. & Dusserre, L. (1997). A computer model based on superquadrics for the analysis of movement disabilities. *19th International Conference of the IEEE, Engineering in Medicine and Biology Society*. Chicago, IL.

Moeslund, T. B. & Granum, E. (2001). A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding, 81*(3), 231-268.

Nurre, J., Connor, J., Lewark, E. & Collier, J. (2000). On segmenting the three-dimensional scan data of a human body. *IEEE Trans. on Medical Imaging, 19*(8), 787-797.

Okada, R., Shirai, Y. & Miura, J. (2000). Tracking a person with 3D motion by integrating optical flow and depth. *4th IEEE International Conference on Automatic Face and Gesture Recognition.* Grenoble, France.

Park, S. & Aggarwal, J. K. (2000). Recognition of human interaction using multiple features in grayscale images. *15th International Conference on Pattern Recognition.* Barcelona, Spain.

Paul, R. (1981). *Robot manipulators: mathematics, programming and control*. Cambridge, MA: MIT Press.

Pavlovic, V. & Rehg, J. M. (2000). Impact of dynamic model learning on classification of human motion. *IEEE International Conference on Computer Vision and Pattern Recognition*. Hilton Head Island, SC.

Plänkers, R. & Fua, P. (2001). Articulated soft objects for video-based body modelling. *IEEE International Conference on Computer Vision*. Vancouver, Canada.

Preda, M. (Ed.). (2002). MPEG-4 Animation Framework eXtension (AFX) VM 9.0.

Preda, M. & Prêteux, F. (2001). Advanced virtual humanoid animation framework based on the MPEG-4 SNHC Standard. *Euroimage ICAV 3D 2001 Conference*. Mykonos, Greece.

Rosales, R., Siddiqui, M., Alon, J. & Sclaroff, S. (2001). Estimating 3D Body Pose using Uncalibrated Cameras. *IEEE International Conference on Computer Vision and Pattern Recognition.* Kauai Marriott, Hawaii.

Sato, K. & Aggarwal, J. K. (2001). Tracking and recognizing two-person interactions in outdoor image sequences. *IEEE Workshop on Multi-Object Tracking*. Vancouver, Canada.

Sidenbladh, H., Black, M. J. & Sigal, L. (2002). Implicit probabilistic models of human motion for synthesis and tracking. *European Conf. on Computer Vision*. Copenhagen, Denmark.

Sminchisescu, C. & Triggs, B. (2001). Covariance scaled sampling for monocular 3D body tracking. *IEEE International Conference on Computer Vision and Pattern Recognition*. Kauai Marriott, Hawaii.

Solina, F. & Bajcsy, R. (1990). Recovery of parametric models from range images: the case for superquadrics with global deformations. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, *12*(2), 131-147.

Tognola, G., Parazini, M., Ravazzani, P., Grandori, F. & Svelto, C. (2002). Simple 3D laser scanner for anatomical parts and image reconstruction from unorganized range data. *IEEE International Conference on Instrumentation and Measurement Technology*. Anchorage, AK.

Utsumi, A., Mori, H., Ohya, J. & Yachida, M. (1998). Multiple-view-based tracking of multiple humans. *14th International Conference on Pattern Recognition*. Brisbane, Qld., Australia.

Wachter, S. & Nagel, H. (1999). Tracking persons in monocular image sequences. *Computer Vision and Image Understanding*, *74*(3), 174-192.

Weng, N., Yang, Y. & Pierson, R. (1996). 3D surface reconstruction using optical flow for medical imaging. *IEEE Nuclear Science Symposium*. Anaheim, CA.

Werghi, N. & Xiao, Y. (2002). Wavelet moments for recognizing human body posture from 3D scans. *Int. Conf. on Pattern Recognition*. Quebec City, Canada.

Wingbermuehle, J., Weik, S., & Kopernik, A. (1997). Highly realistic modeling of persons for 3D videoconferencing systems. *IEEE Workshop on Multimedia Signal Processing.* Princeton, NJ, USA.

Wren, C. & Pentland, A. (1998). Dynamic models of human motion. *IEEE International Conference on Automatic Face and Gesture Recognition*. Nara, Japan.

Wren, C., Azarbayejani, A., Darrell, T. & Pentland, A. (1997). Pfinder: real-time tracking of the human body. *IEEE Trans. on Pattern Analysis and Machine Intelligence,19*(7), 780-785.

Yamamoto, M., Sato, A., Kawada, S., Kondo, T. & Osaki, Y. (1998). Incremental tracking of human actions from multiple views. *IEEE International Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA.

Yoo, J., Nixon, M. & Harris, C. (2002). Extracting human gait signatures by body segment properties. *5th IEEE Southwest Symposium on Image Analysis and Interpretation*, Santa Fe, CA.

Chapter II

Virtual Character Definition and Animation within the MPEG-4 Standard

Marius Preda
GET/Institut National des Télécommunications, France

Ioan Alexandru Salomie
ETRO Department of the Vrije Universiteit Brussel, Belgium

Françoise Preteux
GET/Institut National des Télécommunications, France

Gauthier Lafruit
MICS-DESICS/Interuniversity MicroElectronics Center (IMEC), Belgium

Abstract

Besides being one of the well-known audio/video coding techniques, MPEG-4 provides additional coding tools dedicated to virtual character animation. The motivation of considering virtual character definition and animation issues within MPEG-4 is first presented. Then, it is shown how MPEG-4, Amendment 1 offers an appropriate framework for virtual human *animation and compression/transmission. It is shown how this framework is extended within the new MPEG-4 standardization process by: 1) allowing the animation of any kind of articulated model, and 2) addressing advanced modeling and animation concepts, such as "Skeleton, Muscle and Skin"-based approaches. The new syntax for node definition and animation stream is presented and discussed in terms of a generic representation and additional functionalities. The biomechanical properties, modeled by means of the character skeleton that defines the bone influence on the skin region, as well as the local spatial deformations simulating muscles, are supported by specific nodes. Animating the virtual character consists in instantiating bone transformations and muscle control curves. Interpolation techniques, inverse kinematics, discrete cosine transform and arithmetic encoding techniques make it possible to provide a highly compressed animation stream. Within a dedicated modeling approach — the so-called MESHGRID — we show how the bone and muscle-based animation mechanism is applied to deform the 3D space around a humanoid.*

Context and Objectives

The first 3D virtual human model was designed and animated by means of the computer in the late 70s. Since then, virtual character models have become more and more popular, making a growing population able to impact the every day, real world. Starting from simple and easy-to-control models used in commercial games as those produced by Activision or Electronic Arts, to more complex virtual assistants for commercial[1] or informational[2] web sites, to the new stars of virtual cinema[3], television[4] and advertising[5], the 3D character model industry is currently booming.

Moreover, the steady improvements within the distributed network area and advanced communication protocols have promoted the emergence of 3D communities[6] and immersion experiences (Thalmann, 2000) in distributed 3D virtual environments.

Creating, animating and, most of all, sharing virtual characters over Internet or mobile networks requires unified data formats. If some animation industry leaders try — and sometimes succeed[7,8] — to impose their own formats in the computer world, the alternative of an open standard is the only valid solution ensuring interoperability requirements, specifically when hardware products are to be built.

A dream of any content producer can be simply formulated as "creating once and re-using forever and everywhere, in any circumstances." Nowadays, content is carried by heterogeneous networks (broadcast, IP, mobile), available anywhere and for a large scale of devices (PCs, set-top boxes, PDAs, mobile phones) and profiled with respect to the user preferences. All these requirements make the chain where content is processed more and more complicated and a lot of different actors must interfere: designers, service providers, network providers, device manufacturers, IPR holders, end-users and so on. For each one, consistent interfaces should be created on a stable and standardized basis.

Current work to provide 3D applications within a unified and interoperable framework is materialized by 3D graphics interchange standards such as VRML[9] and multimedia 2D/3D standards, such as MPEG-4 (ISOIEC, 2001). Each one addresses, more or less in a coordinated way, the virtual character animation issue. In the VRML community, the H-Anim[10] group released three versions of their specifications (1.0, 1.1 and 2001), while the SNHC[11] sub-group of MPEG also released three versions: MPEG-4 Version 1 supports face animation, MPEG-4 Version 2 supports body animation and MPEG-4 Part 16 addresses the animation of generic virtual objects. In MPEG-4 the specifications dealing with the definition and animation of avatars are grouped under the name FBA — Face and Body Animation — and those referring to generic models under the name BBA — Bone-based Animation. The next section analyses the main similarities and differences of these two standardization frameworks.

The VRML standard deals with a textual description of 3D objects and scenes. It focuses on the spatial representation of such objects, while the temporal behaviour is less supported. The major mechanism for supporting animation consists of defining it as an interpolation between key-frames.

The MPEG-4 standard, unlike the previous MPEG standards, does not only cope with highly efficient audio and video compression schemes, but also introduces the fundamental concept of media objects such as audio, visual, 2D/3D, natural and synthetic objects to make up a multimedia scene. As established in July 1994, the MPEG-4 objectives are focused on supporting new ways (notably content-based) of communicating, accessing and manipulating digital audiovisual data (Pereira, 2002). Thus, temporal and/or spatial behaviour can be associated with an object. The main functionalities proposed by the standard address the compression of each type of media objects, hybrid encoding of the natural and synthetic objects, universal content accessibility over various networks and interactivity for the end-user. In order to specify the spatial and temporal localisation of an object in the scene, MPEG-4 defines a dedicated language called BIFS — Binary Format for Scenes. BIFS inherits from VRML the representation of the scene, described as a hierarchical graph, and some dedicated tools, such as animation procedures based on interpolators, events routed to the nodes or sensor-based interactivity. In addition, BIFS introduces some new and advanced mechanisms, such as compression schemes to encode the scene, streamed animations, integration of 2D objects and advanced time control.

In terms of functionalities related to virtual characters, both VRML and MPEG-4 standards define a set of nodes in the scene graph to allow for a representation of an avatar. However, only the MPEG-4 SNHC specifications deal with streamed avatar animations. A major difference is that an MPEG-4 compliant avatar can coexist in a hybrid environment and its animation can be natively synchronized with other types of media objects, while the H-Anim avatar can only exist in a VRML world and must be animated by VRML generic, usually non-compressed, animation tools.

Now that the reasons of virtual character standardization within MPEG-4 become clearer, the question is how to find the good compromise between the need for freedom in content creation and the need for interoperability? What exactly should be standardized, fixed, invariant and in the meantime, ideally impose no constraints on the designer creativity? The long-term experience that the MPEG community has makes it possible to formulate a straight and solid resolution: in the complex chain of content producing-transmitting-consuming, the interoperability is ensured by only standardizing the data representation format at the decoder side. Pushing this concept to its extreme, an MPEG ideal tool is that one for which two requirements are satisfied: the designer can use any production tool he/she possesses to create the content and it can be possible to build a full conversion/mapping tool between this content and an MPEG compliant one. The same principle has been followed when MPEG released the specifications concerning the definition and the animation of the virtual characters, and specifically human avatars: there are no “limits” on the complexity of the avatar with respect to its geometry or appearance and no constraints on the motion capabilities.

The animation method of a synthetic object is strongly related to its definition model. A simple approach, often used in cartoons, is to consider the virtual character as a hierarchical collection of rigid geometric objects called segments, and to obtain the animation by transforming these objects with respect to their direct parents. The second method consists in considering the geometry of the virtual character as a unique mesh and to animate it by continuously deforming its shape. While the former offers low animation complexity, with the price of the seams at the joints between the segments, the latter ensures a higher realism of the representation, but requires more computation. Both modeling/animation methods are supported by the MPEG-4 standard, as will be extensively shown in this chapter. Its structure is as follows. The first section presents the tools adopted in the MPEG-4 standard related to the specification and encoding of the synthetic object’s geometry in general. Specifically, techniques based on IndexedFaceSet, Wavelet Subdivision Surfaces and MeshGrid are briefly described. The next section describes in detail the first avatar animation framework, adopted in MPEG-4 in 1998, i.e., the FBA framework. Here, the avatar body is structured as a collection of segments individually specified by using IndexedFaceSet. The avatar face is a unique object animated by deformation controlled by standardized feature points. The section, *Virtual Characters in MPEG-4 Part 16*, introduces a generic deformation model, recently adopted by MPEG-4 (December, 2002), called BBA. It is shown how this model is implemented through two deformation controllers: bones and muscles. The generality of the model allows it to directly animate the seamless object mesh or the space around it. Moreover, hierarchical animation is possible when considering the BBA technique and specific geometry representations, such as Subdivision Surfaces or MeshGrid. This advanced animation is presented in the section, Hierarchic Animation: Subdivision Surface and MeshGrid.

MPEG-4's Geometry Tools in a Nutshell

The simplest and most straightforward representation of 3D objects, dating from the early days of computer graphics, is the IndexedFaceSet model. It consists in approximating the geometry as a collection of planar polygons defined with the aid of a list of vertex coordinates. Unfortunately, IndexedFaceSet has not been designed to deal efficiently with highly detailed and complex surfaces, consisting of ten to hundreds of thousands of triangles, necessary to achieve realistic rendering of objects found in daily life. Even more important than compact storage is the possibility to scale the complexity of the surface representations according to the capacity of the digital transmission channels or to the performance of the graphics hardware on the target platform. Another vital issue for the animation of objects is the support for free-form modeling or animation, offered by the representation method.

As a response to these new demands, several compact surface encoding techniques have been devised during the last years. A *first category* of techniques tries to respect as much as possible the vertex positions and their connectivity as defined in the initial IndexedFaceSet description. The second category opts for an alternative surface representation method, enabling higher compression ratios and extra features, such as support for animation. The second approach is more complex, certainly at the encoding stage, since a surface described with the alternative surface representation will have to be fitted within certain error bounds to the initial mesh description.

A representative for the first category of techniques is the Topological Surgery (TS) representation (Taubin, 1998a), which compresses the connectivity of manifold polygonal meshes of arbitrary topological type, as well as the vertex locations. In order to support multi-resolution and fast interaction via progressive transmission, the TS method has been combined with the Progressive Forest Split scheme (PFS) described in Taubin (1998b). TS is used to encode the base mesh, while the PFS being applied to encode a sequence of the forest split refinement operations is allowed to generate higher resolutions of the base mesh. TS and PFS approaches have been promoted to the MPEG-4 and are known as 3D Mesh Coding (3DMC).

The second category is represented by WAVELET SUBDIVISION SURFACES (WSS), a method recently introduced in MPEG-4 (ISOIEC, 2003). A base mesh is used as the seed for a recursive subdivision process, during which the 3D details (i.e., the wavelet coefficients) needed to obtain finer and finer approximations to the original shape are added to the new vertex positions predicted by the subdivision scheme. WSS does not attempt to encode the base mesh—a method like TS can be used for that purpose. Instead, the focus is on parameterization of the underlying surface's shape over a triangular or quadrilateral base domain, in order to obtain a multi-resolution mesh. Therefore, the main problem of these approaches lies in finding an optimal base mesh, which is in general a computing intensive process.

Another way of representing shapes in MPEG-4 is the MESHGRID compression tool (ISOIEC, 2003), which is an arbitrary, cutting plane-based representation scheme suited for encoding the surfaces obtained from discrete 3D data sets (e.g., 3D medical images, processed range scanner data, quadrilateral meshes, or generic models defined by means of implicit surfaces). The resulting hierarchical surface representation defines the wireframe of the object's skin by: (1) describing the connectivity between the vertices in an efficient implicit way, called the connectivity-wireframe (CW); and (2) positioning these vertices in relation to a regular 3D grid of points that characterizes the space inside and outside the skin, called the reference-grid (RG). Therefore, the MESHGRID surface representation lies somewhat in between the two categories: (1) it has features common to the first category in the way the connectivity-wireframe is encoded; and (2) it exploits wavelet-based, multi-resolution techniques for refining the shape, through the RG.

Based on these classes of geometry representation, MPEG-4 has defined dedicated tools for the definition and the animation of virtual characters. The next sections describe these tools in more detail.

Virtual Character in MPEG-4 Version 1 and 2: Face and Body Animation

First efforts to standardize the animation of a human-like character (an avatar) within MPEG-4 were finalized at the beginning of 1999. Published under the name FBA, they dealt with specifications for defining and animating the avatar. This section first describes the node specification for Face and Body objects, and then describes how to create and animate an FBA compliant avatar model. The compression schemes of animation parameters are presented in the third subsection. Finally, local deformation issues are tackled.

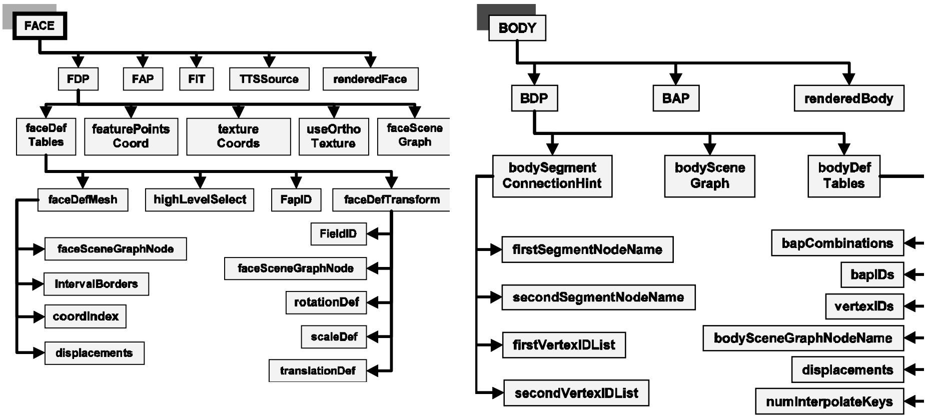
Face and Body Animation Nodes Specification

A key concept in the MPEG-4 standard is the definition of the scene, where text, 2D and 3D graphics, audio and video data can (co)exist and (inter)act. A scene is represented as a tree, where each object in the scene is the instantiation of a node or a set of nodes. Compression representation of the scene is done through the BInary Format for Scene (BIFS) specification (ISOIEC, 2001). Special transformations and grouping capabilities of the scene make it possible to cope with spatial and temporal relationships between objects.

The first version of the standard addresses the animation issue of a virtual human face, while Amendment 1 contains specifications related to virtual human body animation. In order to define and animate a human-like virtual character, MPEG-4 introduces the so-called FBA Object. Conceptually, the FBA object consists of two collections of nodes in a scene graph grouped under the so-called Face node and Body node (Figure 1), and a dedicated compressed stream. The next paragraph describes how these node hierarchies include the definition of the geometry, the texture, the animation parameters and the deformation behaviour.

The structure of the Face node (Figure 1a) allows the geometric representation of the head as a collection of meshes, where the face consists of a unique mesh (Figure 2a). The shape and the appearance of the face is controlled by the FDP (Facial Definition Parameter) node through the faceSceneGraph node for the geometry, and the *textureCoord* and *useOrthoTexture* fields for the texture. Moreover, a standardized number of control points are attached to the face mesh through the *featurePointsCoord* field as shown in Figure 3. These points control the face deformation. The deformation model is enriched by attaching parameterisation of the deformation function within the neighbourhood of the control points through the faceDefTables node.

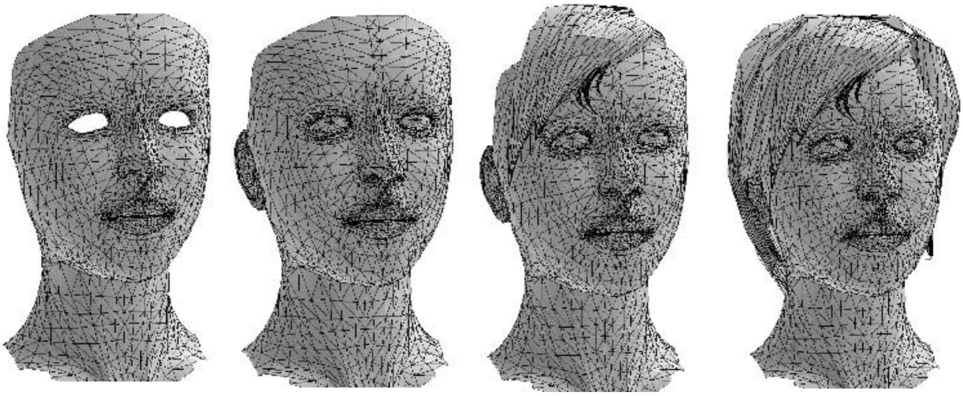
Figure 1. The MPEG-4 FBA related nodes.



(a) The Face scene graph hierarchy.

(b) The Body scene graph hierarchy.

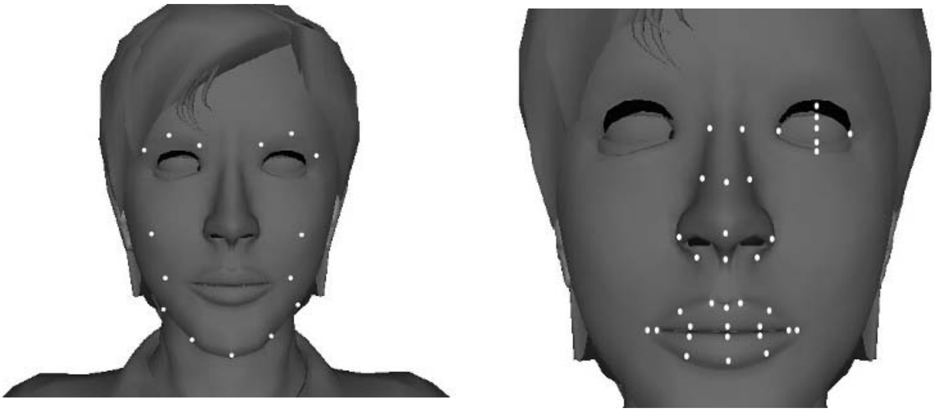
Figure 2. The face as a unique mesh (a) and the head as a collection of meshes (b), (c) and (d).



(a) *(b)* *(c)* *(d)*

The face expressions and animation are controlled by the FAP (Face Animation Parameter) node, which is temporal and updated by the FBA decoder. Animations can be performed at a high level, using a standardized number of expressions and visemes, as well as at a low level by directly controlling the feature points. In this case, a standardized number of key points (84), corresponding to the human features (e.g., middle point of upper lip) is defined on the face surface (Figure 3a and b). The complete animation is then performed by

Figure 3. Examples of standardized key-points on the face object.



(a) (b)

deforming the mesh in the vicinity of the key points (Doenges, 1997; Escher, 1998; Lavagetto, 1999).

A virtual body object is represented in the scene graph as a collection of nodes grouped under the so-called Body node (Figure 1b).

The BDP (Body Definition Parameters) node controls the intrinsic properties of each anatomical segment of the avatar body. It includes information related to the avatar body representation as a static object composed by anatomical segments (bodySceneGraph node), and deformation behaviour (bodyDefTables and bodySegmentConnectionHint nodes) (ISOIEC, 2001). Body definition parameters are virtual character-specific. Hence the complete morphology of an avatar can readily be altered by overriding the current BDP node. Geometrically, the static definition of the virtual body object is a hierarchical graph consisting of nodes associated with anatomical segments and edges. This representation could be compressed using the MPEG-4 3D Mesh coding (3DMC) algorithm (Taubin, 1998a) defining subpart relationships, grouped under the bodySceneGraph node. The MPEG-4 virtual avatar is defined as a segmented virtual character, using the H-Anim V2.0 nodes and hierarchy: Humanoid, Joint, Segment, and Site nodes.

The BAP (Body Animation Parameters) node contains angular values and defines the animation parameters as extrinsic properties of an anatomical segment, i.e., its 3D pose with respect to a reference frame attached to the parent segment. The orientation of any anatomical segment is expressed as the composition of elementary rotations, namely twisting, abduction and flexion. Here, 296 angular joint values are enough to describe any 3D posture of a virtual human-like character. The angular values are specified with respect to the local 3D coordinate system of the anatomical segment. The origin of the local coordinate system is defined as the gravity centre of the joint contour common to the considered anatomical segment and its parent. The rotation planes are specified and/or anatomical segment rotation axes are standardized. Contrary to BDPs, BAPs are meant to be generic, i.e., independent of, or poorly dependent upon, the avatar geometry.

Figure 4 and Figure 5 illustrate the rotation axes for arm, forearm and fingers. For a complete definition of the axes associated with all body segments, one is referred to ISOIEC (2001).

Figure 4. Standardized rotation axes attached to shoulder, elbow and wrist.

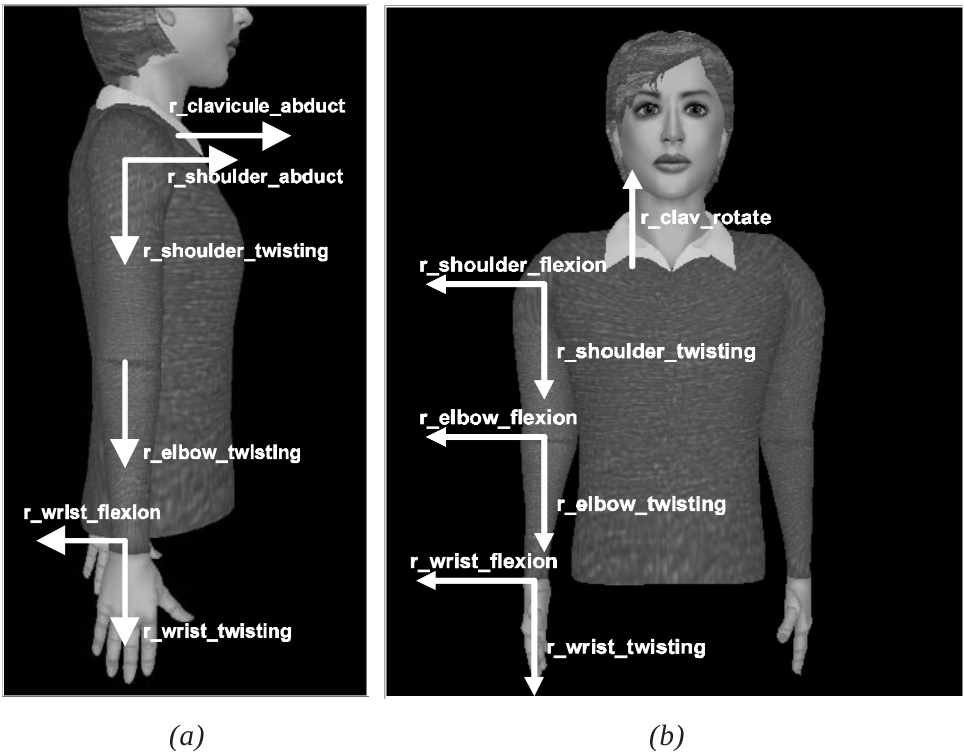
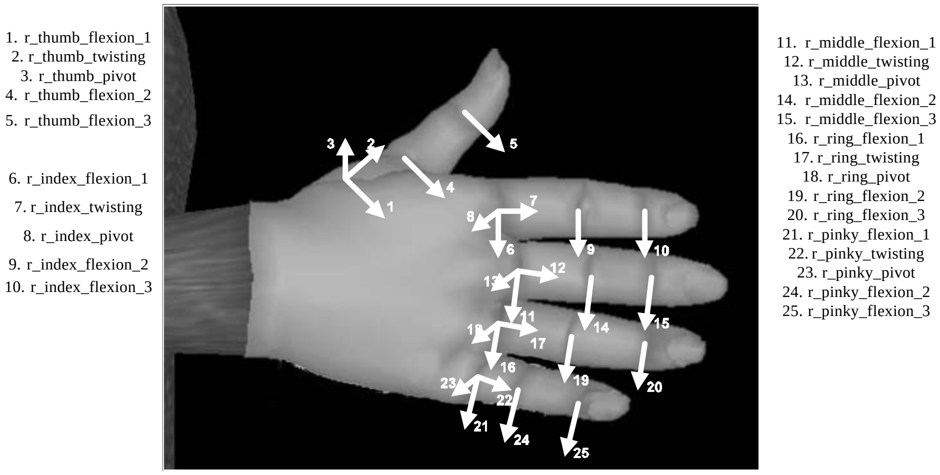


Figure 5. Standardized rotation axes attached to fingers.

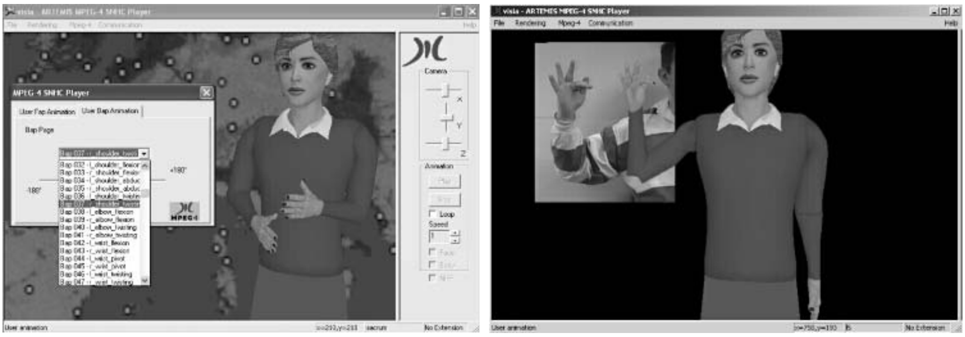


Creating FBA Compliant Data: Avatar Model and Animation Parameters

Since the manners for obtaining the content are various and quickly evolve in time, the FBA specifications do not mandate any specific methods for obtaining a real description of a 3D human body and its associated animation parameters. Defining only the representation format, the specifications allow a free development of content creation. To address the avatar modeling issue, we developed the Virtual Human Modeling (VHM) authoring tool to help a designer to obtain — from a scanned geometric model of a human-like avatar — an articulated version, compliant with the FBA specifications. The authoring tool is made up of three parts:

- a *Segmentation Module* to split the original object into a set of 3D objects using the geodesical segmentation algorithm described in Preda (2002).
- a *Building Module* to build the articulated character by using the FBA predefined hierarchy by setting up parent-child property of the anatomical segments previously segmented.
- a *Face Parameterisation Module* to specify the control points of the face mesh and to define the face influence region associated with each control point.

Figure 6. The 3AI: (a) BAP editing by using a dedicated user interface allowing to tune the value for each BAP, (b) interactive tracking of gestures in image sequences.


MPEG-4 SNHC Player
User Fap Animation
User Bap Animation
Bap Page
Bap 032 - l_shoulder_flexion
Bap 033 - r_shoulder_flexion
Bap 034 - l_shoulder_abduc
Bap 035 - r_shoulder_abduc
Bap 036 - l_shoulder_twistin
Bap 038 - l_elbow_flexion
Bap 039 - r_elbow_flexion
Bap 040 - l_elbow_twisting
Bap 041 - r_elbow_twisting
Bap 042 - l_wrist_flexion
Bap 043 - r_wrist_flexion
Bap 044 - l_wrist_pivot
Bap 045 - r_wrist_pivot
Bap 046 - l_wrist_twisting
Bap 047 - r_wrist_twisting
+180°
-180°
MPEG-4
Camera
X
Y
Z
Animation
Loop
Speed
User animation
No Extension
visia - ARTEMIS MPEG-4 SNHC Player
File
Rendering
Mpeg-4
Communication
Help

(a) (b)

In order to address the avatar animation issue, we developed a dedicated authoring tool named ARTEMIS Animation Avatar Interface (3AI), to ease the editing and extraction of face and body animation parameters. The 3AI authoring tool also provides the following functionalities: (1) loading an FBA compliant avatar; (2) 3D composition of objects such as images, video sequences, avatars or anatomical part models (face, hand, arm, and 3D scenes; (3) calibration of 3D face and body models according to the anthropometric characteristics of the actor in a video sequence (dimensions of the palm, length of the fingers); (4) face tracking in natural video sequences and FAP instantiation (Malciu, 2000); (5) interactive extraction of BAPs to describe any posture or corresponding to the posture shown in the video sequence (see Figure 6b); (6) animation parameters editing through selection of key-frames and end-effector positioning; and finally (7) avatar animation according to a FAPs/BAPs file source and network resource (e.g., UDP server). Some of these functionalities are visually represented in Figure 6.

Animation Parameters Compression

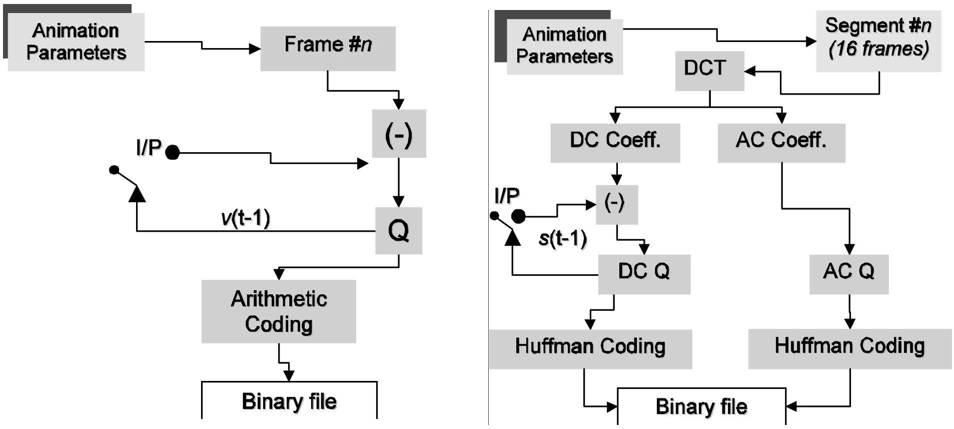
The FBA specifications provide, for both face and body animation parameters, two encoding methods (predictive and DCT-based).

In the first method (Figure 7a), FAPs/BAPs are coded with a predictive coding scheme. For each parameter to be coded in frame n, the decoded value of this parameter in frame n-1 is used as a prediction. Depending on precision requirements on FAPs/BAPs, different quantization step sizes could be applied. They consist of a local (FAP/BAP specific) step size and a global one (used for bit-rate control). The quantized prediction error is then encoded by arithmetic coding. Taking into account the fact that natural human motions are constrained by physical properties, the range of each FAP/BAP is limited to a specific interval. Using this property, the coding efficiency is increased.

The DCT-based coding method (Figure 7b) splits FAP/BAP time sequences into segments made of 16 consecutive frames. Three steps are necessary to encode each segment: (1) the determination of the 16 coefficient values using discrete cosine transform (DCT); (2) quantizing and coding the alternative coefficients (AC); and (3) predictively coding and quantizing the continuous component (DC) coefficients. The global quantization step Q for the DC coefficients can be controlled and the AC coefficients quantization step is set to 1/3 of Q. The DC coefficients of an intra-coded segment are encoded as it is and, for an inter-coded segment, the DC coefficient of the previous segment is used as a prediction of the current DC coefficient. The prediction error and alternative component coefficients (AC), (for both inter and intra-coded segments), are coded using Huffman tables.

The current FBA encoder implementation from the MPEG reference software,[12] as well as commercial[13] and academic implementations (Capin, 1999; Preda 2002) shows a very low bit rate for compression of the animation parameters, ranging from 2kbps for the face, up to 40 kbps for the entire body.

Figure 7. Decoding block diagram.



(a) Frame predictive-based method. *(b) DCT-based method.*

Table 1. Bit-rates [kbps] for the DCT-based coding scheme. Q denotes the global quantization value.

Sign	Q=1	Q=2	Q=4	Q=8	Q=12	Q=12	Q=24	Q=31
Bitrate[kbps]	12.08	10.41	8.5	7.29	6.25	5.41	3.95	3.33

We have tested the FBA compression techniques on a data set representing real sign language content. In this sequence, both arms and the body of the avatar are animated. The frame-rate for this content is 20 fps. When dealing with a wide range of target bit rates, the DCT-based method has to be used. Since the DCT-based method uses a 16 frames temporal buffer, an animation delay occurs. For sign language application, this delay introduces a small non-synchronisation, but does not affect the message comprehension. In the case of applications that require near loss-less compression and exact synchronization of the animation with another media, the use of the frame predictive-based method is recommended. In order to increase the efficiency of the arithmetic encoder, the MPEG-4 FBA specifications standardize a set of ranges for each animation parameter. The global quantization step is used here for scaling the value to be encoded in the corresponding range. Each animation parameter is encoded with the same number of bits inside this range. If the obtained scaled value is outside of the range, a higher quantization step has to be used.

In our tests related to sign language, when using the frame predictive-based method, a quantization value bigger than four has to be used and the obtained bit-rate is close to 6.2 kbps. The compression results for the DCT-based method are presented in Table 1.

The low bit-rate, less than 15 kbps, obtained by compressing the animation parameters, while keeping visual degradation at a satisfactory level, allows animation transmission in a low bit-rate network.

Local Deformations

The segmented nature of an FBA compliant avatar has the main disadvantage that during the animation seams will occur at the joints between the segments. To overcome this limitation, a special tool based on the so-called Body Deformation Tables (BDTs) has been introduced in MPEG-4. The principle consists in adding small displacements for the vertices near the joint. Thereby, during the animation the borders of two segments remain connected. BDTs specify a list of vertices of the 3D model, as well as their local displacements as functions of BAPs (ISOIEC, 2001). An example of BDTs' use is described in Preda (2002).

The generation of the deformation tables requires additional animation methods and the size of the deformation tables can comprise up to 50% of the size of the entire model, depending on the requested deformation accuracy. To overcome this limitation, new deformation tools have been adopted by the MPEG-4 standard, as part of the AFX specifications. These tools are generic and support the animation of any kind of 2D/3D synthetic objects. The next section shows how one can use them for defining and animating virtual characters.

Virtual Characters in MPEG-4 Part 16: The Skeleton, Muscle and Skin (SMS) Framework

The purpose of this section is to introduce the new animation framework for generic virtual objects as specified in the Part 16 of the MPEG-4 standard. This framework is founded on a generic deformation model (Preda, 2002c), which relies on a deformation controller defined by means of a geometric support, an influence volume around this support, and a measure of affectedness within the influence volume. With respect to these elements, a classification of the main deformation techniques reported in the literature is presented.

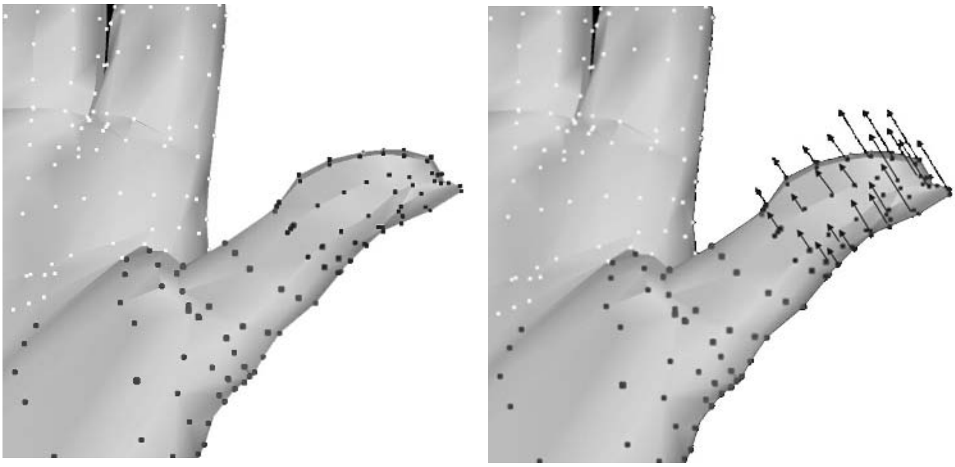
In the following, we introduce two instances of the 1D deformation model, which offer efficient control of the geometrical support and appropriate volume specification: (1) bone controller and (2) muscle controller. The Skeleton, Muscle and Skin (SMS) framework is built around these concepts: bone and muscle. We show how they can be used to define and animate generic virtual characters. In order to embed generic virtual characters into a 3D scene, the SMS architecture is provided with the scene graph nodes. The efficient representation of the animation parameters is addressed by: (1) enriching the animation capabilities with temporal interpolation and inverse kinematics support; and (2) adapting two data compression techniques to the SMS animation data, namely the predictive and DCT-based methods.

Synthetic Object Deformation: Toward a Unified Mathematical Model

A key issue pointed out in the previous section refers to realistic animation that FBA tools cannot efficiently achieve. One of the main reasons for this comes from considering the avatar as a segmented mesh and performing the animation by applying a rigid geometric transformation to each segment. In order to overcome this limitation, the object should be considered as a seamless mesh and animated by means of deformations. The generic principle is described and illustrated below.

Let $M(\Omega)$ be a seamless mesh, where $\Omega = \{v_0, v_1, ... v_n\}$ is the set of the mesh vertices and let $(\Omega_i)_i$ be a family of non-empty subsets of Ω (Figure 8a). A local deformation function $\varphi_i : \Omega \to \mathbf{R}^3$ makes it possible to move a vertex $v \in \Omega_i$ into the new position expressed as $v + \varphi_i(v)$ (Figure 8b and c). Here, φ_i is extended from Ω_i to Ω as the null function, i.e., $\forall v \in \Omega \setminus \Omega_i, \varphi_i(v) = 0$. Note that the family $(\Omega_i)_i$ is not necessarily a partition of Ω. In particular, $\bigcup \Omega_i$ can be a strict subset of Ω (some vertices may remain unchanged) and for two distinct subsets Ω_i and Ω_j, the intersection $\Omega_i \cap \Omega_j$ can be non-empty. The deformation satisfies the superposition principle, i.e., the deformation induced by both φ_i and φ_j at a vertex v belonging to $\Omega_i \cap \Omega_j$ is expressed as the sum $\varphi_i(v) + \varphi_j(v)$ (Figure 8d). In order to achieve a compact description and an efficient implementation of a deformation model, the notion of a *deformation controller* is introduced. It is defined as a triplet made of: (1) the support S associated with a n dimensional (nD) geometric object ($n \in \{0, 1, 2, 3\}$); (2) an influence volume $V(S)$ associated with S; and (3) the affectedness measure μ, defined on $V(S)$ and characterizing the intrinsic deformation properties of the influence volume.

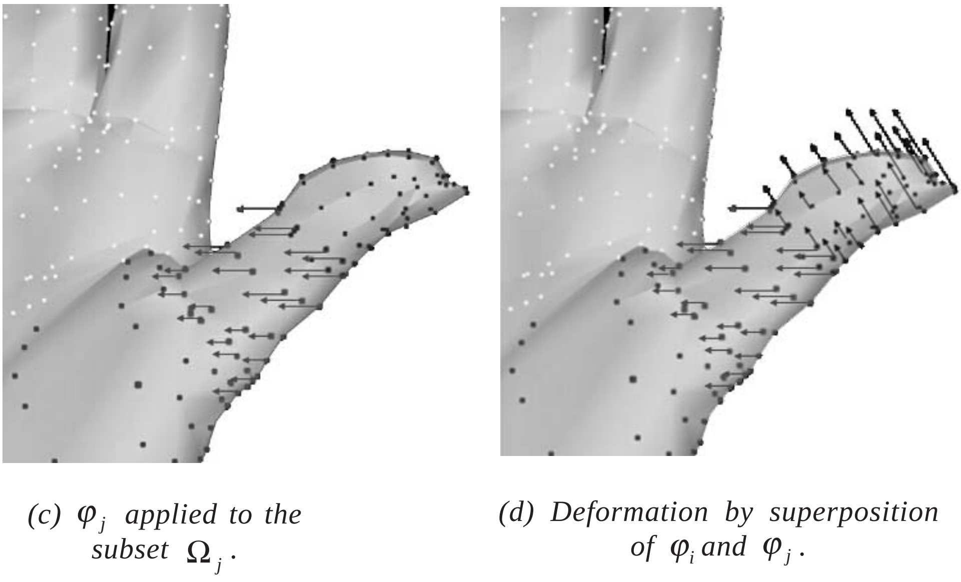
Figure 8. Mesh deformation principle.



(a) Mesh partition.

(b) Deformation function φ_i applied to the subset Ω_i.

Figure 8. Mesh deformation principle. (continued)

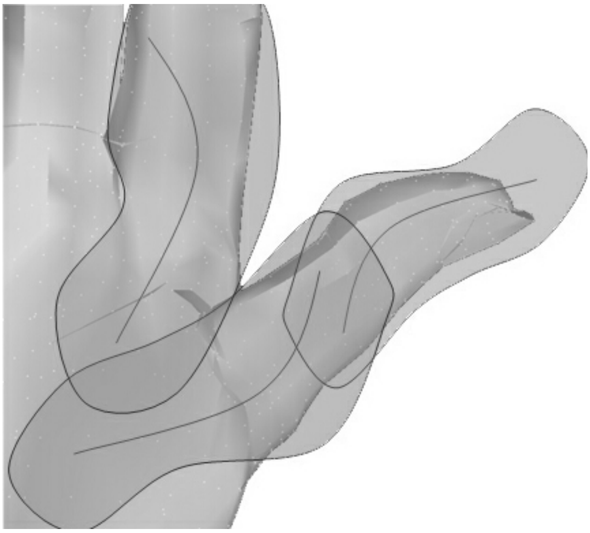


(c) φ_j applied to the subset Ω_j.

(d) Deformation by superposition of φ_i and φ_j.

A family $(C_i = \{S_i, V(S_i), \mu_i\})_i$ of controllers is said to be associated with a mesh $M(\Omega)$ (Figure 9) if and only if the following relationships are fulfilled: (1) $\forall i \in \{0, 1, ..., n\}$ $\Omega_i = \Omega \cap V(S_i)$ and (2) there exists a mapping ψ_i from Ω_i to S_i, such that any vertex $v \in \Omega_i$ is linked to a set of the elements of S_i. Applying

Figure 9. Deformation controllers associated with the mesh.



an affine (or not) transformation, T_i to C_i is equivalent in defining a deformation function φ_i on Ω_i such that:

$$\forall v \in \Omega_i, \varphi_i(v) = \mu_i(v) \sum_{\xi_k \in \psi_i(v)} \omega_k \left[T_i(\xi_k) - \xi_k \right], \tag{1}$$

where μ_i is the affectedness measure associated with C_i and ω_k is a weigh coefficient.

In practice, the transformation T_i is applied to the controller and is propagated within the influence volume $V(S_i)$ according to the affectedness measure μ_i, which plays the role of a weighting function. When $\psi_i(v)$ is reduced to a single element, Equation (1) is simplified:

$$\forall v \in \Omega_i, \varphi_i(v) = \mu_i(v) \left[T_i(\psi_i(v)) - \psi_i(v) \right]. \tag{2}$$

This controller-based deformation framework enables the unification of the different deformation techniques reported in the literature with respect to the dimension of the controller support. Typically, the most representative technique of a volume controller-based approach is the lattice-based deformation model. In this case, the 3D grid is considered as the controller volume. The 1D controller-based approach covers most of the deformation techniques currently used, namely: spline-based and skeleton-based. The 0D controller principle is used in the case of deformation tables (a particular case being described in the previous section in the case of FBA), cluster-based and morphing-based approaches.

In practice, choosing an appropriate controller results from a trade-off between: (1) the complexity of representing the controller directly linked to its dimension; and (2) the distribution of mesh vertices affected by the controller, specifically by choosing the most appropriate influence volume.

An optimal balance is obtained by using a 1D controller. The support of the controller is thus easy to control (the number of parameters is small) and the corresponding influence volume covers a large class of configurations. The new specifications of the MPEG-4 standard support this approach for animating an articulated virtual character in the case of two specific 1D controllers, namely a segment and a curve defined as a NURBS, referred to as bone and muscle-based deformation, respectively.

The following sections describe in details each of these 1D controllers.

Skeleton, Muscle and Skin Framework

Bone and muscle controllers for animating an articulated object

An articulated virtual character and, generally, an articulated synthetic object, also called kinematics linkage, is composed of a set of rigid links which are connected at the joints. In the case of a seamless virtual character, a rigid link is associated with each anatomical bone of the skeleton.

In order to define the static 3D posture of an articulated virtual character, including geometry, colour and texture attributes, the approach here proposed consists of considering the entire virtual character as a single 3D mesh referred to as skin.

During the animation stage, the anatomical bones can only be affected by rigid transformations and cannot be locally deformed. Nevertheless, realistic animations can be obtained by local skin deformations for simulating muscular activity effects. In order to fulfil this requirement, curve-based entities are attached at an arbitrary level of the skeleton.

Two issues are addressed by the SMS framework. The first one deals with the definition of the skinned model as a static model described by its geometry and its appearance. In order to perform this task, a hierarchical skeleton and a collection of muscles are introduced. The second issue deals with the animation of articulated models together with a compressed representation of the animation parameters.

Defining a SMS virtual character requires us to specify a set of static attributes (geometry, texture, etc.), as well as a deformation model. From a geometric point of view, an SMS synthetic model is represented in such a way that the set of vertices which belong to the skin of the virtual character is specified as a unique list. In order to define various appearances at different levels of the SMS virtual character, distinct shapes can be concatenated, provided that all of the shapes composing the skin share the same list of vertices. This type of representation avoids getting seams on the skin during the animation stage, while preserving the possibility to define various sets of colour, texture and material attributes at different levels of the skin.

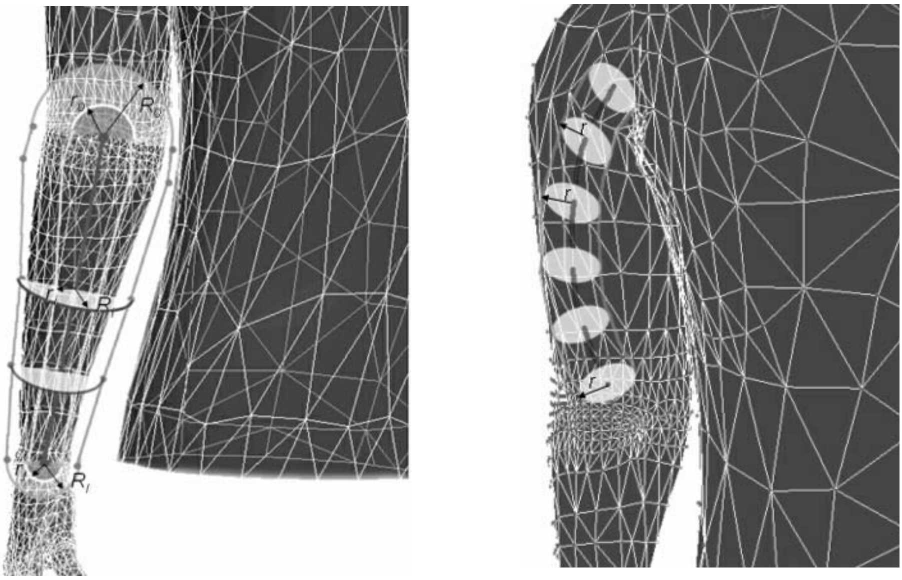
The animation behaviour of a skinned model is defined by means of skeleton and muscle layers together with their properties. The skeleton is a hierarchical structure built from bones. A bone is defined as a segment of length *l* by means of: (1) a geometric transformation of the bone, defined with respect to its parent in the skeleton hierarchy; (2) a model of influence of the bone movement on the surface of the articulated model; and (3) inverse kinematics constraints. A muscle layer is a collection of individual muscles, each one being attached to a bone. Specifically, it is defined by means of: (1) a curve represented as a NURBS which can be deformed; and (2) a model of influence of the curve deformation on the skin of the model.

Each bone and each muscle influences the skin deformation. Thus, by changing a bone position or orientation or by deforming the muscle curve, the skin vertices belonging to the associated influence volume will be displaced accordingly. Here, the skinned virtual character definition consists in specifying for each bone and for each muscle an associated influence volume, i.e., the subset of the affected skin vertices together with the related measure of affectedness through weighting coefficients. The influence can be directly specified vertex-by-vertex by the designer of the synthetic model, or computed before performing the animation. In the first case, the list of affected vertices and the weighting coefficients are included in the bone/muscle definition. In the second case, distinct approaches are proposed for bones and muscles, respectively. The following sub-section present different strategies for computing the bone and the muscle influence volume, respectively.

Bone and muscle-based modeling and animation

The bone influence volume is defined as the support of the affectedness measure μ. Here μ is expressed as a family of functions $(\mu_d)_{d\in[0,l]} \cup \mu_{0^-} \cup \mu_{l^+}$. μ_d is

Figure 10. Bone and muscle modeling.



(a) Forearm bone influence volume. *(b) The muscle influence volume.*

defined on the perpendicular plane located at distance d from the bone origin. The support of μ_d is partitioned into three specific zones ($Z_{d,in}$, $Z_{d,mid}$ and $Z_{d,ext}$) by two concentric circles characterised by their respective radius r_d and R_d (Figure 10). μ_d is then defined as follows:

$$\mu_d(x) = \begin{cases} 1, & x \in Z_{\text{int}} \\ f\left(\dfrac{\delta(x, Z_{ext})}{R_d - r_d}\right), & x \in Z_{mid}, \\ 0, & x \in Z_{ext} \end{cases} \tag{3}$$

where $\delta(x, Z_{ext})$ denotes the Euclidean distance from x to Z_{ext} and $f(\cdot)$ is a user-specified fall-off to be chosen among the following functions: $x^3, x^2, x, \sin(\frac{\pi}{2}x), \sqrt{x}$ and $\sqrt[3]{x}$. This set of functions allows a large choice for designing the influence volume and ensures the generality of the model.

The affectedness measure μ_{0^-} (respectively μ_{l^+}) is defined in the same manner, but using two half-spheres of radius r_0 and R_0 (respectively r_l and R_l) as illustrated in Figure 10a.

The bone influence volume being defined, animating the virtual character consists of deforming its mesh by translating its vertices according to the bone transformation.

Here only affine transformations are applied to the bone controller. In virtual character animation, the most widely used geometric transformation consists in changing the orientation of the bone with respect to its parent in the skeleton hierarchy. Thus, the bone can be rotated with respect to an arbitrary axis. However, when special effects are needed, the bone can also be translated. For instance, in cartoon-like animations, thinning and thickening the skin envelope are frequently used. For such effects, the bone transformation must contain a scale component specified with respect to a pre-defined direction.

The general form of the geometric transformation of a bone b is expressed as a 4×4 element matrix T obtained as follows:

$$T = TR^w b \cdot R^w b \cdot S^w b \tag{4}$$

where $TR^w b$, $R^w b$, $S^w b$ give the bone translation, rotation and scale, respectively, expressed in the world coordinate system.

In practice, the computations are performed in the local coordinate system attached to the bone. In order to perform the scaling with respect to a pre-defined direction, the matrix SR_b performs a pre-orientation of the bone. Thus, in this system, the T transformation is expressed as:

$$T = TR_b \cdot C_b \cdot R_b \cdot SR_b \cdot S_b \cdot (SR_b)^{-1} \cdot (C_b)^{-1} \qquad (5)$$

where matrix C_b allows it to pass from the bone local coordinate system to the world coordinate system.

Once the bone geometric transformation is computed, it is possible to compute the new position of the skin vertices in the bone influence volume according to Equation (2).

The SMS framework does not limit the number of bones used for animating a virtual character. Consequently, local deformation effects can be obtained by creating and animating additional bones. If such an approach can be used with a certain success for simulating limited local deformation, the efficiency may strongly decrease when considering realistic animations. To address this kind of animation, the second controller, ensuring muscle-like deformation was introduced.

The muscle influence volume is constructed as a tubular surface generated by a circle of radius r moving along the NURBS curve (Figure 10b). The affectedness function is then defined as follows:

$$\mu(v) = \begin{cases} 0 & \delta\left(v_i, \psi\left(v_i\right)\right) > r \\ f\left(\dfrac{r - \delta\left(v_i, \psi\left(v_i\right)\right)}{r}\right) & \delta\left(v_i, \psi\left(v_i\right)\right) \le r\text{'} \end{cases} \qquad (6)$$

where δ denotes the Euclidean distance, $f(\cdot)$ is to be chosen among the following functions: $x^3, x^2, x, \sin(\frac{\pi}{2}x), x^{1/2}$ and $x^{1/3}$, and ψ is the function assigning to v its correspondent point on the muscle curve.

A muscle is designed as a curve, together with an influence volume on the virtual character's skin. In order to build a flexible and compact representation of the muscle shape, a NURBS-based modeling is used. Animating the muscle consists here of updating the NURBS parameters. The main advantages of NURBS-based modeling are the accurate representation of complex shapes. The control of the curve shape is easily addressed. A set of control points coordinates, weights and knots, ruling the shape of the curve, can be directly manipulated in order to control the local curvature.

As NURBS theory is widely reported in the literature, we shall not get into details here, however, the interested reader is referred to Piegl (1997). NURB-based modeling is fully supported within the SMS framework. Animating a muscle consists of updating the values of its NURBS parameters. Once the muscle transformation is computed, it is possible to compute the new position of the skin vertices in the muscle influence volume according to Equation (2).

Skeleton, Muscle and Skin Nodes Specification

One of the main purposes of the SMS framework is to allow the definition and the animation of a virtual character within a hybrid 3D scene. In this context, a scene graph architecture to describe the SMS elements is proposed. This architecture is built according to the VRML and MPEG-4 scene graph definition rules. The structure of the proposed architecture, therefore, relies on the definition of scene graph nodes.

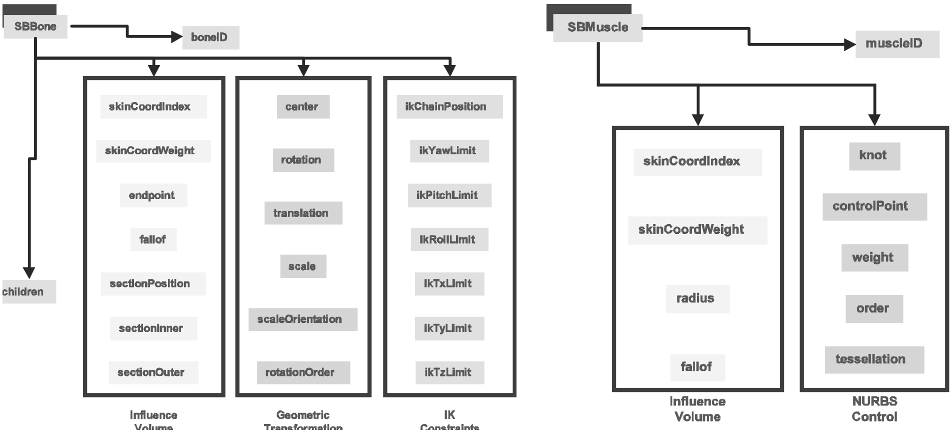
At the root of the SMS-related node hierarchy, a SBVCAnimation node is defined. The main purpose of this node is to group together a subset of virtual characters of the scene graph and to attach to this group an animation resource (textual or binary). An SMS virtual character is defined as a SBSkinnedModel node and it is related to a collection of bones, each one defined as a SBBone node, together with a collection of muscles, defined as SBMuscle nodes. An optimal modeling issue is addressed by defining the SBSegment node. In addition, the SBSite node allows defining semantic regions in the space of virtual character. The extensive description of the node interfaces is outside of the goal of this chapter and one can find them in the MPEG-4 standard Part 16 published by ISO (ISOIEC, 2003). Nevertheless, we briefly present the SBBone, SBMuscle and SBSkinnedModel in order to illustrate the concepts discussed above.

The fields of the SBBone node are illustrated in Figure 11a.

The SBBone node specifies four types of information, namely: semantic data, bone-skin influence volume, bone geometric transformation, and bone IK constraints. Each of these components is further detailed in the following.

The SBBone node is used as a building block in order to describe the hierarchy of the articulated virtual character by attaching one or more child objects. The children field has the same semantic as in MPEG-4 BIFS. During the animation, each bone can be addressed by using its identifier, *boneID*. This field is also present in the animation resource (textual or binary). If two bones share the same

Figure 11. The fields of the SBBone (a) and SBMuscle (b) node.



(a) (b)

identifier, their geometric transformations have the same set of parameter values.

The bone-skin influence volume described is implemented as follows: the specification of the affected vertices and the measure of affectedness are obtained by instantiating the *skinCoordIndex* and *skinCoordWeight* fields. The *skinCoordIndex* field enumerates the indices of all skin vertices affected by the current bone. The *skinCoordWeight* field is a list of values of affectedness measure (one for each vertex listed in *skinCoordIndex*). The influence volume specified with respect to a certain number of planes (discretisation step) is computed as follows. The *sectionInner* (respectively, *sectionOuter*) field is a list of inner (respectively, outer) radii of the influence volume corresponding to different planes. The *sectionPosition* field corresponds to the distance *d*. The *falloff* field specifies the choice of the measure of affectedness function as follows: -1 for x^3, 0 for x^2, 1 for x, 2 for $\sin(\frac{\pi}{2}x)$, 3 for $\sqrt{x}$ and 4 for $\sqrt[3]{x}$. The location of the bone is specified by the *center* and *endpoint* fields.

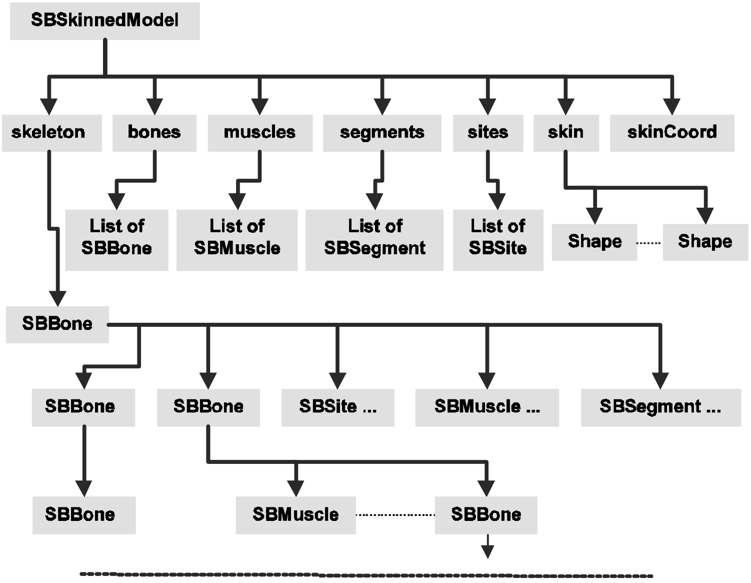
The possible 3D geometric transformation consists of (in this order): (1) (optionally) a non-uniform scale; (2) a rotation with respect to an arbitrary point and axis; and (3) a translation. The transformation is obtained through the fields *rotation*, *translation*, *scale* and *scaleOrientation*. The global geometric transformation of a given child of a bone is obtained by composing the bone transformation of the child with the parent. The *rotationOrder* field contains information related to the conversion from the quaternion representation, used in the node data, to the Euler angles-based representation, used in the animation resource (file or stream), and *vice versa* (Shoemake, 1994).

The Inverse Kinematics information related to a bone refers to the position of the bone in a kinematics chain and to the definition of possible movement constraints of this one bone.

The fields of the SBMuscle node are illustrated in Figure 11b. The SBMuscle node enables us to add information relative to local deformation for simulating muscle activity at the skin level. Mainly two kinds of information are represented by this node: the influence volume and the curve form. The muscle influence volume described is supported as follows. The specification of the affected vertices and of the measure of affectedness is performed by instantiating the vertices list (*skinCoordIndex*) and the affectedness list (*skinCoordWeight*). The influence volume is computed by using the *radius* and *falloff* fields. The radius field specifies the maximum distance for which the muscle will affect the skin. The *falloff* field specifies the choice of the measure of affectedness function as follows: -1 for x^3, 0 for x^2, 1 for x, 2 for $\sin(\frac{\pi}{2}x)$, 3 for $\sqrt{x}$ and 4 for $\sqrt[3]{x}$.

The animation of the muscle curve is based on NurbsCurve structure as defined in Grahn (2001) and uses the following fields: *controlPoint* — containing the coordinates and the weights — and knot. The main fields of the SBSkinnedModel node are illustrated in Figure 12.

Figure 12. The fields of the SBSkinnedModel node.



The SBSkinnedModel node is the root used to define one SMS virtual character and it contains the definition parameters of the entire seamless model or of a seamless part of the model. Mainly, this node contains the model geometry and the skeleton hierarchy. The geometry is specified by *skinCoord* field — (a list containing the 3D coordinates of all the vertices of the seamless model) and the skin field — (a collection of shapes which share the same *skinCoord).* This mechanism allows us to consider the model as a continuous mesh and, at the same time, to attach different attributes (e.g., colour, texture, etc.) to different parts of the model. The skeleton field contains the root of the bone hierarchy. All the bones and muscles belonging to the skinned model are contained in dedicated lists.

Once the skinned model is defined in a static position, the animation is obtained by updating, at time samples, the geometric transformation of the bones and muscles. In order to ensure a compact representation of these parameters, the MPEG-4 standard specifies a dedicated stream, the so-called BBA stream.

Skeleton, Muscle, and Skin Animation Stream

Animation principle and resource representation

To address streamed animation, MPEG-4 considers the animation data independent of the model parameters. Thus, the model is transmitted or loaded at the beginning of the animation session and the animation parameters are sequentially transmitted at each frame. Two representation techniques for the animation data format are supported: the first one corresponds to a non-compressed (human readable) format. This is useful when editing the animation parameters. In this case, the file format is XMT (Kim, 2000) compliant in order to allow easy editing and data exchange. This representation is called "SMS textual." The second representation is a compressed data format. By using appropriate compression schemes, low bit-rate animation transmission is performed. This representation is called "SMS binary."

Conceptually, both animation data formats use the same representation of the geometrical transformation parameters. The next sections describe this representation.

Animation parameter representation

The key point for ensuring a compact representation of the SMS animation parameters consists of decomposing the geometric transformations into elementary motions. Thus, when only using, for example, the rotation component of the bone geometric transformation, a binary mask indicates that the other components are not involved. In order to deform a muscle only by translating a control point, a binary mask has to specify that weight factors and basis functions are not used. Since the animation system does not systematically use all of the elements of the transformations associated with bones and muscles, this approach produces a very compact representation of the animation stream. Moreover, the compactness of the animation stream can still be improved when dealing with rotations. During the animation, the rotation of a bone with respect to its parent is a typically used technique. In the definition of the bone node, the rotation is represented as a quaternion. However, many motion editing systems use the rotation decomposition with respect to the Euler's angles. In practice, when less than three angles describe a joint transformation due to the nature of the joint, a Euler's angle-based representation is more appropriate. Thus, to get a more compact animation stream, a rotation is represented, in the animation resource, as Euler's angles-based decomposition.

In Craig (1989), it is shown that there are 24 different ways to specify a rotation by using a triplet of angles. By introducing a parameter characterizing the 24 possible combinations of the Euler's angles, Shoemake (1994) demonstrates that there is a one-to-one mapping between the quaternion (or rotation matrix) representation and the pair given by the Euler's angles and the introduced parameter. In order to take this into account, a parameter called *rotationOrder* has been introduced into the bone node.

For the rest of the bone transformation components (translation, scale, etc.), the representation in the animation resource is identical to the representation in the nodes.

Temporal frame interpolation

The issue of temporal frame interpolation has been often addressed in the computer animation literature (Foley, 1992; O'Rourke, 1998). From simple linear interpolation, appropriate for translations, to more complex schemes based on high-degree polynomials, or quaternions, which take orientation into account, a large number of techniques are available. The advantages and the disadvantages of each one are well known. Many of these techniques are supported by most of the current animation software packages. Temporal frame interpolation is intensively used to perform animation from textual description or from interactive authoring. One in order to reduce the size of the transmitted data ,and the second to ease authoring, it is allowed to specify the animation parameters for the key-frames and not only frame-by-frame. However, in order to ensure the consistency of the data over different decoder implementation, the interpolation schemes were also standardized.

For real-time purposes, a linear interpolation is used for the translation and scale components and a spherical linear quaternion interpolation is used for the rotation and *scaleOrientation* components.

Animation frame

In an SMS textual or binary format, for each key frame, two types of information are defined: a vector corresponding to the animation mask, called *animationMaskVector*, which indicates the components of the geometrical transformation to be updated in the current frame; and a vector corresponding to the animation values called *animationValueVector* which specifies the new values of the components to be updated.

Let us describe the content of each of these vectors. For the exact syntax, one can refer to the ISOIEC (2003).

- animationMaskVector

  In the animation mask of a key-frame, a positive integer *KeyFrameIndex* indicates to the decoder the number of frames which have to be obtained by temporal interpolation. If this number is zero, the decoder sends the frame directly to the animation engine. Otherwise, the decoder computes n intermediate frames (n=*KeyFrameIndex*) and sends them to the animation engine, together with the content of the received key-frame.

  Some bones or muscles of the SMS virtual character may not be animated in all frames. The *boneIDs* and *muscleIDs* of the updated bones and muscles, respectively, are parts of the *animationMaskVector*. In addition, *animationMaskVector* contains the animation mask of each bone, *boneAnimationMaskVector*, and the animation mask of each muscle, *muscleAnimationMaskVector*. These vectors are detailed below.

- animationValueVector

  The *animationValueVector* contains the new values of each bone and muscle geometric transformation that have to be transmitted and it is obtained by concatenation of all the *boneAnimationValueVector* and *muscleAnimationValueVector* fields.

  For compression efficiency, SMS stream specifications limit the maximum number of bones and muscle nodes to 1,024 each. These bone and muscle nodes can belong to one or more skinned models and are grouped in a SBVCAnimation node. Thus, the fields *boneID* and *muscleID* must be unique in the scene graph and their values must lie in the interval [0, …1,023].

- boneAnimationMaskVector

  To address high compression efficiency, a hierarchical representation of the bone motion is used. At the first level, the bone motion is decomposed into translation, rotation, scale, scale orientation and center transformation. At the second level, all of these components that are set to 1 in the bone mask, are individually decomposed in elementary motions (e.g., translation along the X axis, rotation with respect to Y axis). This hierarchical processing makes it possible to obtain short mask vectors. The size of the *boneAnimationMaskVector* can vary from two bits (corresponding to a single elementary motion) to 21 bits (all the components of the local transformation of the bone change with respect to the previous key-frame).

- boneAnimationValueVector

  The *boneAnimationValueVector* contains the values to be updated corresponding to all elementary motions with a mask value of 1. The order of the elements in the *boneAnimationValueVector* is obtained by analyzing *boneAnimationMaskVector*.

- muscleAnimationMaskVector

  The muscle animation parameters in the SMS stream are coordinates of the control points of the NURBS curve, weights of the control points and/or knot values.

  The number of control points and the number of elements of the knot sequence are integers between 0 and 63 and they are encoded in the *muscleAnimationMaskVector*, after the *muscleID* field. As in the case of the bone, a hierarchical processing is used to represent the mask.

- muscleAnimationValueVector

  The *muscleAnimationValueVector* contains the new values of the muscle animation parameters. As in the case of a bone, this vector is ordered according to the *muscleAnimationMaskVector*.

Since the compression schemes developed in the MPEG-4 FBA framework offer good performances, the two algorithms (predictive and DCT-based) have been adopted for compressing the SMS animation data.

SMS versus FBA

A comparative analysis of the FBA and SMS frameworks is synthesized in Table 2 below.

While the FBA is founded on the representation of avatars as segmented characters, that makes it an appropriate framework for cartoon-like applications, SMS offers a higher degree of realistic representation, dealing with the concept of skeleton-driven animation.

When dealing with the avatar body, the FBA standardizes a fixed number of animation parameters (296) by attaching to each anatomical segment up to three rotation angles. The SMS framework does not limit the number of animation parameters (bones and muscles). Moreover, the animation parameters refer to an extended set of geometrical transformations (rotations, translations, scaling factors).

Shape deformations are present in both frameworks. For example, the FBA standardizes a number of control points in order to perform facial deformations, while the SMS allows us to add curve-based deformers at any level of the skin. In FBA, the deformation tools are cluster-based. In SMS, they are curve-based. The FBA standardizes the number and location of the control points, while SMS

Table 2. Main FBA and SMS features (Preda, 2002b).

Criteria	FBA	SMS
Model type	Virtual human character	Generic virtual character
Geometry definition	Segmented character	Seamless character
Hierarchy	Standardized Hierarchy	Hierarchy build on a generic skeleton
Local deformation	Cluster based for face, deformation tables for body	Curve-based deformation
Scene graph nodes	Define a Face and Body Node and use H-Anim PROTOs for specifying the model geometry	Define a own set of 6 nodes
Animation parameters	296 for body, 68 for face	Undefined number of parameters, arbitrary number of bones and muscles are supported
Animation editing support	Forward kinematics	Forward kinematics, inverse kinematics, temporal frame interpolation
Compression	Frame predictive-based, DCT based	Frame predictive-based, DCT based

gives this freedom to the designer, being thus possible to achieve muscle-like deformations on any part of the virtual character's skin in order to get a realistic animation.

Both frameworks address streaming animation and provide low-bit-rate compression schemes. Both FBA and SMS allow the above-mentioned compression methods, frame-based and DCT-based. Moreover, to improve the compression ratio, SMS supports advanced animation techniques, such as temporal frame interpolation and inverse kinematics. For both FBA and SMS, the bit-rate of the compressed stream depends on the movement complexity (number of segments/joints involved in motion) and generally lies in the range of 5-40 kbps, for a frame rate of 25fps.

In the FBA framework, the animation stream contains information relative to the animation of a single human virtual character, while, in the SMS framework, it is possible to animate several characters by using a unique stream. Moreover, the SMS supports the definition and animation of generic 2D/3D objects. This property is very useful when dealing with a scene where a large number of avatars or generic objects are present.

In SMS animation, more complex computations are required than in the case of FBA animation. Thus, concerning the terminal capabilities, dedicated 3D hardware or software optimization is well-suited for implementing SMS animation. However, the SMS deformation mechanism is in line with the development of graphics APIs and graphics hardware.

The deformation based on the bones and muscles controllers can be applied in relation with advanced geometry definition techniques, allowing hierarchical animation. The following section describes such representation techniques (Subdivision Surfaces and MeshGrid) and shows the use of BBA to control the animation of a synthetic object by affecting its surrounding space.

Hierarchic Animation: Subdivision Surfaces and MeshGrid

In this section, methods are presented for performing hierarchical animation, where the displacement of a number of key points is extended to all vertices through an automatic iterative construction/displacement scheme. Subdivision Surfaces and MeshGrid both achieve this goal, but since MeshGrid is more appropriate for defining a "surrounding influence volume" in the skin deformation control presented in previous section, more attention will be devoted to MeshGrid.

Subdivision Surfaces

Subdivision surfaces, originally introduced by Catmull and Clark (1978) and Doo and Sabin (1978), have recently emerged as a useful tool for modeling free-form surfaces. A number of other subdivision schemes have been devised over the years, including Loop's (1987), Dyn et al.'s (known as the "butterfly" scheme) (1990) or Kobbelt's (2000). Subdivision is a recursive refinement process that splits the facets or vertices of a polygonal mesh (the initial "control hull") to yield a smooth limit surface. The refined mesh obtained after each subdivision step is used as the control hull for the next step, and so all successive (and hierarchically nested) meshes can be regarded as control hulls. The refinement of a mesh is performed both on its topology, as the vertex connectivity is made richer and richer, and on its geometry, as the new vertices are positioned in such a way that the angles formed by the new facets are smaller than those formed by the old facets. The interest in considering subdivision surfaces for animation purposes are related to the hierarchical structure: the animation parameters directly affect only the base mesh vertex positions and, for higher resolutions, the vertices are obtained through a subdivision process. Three subdivision surfaces schemes are supported by the MPEG-4 standard: Catmull-Clark, Modified Loop and Wavelet-based. For a detailed description of these methods and how they are implemented in MPEG-4, the reader is referred to ISOIEC (2003).

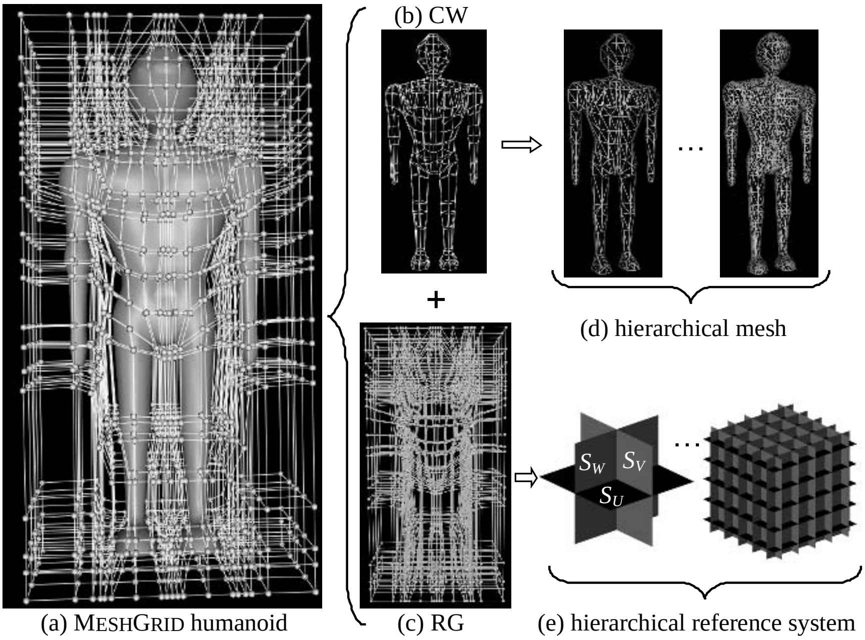
MeshGrid

MeshGrid is a novel surface representation typical for describing surfaces obtained by scanning the shape of "real" or "virtual" objects according to a certain strategy. The scanning strategy should provide a description of the object as a series of open or closed contours.

Virtual Character definition in MeshGrid format

The particularity of the MeshGrid representation lies in combining a wireframe, i.e., the connectivity-wireframe (CW), describing the connectivity between the vertices located on the surface of the object, with a regular 3-D grid of points, i.e., the reference-grid (RG) that stores the spatial distribution of the vertices from the connectivity-wireframe. The decomposition of a MeshGrid object into its components is illustrated in Figure 13 for a multi-resolution humanoid model. Figure 13a shows the MeshGrid representation of the model, which consists of the hierarchical connectivity-wireframe (Figure 13b) and the hierarchical reference-grid (Figure 13c). The different resolutions of the mesh (Figure 13d), which

Figure 13. MeshGrid representation of a humanoid model.



can be used to render the object at the appropriate level of detail, can be obtained from the connectivity-wireframe. The reference-grid is the result of a hierarchical reference system as shown in Figure 13e.

Starting from the humanoid model of Figure 13a, the following sections will discuss the design particularities of the connectivity-wireframe, reference-grid, and their relationship, such that the model can be animated using a hierarchical skeleton-based approach.

The components of the MeshGrid model: RG and CW

The original surface of the humanoid model has been designed by means of implicit functions. A typical way to obtain a MeshGrid object from such an implicit surface definition is to apply a method called TriScan (Salomie, 2002a), which performs the contouring of the surface of the object at specified scanning positions. The scanning positions are defined by the reference system specified for the object.

The reference system consists of three sets of reference surfaces S_U, S_V, S_W, as labeled in Figure 13e. For a better understanding, the reference system has been chosen uniformly distributed. Notice that, usually in a real case, as shown in Figure 13c, the reference grid is non-uniformly distributed. The reference grid is defined by the intersection points between the three sets of reference surfaces S_U, S_V, S_W, as given by Equation (7).

$$RG = \bigcap \left\{ \sum_U S_U, \sum_V S_V, \sum_W S_W \right\} \tag{7}$$

$$CW = \bigcup \left\{ \sum_U C(S_U), \sum_V C(S_V), \sum_W C(S_W) \right\} \tag{8}$$

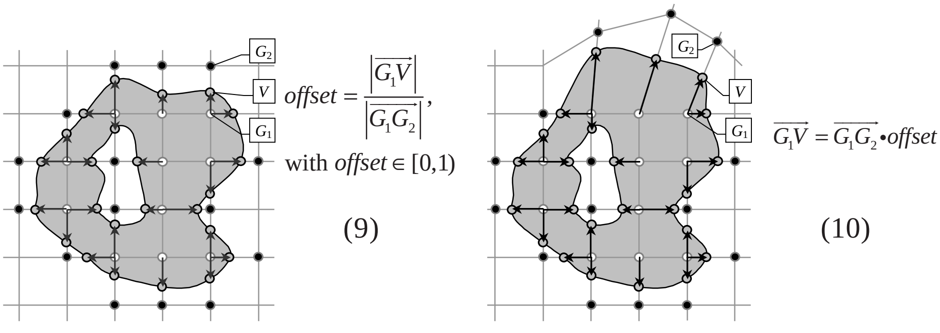
The discrete position (u, v, w) of a reference grid point represents the indices of the reference surfaces $\{S_U, S_V, S_W\}$ intersecting at that point, while the coordinate (x, y, z) of a reference grid point is equal to the coordinate of the computed intersection point.

There is a constraint imposed on the reference surfaces, however. They must be chosen in such a way that the reference surfaces from one set do not intersect each other, but intersect the reference surfaces from the other sets. To obtain the connectivity-wireframe, the TriScan method performs the contouring of the object in each of the reference surfaces S_U, S_V, S_W. Any intersection between two contours defines a vertex. The connectivity-wireframe consists of the set of all vertices generated by the intersections between contours, and the connectivity between these vertices. A mathematical definition of the connectivity-wireframe is given by Equation (8).

In the general case, the connectivity-wireframe is heterogeneous and can be seen as a net of polygonal shapes ranging from triangles to heptagons, where, except for the triangles, the other polygons may not be planar. Therefore, to triangulate the connectivity-wireframe in a consistent way, a set of connectivity rules has been designed especially for that purpose, as explained in Salomie (2002a; Salomie, 2002b).

As one might have noticed, there exists a relationship between the vertices and the reference grid, since a vertex is the intersection point between two contours, therefore, belonging to two reference surfaces from different sets. This relationship can be followed in the 2D cross-section (see Figure 14), inside a reference surface, intersecting the object. Any vertex (label 4), lying on a contour of the object (label 5), is located on a reference grid line (label 1) in between two reference grid points, one inside the object (label 3) and one outside the object (label 2). Notice that a reference grid line is the intersection curve of two

Figure 14. A 2D cross-section through the object.



$$offset = \frac{\left|\overrightarrow{G_1V}\right|}{\left|\overrightarrow{G_1G_2}\right|}, \text{ with } offset \in [0,1) \qquad (9)$$

$$\overrightarrow{G_1V} = \overrightarrow{G_1G_2} \bullet offset \qquad (10)$$

(a) *(b)*

reference surfaces from different sets. As illustrated in Figure 14a, each vertex is attached to a grid position G_1, and the relative position of vertex V with respect to G_1 and G_2 is given by the scalar *offset* (see Equation (9)). When either G_1 or G_2 moves during the animation (shown in Figure 14b), the coordinates (x, y, z) of V can be updated as given by Equation (10).

A multi-resolution model can be designed by choosing a multi-resolution reference system, each resolution level having its corresponding reference grid. The multi-resolution reference system has a hierarchical structure (see Figure 13e), which allows obtaining from the last resolution level reference system any lower-resolution-level reference system by removing the appropriate reference surfaces.

The connectivity-wireframe obtained from a model (Figure 13b), by scanning it according to a hierarchical reference system (Figure 13e), has a hierarchical structure, as well. This can be proven considering that all the vertices from any lower-resolution-level R^l are preserved in the immediate higher-resolution-level R^{l+1}, since the reference system of resolution level R^l is a sub-set of the reference system of resolution level R^{l+1}. In addition, resolution level R^{l+1} will insert new vertices and, therefore, alter the connectivity between the vertices of resolution level R^l. A hierarchical connectivity-wireframe can be decomposed into single-resolution connectivity-wireframes, and each of them can be triangulated to obtain the corresponding mesh, as show in Figure 13d.

Hierarchical reference grid animation

The MeshGrid model is very flexible for animation purposes, since, in addition to the vertex-based animation typical for IndexedFaceSet or SubdivisionSurface representations, it allows for specific animation types, such as: (1) rippling effects by modifying the value of the *offset*, see Equation (10); and (2) reshaping of the regular reference grid. The latter form of animation can be done on a hierarchical multi-resolution basis, and will be exploited for the bone-based animation of the humanoid.

A particularity of the MeshGrid representation is that the hierarchical structure of the reference grid allows the coordinates of the reference grid points of any resolution-level R^{l+1} to be recomputed whenever the coordinates of the reference grid points of the lower resolution-level R^{l} are modified, for instance by the bone-based animation script. For that purpose, "Dyn's four-point scheme for curves" interpolation (Dyn, 1987) is applied.

The compact and scalable MeshGrid stream

The particularities of the MeshGrid representation allow a very compact encoding of the model. In addition, the information inside the compressed stream is organized in regions of interest and levels of refinement such that the needed portions of the mesh can be retrieved at the appropriate resolution and quality level. The high encoding performance is achieved by combining different coding techniques for the different components of the MeshGrid representation, as follows:

1. The surface mesh can be obtained from the connectivity-wireframe by performing a triangulation procedure using some of the same connectivity rules as described in Salomie (2002a). It is, however, more efficient to encode the connectivity-wireframe than the triangulated mesh because of the smaller number of edges that have to be described. For encoding the connectivity-wireframe, a new type of 3D extension of the Freeman chain-code is used, requiring only between one and two bits per edge.
2. The reference grid is a smooth vector field defined on a regular discrete 3D space, each reference grid point being identified by a discrete position (u,v,w). The (x,y,z) coordinates are efficiently compressed using an embedded 3D wavelet-based multi-resolution intra-band coding algorithm.

By combining and applying these coding techniques to multi-resolution objects, the compressed MeshGrid stream can be 2.5 times more compact than the corresponding multi-resolution 3DMC (Taubin, 1998a) stream. When MeshGrid is configured in homogeneous triangular or quadrilateral mesh mode, its compression performances are close to that of WSS, dedicated to triangular or quadrilateral meshes. The encoding performance of the WSS 3D-detail information is, indeed, very high and difficult to surpass. Nevertheless, MeshGrid does not need a separate compression tool, in contrast to WSS for which the base mesh is coded separately, typically with low compression tools like IndexedFaceSet. Therefore, and together with its dedicated animation capabilities, MeshGrid is preferred over WSS for virtual character animation.

Design and Animation of a Virtual Character defined in MeshGrid format

Design of the Virtual Character

According to the bone-based animation requirements, the humanoid model must consist of a global seamless mesh of the entire figure for each resolution level, which should virtually be split into anatomical parts, e.g., shoulder, elbow, wrist, etc., such that the motion of the skeleton can drive the appropriate parts of the mesh.

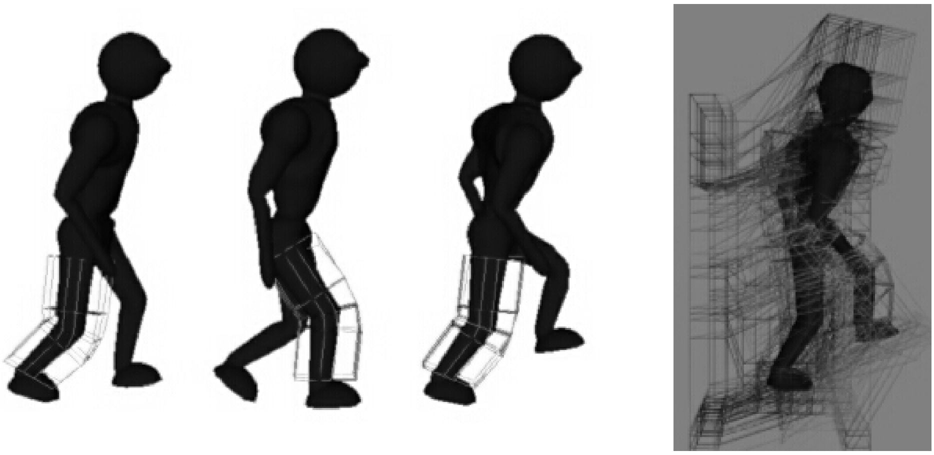
Applying the TriScan method on a humanoid model defined by implicit functions yields as a final result a seamless mesh of the entire object at each resolution level, as shown in Figure 13d. In order to meet the virtual-split requirement for the mesh, the reference system for the humanoid model has to be designed accordingly. As shown in Figure 13a and c, the reference surfaces defining the reference grid have been chosen such that they pass through the anatomical articulations (joints) of the body. Consequently, the single mesh is virtually split into meaningful anatomical parts, which can be driven by the hierarchical skeleton definition from the bones-based animation script. Notice in Figure 13a that the density of the reference grid is higher in the areas belonging to the joints, which will generate a denser mesh for allowing smoother deformations and modeling. The reference system is hierarchical, providing a humanoid model with three resolution levels, as shown in Figure 13d.

Animation of a Virtual Character defined in MeshGrid format

For the hierarchical humanoid model shown in Figure 13, the reference grid is organized in three resolution levels: the first level contains 1,638 points, the second level 11,056 points and the third level contains 83,349 points. The lowest resolution level is chosen as the base level for the animation, while higher mesh resolution levels are improving the smoothness of the deformations at the joints and of the overall shape. It is possible, as well, to simultaneously animate different resolution levels, in case some details only appear at higher resolution levels. An animation sequence of the humanoid model is shown in Figure 15a. In addition to the shaded surface mesh, the reference grid attached to the right leg is displayed, in order to illustrate the path followed and the deformation to which the model is constrained. Each deformation of the reference grid triggers the hierarchical update of the reference grid points belonging to higher resolution levels, and the computation of the vertices coordinates according to Equation (10) is such that the mesh follows the movements applied to the reference grid. As illustrated in Figure 15b, not all of the reference grid points have to be animated, only those where vertices are attached to.

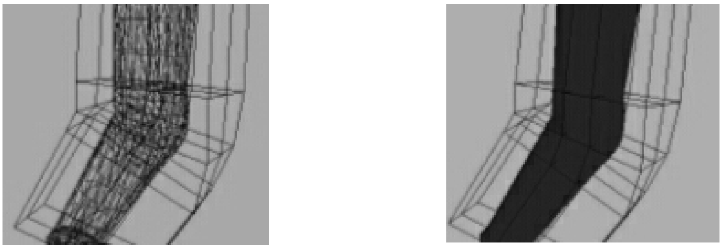
The benefits of specifying an animation in terms of a hierarchical reference grid are even more pronounced when comparing this approach to animation methods that are directly applied to the vertices, as is the case for the IndexedFaceSet representation. This difference in complexity is illustrated by Figure 16, which depicts the knee of the humanoid model. As one can see in Figure 16a, the

Figure 15. Bone-based animation of the reference grid of the humanoid model.



(a) *(b)*

Figure 16. Snapshot of the humanoid knee during the animation. The reference grid lines are displayed in black. The surface at the second resolution level is displayed as a wireframe in (a) and Gouraud-shaded in (b).



(a) *(b)*

number of vertices contained in the surface mesh of the second resolution level is already quite high (216 vertices), while the reference grid at the lowest level only consists of 27 points at the height of the knee (three planes defined by nine points each). Although the number of grid points will be higher when applying the long interpolation filter, the fact that the grid is defined on a regular space, seriously simplifies the interactive selection of the grid points, since it is possible to determine the majority of these points automatically once a few key points have been chosen. Moreover, the same animation script can animate: (1) any resolution of the MeshGrid model, due to its hierarchical construction; and (2) any model with a reference grid that is defined in a compatible way with the reference model used in the animation script. Compatible MeshGrid models are characterized by: (1) the same number of reference surfaces defining the reference system; and (2) the same reference surfaces passing through the same anatomical positions in the different models. The drawback when animating an IndexedFaceSet model is the need for a different script at each resolution level of each model.

In practice, for achieving efficient hierarchical animation with the BBA approach, an appropriate technique is to animate the reference grid points, i.e., the space around the model. Moreover, the BBA hierarchical animation technique based on the MeshGrid representation method offers serious advantages in terms of compactness, design simplicity and computational load, compared to a bone-based animation defined for a complex single resolution model which is described as an IndexedFaceSet.

The major advantage compared to animation techniques defined for other hierarchical models (e.g., Subdivision Surfaces), is that it is more intuitive to address the regularly defined grid points than the vertices and that it is possible, with the same script, to animate compatible MeshGrid models. The reference grid points contained in the lowest resolution level represent only a small percentage of the total number of points present in the final level and, in addition, only a limited number of grid points are animated, i.e., only those related to vertices. Hence, such a hierarchical approach is very advantageous, since the animation can be designed in terms of a limited number of points and since the number of computations needed to apply the BBA transformation to each of these points will be reduced.

Conclusions

This chapter is devoted to the standardization of virtual character animation. In particular, the MPEG-4 Face and Body, as well as the Skeleton, Muscle, and Skin animation frameworks have been presented. A generic deformation model and its implementation in the MPEG-4 standard through the bone and muscle controllers has been introduced. This generic concept, in relation with dedicated surface representation tools, like Subdivision Surfaces and MeshGrid, recently standardized by MPEG-4, allows hierarchical animation. It provides support for explicitly animating only a limited set of key vertices by the designer, out of which the animation of all other vertices is automatically calculated through an iterative calculation scheme.

In recent years, major improvements have been reported in the field of virtual character animation, ranging from the creation of realistic models used in cinema movies and the development of animation production tools (e.g., motion capture systems) to the production of on-line animations in television shows and content streaming in distributed environments. However, research in this field is still in an initial stage and presents challenging issues. Despite of large on-going efforts, computer vision technologies for tracking human motion have not yet reached a level of maturity that is satisfactory for commercial use. In order to decrease the production cost of 3D content, re-targeting motion from motion capture data set to different avatars is still a hot topic of research. Other important research activities are oriented towards behavioural models for avatar and/or crowds, building autonomous agents able to own intelligence and making virtual characters "live" with the appropriate emotional content. Even if MPEG-4 did not explicitly analyse these issues closely, its generic framework makes extensions possible, providing the means to an ever-evolving, living standard.

Acknowledgments

This chapter has grown out of numerous discussions with members of the MPEG-4 AFX standardization committee and fruitful suggestions proposed by numerous colleagues, who have provided inspiration during the germination of the chapter. In particular, Prof. Jan Cornelis, Dr. Rudi Deklerck and Dr. Augustin Gavrilescu from the ETRO Department of the Vrije Universiteit Brussel have contributed to many improvements in the Humanoid animation framework, based on the MeshGrid representation. Some parts of the work, like the FBA and SMS have matured thanks to assessments through projects, amongst which the IST European Project ViSiCAST has probably contributed the most to the quality of the outcome.

References

Capin, T. K., & Thalmann, D. (1999). Controlling and Efficient coding of MPEG-4 Compliant Avatars, *Proceedings of IWSNHC3DI'9*9, Santorini, Greece.

Catmull, E. & Clark, J. (1978). Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design, 10*, 350-355.

Craig, J., Jr. (1989). *Introduction to Robotics: Mechanics and Control, 2nd edition*. Reading, MA: Addison Wesley.

Doenges, P., Capin, T., Lavagetto, F., Ostermann, J., Pandzic, I. & Petajan, E. (1997). Mpeg-4: Audio/video and synthetic graphics/audio for real-time, interactive media delivery. *Image Communications Journal, 5*(4), 433-463.

Doo, D. & Sabin, M. (1978). Behaviour of recursive division surfaces near extraordinary points. *Computer-Aided Design, 10*(6), 356-360.

Dyn, N., Levin, D. & Gregory, J. A. (1987). A four-point interpolatory subdivision scheme for curve design. *Computer-Aided Geometric Design, 4*, 257-268.

Dyn, N., Levin, D. & Gregory, J. A (1990). A Butterfly Subdivision Scheme for Surface Interpolation with Tension Control. *ACM Transactions on Graphics*, *9*(2), 160-169.

Escher, M., Pandzic, I. & Magnenat-Thalmann, N. (1998). Facial Animation and Deformation for MPEG-4, *Proceedings of Computer Animation'98.*

Foley, J. D., Damm , A., Feiner, S. K., & Hughes, J. F. (1992). *Computer Graphics – Principles and Practice, 2nd edition*. Reading, MA: Addison Wesley.

Grahn, H., Volk, T. & Wolters, H. J. (2001). NURBS Extension for VRML97, 2/2001, © Bitmanagement Software. Retrieved from the World Wide Web at: http://www.bitmanagement.de/developer/contact/nurbs/overview.html.

ISOIEC 14496-1:2001 (2001). Information technology. Coding of audio-visual objects. Part 1: Systems, International Organization for Standardization, Switzerland.

ISOIEC 14496-1:2003 (2003). Information technology. Coding of audio-visual objects. Part 16: Animation Framework eXtension. International Organization for Standardization, Switzerland.

Kim, M., Wood S. & Cheok, L. T. (2000). Extensible MPEG-4 Textual Format (XMT), in *Proceedings of the 2000 ACM workshops in Multimedia*, 71-74, Los Angeles, CA.

Kobbelt, L. (2000). $\sqrt{3}$-Subdivision. *SIGGRAPH'00 Conference Proceedings,* 103-112.

Lavagetto, F. & Pockaj, R. (1999). The Facial Animation Engine: Toward a High-level Interface for the Design of Mpeg-4 Compliant Animated Faces, *IEEE Transaction Circuits Systems Video Technology, 9*(2), 277-289.

Loop, C. (1987). Smooth subdivision surfaces based on triangles. Master's thesis, Department of Mathematics, University of Utah.

Malciu, M. & Preteux, F. (2000). Tracking facial features in video sequences using a deformable model-based approach. *Proceedings of SPIE Conference on Mathematical Modeling, Estimation and Imaging, San Diego, CA*, 4121.

O'Rourke, M. (1998). *Principles of Three-Dimensional Computer Animation: Modeling, Rendering, and Animating With 3d Computer Graphics*, Hardcover Edition.

Pereira, F. & Ebrahimi, T. (2002). *The MPEG-4 Book*. Prentice Hall.

Piegl, L. & Tiller, W. (1997). *The NURBS Book, 2nd edition*. Springer.

Preda, M. (2002c). Système d'animation d'objets virtuels : De la modélisation à la normalisation MPEG-4. Ph.D. Dissertation Université Paris V - René Descartes, France.

Preda, M. & Prêteux, F. (2002a). Insights into low-level animation and MPEG-4 standardization, *Signal Processing: Image Communication, 17*(9), 717-741.

Preda, M. & Prêteux, F. (2002b). Critic review on MPEG-4 Face and Body Animation. *Proceedings IEEE International Conference on Image Processing (ICIP'2002), Rochester, NY.*

Salomie, I. A., Deklerck, R., Munteanu, A. & Cornelis, J. (2002a). The MeshGrid surface representation. Technical Report IRIS-TR-0082, Dept. ETRO-IRIS, Vrije Universiteit Brussel. Retrieved from the World Wide Web: http://www.etro.vub.ac.be/MeshGrid_TR_0082_2002.pdf.

Salomie, I. A. et al. (2002b). MeshGrid – A compact, multi-scalable and animation-friendly surface representation. *Proceedings of IEEE International Conference on Image Processing (ICIP 2002)*, Rochester, NY, 3, 13-16.

Shoemake, K. (1994). Euler Angle Conversion. Gramphics Gems IV, Academic Press Professional, Toronto.

Taubin, G. & Rossignac, J. (1998a). Geometric compression through topological surgery. *ACM Transactions on Graphics, 17*(2), 84-115.

Taubin, G., Guéziec, A., Horn, W. & Lazarus, F. (1998b). Progressive forest split compression. *Proceedings of SIGGRAPH'98*, 123-132.

Thalmann-Magnenat, N. & Thalmann, D. (2000). Virtual Reality Software and Technology. *Encyclopedia of Computer Science and Technology*. Marcel Dekker, 41.

Endnotes

[1] Living actor technology, http://www.living-actor.com/.

[2] Scotland government web page, http://www.scotland.gov.uk/pages/news/junior/introducing_seonaid.aspx.

[3] Walt Disney Pictures & Pixar. Geri's game, Toy Story (1995), A Bug's Life (1998), Toy Story 2 (1999) and Monsters, Inc. (2001).

[4] Vandrea news presenter, Channel 5, British Broadcasting Television.

[5] Eve Solal, Attitude Studio, www.evesolal.com.

[6] blaxxun Community, VRML - 3D - Avatars - Multi-User Interaction, http://www.blaxxun.com/vrml/home/ccpro.htm.

[7] 3D Studio Max™ Discreet, http://www.discreet.com/index-nf.html.

[8] Maya™ Alias/Wavefront, http://www.aliaswavefront.com/en/news/home.shtml.

[9] The Virtual Reality Modeling Language, International Standard ISO/IEC 14772-1:1997, www.vrml.org.

[10] H-Anim – Humanoid AnimationWorking Group, www.h-anim.org.

[11] SNHC - Synthetic and Natural Hybrid Coding, www.sait.samsung.co.kr/snhc.

[12] MPEG Page at NIST, mpeg.nist.gov.

[13] Face2Face Inc. www.f2f-inc.com.

Chapter III

Camera Calibration for 3D Reconstruction and View Transformation

B. J. Lei
Delft University of Technology, The Netherlands

E. A. Hendriks
Delft University of Technology, The Netherlands

Aggelos K. Katsaggelos
Northwestern University, USA

Abstract

This chapter presents an extensive overview of passive camera calibration techniques. Starting with a detailed introduction and mathematical description of the imaging process of an off-the-shelf camera, it reviews all existing passive calibration approaches with increasing complexity. All algorithms are presented in detail so that they are directly applicable. For completeness, a brief counting about the self-calibration is also provided. In addition, two typical applications are given of passive camera calibration methods for specific problems of face model reconstruction and telepresence and experimentally evaluated. It is expected that this chapter can serve as a standard reference. Researchers in various fields in which passive camera calibration is actively or potentially of interest can use this chapter to identify the appropriate techniques suitable for their applications.

Camera calibration is the process of determining the internal physical characteristics of a camera and its 3-D position and orientation with respect to a world coordinate system using some predefined objects or automatically detected features. The result of camera calibration is the establishment of a mathematical relationship between the 3-D coordinates of a point in the 3-D scene and the 2-D coordinates of its projection onto the image recorded by the camera.

Camera calibration is an important preliminary step towards many vision-related applications. Passive calibration, active calibration, and self-calibration are the most frequently referred to camera calibration algorithm categories. Active calibration methods were developed mainly for robotic systems. Recently, algorithms for active calibration purposes have been investigated that fall in the more general self-calibration category (Lamiroy, Puget & Ho-raud, 2000). While detailed discussions about self-calibration are given in Faugeras & Luong (2001), Hartley & Zisserman (2000) and Fusiello (2000), this paper intends to give an overview of passive calibration. However, for completeness, a brief counting about the self-calibration will also be presented.

Passive calibration has been used extensively in the synthesis and analysis of the human body for telepresence (Xu, Lei, & Hendriks, 2002) and in 3-D face modeling (Liu, Zhang, Jacobs, & Cohen, 2001). However, despite its wide range of applications and extensive investigations, no comprehensive overview of this topic exists. This chapter attempts to fill this gap by providing such an overview in a systematic and unified manner and by comparing and evaluating existing approaches. In addition, two typical applications are given of passive camera calibration methods for specific problems of face model reconstruction and telepresence and then experimentally evaluated. It is expected that this chapter can serve as a standard reference. Researchers in various fields in which passive camera calibration is actively or potentially of interest can use this chapter to identify the appropriate techniques suitable for their applications.

The chapter is organized as follows. In the next section, a detailed introduction and mathematical description is provided of the imaging process of an off-the-shelf camera. In the next section, all existing camera calibration techniques are classified based on several different points of view. The nonlinear component of the camera, responsible for distortion, is then modeled using two alternative methods and discussed in a following section. Key passive camera calibration algorithms are reviewed in detail, followed by a brief overview of self-calibration algorithms. Finally, two applications for which calibrated cameras are required are analyzed, and a summary and conclusions are presented.

Camera Imaging Process

In the perfect case, a camera can be modeled linearly as a pinhole. However, to compensate for nonlinear effects in the imaging process, certain distortion coefficients have to be added to the simple pinhole model.

Coordinate Systems

In pinhole modeling, five relevant *coordinate systems* (CSs) are needed to transform positions of the world points in the 3-D space into their projections in the image plane of the camera, as described next.

1. **Object Coordinate System** (**OCS**): This CS is fixed to an object. This means that for each object there is a unique OCS. In this CS, the position of each point in the corresponding object is denoted by $\mathbf{x}^o = \begin{bmatrix} x^o & y^o & z^o \end{bmatrix}^T$.
2. **World Coordinate System** (**WCS**): This is a common CS. All other CSs are defined in reference to it. Any point in the 3-D scene has coordinates denoted by $\mathbf{x}^w = \begin{bmatrix} x^w & y^w & z^w \end{bmatrix}^T$.
3. **Camera Coordinate System** (**CCS**): This CS is connected to the camera. The x-y plane is parallel to the image plane with its origin at the projection center, and its z-axis along the optical axis (ref. Figure 1a). A point in the 3-D scene has coordinates denoted by $\mathbf{x}^c = \begin{bmatrix} x^c & y^c & z^c \end{bmatrix}^T$.
4. **Projection Coordinate System** (**PCS**): This CS is a 2-D CS. It records the **metric** coordinates of the projection of a 3-D scene point through the pinhole onto the camera image plane. The x and y axes of this system are always set to be parallel with those of the corresponding CCS (ref. Figure 1a). Each projection has coordinates in this CS denoted by $\mathbf{x}^m = \begin{bmatrix} x^m & y^m \end{bmatrix}^T$.
5. **IMage Coordinate System** (**IMCS**): Each coordinate in this system is the *pixel coordinate* that is actually measured in the image. If the imaging process has no nonlinear component, then the coordinates in this system are denoted by $\mathbf{x}^{im} = \begin{bmatrix} x^{im} & y^{im} \end{bmatrix}^T$; otherwise, they are denoted by $\hat{\mathbf{x}}^{im} = \begin{bmatrix} \hat{x}^{im} & \hat{y}^{im} \end{bmatrix}^T$. In the nonlinear case, $\hat{\mathbf{x}}^{im}$ is always modeled as $\mathbf{x}^{im}$ plus some nonlinear elements (*distortion*). Sometimes, to simplify computations, it is assumed that $\hat{\mathbf{x}}^{im} = \mathbf{x}^{im}$, which means that a linear model is used and is fit to a nonlinear imaging system (as in the most recent self-calibration methods).

Linear Coordinate Transformation

The relation between $\mathbf{x}^o$ and the corresponding $\mathbf{x}^w$ can be expressed as:

$$\mathbf{x}^w = \mathbf{R}_o^w \mathbf{x}^o + \mathbf{t}_o^w, \tag{1}$$

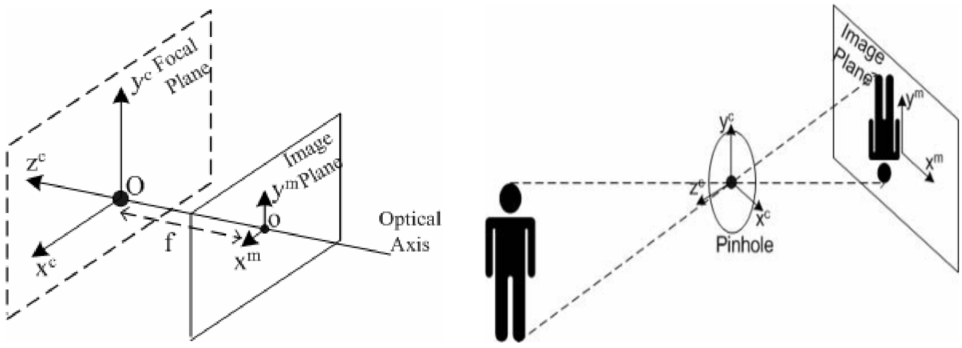
where the rotation matrix $\mathbf{R}_o^w = \mathbf{R}(\phi, \varphi, \psi)$ and the translation vector $\mathbf{t}_o^w = \begin{bmatrix} x_o^w & y_o^w & z_o^w \end{bmatrix}^T$ determine the *pose* (including position and orientation) of the OCS in the WCS. $\mathbf{R}_o^w$ is defined as the product of three separate rotations around the respective axes, that is, $\mathbf{R}_o^w = \mathbf{R}_\mathbf{z}(\psi) \cdot \mathbf{R}_\mathbf{y}(\varphi) \cdot \mathbf{R}_\mathbf{x}(\phi)$. Equation 1 is a rigid body transformation, in which only rotation and translation are permitted, but scaling is not allowed (*Euclidean geometry*). This kind of transformation is called *Euclidean transformation*.

A similar relation exists between $\mathbf{x}^w$ and the corresponding $\mathbf{x}^c$:

$$\mathbf{x}^w = \mathbf{R}_c^w \mathbf{x}^c + \mathbf{t}_c^w \text{ or } \mathbf{x}^c = \left(\mathbf{R}_c^w\right)^T \left(\mathbf{x}^w - \mathbf{t}_c^w\right) \tag{2}$$

where the rotation matrix $\mathbf{R}_c^w = \mathbf{R}(\alpha, \beta, \gamma)$ and the translation vector $\mathbf{t}_c^w = \begin{bmatrix} x_c^w & y_c^w & z_c^w \end{bmatrix}^T$ determine the pose of the CCS with respect to the WCS. They actually represent the *extrinsic parameters* of the camera. And they have in total six *degrees of freedom* (DOFs).

Figure 1. The pinhole camera model.



(a) Relation between a CCS and the corresponding PCS.

(b) The principle of a pinhole projection.

The relation between the camera coordinates (in CCS) and the metric projection coordinates (in PCS) is inferred from the principle of lens projection (modeled as a pinhole, see Figure 1b). This perspective transformation is a kind of projective mapping (ref. Figure 1a).

In Figure 1a, the *optical center*, denoted by **O**, is the center of the focus of projection. The distance between the image plane and **O** is the *focal length*, which is a camera constant and denoted by *f*. The line going through **O** that is perpendicular to the image plane is called the *optical axis*. The intersection of the optical axis and the image plane is denoted by **o**, and is termed the *principal point* or *image center*. The plane going through **O** that is parallel to the image plane is called the *focal plane*.

The perspective projection from the 3-D space (in CCS) onto the image plane (in IMCS) through the PCS can be formulated as:

$$z^c \begin{bmatrix} x^{im} \\ y^{im} \\ 1 \end{bmatrix} = z^c \begin{bmatrix} 1/s_x & 0 & x_0 \\ 0 & 1/s_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^m \\ y^m \\ 1 \end{bmatrix} = \begin{bmatrix} -f_x & 0 & x_0 \\ 0 & -f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^c \\ y^c \\ z^c \end{bmatrix}, \quad (3)$$

where $f_x = f / s_x$ and $f_y = f / s_y$. $\begin{bmatrix} x_0 & y_0 \end{bmatrix}^T$ is the pixel coordinate of the principal point with respect to the IMCS, and s_x and s_y are the effective pixel distances in the horizontal (x-axis) and vertical (y-axis) directions, determined by the sampling rates for Vidicon cameras or the sensitive distance for CCD and CID cameras. Most CCD cameras do not have square pixels, but rectangular pixels with an *aspect ratio* s_y / s_x of about 0.9 to 1.1. f, x_0, y_0, and are called the *intrinsic parameters* of the camera.

Substituting equation 2 into equation 3 we obtain:

$$\begin{bmatrix} x^{im} \\ y^{im} \\ 1 \end{bmatrix} \cong \begin{bmatrix} -f_x & 0 & x_0 \\ 0 & -f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \left(\mathbf{R}_c^w\right)^T & -\left(\mathbf{R}_c^w\right)^T \cdot \mathbf{t}_c^w \end{bmatrix} \begin{bmatrix} x^w \\ y^w \\ z^w \\ 1 \end{bmatrix},$$

where ≅ means “equal up to a non-zero scale factor”, which is the symbol of equality in the *projective space* (Faugeras, 1993).

Thus, three transform matrices can be identified to project a 3-D world point onto its 2-D correspondence in an image. These three matrices are termed the *intrinsic transform matrix* $\tilde{\mathbf{K}}$ (ITM, encoding all intrinsic camera parameters), the *extrinsic transform matrix* $\tilde{\mathbf{M}}$ (ETM, encoding all extrinsic camera parameters), and the *projection matrix* $\tilde{\mathbf{P}}$ (PM, encoding all linear camera parameters), and are given by:

$$\tilde{\mathbf{K}} = \begin{bmatrix} -f_x & 0 & x_0 \\ 0 & -f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}, \ \tilde{\mathbf{M}} = \begin{bmatrix} \left(\mathbf{R}_c^w\right)^T & -\left(\mathbf{R}_c^w\right)^T \cdot \mathbf{t}_c^w \end{bmatrix}, \text{ and } \tilde{\mathbf{P}} = \tilde{\mathbf{K}}\tilde{\mathbf{M}}. \quad (4)$$

Thus, for the *projective coordinates*[1] $\tilde{\mathbf{x}}^{im} = \begin{bmatrix} x^{im} & y^{im} & 1 \end{bmatrix}^T$ and $\tilde{\mathbf{x}}^w = \begin{bmatrix} x^w & y^w & z^w & 1 \end{bmatrix}^T$, we have:

$$\tilde{\mathbf{x}}^{im} \cong \tilde{\mathbf{P}} \cdot \tilde{\mathbf{x}}^w \cdot \quad (5)$$

Through this projection, a 3-D straight line will map as a 2-D straight line. From this observation, this pure pinhole modeling is called **linear** modeling.

Modeling Nonlinear Components

The perfect pinhole model is only an approximation of the real camera system. It is, therefore, not valid when high accuracy is required. The nonlinear components (skew and distortion) of the model need to be taken into account in order to compensate for the mismatch between the perfect pinhole model and the real situation.

In applications for which highly accurate calibration is not necessary, distortion is not considered. Instead, in this case a parameter *u* characterizing the *skew* (Faugeras, 1993) of the image is defined and computed, resulting in an ITM as:

$$\tilde{\mathbf{K}} = \begin{bmatrix} -f_x & u & x_0 \\ 0 & -f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} x^{im} \\ y^{im} \end{bmatrix} = \begin{bmatrix} \hat{x}^{im} \\ \hat{y}^{im} \end{bmatrix}. \quad (6)$$

If highly accurate calibration results are required or in cases where low-cost or wide-angle lenses are employed, *distortion* has to be accounted for. There exist two ways of modeling the camera distortion, which will be discussed in the following section.

Camera Parameters

The coefficients of the distortion model, together with the intrinsic and extrinsic parameters, emulate the imaging process of a camera with very high accuracy. In addition, the skew parameter could also be included. However, since the modeled distortion already accounts for the interaction between the *x* and *y* components, the image skew does not need to be considered explicitly, but instead it is treated implicitly as part of the distortion.

It can be noticed in the previous equations that f and s_x, respectively f and s_y, are always coupled with each other. This means that only the values of $f_x = f / s_x$ and $f_y = f / s_y$, instead of the actual values of f, s_x, and s_y, are important and can be recovered.

All these camera parameters, which can be recovered directly from a camera calibration procedure, can be grouped into a *linear parameter vector* $\mathbf{p}_l$ and a *distortion coefficient vector* $\mathbf{p}_d$, as follows:

$$\mathbf{p}_l = \begin{bmatrix} \alpha & \beta & \gamma & x_c^w & y_c^w & z_c^w & f_x & f_y & x_0 & y_0 \end{bmatrix}^T, \tag{7}$$

$$\mathbf{p}_d = \left[\textit{distortion coefficients} \right]^T . \tag{8}$$

These two vectors are further combined into one *camera parameter vector* $\mathbf{p}_c$ as follows:

$$\mathbf{p}_c = \begin{bmatrix} \mathbf{p}_l{}^T & \mathbf{p}_d{}^T \end{bmatrix}^T . \tag{9}$$

Classifications of Camera Calibration Techniques

Camera calibration is a fundamental part of 3-D computer vision. Based on it, the original 3-D structure of a scene can be recovered (*3-D reconstruction*) or a geometric-valid representation of the scene in a new image space can be produced directly (*Image Based Rendering*, **IBR**).

During camera calibration, all camera parameters should be estimated from certain observed geometric constraints. These constraints can be expressed by:

1. **Correspondences** between a **known** 3-D structure and **measured** 2-D image contents (Slama, 1980);
2. **Optical flow** or **parallax** embedded in a sequence of views (provided by a video camera or a multiple-baseline system) that are 2-D projections of an **unknown** 3-D scene from different **unknown** (Pollefeys, Koch, & Gool, 1999) or **pre-defined** (Faugeras, Quan, & Sturm, 2000) relative viewpoints; and
3. **Special characteristics** in 2-D image space incurred by certain **known** specific **geometric structures** embedded in the 3-D scene, such as preservation of line straightness through linear projective transformation (Brown, 1971), vanishing points of parallel lines (Caprile & Torre, 1990), and special relations between projections of orthogonal lines (Liebowitz & Zisserman, 1998).

In general, the number of independent constraints should not be less than the DOFs of the camera parameter space. Based on this observation, a counting argument has been presented for self-calibration (Pollefeys et al., 1999). Minimal constraint requirements for different calibration purposes can also be identified (Torr & Zisserman, 1996). Often, however, to improve numerical robustness, many more independent constraints than needed are utilized (*over-constrained problem*) (Hartley & Zisserman, 2000), or several equivalent representations of the same constraint are employed simultaneously (Malm & Heyden, 2001).

Due to the availability of useful geometric constraints, a number of different camera calibration approaches exist. Meanwhile, various types of compromises or trade-offs always have to be made. Such a trade-off is, for instance, the desired accuracy of calibration results and the depth of view that can be supported by the camera calibration results. Requirements also differ from application to application. All existing camera calibration approaches can be categorized from several different points of view.

Given the underlying assumptions about the parameters to be estimated, the following three general types of calibration exist:

Passive (Fixed) Calibration: All camera parameters are assumed fixed and, therefore, the calibration is performed only once. This approach often uses images of objects for which the accurate geometric and photometric properties can be devised, and reconstructs the relationship between such properties and the recorded images. From these relations all camera parameters can be estimated quantitatively through some linear or nonlinear optimization process (Tsai, 1987; Triggs, McLauchlan, Hartley & Fitzgibbon, 1999).

Active Calibration: For active vision purposes, some of the intrinsic camera parameters (typically focus and zoom) are assumed to vary actively, while the extrinsic parameters are either changed on purpose in a controlled fashion (e.g., pure rotation) or not at all (Willson, 1994). The investigation of the relationship between zooming (and/or focus) and other intrinsic parameters is at the center of this approach.

Self-Calibration: Depending on the application requirements, all or some of the camera parameters are assumed to vary independently, while the remaining ones are unknown constants or parameters that have already been recovered via a pre-processing step (e.g., through a passive calibration technique). The purpose of self-calibration is to be able to recover the camera parameters at different settings of the optical and geometrical configurations (Pollefeys et al., 1999).

By considering the appearance of the camera parameters, we can identify:

Explicit Calibration: This represents the traditional approach (Slama, 1980), according to which the values of all individual camera parameters are calculated explicitly.

Implicit Calibration: In this case, only certain relations of the camera parameters are recovered, for example, the projection matrix in equation 4. These relations must contain enough information to support subsequent calculations. The exact value of each individual parameter is not made explicit (Wei & Ma, 1994). However, all individual camera parameters may be calculated from the recovered relations.

Depending on the required complexity and completeness of the computation, the following three types of models can be adopted:

Linear Models: This represents a simplistic approach to calibration. The imaging process is described in terms of a simple linear equation (ref. equation 5). Clearly, typical nonlinear phenomena, like distortion, are rarely taken into account by this model, unless the distortion component can be approximated by a linear function (Fitzgibbon, 2001).

Nonlinear Model: A more complex model of the system (including nonlinear distortion, modeled, for example, as in equation 11) is created for describing the imaging process. There are two possibilities to fit this nonlinear model to the available data: 1) The nonlinear distortion part is first removed by some special technique, such as preservation of line straightness (Brown, 1971), and the linear relation is then easily recovered; and 2) A linear model that does not consider distortion is first fitted to the data, and the outputs are then fed into a nonlinear optimization process to get the best nonlinear fit (Zhang, 2000).

"Black Box" Model: The whole system is treated as a black box. Inputs and outputs are studied together with some specific properties of the camera to predict the nonlinear imaging process (Chen & Jiang, 1991).

In this chapter we focus on passive camera calibration techniques.

Distortion correction is a key issue in the development of camera calibration techniques, since it determines the accuracy of the subsequent applications, such as 3-D reconstruction. Therefore, in the next section, the distortion modeling and estimation issues are first investigated in detail.

Nonlinear Camera Distortion

For an off-the-shelf lens, the deviation of the pixel position in the image plane due to distortion is on average in the order of five pixels (Tsai, 1987) and, in rare cases, it can be up to ten or even 100 pixels (Stein, 1997). If the distortion is modeled, nonlinear methods for estimating the parameters have to be employed; otherwise linear techniques, which are far more efficient and stable, can be applied.

On the other hand, it was found that the radial distortion assumption can linearize the image geometry to an accuracy which is 2×10^{-5} of the image size (Beyer, 1992). Any more elaborate distortion model than a radial one could help in increasing the accuracy, but may incur numerical instability (Tsai, 1987). Thus, most calibration algorithms usually take into account only the radial distortion. However, when wide-angle cameras are used, adding a non-radial distortion component in the distortion model will improve accuracy significantly (Weng, Cohen & Herniou, 1992).

Therefore, the complexity of the distortion model (i.e., the number of distortion coefficients considered) should match the available computation resources and the accuracy required by the application.

Distortion Models

Two different models have been constructed to describe the distortion phenomenon. They were developed for the purpose of projection and that of 3-D reconstruction, respectively.

Imaging-distortion model

For the camera projection purpose, the distortion can be modeled as "imaging distortion" as:

$$\begin{bmatrix} \hat{x}^{im} \\ \hat{y}^{im} \end{bmatrix} = \begin{bmatrix} x^{im} \\ y^{im} \end{bmatrix} + \begin{bmatrix} f_x \cdot \Delta_x^{\mathrm{Im}} \\ f_y \cdot \Delta_y^{\mathrm{Im}} \end{bmatrix}, \qquad (10)$$

where

$$\begin{bmatrix} \Delta_x^{\mathrm{Im}} \\ \Delta_y^{\mathrm{Im}} \end{bmatrix} = \begin{bmatrix} x'r^2 & x'r^4 & x'r^6 & 2x'^2+r^2 & 2x'y' & r^2 & 0 \\ y'r^2 & y'r^4 & y'r^6 & 2x'y' & 2y'^2+r^2 & 0 & r^2 \end{bmatrix} \cdot \mathbf{p}_d^{\mathrm{Im}}, \quad (11)$$

$$\mathbf{p}_d^{\mathrm{Im}} = \begin{bmatrix} k_1^{\mathrm{Im}} & k_2^{\mathrm{Im}} & k_3^{\mathrm{Im}} & P_1^{\mathrm{Im}} & P_2^{\mathrm{Im}} & s_1^{\mathrm{Im}} & s_2^{\mathrm{Im}} \end{bmatrix}^T,$$

$$x' = -\frac{x^c}{z^c} = \frac{x^{im} - x_0}{f_x}, \; y' = -\frac{y^c}{z^c} = \frac{y^{im} - y_0}{f_y}, \text{ and } r^2 = x'^2 + y'^2 .$$

k_1^{Im}, k_2^{Im}, k_3^{Im} (Radial), P_1^{Im}, P_2^{Im} (De-centering), and s_1^{Im}, s_2^{Im} (Thin Prim) represent the imaging-distortion coefficients, while Δ_x^{Im} and Δ_y^{Im} represent distortions in the horizontal (x-axis) and vertical (y-axis) directions, respectively. The radial distortion is caused by the fact that objects at different angular distances from the lens axis undergo different magnifications. The de-centering distortion is due to the fact that the optical centers of multiple lenses are not correctly aligned with the center of the camera. The thin-prim distortion arises from the imperfection in the lens design and manufacturing, as well as the camera assembly.

This distortion model can be simplified by neglecting certain parameters. k_1^{Im} usually accounts for about 90% of the total distortion (Slama, 1980). For example, in some cases, only radial and tangential components were taken into consideration. The effect of the thin prim coefficients (s_1^{Im} and s_2^{Im}) was overlooked without affecting the final accuracy, because this component only causes additional radial and tangential distortions (Weng et al., 1992).

Reconstruction-distortion model

Besides being modeled as imaging distortion, the distortion can also be modeled as *"reconstruction distortion"* as follows:

$$\begin{bmatrix} x^{im} \\ y^{im} \end{bmatrix} = \begin{bmatrix} \hat{x}^{im} \\ \hat{y}^{im} \end{bmatrix} - \begin{bmatrix} f_x \cdot \Delta_x^{\text{Re}} \\ f_y \cdot \Delta_y^{\text{Re}} \end{bmatrix}, \qquad (12)$$

where

$$\begin{bmatrix} \Delta_x^{\text{Re}} \\ \Delta_y^{\text{Re}} \end{bmatrix} = \begin{bmatrix} \hat{x}r^2 & \hat{x}r^4 & \hat{x}r^6 & 2\hat{x}^2 + r^2 & 2\hat{x}\hat{y} & r^2 & 0 \\ \hat{y}r^2 & \hat{y}r^4 & \hat{y}r^6 & 2\hat{x}\hat{y} & 2\hat{y}^2 + r^2 & 0 & r^2 \end{bmatrix} \cdot \mathbf{p}_d^{\text{Re}}, \qquad (13)$$

$$\mathbf{p}_d^{\text{Re}} = \begin{bmatrix} k_1^{\text{Re}} & k_2^{\text{Re}} & k_3^{\text{Re}} & P_1^{\text{Re}} & P_2^{\text{Re}} & s_1^{\text{Re}} & s_2^{\text{Re}} \end{bmatrix}^T,$$

$$\hat{x} = \frac{\hat{x}^{im} - x_0}{f_x}, \ \hat{y} = \frac{\hat{y}^{im} - y_0}{f_y}, \text{ and } r^2 = \hat{x}^2 + \hat{y}^2 .$$

A similar discussion as that on the imaging-distortion coefficients in the previous section also applies to the reconstruction-distortion coefficients $\mathbf{p}_d^{\mathrm{Re}}$.

Principal point vs. distortion center

It has been realized that the *distortion center* $[x_0 \quad y_0]^T$ used in equations 10 and 12 could be different from the principal point employed in equation 3 (Wei & Ma, 1994). On the other hand, under radial distortion with a dominant coefficient k_1^{Im} (or k_1^{Re}), a small shift of the distortion center is equivalent to adding two de-centering distortion terms (Ahmed & Farag, 2001). Therefore, if the distortion is estimated independently of the calibration of other camera parameters, the principal point for the linear perspective transformation should be distinguished from the distortion center. However, if all camera parameters (including distortion coefficients) are calibrated simultaneously, the distortion center and the principal point should be treated as being the same. In this case, it is better to take the de-centering distortion component into consideration.

Discussions

Both imaging and reconstruction-distortion models have advantages and disadvantages (ref. Table 1). In general, the imaging-distortion model is more efficient for distortion correction using the “backward mapping” strategy (Lei & Hendriks, 2002). The reconstruction-distortion model is preferable if the subsequent processing is mainly concerned with the 3-D model recovering.

Both models have been adopted in the calibration literature (Heikkilä, 2000; Tsai, 1987). It was demonstrated by Heikkilä (2000) that they are equivalent with proper compensation. A least-squares method was further proposed for the conversion between the two sets of distortion coefficients. Which model should be adopted for a real situation is application-dependent.

Table 1. The advantages and disadvantages of both imaging and reconstruction-distortion models.

	Imaging-distortion model	Reconstruction-distortion model
Advantages	1. The distortion coefficients are complete in the 3-D CCS and only associated with the 3-D coordinates of the space points; 2. The imaging process can be reproduced; 3. Distortion correction can easily be carried out by employing the efficient “backward mapping” technique (Wolberg, 1990).	1. The distortion coefficients are complete in the 2-D IMCS and only associated with the 2-D coordinates of the image pixels; 2. The 3-D reconstruction problem can be solved more easily; 3. Distortion component in the measured image coordinate can easily be removed.
Disadvantages	From measured image coordinates, it is difficult to get the undistorted 3-D correspondences.	Distortion correction has to be performed by the so-called “forward mapping” technique, which is not so efficient (Wolberg, 1990).

Distortion Estimation Techniques

A survey is given below of distortion estimation techniques mainly developed in the computer-vision field. Among them, there is the elegant “plumb-line” method (Brown, 1971), which plays a quite important role.

Plumb-line method

With a linear camera model (ref. equation 5), a straight line in the 3-D scene is still a straight line in the corresponding image plane. Any deviation from this straightness should, therefore, be attributed to distortion.

Suppose there exists a straight 3-D line in the 3-D scene. Correspondingly, a straight image line should appear in the focal plane of an ideal linear camera. Let $\mathbf{x}^{im} = \left[x^{im} \quad y^{im} \right]^T$ denote an arbitrary point on that image line. The following equation should then be satisfied

$$x^{im} \sin\theta + y^{im} \cos\theta = \rho,$$

where θ is the angle between the image line and the x-axis of the focal plane, and ρ is the perpendicular distance from the origin to this line.

Suppose that the reconstruction-distortion model is employed, and in the distortion model assume that $f_x = f_y$ (Brown, 1971; Devernay & Faugeras, 2001). Substituting equation 12 into equation 14 leads to an expression of the form

$$f\left(\hat{x}^{im}, \hat{y}^{im}, x_0, y_0, s_x, s_y, k_1^{\mathrm{Re}}, k_2^{\mathrm{Re}}, k_3^{\mathrm{Re}}, P_1^{\mathrm{Re}}, P_2^{\mathrm{Re}}, s_1^{\mathrm{Re}}, s_2^{\mathrm{Re}}, \theta, \rho\right) + \varepsilon = 0, \quad (15)$$

where x_0, y_0, s_x, s_y, k_1^{Re}, k_2^{Re}, k_3^{Re}, P_1^{Re}, P_2^{Re}, s_1^{Re}, s_2^{Re}, θ and ρ are all unknown, and ε is a random error.

If enough colinear points are available,

$$\varsigma = \sum_{i=1}^{n} \sum_{j=1}^{m} \left(f\left(\hat{x}_{ij}^{im}, \hat{y}_{ij}^{im}, x_0, y_0, s_x, s_y, k_1^{\mathrm{Re}}, k_2^{\mathrm{Re}}, k_3^{\mathrm{Re}}, P_1^{\mathrm{Re}}, P_2^{\mathrm{Re}}, s_1^{\mathrm{Re}}, s_2^{\mathrm{Re}}, \theta_i, \rho_i\right)\right)^2 \quad (16)$$

can be minimized to recover x_0, y_0, s_x, s_y, k_1^{Re}, k_2^{Re}, k_3^{Re}, P_1^{Re}, P_2^{Re}, s_1^{Re}, s_2^{Re}, θ_i and ρ_i. $\left[\hat{x}_{ij}^{im} \quad \hat{y}_{ij}^{im}\right]^T$ (i = 1 ... n, j = 1 ... m) are distorted image points whose distortion-free correspondences $\left[x_{ij}^{im} \quad y_{ij}^{im}\right]^T$ should lie on the same image line with rotation θ_i and polar distance ρ_i.

However, due to the high inter-correlation between the de-centering distortion coefficients (P_1^{Re}, P_2^{Re}), and the principal point coordinates $\left[x_0 \quad y_0\right]^T$, x_0 and y_0 are always assumed to be known *a priori*. Otherwise, a proper optimization strategy (e.g., coarse-to-fine [Swaminathan & Nayer, 2000]) has to be adopted to get a more stable solution.

When the plumb-line method is applied, “straight” lines, which are distorted straight image lines, need to be extracted first, typically by means of an edge detection technique, before the optimization on equation 16 can be performed. Clearly, the accuracy of the extracted lines determines the accuracy of the estimated parameters. If the calibration set-up is carefully designed so that those “straight” lines can be located accurately, an overall accuracy in the order of 2×10^{-5} can be achieved (Fryer, Clarke & Chen, 1994). However, for irregular natural scenes, it may be difficult to locate “straight” lines very accurately. To tackle this problem, an iterative strategy has been adopted (Devernay & Faugeras, 2001). According to this strategy, edge detection (step 1) and optimization (step 2) are first performed on the original images. Based on the estimated distortion coefficients, images are corrected (undistorted), and then steps 1 and 2 are repeated. This iterative process continues until a small deviation ς is reached. Applying this strategy on natural scenes, a mean distortion error of about pixel (for a 512×512 image) can be obtained (Devernay & Faugeras, 2001). Improved results can be obtained by modifying equation 16 (dividing, for example, the function f (...) by ρ_i [Swaminathan & Nayer, 2000]) and by carefully defining the “straightness” of a line (using, for example, snakes [Kang, 2000]).

Utilization of projective geometry properties

The plumb-line method explores only one invariant of the projective transformation. Other projective invariants or properties, such as converging of parallel lines, can also be employed for estimating distortion in a fashion similar to the plumb-line method. Some of these methods are summarized below. They make use of:

Convergence of parallel lines: Based on the observation that a set of parallel lines should have a common unique vanishing point through linear projective projection, the distortion is estimated by minimizing the dispersion of all possible candidate vanishing points (Becker & Bove, 1995).

Invariance of cross ratio: Since the cross ratio is still an invariant even when radial distortion is present, it is employed to recover the distortion center first, followed by the use of preservation of linearity of the projective geometry to calibrate the distortion coefficients and the aspect ratio (Wei & Ma, 1994).

Linear projection matrix $\tilde{\mathbf{P}}$: Line intersections are accurately detected in the image. Four of them are selected to define a projective basis for the plane. The others are re-expressed in this frame and perturbed so that they are accurately aligned. The recovered distortion corrections from this projective base are then interpolated across the whole image (Brand, Courtney, Paoli & Plancke, 1996).

Linear fundamental matrix embedded in a stereo set-up: A stereo (or triple-camera) setup can also be used for distortion estimation. First, an initial guess of the distortion coefficients is used to calculate the undistorted image point coordinates. These are then used to calculate the so-called fundamental matrix (or tri-linear tensors). Based on this matrix (or tensors), the correspondence error is calculated. This error is then reduced by adjusting the values of distortion coefficients and the process is repeated until a certain optimal point is reached. The optimal values for distortion coefficients are finally estimated independently (Stein, 1997).

In the last three approaches, stereo correspondences or correspondences between 3-D points and their projections are needed. Traditionally, these correspondences are obtained manually. To automatically search for them, view and illumination changes have to be taken into consideration together with the distortion coefficients (Tamaki, Yamamura & Ohnishi, 2002).

Which linear property or invariant is most resistant to noise is still not clear. This needs to be addressed in future work.

Other techniques

In addition to using linear projective geometry, three other interesting techniques have also been proposed for distortion estimation, each of them with their own limitations.

In Perš & Kovačič (2002), by labeling the camera distortion as an inherent geometry property of any lens, the radial distortion defined in equation 11 is remodeled by the following non-parametric equation:

$$\sqrt{\left(\hat{x}^{im}-x_0\right)^2+\left(\hat{y}^{im}-y_0\right)^2} = f\cdot\ln\left(\frac{r}{f}+\sqrt{1+\frac{r^2}{f^2}}\right),$$

where r was as defined in equation 10 and f is the focal length.

This model is much simpler than the one defined in equation 10 and the model in equation 12. Reasonable results have been obtained in Perš & Kovačič (2002), although more research needs be done to model other types of distortions with high accuracy.

In Farid & Popescu (2001), large radial distortion is detected by analyzing the correlation in the frequency-domain of a single image. The method, however, may only work for certain types of scenes. Finding out what types of scenes are suitable for this technique seems to be an interesting research topic.

Contrary to the nonlinear equations 11 and 13, the standard distortion model is modified in Fitzgibbon (2001) to a projective linear, but equivalent, representation assuming only radial distortion. Based on this, an efficient closed-form algorithm is formulated that is guaranteed to estimate the fundamental matrix (Faugeras, 1993).

Passive Camera Calibration

The aim of passive camera calibration is to recover all camera parameters in $\mathbf{p}_c$ by fitting the camera model described in section 1 to a corresponding set of reference points, called *calibration control points*, in the 3-D world and their corresponding projections, called *calibration feature points*, on the image plane.

Much work has been done on passive camera calibration ranging from the classical nonlinear optimization approach (Slama, 1980) to closed-form solutions (Tsai, 1987). Very recently attention has been paid to multi-step schemes that attempt to combine both nonlinear optimization and linear closed-form solutions. In the following, all important and representative approaches developed for passive camera calibration are discussed. Major equations involved are recalculated and reformulated in a uniform way using the camera model introduced.

All algorithms are presented in detail so that they are directly applicable. Only considering the linear imaging model, the simplest directed linear transformation (DLT) approach is first described. Its geometrically valid variations are then discussed. Nonlinear elements are introduced into the DLT approach to also handle the camera distortion. However, the nonlinear system formed from this idea is, in most cases, too complex to be solved efficiently and accurately. Therefore, a method to avoid the possibly large optimization problem is presented. Because the method can only handle the radial distortion, a more general approach, an iterative two-phase strategy, is discussed. Further, to ease the tedious calibration-data-acquisition work, the 2-D planar pattern is introduced as an alternative, but effective, calibration object. To recover the geometry of more than one camera, the linear phase of the iterative two-phase strategy is modified.

After that, some other approaches that utilize special calibration objects or specific phenomena in the 3-D scene are summarized. Finally, the calibration is evaluated and feature extraction issues are discussed.

Direct Linear Transformation (DLT)

Direct linear transformation (**DLT**) (Abdel-Aziz & Karara, 1971) is the simplest version of camera calibration and still plays a relatively important role in computer vision. It can be applied if the distortion can be neglected or has been removed in advance.

Without considering the distortion, from equation 4 the transfer function from a 3-D point $\mathbf{x}^w$ to the corresponding 2-D image pixel $\mathbf{x}^{im}$ can be described as:

$$\begin{bmatrix} t^1 \\ t^2 \\ t^3 \end{bmatrix} \cong \begin{bmatrix} p_1^1 & p_2^1 & p_3^1 & p_4^1 \\ p_1^2 & p_2^2 & p_3^2 & p_4^2 \\ p_1^3 & p_2^3 & p_3^3 & p_4^3 \end{bmatrix} \begin{bmatrix} x^w \\ y^w \\ z^w \\ 1 \end{bmatrix}, \tag{17}$$

where p_j^i is the element of the matrix $\widetilde{\mathbf{P}}$ at the i^{th} row and j^{th} column, and

$$x^{im} = t^1 / t^3 \text{, and } y^{im} = t^2 / t^3 . \tag{18}$$

Substituting equation 18 into equation 17 and expanding the matrix product yields the following two equations with 12 unknowns:

$$p_1^1 x^w + p_2^1 y^w + p_3^1 z^w + p_4^1 - p_1^3 x^w x^{im} - p_2^3 y^w x^{im} - p_3^3 z^w x^{im} - p_4^3 x^{im} = 0, \tag{19}$$

$$p_1^2 x^w + p_2^2 y^w + p_3^2 z^w + p_4^2 - p_1^3 x^w y^{im} - p_2^3 y^w y^{im} - p_3^3 z^w y^{im} - p_4^3 y^{im} = 0, \tag{20}$$

The 12 unknowns can be solved using $N \geq 6$ points at *general positions* (Faugeras, 1993), with 3-D world coordinates $\mathbf{x}_i^w = \begin{bmatrix} x_i^w & y_i^w & z_i^w \end{bmatrix}^T$ and corresponding 2-D image coordinates $\mathbf{x}_i^{im} = \left[x_i^{im} \quad y_i^{im} \right]^T$ (i = 1 ... N) by the following equation:

$$\mathbf{A} \cdot \mathbf{p} = \mathbf{0}_{2N\times1}, \tag{21}$$

where $\mathbf{0}_{2N\times1}$ is a column vector with all $2N$ elements being 0 and

$$\mathbf{A} = \begin{bmatrix} x_1^w & y_1^w & z_1^w & 1 & 0 & 0 & 0 & 0 & -x_1^w x_1^{im} & -y_1^w x_1^{im} & -z_1^w x_1^{im} & -x_1^{im} \\ 0 & 0 & 0 & 0 & x_1^w & y_1^w & z_1^w & 1 & -x_1^w y_1^{im} & -y_1^w y_1^{im} & -z_1^w y_1^{im} & -y_1^{im} \\ x_2^w & y_2^w & z_2^w & 1 & 0 & 0 & 0 & 0 & -x_2^w x_2^{im} & -y_2^w x_2^{im} & -z_2^w x_2^{im} & -x_2^{im} \\ 0 & 0 & 0 & 0 & x_2^w & y_2^w & z_2^w & 1 & -x_2^w y_2^{im} & -y_2^w y_2^{im} & -z_2^w y_2^{im} & -y_2^{im} \\ & & & & & & & \vdots & & & & \\ x_N^w & y_N^w & z_N^w & 1 & 0 & 0 & 0 & 0 & -x_N^w x_N^{im} & -y_N^w x_N^{im} & -z_N^w x_N^{im} & -x_N^{im} \\ 0 & 0 & 0 & 0 & x_N^w & y_N^w & z_N^w & 1 & -x_N^w y_N^{im} & -y_N^w y_N^{im} & -z_N^w y_N^{im} & -y_N^{im} \end{bmatrix},$$

$$\mathbf{p} = \left[p_1^1 \quad p_2^1 \quad p_3^1 \quad p_4^1 \quad p_1^2 \quad p_2^2 \quad p_3^2 \quad p_4^2 \quad p_1^3 \quad p_2^3 \quad p_3^3 \quad p_4^3 \right]^T.$$

Since the overall scaling of the 12 unknowns is irrelevant, a certain constraint should be imposed. This constraint is, in fact, used to get rid of the scale randomicity of the camera projection (multiple 3-D objects with different scales may correspond to the same 2-D projections). A simple form is to let one unknown be equal to one, for example, $p_4^3 = 1$. In this case a simpler linear equation can be derived. The remaining 11 unknowns can thus be calculated from this new equation by employing various methods, e.g., least squares. However, because of the possibility of singularity of this assumption, that is $p_4^3 = 0$, other forms of constraints should be imposed instead. One possibility is the constraint $\left(p_1^3\right)^2 + \left(p_2^3\right)^2 + \left(p_3^3\right)^2 = 1$. In this case, the problem to be solved is reformulated as (ref. equation 21) the minimization of $\|\mathbf{Ap}\|$ subject to the constraint that $\|\mathbf{Cp}\| = 1$. $\mathbf{C}$ is defined such that $c_j^i = 0$ ($i, j = 1 \ldots 12$), where c_j^i represents the element at the i^{th} row and j^{th} column, except $c_9^9 = c_{10}^{10} = c_{11}^{11} = 1$. A closed-form solution to this problem can be obtained by the method described in Faugeras (1993) and Hartley & Zisserman (2000).

Camera parameter recovery

The values of all camera parameters mentioned can be recovered directly from the results of DLT.

Combining the three items in equation 4 gives:

$$\begin{bmatrix} p_1^1 & p_2^1 & p_3^1 & p_4^1 \\ p_1^2 & p_2^2 & p_3^2 & p_4^2 \\ p_1^3 & p_2^3 & p_3^3 & p_4^3 \end{bmatrix} \cong \begin{bmatrix} -f_x(\mathbf{r}_1)^T + x_0(\mathbf{r}_3)^T & -f_x t_x + x_0 t_z \\ -f_y(\mathbf{r}_2)^T + y_0(\mathbf{r}_3)^T & -f_y t_y + y_0 t_z \\ (\mathbf{r}_3)^T & t_z \end{bmatrix}, \qquad (22)$$

where $\mathbf{r}j$ $(j = 1...3)$ is the j^{th} column of matrix $\mathbf{R}_c^w$, and $\begin{bmatrix} t_x & t_y & t_z \end{bmatrix}^T = -(\mathbf{R}_c^w)^T \mathbf{t}_c^w$.

Let:

$$\begin{bmatrix} (\mathbf{q}'^1)^T & q'^1_4 \\ (\mathbf{q}'^2)^T & q'^2_4 \\ (\mathbf{q}'^3)^T & q'^3_4 \end{bmatrix} = \frac{1}{\rho} \begin{bmatrix} (\mathbf{q}^1)^T & p_4^1 \\ (\mathbf{q}^2)^T & p_4^2 \\ (\mathbf{q}^3)^T & p_4^3 \end{bmatrix} = \frac{1}{\rho} \begin{bmatrix} p_1^1 & p_2^1 & p_3^1 & p_4^1 \\ p_1^2 & p_2^2 & p_3^2 & p_4^2 \\ p_1^3 & p_2^3 & p_3^3 & p_4^3 \end{bmatrix}, \qquad (23)$$

where $\rho = \sqrt{(p_1^3)^2 + (p_2^3)^2 + (p_3^3)^2}$ and $\mathbf{q}'^i = \mathbf{q}^i / \rho = \begin{bmatrix} p_1^i & p_2^i & p_3^i \end{bmatrix}^T / \rho$ $(i = 1...3)$.

Then, all original intrinsic and extrinsic camera parameters can be recovered as:

$$t_z = \varepsilon_z q'^3_4, \ \mathbf{r}_3 = \varepsilon_z \mathbf{q}'^3, \ x_0 = (\mathbf{q}'^1)^T \mathbf{q}'^3, \ y_0 = (\mathbf{q}'^2)^T \mathbf{q}'^3,$$

$$f_y = \left\| \mathbf{q}'^2 - y_0 \mathbf{q}'^3 \right\|, \ \mathbf{r}_2 = -\varepsilon_z (\mathbf{q}'^2 - y_0 \mathbf{q}'^3) / f_y, \ \mathbf{r}_1 = -\varepsilon_z (\mathbf{q}'^1 - x_0 \mathbf{q}'^3) / f_x,$$

$$f_x = \left\| \mathbf{q}'^1 - x_0 \mathbf{q}'^3 \right\|,\ t_y = -\left(\varepsilon_z q'^2_4 - y_0 t_z\right)/\ f_y,\ t_x = -\left(\varepsilon_z q'^1_4 - x_0 t_z\right)/\ f_x, \qquad (24)$$

where ε_z (= ±1) is related to the so-called *oriented projective geometry* (Stolfi, 1991). In the current case, ε_z can be determined by judging if the CS's origin lies in front of the camera ($t_z > 0$) or behind it ($t_z < 0$).

However, due to the influence of noise and camera distortion, from equation 24 it is impossible to guarantee that the recovered matrix $\mathbf{R}_c^w$ is orthonormal, which is a requirement that must be met for $\mathbf{R}_c^w$ to be a rotation matrix. The closest orthonormal matrix to $\mathbf{R}_c^w$ can be found by employing one of the methods provided in Weng et al. (1992) or Horn (1987). However, by doing so, the resulting parameters may not fulfill the linear projection model *optimally* anymore. That is why a geometrically valid DLT is needed.

Geometrically Valid DLT

Employing the DLT method described, one can recover 11 independent elements of the matrix $\tilde{\mathbf{P}}$. However, according to a previous section in this chapter, $\tilde{\mathbf{P}}$ has only 10 DOFs. This means that the recovered 11 independent elements may not be geometrically valid. In other words, certain geometric constraints may not be fulfilled by the 10 intrinsic and extrinsic parameters of a camera recovered from the reconstructed $\tilde{\mathbf{P}}$ of the simple DLT. For this problem, there are three solutions.

Camera geometry promotion

In order to match the 11 DOFs of the projection matrix $\tilde{\mathbf{P}}$, one could add one more DOFs to the camera parameter space by taking into account the skew factor u. By this change, substituting equation 6 into equation 5 yields (ref. equation 17):

$$\begin{bmatrix} p_1^1 & p_2^1 & p_3^1 & p_4^1 \\ p_1^2 & p_2^2 & p_3^2 & p_4^2 \\ p_1^3 & p_2^3 & p_3^3 & p_4^3 \end{bmatrix} \cong \begin{bmatrix} -f_x(\mathbf{r}_1)^T + u(\mathbf{r}_2)^T + x_0(\mathbf{r}_3)^T & -f_x t_x + u t_y + x_0 t_z \\ -f_y(\mathbf{r}_2)^T + y_0(\mathbf{r}_3)^T & -f_y t_y + y_0 t_z \\ (\mathbf{r}_3)^T & t_z \end{bmatrix}, \qquad (25)$$

where $\mathbf{r}_j\,(j = 1...3)$ is the j^{th} column of matrix $\mathbf{R}_c^w$, and $\begin{bmatrix} t_x & t_y & t_z \end{bmatrix}^T = -\left(\mathbf{R}_c^w\right)^T \mathbf{t}_c^w$. Then, from equation 25, it turns out immediately (ref. equation 23) that:

$$u = -\left(\mathbf{q}'^1 \times \mathbf{q}'^2\right)^T \left(\mathbf{q}'^2 \times \mathbf{q}'^3\right),\ t_z = \varepsilon_z q'^3_4,\ \mathbf{r}_3 = \varepsilon_z \mathbf{q}'^3,\ x_0 = \left(\mathbf{q}'^1\right)^T \mathbf{q}'^3,\ y_0 = \left(\mathbf{q}'^2\right)^T \mathbf{q}'^3,$$

$$f_y = \left\|\mathbf{q}'^2 - y_0 \mathbf{q}'^3\right\|,\ \mathbf{r}_2 = -\varepsilon_z \left(\mathbf{q}'^2 - y_0 \mathbf{q}'^3\right) / f_y,\ \mathbf{r}_1 = \mathbf{r}_2 \times \mathbf{r}_3,$$

$$f_x = \left\|\varepsilon_z \mathbf{q}'^1 - u\mathbf{r}_2 - x_0 \mathbf{r}_3\right\|,\ t_y = -\left(\varepsilon_z q'^2_4 - y_0 t_z\right) / f_y,$$

$$t_x = -\left(\varepsilon_z q'^1_4 - u t_y - x_0 t_z\right) / f_x, \qquad (24)$$

where ε_z (= ±1) is the same as in equation 24.

Constrained DLT

In some cases, the skew factor need not be considered, while in others the nonlinear effects have been eliminated in advance by means of the techniques. Then an additional constraint $\left(\mathbf{q}'^1 \times \mathbf{q}'^2\right)^T \left(\mathbf{q}'^2 \times \mathbf{q}'^3\right) = 0$, which guarantees the orthonormality of the matrix $\mathbf{R}_c^w$, can be added to the calculation of DLT. Taking this into consideration, the original DLT problem should be redefined as (ref. equations 21 and 23):

Minimize $\|\mathbf{Ap}\|$ *subject to the constraints* $\left(p_1^3\right)^2 + \left(p_2^3\right)^2 + \left(p_3^3\right)^2 = 1$ *and* $\left(\mathbf{q}'^1 \times \mathbf{q}'^2\right)^T \left(\mathbf{q}'^2 \times \mathbf{q}'^3\right) = 0$. (27)

However, because the *algebraic distance* $d_{a\lg} = \|\mathbf{Ap}\|$ is neither geometrically nor statistically meaningful (Hartley & Zisserman, 2000), it is better to minimize the following *geometric distance*:

$$d_{geo} = \sum_{i=1}^{N} \left\{ \left(\frac{\left(\mathbf{q}^1\right)^T \mathbf{x}^w + q_4^1}{\left(\mathbf{q}^3\right)^T \mathbf{x}^w + q_4^3} - x_i^{im} \right)^2 + \left(\frac{\left(\mathbf{q}^2\right)^T \mathbf{x}^w + q_4^2}{\left(\mathbf{q}^3\right)^T \mathbf{x}^w + q_4^3} - y_i^{im} \right)^2 \right\}.$$

Then the problem to be solved becomes:

Minimize d_{geo} *subject to the constraints in equation 27.* (28)

After solving the optimization problem 28, all original camera parameters can be recovered by applying equation 24.

Modified DLT

To decrease the number of constraint equations required in the nonlinear optimization process (so as to increase its efficiency), one can incorporate the constraints in equation 27 in an alternative way.

According to Hatze (1988), equation 17 can be rewritten as:

$$x^{im} - x_0 = -f_x \cdot \frac{r_1^1 x^w + r_1^2 y^w + r_1^3 z^w + t_x}{r_3^1 x^w + r_3^2 y^w + r_3^3 z^w + t_z} = \frac{a_1 x^w + a_2 y^w + a_3 z^w + a_4}{a_9 x^w + a_{10} y^w + a_{11} z^w + a_{12}},$$

$$y^{im} - y_0 = -f_y \cdot \frac{r_2^1 x^w + r_2^2 y^w + r_2^3 z^w + t_y}{r_3^1 x^w + r_3^2 y^w + r_3^3 z^w + t_z} = \frac{a_5 x^w + a_6 y^w + a_7 z^w + a_8}{a_9 x^w + a_{10} y^w + a_{11} z^w + a_{12}}, \qquad (29)$$

where r_j^i $(i, j = 1\ldots3)$ is the element of $\mathbf{R}_c^w$ at the i^{th} row and j^{th} column and $\begin{bmatrix} t_x & t_y & t_z \end{bmatrix}^T = -\left(\mathbf{R}_c^w\right)^T \mathbf{t}_c^w$.

Therefore, as $\left(\mathbf{R}_c^w\right)^T \mathbf{R}_c^w = \mathbf{I}$, we know immediately that:

$$a_1 a_5 + a_2 a_6 + a_3 a_7 = 0, \text{ and } a_1 a_9 + a_2 a_{10} + a_3 a_{11} = 0, \text{ and}$$
$$a_1 a_5 + a_2 a_6 + a_3 a_7 = 0. \qquad (30)$$

By further requiring that $a_{12} = 1$ or $(a_9)^2 + (a_{10})^2 + (a_{11})^2 = 1$, we can carry out a constrained non-linear search to obtain the 10 independent parameters (x_0, y_0, a_3, a_4, a_6, ..., a_{11}; in case $a_{12} = 1$) or (x_0, y_0, a_3, a_4, a_6, ..., a_{10}, a_{12}; in case $(a_9)^2 + (a_{10})^2 + (a_{11})^2 = 1$).

Up to this point, only linear relations in the imaging process were considered. This already suffices if one aims more at efficiency than accuracy. However, for highly accurate measurements, the distortion should also be taken into account. A straightforward way to achieve this is to incorporate the distortion coefficients directly into the DLT calculation by adding some nonlinear elements. This is the topic of the next section.

DLT Considering Distortion

Both simple DLT and geometrically valid DLTs cannot take the distortion component into account. In this section, an innovative way of incorporating the distortion into the DLT is discussed. Its advantages and disadvantages are addressed, as well.

Because of the existence of two distortion models, the 3-D reconstruction and the projection have to be handled differently.

Two-plane approach using reconstruction-distortion model

This approach is used for 3-D reconstruction.

Rearranging equation 5, we obtain:

$$\mathbf{x}^w \cong \mathbf{R}_c^w \tilde{\mathbf{K}}^{-1} \tilde{\mathbf{x}}^{im} + \mathbf{t}_c^w, \qquad (31)$$

where

$$\tilde{\mathbf{K}}^{-1} = \begin{bmatrix} -1/f_x & 0 & x_0/f_x \\ 0 & -1/f_y & y_0/f_y \\ 0 & 0 & 1 \end{bmatrix}. \qquad (32)$$

Now, by assuming that the camera distortion is subject to the reconstruction-distortion model, and by taking equation 12 into equation 31, we obtain:

$$\mathbf{x}^{w} \cong \mathbf{D}\tilde{\hat{\mathbf{u}}}^{im}, \tag{33}$$

where $\mathbf{D}$ is a matrix with 3 rows and 36 columns. $\tilde{\hat{\mathbf{u}}}^{im}$ is a column vector with 36 elements in the form $\left(\hat{x}^{im}\right)^{i}\left(\hat{y}^{im}\right)^{j}$ ($i, j = 0\ldots7$ and $i + j \leq 7$).

In fact, the dimensions of $\mathbf{D}$ and $\tilde{\hat{\mathbf{u}}}^{im}$ depend on the distortion (coefficients) considered. Several examples are shown in Table 2, where the last case is the same linear situation that was just handled in the previous section.

If enough pairs of $\mathbf{x}^{w}$ and corresponding $\hat{\mathbf{x}}^{im}$ (and thus $\tilde{\hat{\mathbf{u}}}^{im}$) are available, $\mathbf{D}$ can be calculated from equation 33 in the same way as illustrated. Subsequently, from $\mathbf{D}$, the line of sight of any image pixel in the current camera can be computed. Furthermore, assume that in another camera, which has transformation matrix $\mathbf{D}'$, a point $\hat{\mathbf{x}}'^{im}$ (and thus $\tilde{\hat{\mathbf{u}}}'^{im}$) can be located which is the projection of the same world point $\mathbf{x}^{w}$ as the point $\hat{\mathbf{x}}^{im}$. Then:

$$\mathbf{x}^{w} \cong \mathbf{D}'\tilde{\hat{\mathbf{u}}}'^{im}. \tag{34}$$

By combining equations 33 and 34, one can easily reconstruct $\mathbf{x}^{w}$ (Faugeras, 1993).

Table 2. The dimensions of the transformation matrix and vector used in DLT considering distortion.

Considered distortion coefficients	Dimension of $\mathbf{D}$	Dimension of $\tilde{\hat{\mathbf{u}}}^{im}$
k_1^{Re}, k_2^{Re}, k_3^{Re}, P_1^{Re}, P_2^{Re}, s_1^{Re}, s_2^{Re}	3×36	36×1
k_1^{Re}, k_2^{Re}, P_1^{Re}, P_2^{Re}, s_1^{Re}, s_2^{Re}	3×21	21×1
k_1^{Re}, P_1^{Re}, P_2^{Re}, s_1^{Re}, s_2^{Re}	3×10	10×1
None	3×3	3×1

Distorted projection approach with imaging-distortion model

This approach is used for the projection purpose.

Similarly, assume that the camera distortion is subject to the imaging-distortion model. Substituting equation 5 into equation 10 gives:

$$\tilde{\hat{\mathbf{x}}}^{im} \cong \bar{\mathbf{D}}\tilde{\mathbf{u}}^{w}, \tag{35}$$

where $\tilde{\hat{\mathbf{x}}}^{im} = \begin{bmatrix} \hat{x}^{im} & \hat{y}^{im} & 1 \end{bmatrix}^T$; $\bar{\mathbf{D}}$ is a matrix with 3 rows and 120 columns. $\tilde{\mathbf{u}}^{w}$ is a column vector with 120 elements in the form $\left(\hat{x}^{w}\right)^{i}\left(\hat{y}^{w}\right)^{j}\left(\hat{z}^{w}\right)^{k}$ $(i, j, k = 0\ldots 7$ and $i + j + k \le 7$).

If all world points $\mathbf{x}^{w} = \begin{bmatrix} x^{w} & y^{w} & z^{w} \end{bmatrix}^T$ belong to the same plane, it can be assumed that without loss of generality all z^{w} equal 0. If at the same time only distortion coefficients k_1^{Im}, P_1^{Im}, P_2^{Im}, s_1^{Im}, and s_2^{Im} are considered, then the dimension of $\bar{\mathbf{D}}$ decreases to 3×10 and that of $\tilde{\mathbf{u}}^{w}$ to 10×1.

Again, if we have enough pairs of $\tilde{\hat{\mathbf{x}}}^{im}$ and corresponding $\mathbf{x}^{w}$ (and thus $\tilde{\mathbf{u}}^{w}$), $\bar{\mathbf{D}}$ can be calculated from equation 35. From $\bar{\mathbf{D}}$, the projection of any world point into the current camera can be computed by means of equation 35 in a linear fashion.

Perspectivity

For an arbitrary point with coordinates $\breve{\mathbf{x}}^{w}{}_{1} = \begin{bmatrix} x^{w}{}_{1} & y^{w}{}_{1} \end{bmatrix}^T$ in a world plane Π_1, its coordinates in the WCS, whose x and y axes are the same as those of Π_1, are obviously $\tilde{\breve{\mathbf{x}}}^{w}{}_{1} = \begin{bmatrix} x^{w}{}_{1} & y^{w}{}_{1} & 1 \end{bmatrix}^T$. Following the same procedure as the one leading to equation 33, it can be derived that:

$$\tilde{\breve{\mathbf{x}}}_1^{w} \cong \mathbf{E}_1 \tilde{\hat{\mathbf{u}}}^{im}, \tag{36}$$

where $\tilde{\breve{\mathbf{x}}}^{w}{}_{1} = \begin{bmatrix} x^{w}{}_{1} & y^{w}{}_{1} & 1 \end{bmatrix}^T$, $\mathbf{E}_1$ is a matrix with 3 rows and 36 columns, $\begin{bmatrix} \hat{x}^{im} & \hat{y}^{im} \end{bmatrix}^T$ is the projection of $\breve{\mathbf{x}}^{w}{}_{1}$ onto the image plane of the current camera, and $\tilde{\hat{\mathbf{u}}}^{im}$ is a column vector with 36 elements in the form $\left(\hat{x}^{im}\right)^i \left(\hat{y}^{im}\right)^j$ ($i, j = 0\ldots7$ and $i + j \leq 7$).

At the same time, if a point with coordinate $\breve{\mathbf{x}}_2^w = \left[x^w{}_2 \quad y^w{}_2\right]^T$ in another plane Π_2 projects onto the image plane of the current camera also at $\left[\hat{x}^{im} \quad \hat{y}^{im}\right]^T$, we similarly obtain that:

$$\tilde{\breve{\mathbf{x}}}_2^w \cong \mathbf{E}_2 \tilde{\hat{\mathbf{u}}}^{im}, \tag{37}$$

where $\tilde{\breve{\mathbf{x}}}_2^w = \left[x^w{}_2 \quad y^w{}_2 \quad 1\right]^T$, $\mathbf{E}_2$ is also a matrix with 3 rows and 36 columns.

Since they have the same projection $\left[\hat{x}^{im} \quad \hat{y}^{im}\right]^T$, a special relation called *perspectivity* (Hartley & Zisserman, 2000) should exist between $\breve{\mathbf{x}}_1^w$ and $\breve{\mathbf{x}}_2^w$. This perspectivity relation between all corresponding pairs in Π_1 and Π_2 can be described by a 3×3 matrix $\tilde{\mathbf{C}}$ as:

$$\tilde{\breve{\mathbf{x}}}_2^w \cong \tilde{\mathbf{C}} \tilde{\breve{\mathbf{x}}}_1^w. \tag{38}$$

Of course $\tilde{\mathbf{C}}$ has to fulfill some constraints for it to be a perspective transformation (Wei & Ma, 1994). Otherwise, it is just a plane-to-plane homography.

Combining equation 38 with equations 36 and 37 yields:

$$\mathbf{E}_2 \cong \tilde{\mathbf{C}} \mathbf{E}_1. \tag{39}$$

Therefore, instead of recovering $\mathbf{E}_1$ and $\mathbf{E}_2$ separately, $\mathbf{E}_1$ and $\tilde{\mathbf{C}}$ can be calculated first. Then equation 39 is employed to recover $\mathbf{E}_2$. Thus, it is ensured that the perspective constraint is satisfied.

Discussion

By considering the distortion implicitly, as was done in this section, linear estimation techniques can be utilized for calibration. However, several problems arise: First, for a high degree of distortion, the scale of the model increases dramatically; Second, it is very difficult to take into account projective constraints, as addressed in a previous section for DLT, in the calibration; Third, only when distortion is not considered, can the original camera parameters be computed linearly with this model (Wei & Ma, 1994). Therefore, a more efficient way of estimating the distortion coefficients is needed. For this purpose, Tsai's algorithm is a good and representative example.

Tsai's Algorithm

Assuming that only the radial distortion occurs in the camera and the principal point $\begin{bmatrix} x_0 & y_0 \end{bmatrix}^T$ is known (or can be approximated) in advance, Tsai (1987) proposed a two-stage algorithm for explicitly calibrating the camera. In this case, the imaging equations used are exactly the same as those in equation 12, except that the possible distortion is limited as follows:

$$\begin{bmatrix} \Delta_x \\ \Delta_y \end{bmatrix} = \begin{bmatrix} \hat{x}r^2 & \hat{x}r^4 & \hat{x}r^6 & \cdots \\ \hat{y}r^2 & \hat{y}r^4 & \hat{y}r^6 & \cdots \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \\ k_3 \\ \vdots \end{bmatrix}.$$

With this radial distortion, a pixel in the image is only distorted along the radial direction, thus a *radial alignment constraint* (**RAC**) can be formed (Tsai, 1987) (ref. Equation 12):

$$\begin{bmatrix} \frac{s_y}{s_x} \cdot x^c \\ y^c \end{bmatrix} \times \begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = \left(-\frac{s_y z_c}{f} \cdot \begin{bmatrix} \hat{x}^{im} - x_0 \\ \hat{y}^{im} - y_0 \end{bmatrix} \right) \times \begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = 0 . \tag{40}$$

Expanding equation 2 gives:

$$\mathbf{x}^c = \left(\mathbf{R}_c^w\right)^T \left(\mathbf{x}^w - \mathbf{t}_c^w\right) == \left(\mathbf{R}_c^w\right)^T \mathbf{x}^w + \mathbf{t} = \begin{bmatrix} r_1^1 & r_1^2 & r_1^3 \\ r_2^1 & r_2^2 & r_2^3 \\ r_3^1 & r_3^2 & r_3^3 \end{bmatrix} \begin{bmatrix} x^w \\ y^w \\ z^w \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}, \tag{41}$$

where r_j^i $(i, j = 1\ldots3)$ is the element of $\mathbf{R}_c^w$ at the i^{th} row and j^{th} column, and

$$\mathbf{t} = \begin{bmatrix} t_x & t_y & t_z \end{bmatrix}^T = -\left(\mathbf{R}_c^w\right)^T \mathbf{t}_c^w \ .$$

Then, substituting equation 40 into equation 41 yields (assuming $t_y \neq 0$):

$$\begin{bmatrix} \hat{y}x^w & \hat{y}y^w & \hat{y}z^w & \hat{y} & -\hat{x}x^w & -\hat{x}y^w & -\hat{x}z^w \end{bmatrix} \cdot \mathbf{a} - \hat{x} = 0$$

where **a** contains the seven unknowns:

$$\begin{aligned} \mathbf{a} &= \begin{bmatrix} a^1 & a^2 & a^3 & a^4 & a^5 & a^6 & a^7 \end{bmatrix}^T \\ &= \begin{bmatrix} \frac{s_y}{s_x} t_y^{-1} r_1^1 & \frac{s_y}{s_x} t_y^{-1} r_1^2 & \frac{s_y}{s_x} t_y^{-1} r_1^3 & \frac{s_y}{s_x} t_y^{-1} t_x & t_y^{-1} r_2^1 & t_y^{-1} r_2^2 & t_y^{-1} r_2^3 \end{bmatrix}^T . \end{aligned}$$

With more than seven 3-D world points at general positions, one can estimate the seven unknowns from an over-determined linear system (by stacking multiple equation 42). After that, $\mathbf{R}_c^w$, t_x, and t_y can be calculated as:

$$\left|t_y\right| = \sqrt{\left(a^5\right)^2 + \left(a^6\right)^2 + \left(a^7\right)^2} \ , \ \frac{s_y}{s_x} = \left|t_y\right| \cdot \sqrt{\left(a^1\right)^2 + \left(a^2\right)^2 + \left(a^3\right)^2} \ , \ t_x = \frac{s_x}{s_y} a^4 t_y \ ,$$

$$\begin{bmatrix} r_1^1 & r_1^2 & r_1^3 \end{bmatrix}^T = \frac{s_x}{s_y} t_y \begin{bmatrix} a^1 & a^2 & a^3 \end{bmatrix}^T , \ \begin{bmatrix} r_2^1 & r_2^2 & r_2^3 \end{bmatrix}^T = t_y \begin{bmatrix} a^5 & a^6 & a^7 \end{bmatrix}^T ,$$

$$\begin{bmatrix} r_3^1 & r_3^2 & r_3^3 \end{bmatrix}^T = \begin{bmatrix} r_1^1 & r_1^2 & r_1^3 \end{bmatrix}^T \times \begin{bmatrix} r_2^1 & r_2^2 & r_2^3 \end{bmatrix}^T .$$

The sign of t_y can be determined by requiring that $\hat{x}$ and x^c (respectively $\hat{y}$ and y^c) have opposite signs.

On the other hand, if all available points are co-planar, that is $z^w = 0$, equation 42 becomes:

$$\left[\hat{y}x^w \quad \hat{y}y^w \quad \hat{y} \quad -\hat{x}x^w \quad -\hat{x}y^w\right] \cdot \mathbf{a}' - \hat{x} = 0\,, \tag{43}$$

where $\mathbf{a}' = \left[\frac{s_y}{s_x} t_y^{-1} r_1^1 \quad \frac{s_y}{s_x} t_y^{-1} r_1^2 \quad \frac{s_y}{s_x} t_y^{-1} t_x \quad t_y^{-1} r_2^1 \quad t_y^{-1} r_2^2\right]^T$.

In this case, $\mathbf{R}_c^w$, t_x, and t_y can be recovered from the calculated vector $\mathbf{a}'$ only if the aspect ratio $s_y\ /\ s_x$ is known.

In summary, there are two stages in this algorithm:

1. Compute the 3-D *pose* $\mathbf{R}_c^w$, t_x, t_y, and s_y / s_x (in case enough non-co-planar points are available); and
2. Optimize the effective focal length f, radial distortion coefficients k_1, k_2, ..., and t_z, by employing a simple search scheme.

Because the problem has been split into these two stages, the whole computation becomes much simpler and more efficient.

Further improvements

A requirement of Tsai's algorithm is that the position of the principal point and the aspect ratio (in case only co-planar points are available) are known *a priori*. One practical possibility of finding the principal point accurately is to minimize the left-hand side of equation 42 (in the non-co-planar case) or that of equation 43 (in the co-planar case) (Lenz & Tsai, 1988; Penna, 1991). The horizontal scale factor (and thus the aspect ratio) can be measured by using the difference between the scanning frequency of the camera sensor plane and the scanning frequency of the image capture board frame buffer (Lenz & Tsai, 1988). However, this scale factor estimation method is not so practical due to the difficulty of measuring the required frequencies (Penna, 1991). A more direct way would be to employ the image of a sphere for calculating the aspect ratio (Penna, 1991). The power spectrum of the images of two sets of parallel lines can also be utilized for the same purpose (Bani-Hashemi, 1991).

In addition, the RAC model requires that the angle of incidence between the optical axis of the camera and the calibration plane should be at least 30^0 (Tsai, 1987). This ill-conditioned situation can be avoided by setting cos $\alpha = 1$ and sin $\alpha = \alpha$ when $\alpha \rightarrow 0$ (Zhuang & Wu, 1996).

The above modifications improved the capabilities of the original Tsai's algorithm. However, Tsai's algorithm can still only take the radial distortion into consideration. And, the strategy of recovering several subsets of the whole camera parameter space in separate steps may suffer from stability and convergence problems due to the tight correlation between camera parameters (Slama, 1980). It is desirable, therefore, to estimate all parameters simultaneously. Methods based on this idea are discussed below.

Iterative Two-Phase Strategy

During the 1980s, camera calibration techniques were mainly full-scale nonlinear optimization incorporating distortion (Slama, 1980) or techniques that only take into account the linear projection relation as depicted by equation 5 (Abdel-Aziz & Karara, 1971). The *iterative two-phase strategy* was proposed at the beginning of the 1990s to achieve a better performance by combining the above two approaches (Weng et al., 1992). With this iterative strategy, a linear estimation technique such as DLT is applied in phase 1 to approximate the imaging process by $\mathbf{p}_l$, and then in phase 2, starting with the linearly recovered $\mathbf{p}_l$ and $\mathbf{p}_d = \mathbf{0}$, an optimization process is performed iteratively until a best fitting parameter point $\mathbf{p}_c$ is reached.

Because for most cameras, the linear model (ref. equation 5) is quite adequate, and the distortion coefficients are very close to 0, it can be argued that this iterative two-phase strategy would produce better results than pure linear techniques or pure full-scale nonlinear search methods. Camera calibration methods employing the iterative two-phase strategy differ mainly in the following three aspects: 1) The adopted distortion model and distortion coefficients; 2) The linear estimation technique; and 3) The objective function to be optimized. In Sid-Ahmed & Boraie (1990), for example, the reconstruction-distortion model is utilized and k_1^{Re}, k_2^{Re}, k_3^{Re}, P_1^{Re}, and P_2^{Re} are considered. The DLT method introduced is directly employed in phase 1. Then, in phase 2, assuming that $\begin{bmatrix} x_0 & y_0 \end{bmatrix}^T$ is already known, the Marquardt method is used to solve a least-squares problem with respect to $\overline{\mathbf{p}} = \begin{bmatrix} \mathbf{p}^T & k_1^{\text{Re}} & k_2^{\text{Re}} & k_3^{\text{Re}} & P_1^{\text{Re}} & P_2^{\text{Re}} \end{bmatrix}^T$ (ref. equation 21). All camera parameters are made **implicit**.

To further guarantee the geometric validity of the estimated $\overline{\mathbf{p}}$, one extra phase can be introduced between phase 1 and phase 2. In this extra phase, elements in $\overline{\mathbf{p}}$ are modified for fulfilling the orthonormality of the rotation matrix $\mathbf{R}_c^w$. In Zhang (2000), the imaging-distortion model is used and k_1^{Im} and k_2^{Im} are considered. In phase 1, a planar object strategy is employed. Then, in phase 2, the Levenberg-Marquardt method is used to minimize the following objective function over all camera parameters $\mathbf{p}_c$:

$$f(\mathbf{p}_c) = \sum \sqrt{\left(\hat{x}^{im} - x^{im} - f_x \cdot \Delta_x^{\mathrm{Im}}\right)^2 + \left(\hat{y}^{im} - y^{im} - f_y \cdot \Delta_y^{\mathrm{Im}}\right)^2}, \tag{44}$$

where the summation is done over all available data and all variables were defined in equation 10. Instead of the linear estimation method in phase 1, a nonlinear optimization can also be carried out to get a better initial guess of $\mathbf{p}_l$ with all distortion coefficients set to zero.

Optimization issue

At phase 2, each iteration of the optimization can also be split up into the following two steps (Weng et al., 1992):

Step a: The function $f(\mathbf{p}_c)$ in equation 44 is minimized w.r.t. all distortion coefficients in $\mathbf{p}_d$ by a simple linear least-squares method, while $\mathbf{p}_l$ (containing all linear intrinsic and extrinsic parameters) is fixed.

Step b: The function $f(\mathbf{p}_c)$ is minimized by an iterative optimization method w.r.t. $\mathbf{p}_l$ while $\mathbf{p}_d$ remains unchanged.

However, due to the tight interaction between the linear parameters and the distortion coefficients, this two-step optimization converges very slowly.

In Chatterjee, Roychowdhury & Chong (1997), the nonlinear optimization part (phase 2) is further divided into three stages by following the Gauss-Seidel approach. The first stage is similar to the optimization phase in Sid-Ahmed & Boraie (1990), except that the optical center $[x_0 \quad y_0]^T$, the aspect ratio s_y / s_x, and all distortion coefficients are fixed. The second stage is the same as Step "a" in Weng et al. (1992). Finally, in the third stage, the function $f(\mathbf{p}_c)$ in equation 44 is minimized only w.r.t. the optical center $[x_0 \quad y_0]^T$ and the aspect ratio s_y / s_x, while all other camera parameters are fixed. Convergence analysis for this new parameter space partition method has been given in Chatterjee et al. (1997). However, no convergence speed was provided.

Preferably, all camera parameters (including extrinsic and intrinsic parameters, and distortion coefficients) should be optimized simultaneously. To do so, usually a certain iterative technique called *bundle adjustment* is adopted (Triggs et al., 1999). Among them, the Levenberg-Marquardt method is probably the most extensively employed, due to its robustness.

Conclusions

As it combines the linear initialization and the nonlinear full-scale optimization, the iterative two-phase strategy can provide very accurate calibration results with reasonable speed. It is now employed extensively. There exist many variations of it aiming at different compromises between accuracy and efficiency as described above. However, for a complete passive calibration system, the design of the calibration object also plays a quite important role. The next section will introduce a simple but effective calibration object.

Planar Pattern Based Calibration

Various 2-D planar patterns have been used as calibration targets. Compared with 3-D calibration objects, 2-D planar patterns can be more accurately manufactured and fit easier into the view volume of a camera. With **known absolute** or **relative** poses, planar patterns are a special type of 3-D calibration object. In this case, traditional non-co-planar calibration techniques can be applied directly or with very little modification (Tsai, 1987). More often, a single planar pattern is put at several **unknown** poses to calibrate a camera (Zhang, 2000). Each pose of the planar pattern is called a ***frame***. It has been demonstrated that this is already adequate for calibrating a camera. The iterative two-phase strategy discussed can still be applied here. For planar patterns, only phase 1 is different and it is discussed below.

Recovering of linear geometry

Assume a linear camera model. For an arbitrary point $\mathbf{x}^o = \begin{bmatrix} x^o & y^o & 0 \end{bmatrix}^T$ in the calibration plane with orientation $\mathbf{R}_o^w$ and position $\mathbf{t}_o^w$, we obtain from equation 5

$$\begin{aligned}\tilde{\mathbf{x}}^{im} &\cong \tilde{\mathbf{P}} \cdot \tilde{\mathbf{x}}^w = \tilde{\mathbf{K}} \cdot \tilde{\mathbf{M}} \cdot \tilde{\mathbf{x}}^w = \tilde{\mathbf{K}} \cdot \left[\left(\mathbf{R}_c^w\right)^T \quad -\left(\mathbf{R}_c^w\right)^T \cdot \mathbf{t}_c^w \right] \cdot \tilde{\mathbf{x}}^w = \tilde{\mathbf{K}} \cdot \left[\left(\mathbf{R}_c^w\right)^T \cdot \mathbf{x}^w \quad -\left(\mathbf{R}_c^w\right)^T \cdot \mathbf{t}_c^w \right] \\ &= \tilde{\mathbf{K}} \cdot \left[\left(\mathbf{R}_c^w\right)^T \cdot \left(\mathbf{R}_o^w \mathbf{x}^o + \mathbf{t}_o^w\right) \quad -\left(\mathbf{R}_c^w\right)^T \cdot \mathbf{t}_c^w \right] = \tilde{\mathbf{K}} \cdot \left[\left(\mathbf{R}_c^w\right)^T \cdot \mathbf{R}_o^w \quad \left(\mathbf{R}_c^w\right)^T \cdot \left(\mathbf{t}_o^w - \mathbf{t}_c^w\right) \right] \cdot \tilde{\mathbf{x}}^o \\ &= \tilde{\mathbf{K}} \cdot \left[\mathbf{R} \quad \mathbf{t} \right] \cdot \left[x^o \quad y^o \quad 0 \quad 1 \right] = \tilde{\mathbf{K}} \cdot \left[\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t} \right] \cdot \left[x^o \quad y^o \quad 1 \right]\end{aligned} \quad , \quad (45)$$

where $\mathbf{R} = \left(\mathbf{R}_c^w\right)^T \cdot \mathbf{R}_o^w = \left[\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3\right]$($\mathbf{r}_j$ ($j = 1\ldots3$) is the j^{th} column of the matrix $\mathbf{R}$) and

$$\mathbf{t} = \left(\mathbf{R}_c^w\right)^T \cdot \left(\mathbf{t}_o^w - \mathbf{t}_c^w\right).$$

Thus:

$$\tilde{\mathbf{x}}^{im} \cong \tilde{\mathbf{H}} \cdot \left[x^o \quad y^o \quad 1\right] \text{ where } \tilde{\mathbf{H}} = \tilde{\mathbf{K}} \cdot \left[\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}\right]. \tag{46}$$

By applying the simple DLT method introduced, with at least four pairs of corresponding $\left[x^{im} \quad y^{im}\right]^T$ and $\left[x^o \quad y^o\right]^T$, $\tilde{\mathbf{H}}$ can be determined up to a non-zero factor as $\mathbf{H}$, which means $\tilde{\mathbf{H}} \cong \mathbf{H}$.
As the inverse $\tilde{\mathbf{K}}^{-1}$ of $\tilde{\mathbf{K}}$ exists (ref. equation 32), equation 46 can be rewritten as:

$$\left[\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}\right] = \tilde{\mathbf{K}}^{-1} \cdot \tilde{\mathbf{H}} \cong \tilde{\mathbf{K}}^{-1} \cdot \mathbf{H}. \tag{47}$$

Thus:

$$\mathbf{r}_1 = \rho \tilde{\mathbf{K}}^{-1} \cdot \mathbf{h}_1 \text{ and } \mathbf{r}_2 = \rho \tilde{\mathbf{K}}^{-1} \cdot \mathbf{h}_2,$$

where ρ is a non-zero factor and $\mathbf{h}_j$ (j = 1...3) is the j^{th} column of matrix $\mathbf{H}$. Because $(\mathbf{r}_1)^T\mathbf{r}_1 = (\mathbf{r}_2)^T\mathbf{r}_2 = 1$, it turns out that:

$$\left(\mathbf{h}_1\right)^T \cdot \left(\tilde{\mathbf{K}}^{-T}\tilde{\mathbf{K}}^{-1}\right) \cdot \mathbf{h}_1 = \left(\mathbf{h}_2\right)^T \cdot \left(\tilde{\mathbf{K}}^{-T}\tilde{\mathbf{K}}^{-1}\right) \cdot \mathbf{h}_2, \tag{48}$$

$$\left(\mathbf{h}_1\right)^T \cdot \left(\tilde{\mathbf{K}}^{-T}\tilde{\mathbf{K}}^{-1}\right) \cdot \mathbf{h}_2 = 0, \tag{49}$$

where $\tilde{\mathbf{K}}^{-T}\tilde{\mathbf{K}}^{-1}$ is called the *Image of Absolute Conic* (*IAC*), which has been applied successfully in self-calibration (Hartley & Zisserman, 2000). Once the IAC of a camera is located, the geometry of this camera has been determined.

Equations 48 and 49 thus provide two constraints for the intrinsic matrix $\tilde{\mathbf{K}}^{-1}$ with one frame. Since $\tilde{\mathbf{K}}^{-1}$ has four DOFs (ref. equation 32), if two frames (which means two different **H**) are available, $\tilde{\mathbf{K}}^{-1}$ (and all four intrinsic parameters) can then be recovered.

Once $\tilde{\mathbf{K}}^{-1}$ is determined, $\mathbf{r}_1$, $\mathbf{r}_2$, and **t** can be calculated directly from equation 47 under the constraint $(\mathbf{r}_1)^T\mathbf{r}_1 = (\mathbf{r}_2)^T\mathbf{r}_2 = 1$. It follows that $\mathbf{r}_3$ can then be computed as $\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2$. Here it is obvious that, if $\mathbf{r}_1$ and $\mathbf{r}_2$ are solutions of equation 47, $\mathbf{r}_1' = -\mathbf{r}_1$ and $\mathbf{r}_2' = -\mathbf{r}_2$ also satisfy equation 47. Again, the correct solutions can be verified by means of the oriented projective geometry.

In the single-camera case, without loss of generality, it can be assumed that $\mathbf{R}_c^w = \mathbf{I}$ and $\mathbf{t}_c^w = \mathbf{0}$. Then $\mathbf{R}_o^w = \mathbf{R}$ and $\mathbf{t}_o^w = \mathbf{t}$. However, in a multiple-camera configuration, which is discussed in the next section, things are not so simple.

Conclusions

Using the planar pattern as the calibration object may ease the calibration-data-acquisition work quite a lot. The corresponding calibration method is simple and efficient. However, the algorithm described above only holds for a single-camera case. If multiple cameras need to be calibrated in one system, the calibration algorithm should be modified for obtaining higher accuracy (Slama, 1980). This issue is discussed next.

Multiple Camera Configuration Recovering

Suppose that there are *n* cameras (*n* > 1) and *m* frames (*m* > 1) for which the relative poses among all cameras are to be recovered.

In the general situation, let us assume that each frame can be viewed by every camera. The whole camera and frame set in this configuration is called *complete*. By applying the linear geometry estimation techniques discussed, the relative pose between camera *i* and frame *j* can be computed as:

$$\text{Orientation: } \mathbf{R}_{ij} = \left(\mathbf{R}_{ci}^{w}\right)^{T} \cdot \mathbf{R}_{oj}^{w},$$

$$\text{Position: } \mathbf{t}_{ij} = \left(\mathbf{R}_{ci}^{w}\right)^{T} \cdot \left(\mathbf{t}_{oj}^{w} - \mathbf{t}_{ci}^{w}\right),$$

where $i = 1...n$ and $j = 1...m$, $\mathbf{R}_{ci}^{w}$ is the orientation of camera i and $\mathbf{t}_{ci}^{w}$, its position, and $\mathbf{R}_{oj}^{w}$ is the orientation of frame j and $\mathbf{t}_{oj}^{w}$, its position.

Writing all orientation matrices into one large matrix yields:

$$\underbrace{\begin{bmatrix} \mathbf{R}_{11} & \cdots & \mathbf{R}_{1m} \\ \vdots & \ddots & \vdots \\ \mathbf{R}_{n1} & \cdots & \mathbf{R}_{nm} \end{bmatrix}}_{\mathbf{M}} = \underbrace{\begin{bmatrix} \left(\mathbf{R}_{c1}^{w}\right)^{T} \\ \vdots \\ \left(\mathbf{R}_{cn}^{w}\right)^{T} \end{bmatrix}}_{\mathbf{M}_c} \underbrace{\begin{bmatrix} \mathbf{R}_{o1}^{w} & \cdots & \mathbf{R}_{om}^{w} \end{bmatrix}}_{\mathbf{M}_o} .$$

Let $\mathbf{M} = \mathbf{UWV}^{T}$ be the singular value decomposition (**SVD**) (Press, Teukolsky, Vetterling & Flannery, 1992) of **M**. Let **U'** be the matrix consisting of the three columns of **U** that correspond to the three largest singular values in **W**. Then $\left(\mathbf{R}_{ci}^{w}\right)^{T}$ are estimated as the orthonormal matrices that are closest to the corresponding submatrices in **U'** (Horn, Hilden & Negahdaripour, 1988). $\mathbf{R}_{oj}^{w}$ can be computed in the same way from **V**.

After getting all $\left(\mathbf{R}_{ci}^{w}\right)^{T}$, stacking all position vectors on top of each other results in:

$$\underbrace{\begin{bmatrix} 1 & \mathbf{0}_{m-2}^{T} & 0 & \\ \vdots & \vdots & \vdots & -\mathbf{I}_{n\times n} \\ 1 & \mathbf{0}_{m-2}^{T} & 0 & \\ & \ddots & & \vdots \\ 0 & \mathbf{0}_{m-2}^{T} & 1 & \\ \vdots & \vdots & \vdots & -\mathbf{I}_{n\times n} \\ 0 & \mathbf{0}_{m-2}^{T} & 1 & \end{bmatrix}}_{\mathbf{A}} \cdot \underbrace{\begin{bmatrix} \mathbf{t}_{o1}^{w} \\ \vdots \\ \mathbf{t}_{om}^{w} \\ \mathbf{t}_{c1}^{w} \\ \vdots \\ \mathbf{t}_{cn}^{w} \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} \mathbf{R}_{c1}^{w}\mathbf{t}_{11} \\ \vdots \\ \mathbf{R}_{cn}^{w}\mathbf{t}_{n1} \\ \vdots \\ \mathbf{R}_{c1}^{w}\mathbf{t}_{1m} \\ \vdots \\ \mathbf{R}_{cn}^{w}\mathbf{t}_{nm} \end{bmatrix}}_{\mathbf{b}},$$

where $\mathbf{A}$ is a matrix with dimension $(m \cdot n) \times (m+n)$, $\mathbf{x}$ and $\mathbf{b}$ are vectors with dimensions $(m+n)$ and $(m \cdot n)$, respectively. $\mathbf{0}_{m-2}^{T}$ is a row vector with all $(m-2)$ elements being 0, and $\mathbf{I}_{n\times n}$ is a unit matrix with n rows and n columns.

The above equation is over-determined, and can be solved as follows:

$$\mathbf{x} = \left(\mathbf{A}^{T}\mathbf{A}\right)^{-1}\left(\mathbf{A}^{T}\mathbf{b}\right),$$

where the sparsity of the matrix $\mathbf{A}^{T}\mathbf{A}$ can be explored to improve the efficiency of the computation.

However, the camera and frame set is not always complete, which means that some frames are not visible in certain cameras. For this situation there are two solutions: 1) The whole set can be decomposed into several subsets that are complete by themselves. Then, for each subset the above calculations can be done independently and subsequently combined into one WCS; and 2) The problem is treated as a *missing data* system, which can be solved by the interpolation method proposed in Sturm (2000).

Special Camera Calibration Techniques

In addition to the aforementioned approaches for passive camera calibration, some other special techniques, such as those utilizing projective geometry invariants and special calibration objects, have also been developed. For instance, in Liebowitz & Zisserman (1998), instead of using calibration control points with known metric, other types of constraints, such as a known angle, two equal-but-unknown angles, and a known length ratio are utilized. The most important ones are summarized in the following sections.

Vanishing points

By exploring the geometry property of vanishing points, the camera geometry can be obtained to a certain degree. A vanishing point is the common intersection of all image lines whose 3-D-space correspondences are parallel to each other in the same direction before perspective projection. It can be located reliably in an image.

Vanishing points have several interesting properties (Caprile & Torre, 1990): 1) All vanishing points associated with the sets of lines that are parallel to a given plane lie on the same line in the image; this line is called the *vanishing line* (Wang & Tsai, 1991); 2) Given the vanishing points of three sets of mutually orthogonal lines, the orthocenter of the triangle with the three vanishing points as vertices is the principal point; 3) Given the vanishing points $[x_1 \quad y_1]^T$ and $[x_2 \quad y_2]^T$ of two sets of orthogonal lines, it can be shown that: $x_1x_2 + y_1y_2 + f^2 = 0$ (Guillou, Meneveaux, Maisel & Bouatouch, 2000); 4) If the camera moves, the motion of the vanishing points in the image plane depends only on the camera rotation, not on the camera translation. The vanishing points of three non-co-planar sets of parallel lines fully determine the rotation matrix (Guillou et al., 2000); and 5) Given the vanishing points $[x_1 \quad y_1]^T$, $[x_2 \quad y_2]^T$, $[x_3 \quad y_3]^T$ of three sets of mutually orthogonal lines, from equation 5 it can be verified immediately that (Cipolla, Drummond, & Robertson, 1999):

$$\begin{bmatrix} \lambda_1 x_1 & \lambda_2 x_2 & \lambda_3 x_3 \\ \lambda_1 y_1 & \lambda_2 y_2 & \lambda_3 y_3 \\ \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix} = \tilde{\mathbf{P}} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} = \tilde{\mathbf{K}} \left(\mathbf{R}_c^w \right)^T ,$$

where λ_1, λ_2, and λ_3 are unknown scaling factors.

Special shape calibration objects

Under linear perspective projection, the image of a circle is a sphere, whose major axis is on a line passing through the principal point. The eccentricity of the ellipse is a function of the focal length *f*, the distance of the center of the ellipse from the principal point, and the length of the major axis. The principal point can thus be located by intersecting major axes of several spheres in the image space. The focal length is then calculated from the eccentricity. Subsequently, other intrinsic and extrinsic parameters can be recovered. Most recently, concentric circles were employed to get the correct projected circle center (Kim & Kweon, 2001).

Taking the camera distortion into consideration, the sphere is curved. Assuming only k_1 not being zero, the curved sphere is a fourth order polynomial, from which the aspect ratio s_y / s_x can be computed directly. Besides circles, parallelepipeds have also been used (Wilczkowiak, Boyer & Sturm, 2001).

Other techniques

In addition to traditional optimization methods, some other techniques, including the Bayesian probability framework, genetic algorithms, and artificial neural nets, have also been considered in the camera calibration system (Redert, 2000). In the area of structure from motion, n ($n > 1$) cameras are calibrated one after another based on an extension of the Kalman filter (Jebara, Azarbayejani & Pentland, 1999).

Other Calibration-Related Investigations

Besides algorithms for calibrating cameras, other related aspects have been investigated to refine the performance of the camera calibration system. For example, several suggestions on the design of the calibration object and the data acquisition for increasing the calibration performance were given in Tu & Dubuisson (1992).

Feature extraction issue

For passive camera calibration, the calibration object should be carefully designed to ease the extraction of feature points. Several types of calibration object patterns have been used, e.g., a circular pattern in Heikkilä (2000) and a checkerboard pattern in Bouguet (2002). Here sub-pixel feature extraction (Devernay, 1995) is always necessary.

In most cases, features are extracted before the camera parameters are calculated. If feature coordinates are not accurately located, the camera calibration results will not be accurate, either. Unfortunately, when there are distortions in the imaging process, feature extraction will always suffer from systematic errors (Heikkilä, 2000). To circumvent this problem, the calibration can be carried out directly, based on the intermediate characterization of image features, such as maxima of the intensity gradient or zero crossings of the Laplacian. The same idea was also applied to distortion estimation in Ahmed & Farag (2001). Alternatively, the whole process, including the feature extraction and the camera calibration, can be carried out iteratively to yield better accuracy. Within each iteration, the feature extraction is performed after the distortion correction with the distortion coefficients obtained from the calibration (Devernay & Faugeras, 2000).

Performance analysis and evaluation

In Kumar & Hanson (1989), a mathematical analysis and experiments were carried out to develop a closed-form function to express the uncertainty of the calibration process. It was shown theoretically and experimentally in Lai (1993) that the offset of the image center does not significantly affect the determination of the position and orientation of a coordinate frame. An experimental performance analysis of a traditional calibration algorithm was conducted in Scott & Mohan (1995), where the result is evaluated with a geometric interpretation. Recently, the influence of noisy measurements on the camera calibration matrix $\tilde{\mathbf{P}}$ and on all linear camera calibration parameters in $\mathbf{p}_l$ has been analyzed theoretically and verified using Monte Carlo simulations (Kopparapu & Corke, 2001).

Accuracy evaluation is a crucial part in the development of new camera calibration algorithms. The 3-D reconstruction error and the image-plane projection error are probably the most popular evaluation criteria employed. However, due to the differences in image digitization and vision set-up, some normalization should be performed on the criteria to get comparable results for different calibration systems (Hartley & Zisserman, 2000).

Self-Calibration

Self-calibration is the process of determining camera parameters directly from uncalibrated video sequences containing a sufficient number of frames. These video sequences are generated from a single camera wandering in a still 3-D scene, or from multiple cameras at different poses imaging a still 3-D scene, or a single camera observing a moving object. All of these situations are equivalent to the case of "multiple cameras at different poses imaging a still 3-D scene."

If an arbitrary projective transformation is applied on all input data sets, the relative relations among them will not be altered. Thus, with sufficient point correspondence relations among available views, the camera parameters can only be determined up to a projective ambiguity (Hartley & Zisserman, 2000). However, in order to obtain the camera geometry, only a Euclidean ambiguity is allowed. This is why in the self-calibration process certain constraint or *a priori* information has to be assumed to upgrade the projective reconstruction to a Euclidean one (Faugeras, 1994). Generally speaking, the following three types of constraints or information can be employed:

1) Approximating the camera geometry by some simpler model, such as orthographic model (Lee & Huang, 1990), affine model (Tomasi & Kanade, 1991), or paraperspective model (Weinshall, 1993). A unification possibility was discussed in Quan & Triggs (2000);
2) *A priori* information about the possible scene type (Xu, Terai & Shum, 2000) or certain properties of the relative poses of available 3-D scene points (Boufama, Mohr & Veillon, 1993); or
3) Constraints on the variation of the camera parameters. The constraints employed can be represented by:
   - Restricted relative poses of all cameras, such as pure translation, pure rotation, or planar motion (Moons, Gool, Proesmans & Pauwels, 1996).
   - Intrinsic parameters that have been revealed by passive calibration in advance (Spet-sakis & Aloimonos, 1990) or by the first three views on-line (Horn, 1991).
   - The assumption that all (or most) cameras have the same but unknown intrinsic parameters (Triggs, 1998). In this case, Euclidean reconstruction is achievable.
   - Varying intrinsic camera parameters subject to certain restrictions. In Heyden & ÅströM (1999), some of the intrinsic parameters are allowed to vary while others are fixed. Pollefeys et al. (1999) suggest that one can neglect the skew parameter if one wants to vary all intrinsic parameters.

Various combinations of the above-mentioned *a priori* information or constraints have also been investigated for self-calibration. For self-calibration, the most frequently employed equation and concept are the Kruppa equation and the absolute conic, respectively (Hartley & Zisserman, 2000).

Face Model Reconstruction and Telepresence Applications

In this section, two applications that utilize the camera calibration results are presented and analyzed. First, we will discuss 3-D face model reconstruction and, second, a virtual telepresence application. For each application, the necessary techniques involved are introduced and typical outcomes are presented. In these two applications, general-purpose techniques are employed. However, they can be used directly for other applications, as well, such as body, face, and gesture modeling and animation.

3-D Face Model Reconstruction

This application attempts to compute the *facial action parameters* (**FAP**s) automatically from a pair of stereo sequences of the human face. For this purpose, the face is marked with blue dots (see Figure 2).

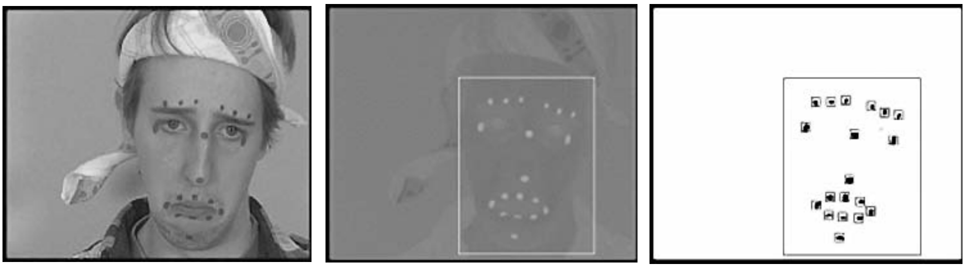
Manipulation process

The whole process includes five stages and can be briefly described as follows:

1) **Calibration**: The employed convergent stereo set-up can be calibrated using any method discussed in the last section. However, since the calibration results are only used for 3-D reconstruction, an **implicit** calibration together with the use of the reconstruction-distortion model would be sufficient. The distortion coefficients are calculated by the plumb-line method (Van Den Eelaart & Hendriks, 1999). After the distortion is removed, the linear DLT method is employed to recover all DLT parameters encoded in $\tilde{\mathbf{P}}$.

2) **Recording**: Without changing the pose or any internal parameters of the two cameras, a person with some special (shape, color, and position) markers on the face sits in front of the cameras, in the common view volume

Figure 2. One example face image. Left: The original image. Middle: The image after a hard thresholding. Right: Markers to be tracked through the sequence.



of both cameras. The person produces various kinds of expressions, such as opening of the mouth or squinting of the eyes. Two stereo sequences are then recorded.

3) **Marker detection and tracking**: Each sequence is processed by some marker detection algorithm (ref. Figure 2) and the markers (currently 19 per frame) are tracked throughout the sequence. Thus, two sequences of 19 moving points are generated.

4) **3-D reconstruction**: At this stage, a pair of stereo sequences of 19 moving points are available. As all parameters of the two utilized cameras are known, a sequence of 3-D coordinates of the 19 points can be reconstructed that reflect the motion of the 19 points in 3-D space.

5) **3-D face model analysis**: From the reconstructed 3-D sequence, FAPs can be calculated to analyze their correspondence relation with the human facial expression.

3-D reconstruction

With the reconstruction-distortion model, the 2-D coordinates of all tracked points can be easily converted to the undistorted coordinates by equation 12 immediately after step 3. On the other hand, if the imaging-distortion model is employed in the calibration software, a distortion correction (Lei & Hendriks, 2002) by simple backward mapping can be applied to all obtained images immediately after step 2. So 2-D distortion-free coordinates of all facial control points are available at step 4. Between them and their corresponding 3-D coordinates exists a linear relation that can be expressed by equation 17.

In terms of matrix notation, equations 19 and 20 can be rewritten as:

$$\begin{bmatrix} p_4^1 - p_4^3 x^{im} \\ p_4^2 - p_4^3 y^{im} \end{bmatrix} = \begin{bmatrix} -p_1^1 + p_1^3 x^{im} & -p_2^1 + p_2^3 x^{im} & -p_3^1 + p_3^3 x^{im} \\ -p_1^2 + p_1^3 y^{im} & -p_2^2 + p_2^3 y^{im} & -p_3^2 + p_3^3 y^{im} \end{bmatrix} \begin{bmatrix} x^w \\ y^w \\ z^w \end{bmatrix}. \quad (50)$$

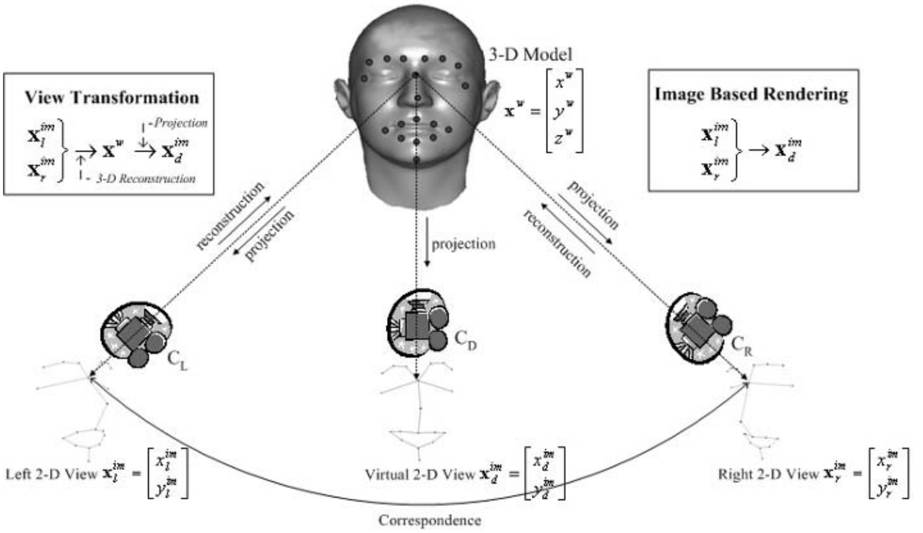
It is easy to see from equation 50 that no unique solution for $\mathbf{x}^w$ can be determined with one known $\mathbf{x}^{im}$ and the calculated p_j^i ($i = 1...3$; $j = 1...4$). That is why two $\mathbf{x}^{im}$ that correspond to the same $\mathbf{x}^w$ are needed to recover it uniquely. Suppose these two $\mathbf{x}^{im}$ are denoted as $\mathbf{x}_r^{im}$ and $\mathbf{x}_l^{im}$ respectively, then:

$$\underbrace{\begin{bmatrix} p_{r4}^{1} - p_{r4}^{3} x_{r}^{im} \\ p_{r4}^{2} - p_{r4}^{3} y_{r}^{im} \\ p_{l4}^{1} - p_{l4}^{3} x_{l}^{im} \\ p_{l4}^{2} - p_{l4}^{3} y_{l}^{im} \end{bmatrix}}_{\mathbf{b}} = \underbrace{\begin{bmatrix} -p_{r1}^{1} + p_{r1}^{3} x_{r}^{im} & -p_{r2}^{1} + p_{r2}^{3} x_{r}^{im} & -p_{r3}^{1} + p_{r3}^{3} x_{r}^{im} \\ -p_{r1}^{2} + p_{r1}^{3} y_{r}^{im} & -p_{r2}^{2} + p_{r2}^{3} y_{r}^{im} & -p_{r3}^{2} + p_{r3}^{3} y_{r}^{im} \\ -p_{l1}^{1} + p_{l1}^{3} x_{l}^{im} & -p_{l2}^{1} + p_{l2}^{3} x_{l}^{im} & -p_{l3}^{1} + p_{l3}^{3} x_{l}^{im} \\ -p_{l1}^{2} + p_{l1}^{3} y_{l}^{im} & -p_{l2}^{2} + p_{l2}^{3} y_{l}^{im} & -p_{l3}^{2} + p_{l3}^{3} y_{l}^{im} \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} x^{w} \\ y^{w} \\ z^{w} \end{bmatrix}}_{\mathbf{x}^{w}}. \quad (51)$$

From equation 51, $\mathbf{x}^{w}$ can be easily calculated using least squares as:

$$\mathbf{x}^{w} = \left(\mathbf{B}^{T}\mathbf{B}\right)^{-1}\left(\mathbf{B}^{T}\mathbf{b}\right). \quad (52)$$

Figure 3. The projection and 3-D reconstruction process of the 3-D face model. From two cameras C_L and C_R we get a pair of stereo images through projection. 3-D reconstruction is then applied on each pair of corresponding points $\mathbf{x}_r^{im}$ and $\mathbf{x}_l^{im}$ to obtain their original 3-D correspondence $\boldsymbol{x}^{w}$. The reconstructed 3-D model can then be projected into an arbitrary virtual camera C_D to form a virtual view. This is just the traditional view transformation process. Alternatively, the 3-D model reconstruction step could be neglected and the virtual view synthesized directed from the two stereo views. This image-based-rendering idea will be explored in the next section.


3-D Model
$\mathbf{x}^{w} = \begin{bmatrix} x^{w} \\ y^{w} \\ z^{w} \end{bmatrix}$
View Transformation
$\left.\begin{matrix} \mathbf{x}_l^{im} \\ \mathbf{x}_r^{im} \end{matrix}\right\} \rightarrow \mathbf{x}^{w} \rightarrow \mathbf{x}_d^{im}$
Projection
3-D Reconstruction
Image Based Rendering
$\left.\begin{matrix} \mathbf{x}_l^{im} \\ \mathbf{x}_r^{im} \end{matrix}\right\} \rightarrow \mathbf{x}_d^{im}$
reconstruction
projection
projection
projection
reconstruction
C_L
C_D
C_R
Left 2-D View $\mathbf{x}_l^{im} = \begin{bmatrix} x_l^{im} \\ y_l^{im} \end{bmatrix}$
Virtual 2-D View $\mathbf{x}_d^{im} = \begin{bmatrix} x_d^{im} \\ y_d^{im} \end{bmatrix}$
Right 2-D View $\mathbf{x}_r^{im} = \begin{bmatrix} x_r^{im} \\ y_r^{im} \end{bmatrix}$
Correspondence

This technique is known as the *pseudo-inverse solution* to the linear least-squares problem. Of course, it should be noted that this technique reduces the precision of the estimated coefficients with a factor of two because of the squared condition data. This flaw can be avoided by employing so-called *Least Squares with Orthogonal Polynomials* or *Weighted Least Squares*.

In fact, the requirement that two pixels that correspond to the same 3-D point are needed to reconstruct the 3-D coordinate of that point is consistent with our intuitive observation on the imaging process discussed in the first section, as two lines are needed to uniquely determine a 3-D point.

The above process is applied to each pair of corresponding points in every frame. Thus, a sequence of 3-D coordinates can finally be obtained.

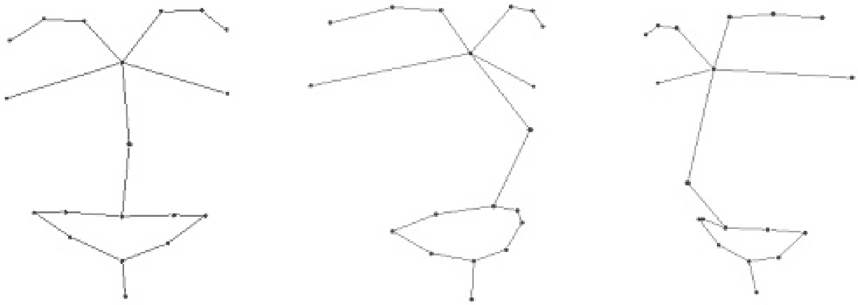
The projection and 3-D reconstruction process on the 3-D face model we used is described intuitively in Figure 3.

3-D face model examples

Since all tracked points are important facial control points, the reconstructed sequence of 3-D coordinates of those points reflects more or less facial actions. Thus, this sequence can be used as a coarse 3-D face model. If more control points are selected, a finer 3-D model can be obtained.

Based on the reconstructed model, FAPs can be calculated. To show the accuracy of the face model qualitatively, a VRML file is generated automatically. Figure 4 shows three example frames of such a VRML file.

Figure 4. Three example frames of the reconstructed 3-D face model. Blue points are control points that were reconstructed.



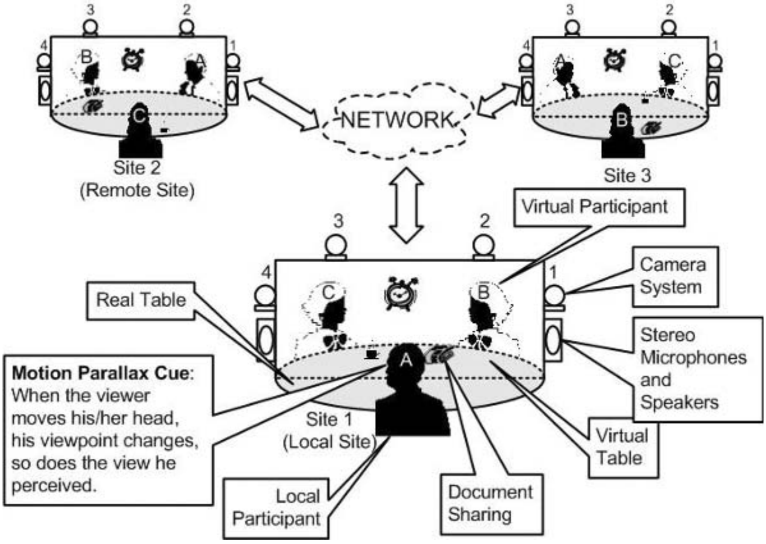
Telepresence

Camera information is also needed in a telepresence application. An IBR toolbox, which explicitly employs camera calibration results, has been developed for producing arbitrary virtual views of the human body, face, and hands, of which detailed 3-D models are difficult to acquire. In this section, a brief introduction is given to this IBR toolbox. Detailed discussions can be found in Lei & Hendriks (2002).

Set-up and processing framework

Figure 5 shows the infrastructure of a three-party telepresence system, in which participants are given a 3-D perception via a 2-D display that provides the motion parallax cue. For example, when participant *A* moves his head, he should be able to perceive different views of participant *B* and his (virtual) environment. The virtual environment can be easily built by current available 3-D graphics techniques. Since the 3-D modeling of a realistic human body and face is still quite difficult, if not impossible, the arbitrary virtual views of the participants are synthesized efficiently in the 2-D image space, ignoring the intermediate 3-D model (Lei & Hendriks, 2002). For instance, the virtual view of participant *C* can

Figure 5. Illustration of a three-party telepresence video-conferencing system.



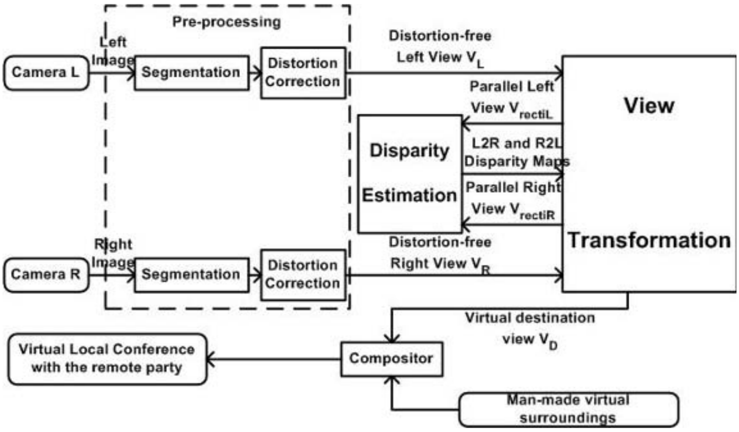
be reconstructed directly from the two views from cameras 1 and 2 of site 2 (remote site) at the local site according to the current viewpoint of participant *A*. This process will be specifically discussed below. Due to the symmetry in the system, the reconstruction for the other participants is similar.

Every 40ms, the fixed stereo set-up at the remote site acquires two images. After segmentation, the pair of stereo views, containing only the remote participant without background, is broadcast to the local site. Locally, the two views, based on the information about the stereo set-up, the local display, and the *pose* (position and orientation) of the local participant, are used to reconstruct a novel view ("telepresence") of the remote participant that is adapted to the current local viewpoint. The reconstructed novel view is then combined with a man-made uniform virtual environment to give the local participant the impression that he/she is in a local conference with the remote participant. The whole processing chain is shown in Figure 6.

Obviously, all parameters of each of the three four-camera set-ups should be computed beforehand. The calibration is done by combining the linear estimation technique and the Levenberg-Marquardt nonlinear optimization method.

With explicitly recovered camera parameters, the view can be transformed in a very flexible and intuitive way, discussed briefly in the next section.

Figure 6. The processing chain for adapting the synthesized view of one participant in line with the viewpoint change of another participant. Based on a pair of stereo sequences, the "virtually" perceived view should be reconstructed and integrated seamlessly with the man-made uniform environment in real time.


Pre-processing
Left Image
Camera L
Segmentation
Distortion Correction
Distortion-free Left View V_L
Parallel Left View V_{rectiL}
View
Disparity Estimation
L2R and R2L Disparity Maps
Parallel Right View V_{rectiR}
Transformation
Right Image
Camera R
Segmentation
Distortion Correction
Distortion-free Right View V_R
Virtual destination view V_D
Virtual Local Conference with the remote party
Compositor
Man-made virtual surroundings

View transformation

The objection of the view transformation is to reconstruct a virtual view V_D for a virtual camera C_D from a pair of stereo views V_L and V_R, which are generated from two cameras, C_L and C_R, respectively.

As a starting point for the following discussion, without loss of generality, the WCS can be selected such that:

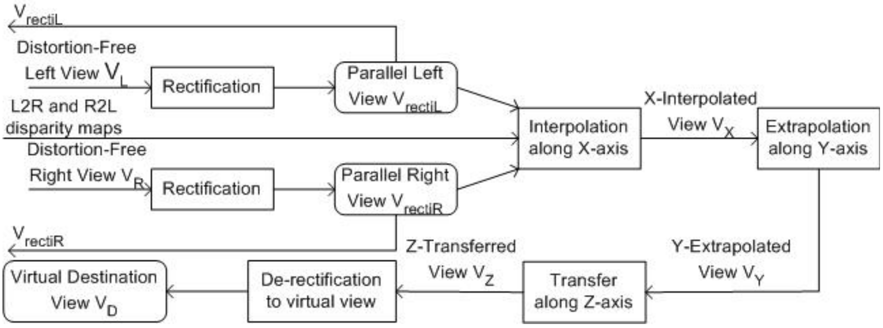
$$\mathbf{t}_{cL} = [1 \quad 0 \quad 0]^T,\ \mathbf{t}_{cR} = [-1 \quad 0 \quad 0]^T,\ \mathbf{t}_{cD} = [x_{cD} \quad y_{cD} \quad z_{cD}]^T,$$

where $\mathbf{t}_{cL}$, $\mathbf{t}_{cR}$, and $\mathbf{t}_{cD}$ are the position vectors of C_L, C_R, and C_D, respectively. This means that the x-axis of the WCS lies on the baseline **b** of C_L and C_R, and points from C_R to C_L. The origin of the WCS is at the middle point on **b**, that is, the unit of distance is $b / 2$.

In the general case, the view transformation process can be divided into five steps (see Figure 7):

1) **Rectification**: Transforming the stereo views V_L and V_R into a pair of new views V_{rectiL} and V_{rectiR}, respectively. The two virtual cameras C_{rectiL} and C_{rectiR}, which generate these two new views, are parallel to each other and

Figure 7. The view transformation framework. Multiple separate steps together eliminate three major differences between the final novel view V_D and the two original views V_L and V_R: 1) Photometric differences, such as focal length, aspect ratio, etc.; 2) Position in 3-D space (x, y, z); 3) Orientation.


V rectiL
Distortion-Free Left View V L
Rectification
Parallel Left View V rectiL
L2R and R2L disparity maps
Distortion-Free Right View V R
Rectification
Parallel Right View V rectiR
Interpolation along X-axis
X-Interpolated View V X
Extrapolation along Y-axis
V rectiR
Z-Transferred View V Z
Y-Extrapolated View V Y
Virtual Destination View V D
De-rectification to virtual view
Transfer along Z-axis

share the same image plane. This process is known as stereo rectification (Hartley, 1999) and is intended to eliminate the photometric and orientation differences between the two source cameras to simplify the correspondence estimation into a 1-D search problem along the scan line and at the same time to provide parallel processing possibilities for later steps.

2) **X-interpolation**: Given the disparity information, the two parallel views V_{rectiL} and V_{rectiR} are combined by interpolation or extrapolation (Seitz & Dyer, 1995) to produce another parallel view V_x. The corresponding camera C_x is located at $\begin{bmatrix} x_{cD} & 0 & 0 \end{bmatrix}$ with the same rotation and intrinsic parameters as C_{rectiL} and C_{rectiR}. The y coordinate of each pixel remains the same, while the x coordinate is transformed by

$$x_p^X = x_p^{rectiL} + \frac{1 - x_{cD}}{2} d_p^{LR}$$

and/or (in case of occlusion)

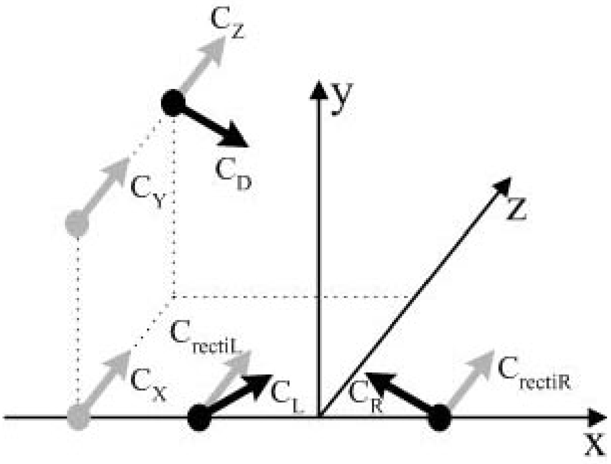
$$x_p^X = x_p^{rectiR} + \frac{1 + x_{cD}}{2} d_p^{RL},$$

where x_p is the x coordinate of pixel p^* in view V_* (* = X, $rectiL$, $rectiR$). p^{rectiL} and p^{rectiR} are projections of the same 3-D point. d_p^{LR} and d_p^{RL} are disparities of p^{rectiL} and p^{rectiR} respectively, where $d_p^{LR} = x_p^{rectiR} - x_p^{rectiL}$ and $d_p^{RL} = x_p^{rectiL} - x_p^{rectiR}$. Note that in the case of occlusion, either p^{rectiL} or p^{rectiR} is not available (Lei & Hendriks, 2002). Through this step, the difference in x position with the final view V_D is eliminated.

3) **Y-extrapolation**: The X-interpolated view V_x is extrapolated (Scharstein, 1999) by shifting pixels in the y direction to produce the view V_Y, which comes from a virtual camera C_Y located at $\begin{bmatrix} x_{cD} & y_{cD} & 0 \end{bmatrix}$ with the same rotation and intrinsic parameters as C_x. In this process, the x coordinate of each pixel remains the same while the y coordinate is transformed by $y_p^Y = y_p^X - y_{cD} \cdot \frac{s_x}{s_y} \cdot d_p^X$, where y_p^* is the y coordinate of pixel p^* in view V_* $(* = X, Y)$. d_p^X is the disparity of p^X, where $d_p^X = d_p^{LR} / 2$ or (in case of occlusion) $d_p^X = -d_p^{RL} / 2$. Through this step, the difference in y position with the final view V_D is eliminated.

4) **Z-transfer**: The Y-extrapolated view V_Y is transferred along the z direction to generate a closer or more distant look V_Z. The corresponding camera C_Z is located at $\begin{bmatrix} x_{cD} & y_{cD} & z_{cD} \end{bmatrix}$ with the same rotation and intrinsic parameters as C_Y. Both the x and y coordinates of each pixel would be transformed

Figure 8. An illustration of the possible camera configurations involved in the multi-step view synthesis process. The direction of each arrow indicates the orientation of the represented camera.



in a similar manner as the X-interpolation and Y-extrapolation. However, the dimension of the view would be maintained as the same (Lei & Hendriks, 2002). The z-position difference to the final view V_D is eliminated. It should be noted that, for different application situations, this Z-transfer step could be simplified or modified in different ways for better computational performance (Lei & Hendriks, 2002).

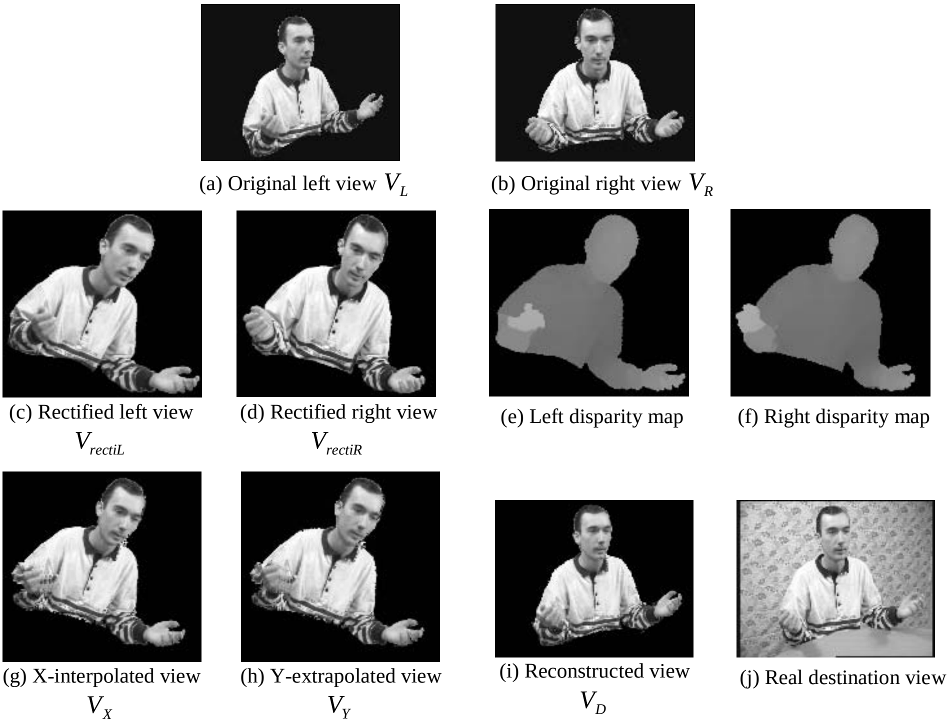
5) **De-rectification**: The Z-transferred view V_Z is rotated and scaled to get the final view V_D.

In Figure 8, an illustration is given of the possible camera configurations involved in the multi-step view synthesis process.

Results

A view transformation example is shown in Figure 9. Note that the Z-transfer step is combined together with the de-rectification step. It can be seen that the reconstructed novel view is comparable with the real view perceived at the virtual viewpoint. The overall visual quality is good.

Figure 9. All intermediate and final results from the view reconstruction are listed in this figure together with the disparity data.



(a) Original left view V_L

(b) Original right view V_R

(c) Rectified left view V_{rectiL}

(d) Rectified right view V_{rectiR}

(e) Left disparity map

(f) Right disparity map

(g) X-interpolated view V_X

(h) Y-extrapolated view V_Y

(i) Reconstructed view V_D

(j) Real destination view

Summary and Conclusions

Camera calibration is fundamental to many vision systems. In this chapter, we attempt to give the "big picture," which may possibly accelerate the application of camera calibration. Based on the pinhole principle, the imaging process of the camera is modeled in the first section. All camera parameters are introduced and their physical meaning is explained. For camera-based vision applications, all of these parameters should be revealed implicitly or explicitly off-line in advance of the application or on-line dynamically. Camera calibration techniques for this purpose are classified from several different points of view in the second section. Due to the importance of the non-linear distortion, this issue is specifically investigated in the third section. After this, passive camera calibration techniques are discussed and grouped into several categories of increasing complexity in the fourth section. Each of them has specific target applications and can provide highly accurate calibration results. Sometimes, however, accuracy is not as important as the flexibility of freely changing the camera configurations. In such cases, self-calibration is needed. A brief survey is, therefore, devoted to self-calibration techniques in the fifth section for completeness. To show how calibration results can be used in specific applications, two such practical and representative applications have also been presented in the sixth section.

As shown in the fourth section, passive camera calibration has been studied extensively during the past 40 years. Recently, due to the interest in image-based modeling (IBM) and IBR, research on self-calibration has intensified. Generally speaking, passive calibration and self-calibration were developed for different goals and circumstances. When the influence of the non-linear distortion component of the camera cannot be neglected or when highly accurate measurements are to be made based on the recovered camera geometry, passive calibration with deliberate modeling of the distortion is necessary. With a fixed camera set-up, as all parameters are recovered beforehand by passive calibration, a real-time vision system can be built (Lei & Hendriks, 2002). On the other hand, if the camera configuration is not fixed and the change is unpredictable, self-calibration is needed to get the values of all parameters whenever needed. However, due to the difficulty of robust feature extraction and correspondence estimation, self-calibration is carried out off-line and, thus, real-time processing cannot be guaranteed.

Within the passive camera calibration approach, different techniques can be applied for different applications. It may or may not be necessary to model distortion. If the accuracy offered by the linear model is acceptable to the problem at hand, distortion does not need to be considered for higher efficiency; otherwise, it has to be estimated and corrected. The complexity of modeling distortion also differs from application to application. This mainly depends on the required accuracy. In some cases, distortion can be estimated in advance of the calibration of other camera parameters, but, in others, it is necessary to estimate all camera parameters and distortion coefficients simultaneously to get highly accurate results. The former is more efficient and versatile while less accurate than the latter. The values of all camera parameters could be revealed explicitly or only certain intermediate expressions need be calculated. It depends on what the subsequent processing requires. From the above discussion, it is easy to see that which calibration technique is adopted for a specific application completely depends on the requirements of accuracy, the vision set-up, and the computation resources. Therefore, of course, compromises can be made between available calibration techniques and the application requirements. Two example vision applications requiring different calibration forms were already introduced in section 6. Because a high level of accuracy is required in both cases, distortion is modeled in both of them. However, because we want to later on utilize dynamic stereo set-up in the face model reconstruction application, which would be calibrated by self-calibration, we estimate the distortion in advance of the calibration of other camera parameters. On the other hand, because in the telepresence application the subsequent view reconstruction processing needs the value of each individual camera parameter, all parameters have to be calibrated explicitly.

Despite its wide range of applicability and extensive research, camera calibration is a subject for which there still remains some interesting and important issues to be investigated, some of which are:

Optimal calibration pattern: For passive camera calibration, a calibration target containing some control points with known geometry is needed. For highly accurate calibration, the projections of these control points in the image should be very efficiently and accurately located. Therefore, the pattern of the control points should be carefully designed. In the past, several kinds of patterns (e.g., circular dots and intersections of lines) have been adopted. However, it is still not clear which existing pattern is the best.

Clear understanding of camera parameter mutual relation: Very little attention has been paid to the analysis of the mutual dependencies among all camera parameters. Due to the existing coupling, all camera parameters should not be treated independently when one describes a 3-D scene. If their mutual dependency and relative importance is clear, the more important parameters can be treated "differently" during calibration. On the other hand, the analysis of the uncertainty of each camera parameter could also lead to better calibration algorithms.

Multiple camera configuration: The multiple camera configuration becomes more and more popular in vision-based applications (Pedersini, Sarti, & Tubaro, 1999), such as telepresence (Xu et al., 2002) and 3-D visual servo systems (Stavnitzky & Capson, 2000). However, the passive calibration problem of a multi-camera set-up has rarely been addressed. Self-calibration, on the other hand, always concentrates on multi-camera (or equivalent) configurations. Therefore, passive calibration may borrow certain techniques from self-calibration, especially for recovering the multi-camera set-up. Similarly, self-calibration can benefit from passive calibration on topics such as, the handling of nonlinear distortions.

References

Abdel-Aziz, Y. & Karara, H. (1971). Direct linear transformation into object space coordinates in close-range photogrammetry. In *Proc. symposium on close-range photogrammetry*, Urbana, IL, 1-18.

Ahmed, M. & Farag, A. (2001). Differential methods for nonmetric calibration of camera lens distortion. In *Proc. CVPR'2001*, Hawaii, 477-482.

Bani-Hashemi, A. (1991). Finding the aspect-ratio of an imaging system. In *Proc. CVPR'91*, Hawaii, 122-126.

Becker, S. & Bove, V. (1995). Semiautomatic 3-d model extraction from uncalibrated 2-d camera views. In *Proc. SPIE visual data exploration and analysis II*, San Jose, CA, 447-461.

Beyer, H. (1992). *Geometric and radiometric analysis of a ccd-camera based photogrammetric close-range system*. Unpublished doctoral dissertation, ETH-Zurich.

Boufama, B., Mohr, R. & Veillon, F. (1993). Euclidian constraints for uncalibrated recon- struction. In *Proc. ICCV'93*, Berlin, Germany, 466-470.

Bouguet, J. (2002). *Complete camera calibration toolbox for matlab*. Retrieved from the World Wide Web: http://www.vision.caltech.edu/bouguetj/calib doc/index.html.

Brand, P., Courtney, P., de Paoli, S. & Plancke, P. (1996). Performance evaluation of camera calibration for space applications. In *Proc. workshop on performance characteristics of vision algorithms*. Cambridge, MA.

Brown, D. (1971). Close-range camera calibration. *Photogrammetric Engineering, 37*(8), 855-866.

Caprile, B. & Torre, V. (1990). Using vanishing points for camera calibration. *International Journal of Computer Vision*, *4*, 127-140.

Chatterjee, C., Roychowdhury, V. & Chong, E. (1997). A nonlinear gauss-seidel algorithm for noncoplanar and coplanar camera calibration with convergence analysis. *Computer Vision and Image Understanding*, *67*(1), 58-80.

Chen, W. & Jiang, B. (1991). 3-d camera calibration using vanishing point concept. *Pattern Recognition*, *24*(1), 56-67.

Cipolla, R., Drummond, T. & Robertson, D. (1999). Camera calibration from vanishing points in images of architectural scenes. In *Proc. BMVC'99*, Nottingham, UK, 382-391.

Devernay, F. (1995). *A non-maxima suppression method for edge detection with sub-pixel accuracy* (Tech. Rep. No. RR2724). INRIA.

Devernay, F. & Faugeras, O. (2001). Straight lines have to be straight: Automatic calibration and removal of distortion from scenes of structured environments. *Machine Vision and Applications*, *13*(1), 14-24.

Farid, H. & Popescu, A. (2001). Blind removal of lens distortions. *Journal of the Optical Society of America*, *18*(9), 2072-2078.

Faugeras, O. (1993). *Three dimensional computer vision: A geometric view* (second ed.). Cambridge, MA: The MIT Press.

Faugeras, O. (1994). Stratification of 3-d vision: Projective, affine, and metric representations. *Journal of the Optical Society of America*, *12*(3), 465-484.

Faugeras, O. & Luong, Q. T. (2001). *The geometry of multiple images.* Cambridge, MA: MIT Press.

Faugeras, O., Quan, L. & Sturm, P. (2000). Self-calibration of 1d projective camera and its application to the self-calibration of a 2d projective camera. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, *22*(10), 1179-1185.

Fitzgibbon, A. (2001). Simultaneous linear estimation of multiple view geometry and lens distortion. In *Proc. CVPR'2001*, Kauai, Hawaii, I, 125-132.

Fryer, J., Clarke, T. & Chen, J. (1994). Lens distortion for simple 'c' mount lenses. *International Archives of Photogrammetry and Remote Sensing*, *30*(5), 97-101.

Fusiello, A. (2000). Uncalibrated euclidean reconstruction: A review. *Image and Vision Computing, 18,* 555-563.

Guillou, E., Meneveaux, D., Maisel, E. & Bouatouch, K. (2000). Using vanishing points for camera calibration and coarse 3d reconstruction from a single image. *The Visual Computer, 16*(7), 396-410.

Hartley, R. (1999). Theory and practice of projective rectification. *International Journal of Computer Vision*, *35*(2), 115-127.

Hartley, R. & Zisserman, A. (2000). *Multiple view geometry in computer vision.* Cambridge, MA: Cambridge University Press.

Hatze, H. (1988). High-precision three-dimensional photogrammetric calibration and object space reconstruction using a modified DLT-approach. *Journal of Biomechanics*, *21,* 533-38.

Heikkilä, J. (2000). Geometric camera calibration using circular control points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(10), 1066-1077.

Heyden, A. & ÅströM, K. (1999). Flexible calibration: Minimal cases for auto-calibration. In *Proc. ICCV'99.* Kerkyra, Corfu, Greece.

Horn, B. (1987). Closed form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, *4*(4), 629-642.

Horn, B. (1991). Relative orientation revisited. *Journal of the Optical Society of America*, *8*(10), 1630-1638.

Horn, B., Hilden, H. & Negahdaripour, S. (1988). Closed form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America*, *5*(7), 1127-1135.

Jebara, T., Azarbayejani, A. & Pentland, A. (1999). 3d structure from 2d motion. *IEEE Signal Processing Magazine*, *16*(3), 66-84.

Kang, S. (2000). Radial distortion snakes. In *Proc. IAPR workshop on machine vision applications (MVA2000).* Tokyo, Japan.

Kim, J. & Kweon, I. (2001). A new camera calibration method for robotic applications. In *Proc. IEEE/RSJ international conference on intelligent robots and systems,* Hawaii, 778-783.

Kopparapu, S. & Corke, P. (2001). The effect of noise on camera calibration parameters. *Graphical Models*, *63*(5), 277-303.

Kumar, R. & Hanson, A. (1989). Robust estimation of camera location and orientation from noisy data having outliers. In *Proc. workshop on interpretation of 3d scenes,* Austin, TX, 52-60.

Lai, J. (1993). On the sensitivity of camera calibration. *Image and Vision Computing*, *11*(10), 656-664.

Lamiroy, B., Puget, C. & Horaud, R. (2000). What metric stereo can do for visual serving. In *Proc. the IEEE/RSJ international conference on intelligent robots and systems,* Takamatsu, Japan, 251-256.

Lee, C. & Huang, T. (1990). Finding point correspondences and determining motion of a rigid object from two weak perspective views. *Computer Vision, Graphics, and Image Processing*, *52*(3), 309- 327.

Lei, B. & Hendriks, E. (2001). Middle view stereo representation - an efficient architecture for teleconference with handling occlusions. In *Proc. ICIP'2001,* Greece, 915-918.

Lei, B. & Hendriks, E. (2002). Real-time multi-step view reconstruction for a virtual teleconference system. *EURASIP Journal on Applied Signal Processing*, *2002*(10), 1067-1088.

Lenz, R. & Tsai, R. (1988). Techniques for calibration of the scale factor and image center for high accuracy 3-d machine. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *10*(5), 713-720.

Liebowitz, D. & Zisserman, A. (1998). Metric rectification for perspective images of planes. In *Proc. CVPR'98,* Santa Barbara, CA, 481-488.

Liu, Z., Zhang, Z., Jacobs, C. & Cohen, M. (2001). Rapid modeling of animated faces from video. *Journal of Visualization and Computer Animation*, *12*(4), 227-240.

Malm, H. & Heyden, A. (2001). Stereo head calibration from a planar object. In *Proc. CVPR'2001.* Kauai, Hawaii.

Moons, T., Gool, L., Proesmans, M. & Pauwels, E. (1996). Affine reconstruction from perspective image pairs with a relative object-camera translation in between. *Pattern Analysis and Machine Intelligence*, *18*(1), 77-83.

Pedersini, F., Sarti, A. & Tubaro, S. (1999). Multi-camera systems: Calibration and applications. *IEEE Signal Processing Magazine*, *16*(3), 55-65.

Penna, M. (1991). Camera calibration: A quick and easy way to determine the scale factor. *IEEE Transaction PAMI*, *12*, 1240-1245.

Perš, J., & Kovačič , S. (2002). Nonparametric, model-based radial lens distortion correction using tilted camera assumption. In H. Wildenauer & W. Kropatsch (Eds.), *Proc. the computer vision winter workshop 2002*, Bad Aussee, Austria, 286-295.

Pollefeys, M., Koch, R. & Gool, L. (1999). Self-calibration and metric reconstruction in spite of varying and unknown intrinsic camera. *International Journal of Computer Vision*, *32*(1), 7-25.

Press, W., Teukolsky, S., Vetterling, W. & Flannery, B. (1992). *Numerical recipes in c* (Second ed.). Cambridge, MA:Cambridge University Press.

Quan, L. & Triggs, B. (2000). A unification of autocalibration methods. In *Proc. ACCV'2000.* Taipei, Taiwan.

Redert, P. (2000). *Multi-viewpoint systems for 3-d visual communication.* Unpublished doctoral dissertation, Delft University of Technology.

Scharstein, D. (1999). *View synthesis using stereo vision* (Vol. 1583). Springer Verlag.

Scott, T. & Mohan, T. (1995). Residual uncertainty in three-dimensional reconstruction using two-planes calibration and stereo methods. *Pattern Recognition*, *28*(7), 1073-1082.

Seitz, S. & Dyer, C. (1995). Physically-valid view synthesis by image interpolation. In *Proc. workshop on representation of visual scenes*, MIT, Cambridge, MA, 18-25.

Sid-Ahmed, M. & Boraie, M. (1990). Dual camera calibration for 3-d machine vision metrology. *IEEE Transactions on Instrumentation and Measurement*, *39*(3), 512-516.

Slama, C. (1980). *Manual of photogrammetry* (Fourth ed.). Falls Church, VA: American Society of Photogrammetry and Remote Sensing.

Spetsakis, M. & Aloimonos, J. (1990). Structure from motion using line correspondences. *International Journal of Computer Vision*, *4*, 171-183.

Stavnitzky, J. & Capson, D. (2000). Multiple camera model-based 3-d visual servo. *IEEE Transactions on Robotics and Automation*, *16*(6), 732-739.

Stein, G. (1997). Lens distortion calibration using point correspondences. In *Proc. CVPR'97*, San Juan, PR, 602-609.

Stolfi, J. (1991). *Oriented projective geometry.* San Diego, CA: Academic Press.

Sturm, P. (2000). Algorithms for plane-based pose estimation. In *Proc. CVPR'2000,* Hilton Head Island, SC, 706-711.

Swaminathan, R. & Nayer, S. (2000). Nonmetric calibration of wide-angle lenses and poly-cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(10), 1172-1178.

Tamaki, T., Yamamura, T. & Ohnishi, N. (2002). Correcting distortion of image by image registration with the implicit function theorem. *International Journal on Image and Graphics*, 309-330.

Tomasi, C. & Kanade, T. (1991). Factoring image sequences into shape and motion. In *Proc. IEEE workshop on visual motion,* Princeton, NJ, 21-29.

Torr, P. & Zisserman, A. (1996). Robust parameterization and computation of the trifocal tensor. In R. Fisher & E. Trucco (Eds.), *Proc. BMVC'96,* Edinburgh: BMVA, 655-664.

Triggs, B. (1998). Autocalibration from planar scenes. In *Proc. ECCV'98,* University of Freiburg, Germany.

Triggs, B., McLauchlan, P., Hartley, R. & Fitzgibbon, A. (1999). Bundle adjustment – a modern synthesis. In *Proc. vision algorithms: Theory and practice,* Corfu, Greece, 298-372.

Tsai, R. (1987). A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Transactions on Robotics and Automation*, *3*(4), 323-344.

Tu, X. & Dubuisson, B. (1992). CCD camera model and its physical characteristics consideration in calibration task for robotics. In *Proc. IEEE international workshop on robot and human communication,* Tsukuba, Japan, 75-77.

Van Den Eelaart, I. & Hendriks, E. (1999). A flexible camera calibration system that uses straight lines in a 3d scene to calculate the lens distortion. In *Proc. ASCI'99,* Heijen, The Netherlands, 443-448.

Wang, L. & Tsai, W. (1991). Camera calibration by vanishing lines for 3-d computer vision. *IEEE Transaction PAMI*, *13*(4), 370-376.

Wei, G. & Ma, S. (1994). Implicit and explicit camera calibration: Theory and experiments. *IEEE Transaction PAMI, 16*(5), 469-480.

Weinshall, D. (1993). Model-based invariants for 3d vision. *International Journal of Computer Vision*, *10*(1), 27-42.

Weng, J., Cohen, P. & Herniou, M. (1992). Camera calibration with distortion models and accuracy evaluation. *IEEE Transaction PAMI*, *14*(10), 965-980.

Wilczkowiak, M., Boyer, E. & Sturm, P. (2001). Camera calibration and 3d reconstruction from single images using parallelepipeds. In *Proc. ICCV'2001*, Vancouver, Canada, 142-148.

Willson, R. (1994). *Modeling and calibration of automated zoom lenses.* Unpublished doctoral dissertation, Department of Electrical and Computer Engineering, Carnegie Mellon University.

Wolberg, G. (1990). *Digital image warping.* Los Alamitos, CA: IEEE Computer Society Press.

Xu, G., Terai, J. & Shum, H. (2000). A linear algorithm for camera self-calibration, motion and structure recovery for multi-planar scenes from two perspective images. In *Proc. CVPR'2000*, Hilton Head Island, SC, 474-479.

Xu, L., Lei, B. & Hendriks, E. (2002). Computer vision for 3d visualization and telepresence collaborative working environment. *BT Technical Journal*, 64-74.

Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transaction PAMI*, *22*(11), 1330-1334.

Zhuang, H. & Wu, W. (1996). Camera calibration with a near-parallel (ill-conditioned) calibration board configuration. *IEEE Transactions on Robotics and Automation*, *12*, 918-921.

Chapter IV

Real-Time Analysis of Human Body Parts and Gesture-Activity Recognition in 3D

Burak Ozer
Princeton University, USA

Tiehan Lv
Princeton University, USA

Wayne Wolf
Princeton University, USA

Abstract

This chapter focuses on real-time processing techniques for the reconstruction of visual information from multiple views and its analysis for human detection and gesture and activity recognition. It presents a review of the main components of three-dimensional visual processing techniques and visual analysis of multiple cameras, i.e., projection of three-dimensional models onto two-dimensional images and three-dimensional visual reconstruction from multiple images. It discusses real-time aspects of these techniques and shows how these aspects affect the software and hardware architectures. Furthermore, the authors present their multiple-camera *system to investigate the relationship between the activity recognition algorithms and the architectures required to perform these tasks in real time. The chapter describes the proposed activity recognition method that consists of a distributed algorithm and a data fusion scheme for two and three-dimensional visual analysis, respectively. The authors analyze the available data independencies for this algorithm and discuss the potential architectures to exploit the parallelism resulting from these independencies.*

Introduction

Three-dimensional motion estimation has a wide range of applications, from video surveillance to virtual animation. Therefore, reconstruction of visual information from multiple cameras and its analysis has been a research area for many years in computer vision and computer graphics communities. Recent advances in camera and storage systems are main factors driving the increased popularity of multi-camera systems. Prices continue to drop on components, e.g., CMOS cameras, while manufacturers have added more features. Furthermore, the evolution of digital video, especially in digital video storage and retrieval systems, is another leading factor.

In this chapter, we focus on real-time processing of multiple views for practical applications, such as smart rooms and video surveillance systems. The increased importance of applications requiring fast, cheap, small and highly accurate smart cameras necessitates research efforts to provide efficient solutions to the problem of real-time detection of persons and classification of their activities. A great effort has been devoted to three-dimensional human modeling and motion estimation by using multi-camera systems in order to overcome the problems due to the occlusion and motion ambiguities related to projection into the image plane. However, introduced computational complexity is the main obstacle for many practical applications.

This chapter investigates the relationship between the activity recognition algorithms and the architectures required to perform these tasks in real time. We focus on the concepts of three-dimensional human detection and activity recognition for real-time video processing. As an example, we present our real-time human detection and activity recognition algorithm and our multi-camera, test bed architecture. We extend our previous 2D method for 3D applications and propose a new algorithm for generating a global 3D human model and activity classification.

Some application areas of the real-time system are:

- Surveillance
- Provide security in a campus, shopping mall, office complex, casino, etc.
- Detect people's movements, gestures and postures from a security checkpoint in an airport, parking garage, or other facility
- Traffic
- Monitor pedestrian activity in an uncontrolled and/or controlled crosswalk
- Smart Environments
- Entertainment

Different applications require different levels of modeling-related performance parameters, e.g., accuracy, speed and robustness, hence, different 3D techniques. First, we revise the main components of 3D techniques and give a brief overview of previous work on basic 3D algorithm steps, such as disparity map generation, reconstruction and rendering. Then, we review the state of the art of human detection/activity recognition methods while placing emphasis on multi-camera systems. Specifically, general stereo vision issues and 2D/3D human activity recognition issues are reviewed with respect to their real-time applicability. In Section "Real Time 3D Analysis," we present our multi-camera system developed for practical applications, such as video surveillance and human-computer interaction. A novel 3D method is proposed to increase accuracy by keeping the complexity level low enough to run real-time applications. The section "Architectures for 3D Video Processing" further investigates the architectures required to perform these tasks in real-time. We conclude the chapter with a brief presentation of the major contributions of practical 3D methods as proposed in this chapter and discuss future directions.

3D Human Detection and Activity Recognition Techniques

Three-dimensional representation of the human body enables us to recover the general location and orientation of the human body parts, as well as three-dimensional activity of the body. The determination of three-dimensional information from two-dimensional digital images is a fundamental task. Traditional monocular and stereo vision methods have been widely used in computing 3D structure for a variety of applications, from robot navigation to visual inspection.

Basic Algorithm Steps

3D scene synthesis and analysis, by using visible light and multiple cameras, has been studied by many researchers. Before considering some of these methods, it is beneficial to review general stereo vision issues with respect to their real-time applicability. There are three basic problems, namely correspondence (disparity map), reconstruction, and rendering.

Disparity map generation

One well-known technique for obtaining depth information from digital images is the stereo technique. In stereo techniques, the objective is to solve the correspondence problem, i.e., to find the corresponding points in the left and right image. For each scene element in one image, a matching scene element in the other image is identified. The difference in the spatial position of the corresponding points, namely disparity, is stored in a disparity map. Whenever the corresponding points are determined, the depth can be computed by triangulation. Attempts to solve the correspondence problem have produced many variations, which can be grouped into matching pixels and matching features, e.g., edges. The former approach produces dense depth maps while the latter produces sparse depth maps. The specific approach desired depends on the objective of the application. In some applications, e.g., the reconstruction of complex surfaces, it is desirable to compute dense disparity maps defined for all pixels in the image. Unfortunately, most of the existing dense stereo techniques are very time consuming.

Even though stereo vision techniques are used in many image processing applications, the computational complexity of matching stereo images is still the main obstacle for practical applications. Therefore, computational fast stereo techniques are required for real-time applications. Given the algorithmic complexity of stereo vision techniques, general purpose computers are not fast enough to meet real-time requirements which necessitate the use of parallel algorithms and/or special hardware to achieve real-time execution.

Two main performance evaluation metrics are throughput, that is, frame rate times frame size, and range of disparity search that determines the dynamic range of distance measurement. There is still a great deal of research devoted to develop stereo systems to achieve the desired performance. The PRISM3 system (Nishihara, 1990), developed by Teleos, the JPL stereo implemented on DataCube (Matthies, 1992), CMU's warp-based multi-baseline stereo (Webb, 1993), and INRIA's system (Faugeras, 1993) are some of the early real-time stereo systems. Yet, they do not provide a complete video-rate output of range as dense as the input image with low latency. Another major problem is that the depth maps obtained by current stereo systems are not very accurate or reliable.

At Carnegie Mellon, a video rate stereo machine was developed (Kanade et al., 1996) where multiple images are obtained by multiple cameras to produce different baselines in lengths and in directions. The multi-baseline stereo method consists of three steps. The first step is the Laplacian of Gaussian (LOG) filtering of input images. This enhances the image features, as well as removes the effect of intensity variations among images due to the difference in camera gains, ambient light, etc. The second step is the computation of sum-of-squares differences (SSD) values for all stereo image pairs and the summation of the SSD values to produce the sum-of-sum-of-squares differences (SSSD) function. Image interpolation for sub-pixel re-sampling is required in this process. The third and final step is the identification and localization of the minimum of the SSSD function to determine the inverse depth. Uncertainty is evaluated by analyzing the curvature of the SSSD function at the minimum. All these measurements are done in one-tenth sub-pixel precision. One of the advantages of this multi-baseline stereo technique is that it is completely local in its computation without requiring any global optimization or comparison.

Schreer et al. (2001) developed a real-time disparity algorithm for immersive teleconferencing. It is a hybrid and pixel recursive disparity analysis approach, called hybrid recursive matching (HRM). The computational time is minimized by the efficient selection of a small number of candidate vectors, guaranteeing both spatial and temporal consistency of disparities. The authors use cameras, mounted around a wide screen, yielding a wide-baseline stereo geometry. The authors compare the real-time performance of their algorithm with a pyramid approach, based on multi-resolution images, and with a two stage hierarchical block-matching algorithm. The proposed method can achieve a processing speed of 40 msecs per frame for HRM algorithm in the case of sparse fields with block sizes of 8 by 8 pixels.

In Koschan & Rodehorst's (1995) work, parallel algorithms are proposed to obtain dense depth maps from color stereo images employing a block matching approach. The authors compare single processor and multiple processor performance to evaluate the profit of parallel realizations. The authors present computing times for block matching and edge-based stereo algorithms for multiple processing units that run in parallel on different hardware configurations.

A commercial system with small-baseline cameras has been developed by Videre Design. From two calibrated cameras, the system generates a disparity image in real-time by using area based stereo matching (Konolige, 1997). Their algorithm has four major blocks, namely LOG transform, variable disparity search, post-filtering, and interpolation. The special purpose hardware consists of two CMOS 320x240 grayscale imagers and lenses, low power A/D converters, a digital signal processor, and a small flash memory for program storage. The board communicates with the host PC via the parallel port. Second generation hardware uses a DSP from Texas Instruments (TMS320C60x).

Reconstruction and calibration

Reconstruction involves computing a point in space for each corresponding point pair in the images. This requires calibration of the cameras. There are two major parameter sets for cameras, namely intrinsic and extrinsic parameters. If both of the parameter sets are known, then the cameras are fully calibrated. By using the intrinsic parameters, the 3D depth map can be converted into (x,y,z) coordinates. The depth values give the z-coordinates and (x,y) coordinates are calculated from camera's intrinsic parameters. The extrinsic parameters are used to convert the camera centered (x,y,z) position into a world coordinates position (Narayanan et al., 1998; Kanade et al., 1997). These 3D points are converted into a surface representation via a triangular mesh. Since there is no exact solution, the algorithm calculates the correspondence that minimizes the geometric error subject to the epipolar constraint. In this chapter, for our experiments we assume that the cameras are fully calibrated. Detailed information about cameras and camera calibration can be found in Hartley & Zisserman's work (Hartley, 2000).

An exemplar application for scene reconstruction is Narayanan et al.'s (1998) work. The authors use 51 synchronized and calibrated video cameras to extract the depth map, polygonize it into triangles in 3D space, and apply texture maps onto the mesh. Another 3D scene reconstruction method is volumetric reconstruction. In this method, the reconstruction volume is divided into voxels where volumetric intersection algorithms reconstruct surface and voxels from the silhouette of an object (Cheung et al., 2000).

Pollefeys et al. (1999) developed a structure from the motion method to reconstruct a scene from uncalibrated cameras. Structure from motion was also used by Zisserman et al. (1999) for scene reconstruction. In their method, the authors locate corners in the images and estimate the fundamental matrix.

Although many algorithms are proposed for more accurate and reliable 3D object reconstruction, they are not suitable for practical applications due to their computational complexity. Depending on the application type, algorithm and hardware-related solutions are proposed. In Li et al. (2001), the authors reduce the complexity of finding spatio-temporal correspondence by using constraints from prior information. In Yang et al. (2002), the authors use a graphics hardware that effectively combines a plane-sweeping algorithm with view synthesis for real-time, 3D scene acquisition.

Rendering

Rendering is the process of producing realistic 3D images. The rendering issues are related to the interaction between light and surface, the intersection of viewing rays and objects sampling of the scene and displaying techniques. There are four main rendering methods used in visualization, i.e., ray tracing, volume rendering, radiosity and polygon rendering (Crockett, 1997). Due to the high computational requirements of traditional computer graphics, general purpose computers are not efficient in rendering applications. Consequently, special-purpose graphics engines are developed, primarily for polygon rendering. Similarly, special-purpose volume rendering architectures are developed to meet the special needs of volume rendering in order to compute rapidly and repeatedly from a volume dataset. To provide real-time volume rendering on standard computers, volume rendering is separated from general-purpose computing by using a dedicated accelerator. Another approach is to use volume visualization hardware that can be integrated with real-time acquisition devices.

3D reconstruction for image-based rendering is still an open research area. The visual hull concept is introduced to describe the maximal volume that reproduces the silhouettes of an object. In Matusik et al. (2000), an on-line, image-based approach is described to compute and shade visual hulls from silhouette image data. The maximal volume is constructed from all possible silhouettes. Computational complexity is reduced and a constant rendering cost per rendered pixel is achieved. In Matusik et al. (2001), new algorithms are proposed to render visual hulls in real-time. Unlike voxel or sampled approaches, an exact polyhedral representation is computed for the visual hull directly from the silhouettes.

Several other methods are proposed for real-time rendering. Volume carving is a common method used to convert silhouette contours into visual hulls by removing unoccupied regions from an explicit volumetric representation. Another method is Constructive Solid Geometry rendering. To avoid the complexity in computing the solid, ray tracing is used to render an object by defining a tree of CSG operations. Although an image-based rendering method yields higher realism, data acquisition and preprocessing requirements increase the complexity.

In Goldlücke et al. (2002), a method based on warping and blending images recorded from multiple synchronized video cameras is proposed to render dynamic scenes. Image quality is increased with the accuracy of the disparity maps provided with the recorded video streams. In Li et al. (2003), a simultaneous visual hull reconstruction and rendering algorithm is proposed by exploiting off-the-shelf graphics hardware.

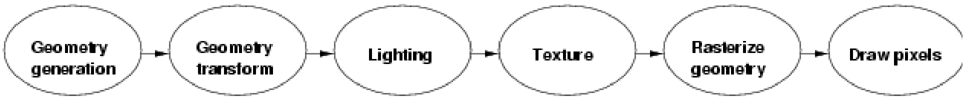
Beside special hardware, the use of parallel algorithms can't be avoided to achieve high-speed rendering applications. Early systems, such as Pixar's CHAP (Levinthal & Porter, 1984) and the commercially available Ikonas platform (England, 1986), had SIMD processors that could process vertex and pixel data in parallel. Programmable MIMD machines that could process triangles in parallel, such as the Pixel Planes (Fuchs et al., 1989) and the SGI InfiniteReality, had complex low-level custom microcodes and were rarely used. CPU vendors began to introduce graphics-oriented SIMD processor extensions into general purpose CPU designs. Examples of these extensions include Intel's MMX/SSE instructions, AMD's 3DNow architecture, and Motorola's AltiVec technology. Although such extensions accelerate several graphics operations, more sophisticated graphics coprocessors, e.g., processors that can support rendering pipelines, are needed. Such a system has been developed by Sony. The company designed a custom dual-processor SIMD architecture for graphics called the Emotion Engine (Kunimatsu et al., 2000).

A detailed survey on graphics hardware can be found in Thompson et al. (2002). The basic steps for image rendering are shown in Figure 1. The input of the graphics hardware is raw geometry data specified in some local coordinate system. The hardware transforms this geometry into world space and performs lighting and color calculations followed by a texture step. The hardware converts the vector-based geometry to a pixel-based raster representation, and the resulting pixels are sent into the screen buffer.

Human Detection and Activity Recognition

In this section, we present related work by classifying the research in terms of visual analysis of multiple cameras, i.e., projection of 3D models onto 2D images versus 3D visual reconstruction from stereo images. The former involves

Figure 1. Graphics pipeline.



extraction of correspondences between images from different views and projections of a 3D model while the later yields extraction of correspondences between 3D articulated models and reconstructed visual input. Gavrila (1999), Aggarwal & Cai (1999), and Moeslund & Granum (2001) presented overviews of various methods used for articulated and elastic non-rigid motion detection, human motion estimation, tracking, recognition, pose estimation and various other issues based on human detection and activity recognition. More background information on gesture recognition can be found in Wu & Huang (1999), Kohler & Schroter (1998) and LaViola (1999).

Luck et al. (2002) and Cheung et al. (2000) obtain 3D models of the moving human body by extracting the silhouettes from multiple cameras. Although our approach is similar from this point, the main algorithm used for the modeling is different.

In Luck et al. (2002), the authors use a physics-based approach for tracking 3D human models. The voxels obtained from the silhouettes exert attractive forces on a kinematic model of the human body to align the model with the voxels. Although this method enables very accurate modeling, it requires the human body model to be acquired from a specific initialization pose.

Our main aim is to use 3D info for our HMM-based activity recognition in real-time for different applications without requiring any specific pose or user interaction. Another major difference is architectural, as the authors use one PC where all the processing is done in a centralized way, while our architecture is distributed with local processors.

In Cheung et al. (2000), the authors use a similar approach to perform 3D voxel-based reconstruction by using silhouette images from multiple cameras. The local processing is used only for silhouette extraction. The five silhouette images are then sent to a host computer to perform 3D voxel-based reconstruction. The proposed algorithm first reconstructs 3D voxel data and then finds ellipsoids that model the human body. Our algorithm, on the other hand, first finds 2D ellipses that model the human body via graph matching at each local processor and then reconstructs 3D ellipsoids at a host computer. Note that 2D processing such as pose estimation is independent of the 3D modeling and activity recognition.

In Cohen & Lee (2002), the authors propose an approach for capturing 3D body motion and inferring human body posture from detected silhouettes. 3D body reconstruction is based on the integration of two or more silhouettes and the representation of body parts using generalized cylinders and a particle filter technique. Each silhouette is also used to identify human body postures by using support vector machine. In Kakadiaris & Metaxas (1998), a human body part identification strategy that recovers all the body parts of a moving human is employed by using the spatio temporal analysis of its deforming silhouette. 2D shape estimation is achieved by employing the supervisory control theory of discrete event systems. The 3D shape of the body parts is reconstructed by selectively integrating the apparent contours from three mutually orthogonal views.

Several methods are proposed for 3D motion recovery from monocular images. DiFranco et al. (2001) describe a method for computing the 3D motion of articulated models from 2D correspondences. The authors use kinematic constraints based on a 3D kinematic model, joint angle limits, dynamic smoothing and 3D key frames which can be specified by the user. In Sminchisescu & Triggs (2001), the authors propose a method for recovering 3D human body motion from monocular video sequences using robust image matching, joint and non-self-intersection constraints. To reduce correspondence ambiguities, the authors use a matching cost metric that combines robust optical flow, edge energy, and motion boundaries. In Howe et al. (2000), the authors present a 3D motion capture via a single camera. The method depends on prior knowledge about human motion to resolve the ambiguities of the 2D projection.

A geometric model is an approximation of the shape and of the deformations of the object. This model can be two-dimensional (modeling the contours of the projections of the object in the images), or three-dimensional (modeling the surfaces of the object). 2D shape models are generally made of curves, snakes, segments, sticks, etc., whereas 3D shape models are either systems of rigid bodies (spheres, superquadrics, etc.) or deformable surfaces (mesh). The articulations may be modeled by joints or by the motion of control points of B-splines. The choice between a 2D or a 3D model depends on the application, e.g., needed precision, number of cameras, and type of motion to be recognized.

2D

Several researchers work with 2D features to recognize human movement. Gavrila (1999), Goddard (1994) and Guo et al. (1994) use model-based recognition techniques, namely stick-figures, for this purpose. Other researchers who used 2D models are Papageorgiu & Poggio (1999), Comaniciu et al. (2000) and Isard & MacCormick (2001). Wachter & Nagel (1999) proposed a method to track the human body in monocular sequences. Their method depends on contour information and moving regions between frames.

Most of the work in this area is based on the segmentation of different body parts. Wren et al. (1999) proposed a system, Pfinder, to track people in 2D by using blobs that represent different body parts. The system uses a Maximum A Posteriori Probability (MAP) approach to detect and track people. The authors extend their work to obtain 3D estimates of the hands and head by applying two Pfinder algorithms (Wren et al., 2000). Pfinder uses blob features to detect a single moving person while our hierarchical and parallel graph matching and HMM-based activity recognition algorithms enable multi-person detection and activity recognition.

W4 is another real-time human tracking system (Haritaoglu et al., 1998) where the background information should be collected before the system can track foreground objects. The individual body parts are found using a cardboard model of a walking human as reference. There are a few works that aim to obtain a more compact representation of the human body without requiring segmentation. Oren et al. (1997) used wavelet coefficients to find pedestrians in the images, while Ozer & Wolf (2001) used DCT coefficients that are available in MPEG movies to detect people and recognize their posture.

Self-occlusion makes the 2D tracking problem hard for arbitrary movements and some of the systems assume *a priori* knowledge of the type of movement. The authors (Wolf et al., 2002) developed a system by using ellipses and a graph-matching algorithm to detect human body parts and classified the activity of the body parts via a Hidden Markov Model-based method. The proposed system can work in real-time and has a high correct classification rate. However, a lot of information has been lost during the 2D human body detection and activity classification. Generating a 3D model of the scene and of the object of interest by using multiple cameras can minimize the effects of occlusion, as well as help to cover a larger area of interest.

3D

One of the early works on tracking articulated objects is by O'Rourke & Badler (1980). The authors used a 3D model of a person made of overlapping spheres. They synthesized the model in images, analyzed the images, estimated the pose of the model and predicted the next pose. Hogg (1983) tracked human activity and studied periodic walking activity in monocular images. Rehg & Kanade (1995) built a 3D articulated model of a hand with truncated cones. The authors minimized the difference between each image and the appearance of the 3D model. Kakadiaris & Metaxas (1995; 1996) proposed a method to generate the 3D model of an articulated object from different views. The authors used an extended Kalman filter for motion prediction. Kuch & Huang (1995) modeled the hand with cubic B-splines and used a tracking technique based on minimization. Gavrila & Davis (1996) used superquadrics to model the human body. They used dynamic time warping to recognize human motion.

Munkelt et al. (1998) used markers and stereo to estimate the joints of a 3D articulated model. Deutscher et al. (1999) tracked the human arm by using a Kalman filter and the condensation algorithm and compared their performances.

Bregler & Malik (1998) proposed a new method for articulated visual motion tracking based on exponential maps and twist motions.

Most of the previous work for human detection depends highly on the segmentation results and mostly motion is used as the cue for segmentation. Most of the activity recognition techniques rely on successful feature extraction and proposed approaches are generally only suitable for a specific application type. The authors have developed a system that can detect a wide range of activities for different applications. For this reason, our scheme detects different body parts and their movement in order to combine them at a later stage that connects to high-level semantics.

Real-Time 3D Analysis

This section is devoted to our proposed method of real-time 3D human motion estimation. Multi-camera systems are used to overcome self-occlusion problems in the estimation of articulated human body motion. Since many movements become ambiguous when projected into the image plane and 2D information alone can not represent 3D constraints, we use multiple views to estimate 3D human motion. First, we discuss real-time aspects of 3D human detection and activity recognition. In the following two subsections we show how these aspects affect the software and hardware architectures. A detailed analysis of our 3D human detection/activity recognition algorithm and a testbed architecture for this particular algorithm are given in the last subsection.

Real-Time Aspects

Real-time aspects are critical for the success of the algorithm. The authors analyze various aspects and challenges of 3D human detection and activity recognition algorithms. These include: the instruction statistics, branch behavior, and memory access behavior of different program parts, e.g., stereo matching, disparity map generation, reconstruction, projection, 2D/3D human-body part detection, 2D/3D tracking, 2D/3D activity recognition, etc., in the Section "Algorithmic Issues." Note that it is essential to understand the application behavior to develop efficient hardware for a 3D camera system. Hardware related aspects and challenges for real-time applications are discussed in the Section "Hardware Issues." Decisions such as the number of processors in the system, the topology of the processors, cache parameters of each processor, the number of arithmetic logic units, ISA (instruction set architecture), etc., all rely on the characteristic of the application running in the system. For this purpose, we focus on a specific algorithm, our proposed 3D human detection/activity recognition system, and evaluate some extended aspects that are presented in this section.

Algorithmic issues

In the Section "3D Human Detection and Activity Recognition Techniques," we presented previous work on the basic steps of stereo vision algorithms and their real-time applicability for different applications. In general, we can divide 3D human detection and activity recognition methods into two categories (Cheung et al., 2000): off-line methods, where the algorithms focus on detailed model reconstruction (e.g., wire-frame generation), and real-time methods with global 3D human model reconstruction (Bregler & Malik, 1998; Delamarre & Faugeras, 2001).

The major challenge in many 3D applications is to compute dense range data at high frame rates, since participants cannot easily communicate if the processing cycle or network latencies are long. As an example of non-real-time methods, we can give Mulligan et al.'s (2001) work. In their work, to achieve the required speed and accuracy, Mulligan et al. use a matching algorithm based on the sum of modified, normalized cross-correlations, and sub-pixel disparity interpolation. To increase speed, they use Intel IPL functions in the pre-processing steps of background subtraction and image rectification, as well as a four-processor parallelization. The authors can only achieve a speed of 2-3 frames per second. Another non-real-time method (Kakadiaris & Metaxas, 1995) has been presented in the previous section.

Most of the real-time methods use a generic 3D human model and fit the projected model to the projected silhouette features. Another silhouette-based method is proposed by Cheung et al. (2000) and, recently, by Luck et al. (2002), where the human model is fit in real-time and in the 3D domain. The first method can reach a speed of 15 frames per second, whereas the second one runs at 20 frames per second. The speed of the systems highly depend on the voxel resolution. None of these methods tried to use 2D information obtained from each camera and combine the high-level information, e.g., head, torso, hand locations and activities, as well as the low-level information, e.g., ellipse parameters, to generate a global 3D model of the human body parts and recognize their activities in 3D. 2D information in terms of human image position and body labeling information is a very valuable component for higher level modules. In our system, it forms the basis for constructing the 3D body and activity model.

Our algorithmic pipeline clearly performs a wide range of disparate operations:

- pixel-by-pixel operations, such as color segmentation;
- pixel region operations, such as region identification;
- mixed operations, such as ellipse fitting; and
- non-pixel operations, such as graph matching and Hidden Markov Models.

We start with operations that are clearly signal-oriented and move steadily away from the signal representation until the data are very far removed from a traditional signal representation. In general, the volume of data goes down as image processing progresses.

Hardware issues

Real-time implementation of image/video processing algorithms necessitates data and instruction-level parallelism techniques to achieve the best performance for several application types. In this part, we will give an overview of some multimedia processing hardware and give a detailed description of our testbed architecture. Besides the algorithm development, hardware design is one of the most important issues for a real-time system. Watlington & Bove (1997) proposed a data-flow model for parallel media processing. Davis et al. (1999) developed a multi-perspective video system at the University of Maryland. Fritts et al. (1999) evaluated the characteristics of multimedia applications for media processors. Researchers also pay attention to multiprocessor architecture. Simultaneous multi-threading is proposed by Tullsen et al. (1995). Hammond et al. (1997) proposed single-chip multiprocessor architecture. An IMAGINE processor is being developed at Stanford University which has explicit programmable communication structure (Khailany et al., 2001).

Many different image/video-processor architectures exist with their own advantages and disadvantages. The selection of a processor must be based on a number of issues, including power, cost, development tools, and performance-related features. Texas Instruments has two DSPs (the TMS320C6201 and C6701), using a VLIW (Very Long Instruction Word) architecture, which means that they are able to select at compilation time instructions that can be executed in parallel, with a maximum of eight per clock cycle. The TMS320C80 has a MIMD (Multiple Instructions Multiple Data) architecture, and it can achieve the performances of the C6201, although its clock frequency is much slower.

Another VLIW processor, the Philips TriMedia processors, can execute up to five instructions per cycle. Besides the main CPU, there are other peripherals which can take the load from the main CPU for particular computations. Another advantage of this processor is the price, as well as the interfacing capabilities (PCI bus, serial link, video input/output, MPEG encoding/decoding, etc.) and the programming environment. In our application, we use TriMedia video capture boards, with TM1300 processors. A detailed description of the testbed will be given in the next subsection. Trimedia processors are media processors, which have wider data paths, wider registers, and more flexible memory interfaces than the regular DSPs. They can use data paths and register files to support SIMD (single instruction multiple data) types of operations, which is very useful when dealing with different real-time data inputs with varying dynamic range demands.

The Sharc ADSP 21160M is a SIMD processor that can be used for low-power, image/video-processing applications, but its performance is below the others. General purpose processors' (GPP) high power consumption and large size are the main disadvantages for portable image/video-processing applications. Another important factor is cost. For cost sensitive applications, DSP devices are significantly less expensive than GPPs. Code generation and debugging tools have a major impact on the efficiency of the development effort. Unlike GPP tools, some DSP development tools implicitly support the debugging of multiprocessor systems and provide unique data visualization tools.

In general, DSPs, unlike GPPs, are generally optimized for high throughput, data streaming applications. Some of the key features that support this include multiple bus architectures, multiple execution units that function in parallel, multiple data and instruction memories (both internal and external), multiple DMA channels for high speed data movement without processor involvement, special addressing modes, such as circular or bit reversed addressing, and specialized instructions to handle saturation and normalization. Interrupt handling on DSPs is efficient and uncomplicated. Finally, some DSP families support high-speed interprocessor communications which can directly link together multiple DSP devices without any intervening logic. The complexity of integrating external components with a processor is far higher for high performance GPPs than it is with low-end GPPs and DSPs.

Testbed

In this subsection, we give our testbed architecture where a single camera node is composed of a standard camera and a TriMedia video processing board. Designed for media processing, the TriMedia processing board allows Windows and Macintosh platforms to take advantage of the TriMedia processor via PCI interface. Multiple TriMedia processing boards can be installed to one host PC to provide multiprocessing ability. A TriMedia board has a TM1300 TriMedia processor with its own dedicated memory. A 32-bit TM1300 TriMedia processor has a five-issue VLIW (Very Long Instruction Word) CPU together with several coprocessors as shown in Figure 2. The CPU in the processor has multiple functional units and 128 registers. Table 1 shows the major features of a TriMedia CPU.

Besides its complicated hardware, the TriMedia board comes with a set of powerful software tools, which includes a tmsim simulator providing full func-

Figure 2. Structure of a TriMedia processor.

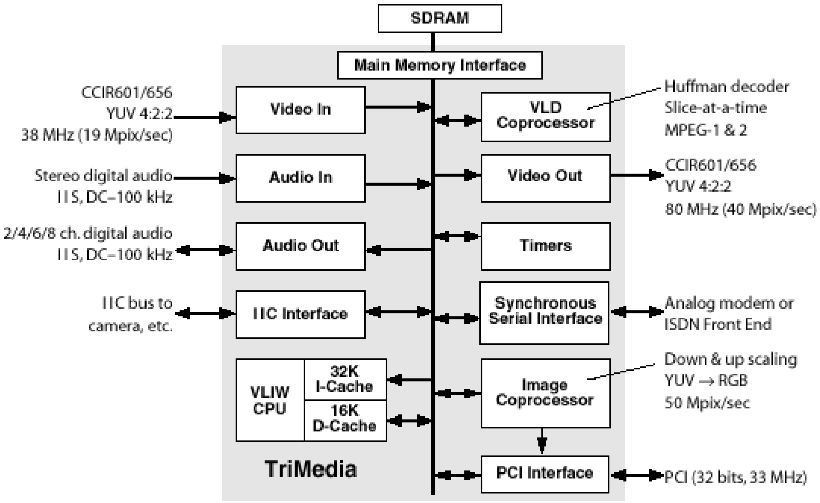


Table 1. TriMedia features.

Number of Functional Units	Constant	5
	Integer ALU	5
	Load/Store	2
	DSP ALU	2
	DSP MUL	2
	Shifter	2
	Branch	3
	Int/Float MUL	2
	Float ALU	2
	Float Compare	1
	Float sqrt/div	1
Number of Registers		128
Instruction Cache		32KB, 8 Way
Data Cache		16KB, 8 Way
Number of Operation Slots-Instruction		5

tional simulation. During the experiment, we use the TriMedia Software Develop Kit version tcs2.20 that includes a compiler tmcc, an assembler tmas, a linker tmld, a simulator tmsim, an execution tool tmrun, and a simulator tmprof. The TriMedia system is running on a Dell Precision-210 computer with two TriMedia reference boards. The TriMedia boards can communicate via shared memory, which enables fast data communication for stereo vision applications, e.g., disparity map generation.

Direct Algorithm for Human Gesture Recognition

In this subsection, we discuss in more detail an exemplar approach for human detection and activity recognition in the light of previously mentioned algorithms and real-time aspects. Most of the activity recognition systems are suitable for a specific application type. The presented example can detect a wide range of activities for different applications. For this reason, the scheme detects different body parts and their movement in order to combine them at a later stage that connects to high-level semantics. Each human body part has its own freedom of motion and the activity recognition for each part is achieved by using several Hidden Markov Models in parallel. Real-time performance of two and three-dimensional activity recognition techniques are compared for this particular example.

2D

A - Low-level Processing:

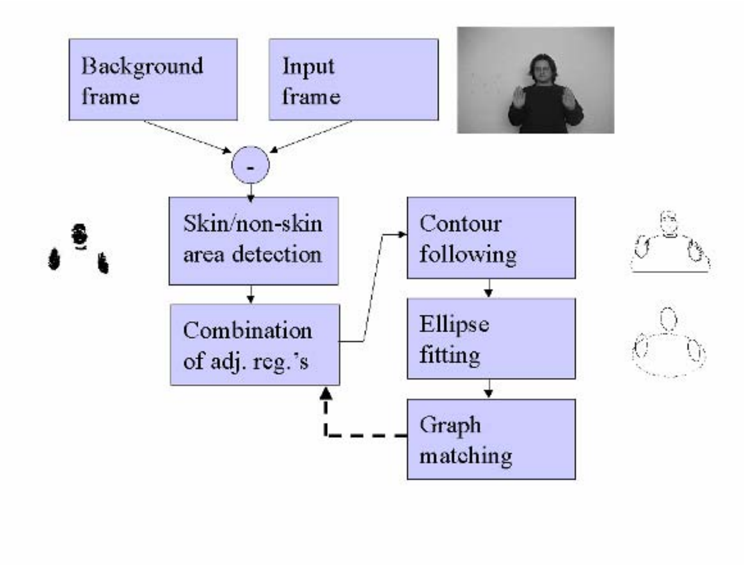
This section presents the proposed algorithm for the detection of the human body parts. The algorithm blocks are displayed in Figure 3. A more detailed explanation of our algorithm can be found in Ozer et al. (2000).

Background elimination and color transformation: The first step is the transformation of pixels into another color space regarding to the application. Background elimination is performed by using these transformed pixel values for the current and background images.

Skin area detection: Skin areas are detected by comparing color values to a human skin model. We use a YUV color model where chrominance values are down-sampled by two.

Segmentation of non-skin areas and connected component algorithm: The foreground regions that are adjacent to detected skin areas are extracted and corresponding connected components are found. We com-

Figure 3. Algorithm blocks and corresponding results of selected steps.



bine the meaningful adjacent segments and use them as the input of the following algorithm steps.

Contour following: We apply the contour following algorithm that uses the 3x3 filter to follow the edge of the component where the filter can move in any of eight directions to follow the edge.

Ellipse fitting: Even when human body is not occluded by another object, due to the possible positions of non-rigid parts, a body part can be occluded in different ways. For example, the hand can occlude some part of the torso or legs. In this case, 2D approximation of parts by fitting ellipses with shape-preserving deformations provides more satisfactory results. It also helps to discard the deformations due to the clothing. Global approximation methods give more satisfactory results for human detection purposes. Hence, instead of region pixels, parametric surface approximations are used to compute shape descriptors. Details of the ellipse fitting can be found in Ozer & Wolf (2002b).

Object modeling by invariant shape attributes: For object detection, it is necessary to select part attributes which are invariant to two-dimensional transformations and are maximally discriminating between objects. Geometric descriptors for simple object segments such as area, compactness (circularity), weak perspective invariants, and spatial relationships are computed (Ozer et al., 2000). These descriptors are classified into two groups: unary and binary attributes. The unary features for human bodies are: a) compactness, and b) eccentricity. The binary features are: a) ratio of areas, b) relative position and orientation, and c) adjacency information between nodes with overlapping boundaries or areas.

Graph matching: Each extracted region modeled with ellipses corresponds to a node in the graphical representation of the human body. Face detection allows the formation of initial branches to start efficiently and reduces their complexity. Each body part and meaningful combinations represent a class *w* where the combination of binary and unary features are represented by a feature vector *X* and computed off-line. Note that feature vector elements of a frame node computed online by using ellipse parameters change according to body part and the nodes of the branch under consideration. For example, for the first node of the branch, the feature vector consists of unary attributes. The feature vector of the following nodes also includes binary features dependent on the previously matched nodes in the branch. For the purpose of determining the class of these feature vectors, a piecewise quadratic Bayesian classifier with discriminate function *g(X)* is used. The generality of the reference model attributes allows the detection of different postures while the conditional rule generation *r* decreases the rate of false alarms. The computations needed for each node matching are then a function of the feature size and the previously matched nodes of the branch under consideration. The marked regions are tracked by using ellipse parameters for the consecutive frames and a graph matching algorithm is applied for new objects appearing in the other regions. Details of the graph matching algorithm can be found in Ozer & Wolf (2002b).

B - High-level Processing:

This section covers the proposed real-time activity recognition algorithm based on Hidden Markov Models (HMMs). HMM is a statistical modeling tool that helps to analyze time-varying signals. Online handwriting recognition (Sim & Kim, 1997), video classification and speech recognition (Rose, 1992) are some of the application areas of HMMs. Only a few researchers have used the HMM to recognize activities of the body parts. It is mainly used for hand gestures (Starner & Pentland, 1995). Parameterized HMM (Wilson & Bobick, 1999) can recognize complex events such as an interaction of two mobile objects, gestures made with two hands (e.g., so big, so small), etc. One of the drawbacks of the parameterized HMM is that for complex events (e.g., a combination of sub-events) parameter training space may become very large. In our application, we assume that each body part has its own freedom of motion and the activity recognition for each part is achieved by using several HMMs in parallel. Combining the outputs of the HMMs to generate scenarios is an application dependent issue. In our application environment, smart room, we use the Mahalanobis distance classifier for combining the activities of different body parts by assigning different weights for each activity. An HMM can be represented by using the notation $\lambda=(A,B,\pi)$ (Huang et al., 1990), where A, B, and π represent the transition, output, and initial probabilities, respectively. The movement of the body parts is described as a spatio-temporal sequence of feature vectors that consist of the direction of the body part movement. Since we use discrete HMMs, we generate eight directional code words. We check the up, down, right, left, and circular movements of the body parts. Our models are trained using the Baum-Welch algorithm. Note that the detected movement of the body part may be a part of a more complex activity. We check the current pattern and combine it with the immediately following one and generate a new pattern. Using dynamic programming, we calculate the probabilities for the first and combined patterns and choose the pattern with the highest probability as the recognized activity. If the probability of the observed activity is below a threshold, we reject the activity. Furthermore, we use the gap between different gestures/activities, e.g., moving the hand out of camera, stopping the body for a while. Another feature in the activity recognition is the speed of the body parts. We use the speed of each body part (slow/fast) for one activity period as an additional input for the classification. The next step is the generation of a feature vector by using the observed activities of the body parts. The activity feature vector is compared with the known activities via a distance classifier, based on the Mahalanobis metric. The output of the classifier detects the overall activity of the person. The proposed activity classification algorithm is given in Figure 4.

Figure 5 displays some of the activity patterns, namely waving one hand, opening and closing hands, and left-right movement of the body. It displays cumulative

Figure 4. Overview of the activity classification.

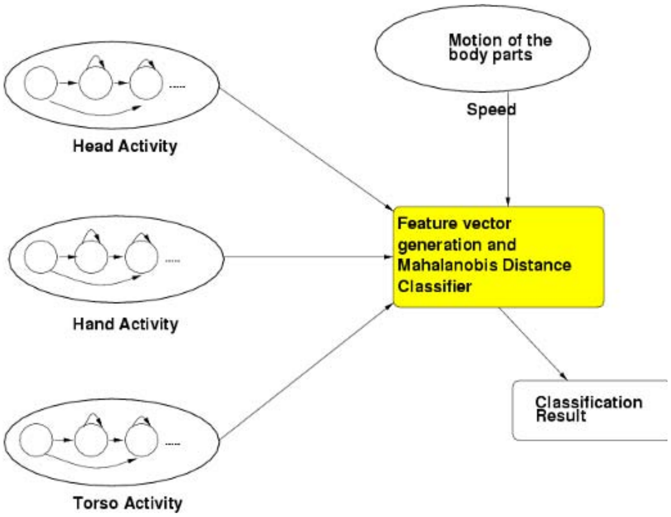
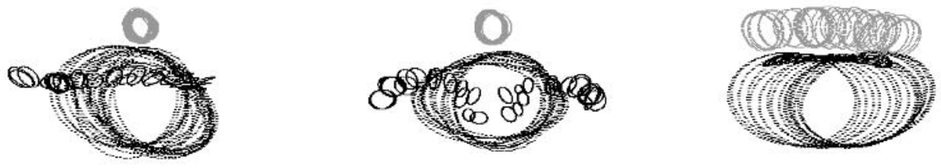


Figure 5. Cumulative motion of body parts for different activity patterns: Waving one hand, opening and closing arms, left-right movement.



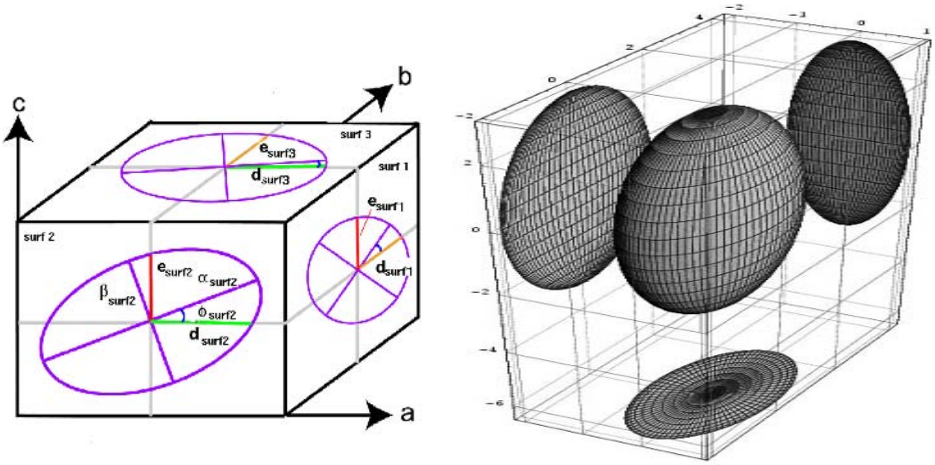
motion of the body parts. We observe that different activity patterns can have overlapping periods (same or similar patterns for a period) for some body parts. Hence, the detection of start and end times of activities is crucial. To detect the start and end time of a gesture, we use the gap between different gestures/ activities.

Eighty-six percent (86%) of the body parts in the processed frames and 90% of the activities are correctly classified, with the rest considered the miss and false classification. Details of the gesture recognition algorithm can be found in Ozer & Wolf (2002a).

From 2D to 3D

In this subsection, we present our algorithm that generates a real-time 3D model of the human body by combining 2D information from multiple cameras located at 90 degrees to each other. We propose a new 3D method for activity recognition in real-time. The proposed method that combines valuable 2D information is fast, robust and accurate. It doesn't require any disparity map and wire-frame information for model generation. We generate a global 3D ellipsoid model of the human body parts from 2D ellipses and use the resulting 3D information to verify the fit of the real body parts with the actual model. We can process approximately 25 frames per second on each TriMedia board. Figure 7 shows the architecture for 3D model generation. Camera calibration and data synchronization are main issues in data fusion from multiple cameras. Visual reconstruction for virtual reality requires high accuracy, while real-time activity recognition and trajectory estimation require high-speed techniques (Dockstader & Tekalp's, 2001; Focken & Stiefelhagens, 2002; Schardt & Yuan, 2002). Note that our system uses static cameras that do not require dynamic calibration.

Figure 6. Orthogonal ellipses and best-fit ellipsoid.



After graph matching, head, torso and hand ellipses and their corresponding attributes are sent from each processing board to the other one via the shared memory. High-level information (ellipses corresponding to head, torso, and hand areas) and low-level information (ellipse attributes) are used to model the best-fit ellipsoids for each body part as shown in Figure 8. The best-fit ellipsoid algorithm is based on Owens's (1984) work. Figure 6 displays the orthogonal ellipses, their attributes, and best-fit ellipsoid after iterative approximation.

The equation of an ellipse is given by:

$$\frac{x^2}{\alpha^2} + \frac{\mathrm{y}^2}{\beta^2} = 1$$

where α and β are the principal axes of the ellipsoid.

After rotation ϕ the ellipse equation becomes:

$$\frac{(x\ \cos(\phi) + y\ \sin(\phi))^2}{\alpha^2} + \frac{(-x\ \sin(\phi) + y\ \cos(\phi))^2}{\beta^2} = 1$$

After projection of α on the y- and β on the x-axis we get d and e, respectively:

$$d = \sqrt{\frac{\alpha^2}{\cos(\phi)^2} + \frac{(\alpha/\beta)^2}{\sin(\phi)^2}}$$

$$e = \sqrt{\frac{\alpha^2}{\sin(\phi)^2} + \frac{(\alpha/\beta)^2}{\cos(\phi)^2}}$$

For three perpendicular surfaces, namely surf1, surf2, and surf3, the diagonal components λ and off-diagonal components γ are calculated for the 2x2 matrices representing the sectional ellipses:

$$\lambda_{surf1} = 1/2\,(\alpha_{surf2}\,(\cos(\phi_{surf2}))^2 + \beta_{surf2}\,(\sin(\phi_{surf2}))^2) + 1/2(\alpha_{surf3}\,(\cos(\phi_{surf3}))^2 + \beta_{surf3}\,(\sin(\phi_{surf3}))^2)$$

$$\lambda_{surf2} = 1/2\,(\alpha_{surf1}\,(\cos(\phi_{surf1}))^2 + \beta_{surf1}\,(\sin(\phi_{surf1}))^2) + 1/2\,(\alpha_{surf3}\,(\cos(\phi_{surf3}))^2 + \beta_{surf3}\,(\sin(\phi_{surf3}))^2)$$

$$\lambda_{surf3} = 1/2\,(\alpha_{surf1}\,(\cos(\phi_{surf1}))^2 + \beta_{surf1}\,(\sin(\phi_{surf1}))^2) + 1/2\,(\alpha_{surf2}\,(\cos(\phi_{surf2}))^2 + \beta_{surf2}\,(\sin(\phi_{surf2}))^2)$$

$$\gamma_{surf1} = \alpha_{surf1}\,\sin(\phi_{surf1})\,\cos(\phi_{surf1}) - \beta_{surf1}\,\sin(\phi_{surf1})\,\cos(\phi_{surf1})$$

$$\gamma_{surf2} = \alpha_{surf2}\,\sin(\phi_{surf2})\,\cos(\phi_{surf2}) - \beta_{surf2}\,\sin(\phi_{surf2})\,\cos(\phi_{surf2})$$

$$\gamma_{surf3} = \alpha_{surf3}\,\sin(\phi_{surf3})\,\cos(\phi_{surf3}) - \beta_{surf3}\,\sin(\phi_{surf3})\,\cos(\phi_{surf3})$$

Note that the diagonal component λ is doubly defined. To get an initial estimate we average the two doubly defined terms. To get a better best-fit estimate we define a matrix P and calculate the normalized eigenvalues Π and eigenvectors V of the sectional ellipses by using singular value decomposition.

$$P = [(\lambda_{surf1}, \gamma_{surf3}, \gamma_{surf2})\,(\gamma_{surf3}, \lambda_{surf2}, \gamma_{surf1})\,(\gamma_{surf2}, \gamma_{surf1}, \lambda_{surf3})]$$

Sectional ellipses are represented by:

$$[(\lambda_{surf\,2},\gamma_{surf\,1})\ (\gamma_{surf\,1},\lambda_{surf\,3})],\ [(\lambda_{surf\,1},\gamma_{surf\,2})\ (\gamma_{surf\,2},\lambda_{surf\,3})] \text{ and } [(\lambda_{surf\,1},\gamma_{surf\,3})\ (\gamma_{surf\,3},\lambda_{surf\,2})]$$

To find the best-fit ellipsoid, a misfit function *G* is generated:

$$G = (\Pi_{surf\,1,surf\,2,surf\,3} - (\lambda_{surf\,1,surf\,2,surf\,3} / \beta_{surf\,1,surf\,2,surf\,3}))^2 + (\kappa_{surf\,1,surf\,2,surf\,3} - \phi_{surf\,1,surf\,2,surf\,3})^2$$

where κ is the arctangent obtained from major axis eigenvectors. *G* function is minimized with respect to λ's and γ's to find the best-fit ellipsoid matrix. From eigenvectors, eigenvalues and parametric unit sphere, the resulting ellipsoid is generated. Note that, in our application, we use body proportion information and spatial position of the body parts obtained from two calibrated cameras to predict the ellipses on the third projection surface. Figure 9 shows the ellipsoids fitted to torso and head-regions. The system recognizes the activity of the 3D human model by combining the sequential direction information obtained from both of the processors for each body part. Figure 10 and Figure 11 show the recognized activities, namely hand left-right and pointing to camera1. Note that a single

Figure 7. Architecture for 3D model generation.

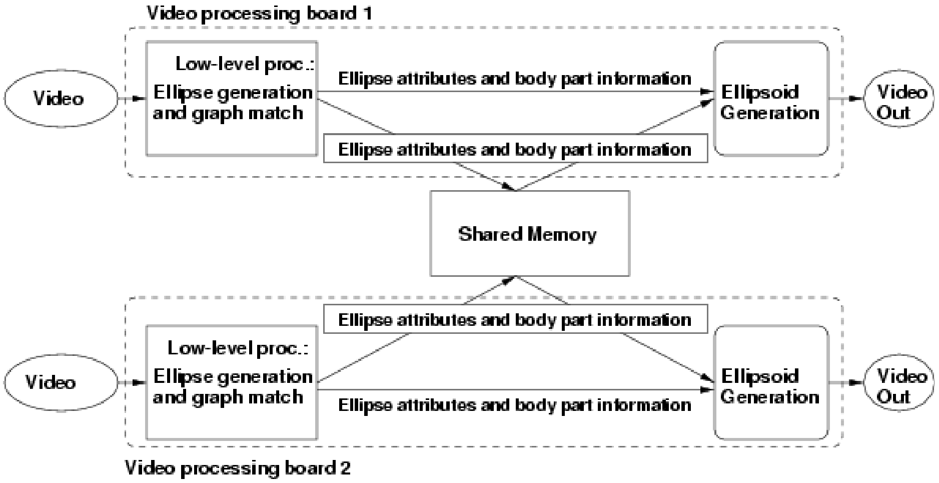

Video processing board 1
Video
Low-level proc.:
Ellipse generation and graph match
Ellipse attributes and body part information
Ellipse attributes and body part information
Ellipsoid Generation
Video Out
Shared Memory
Ellipse attributes and body part information
Video
Low-level proc.:
Ellipse generation and graph match
Ellipse attributes and body part information
Ellipsoid Generation
Video Out
Video processing board 2

Figure 8. Ellipsoid and its projection on the 2D planes. Outer boundaries of the projections represent the 2D ellipses.

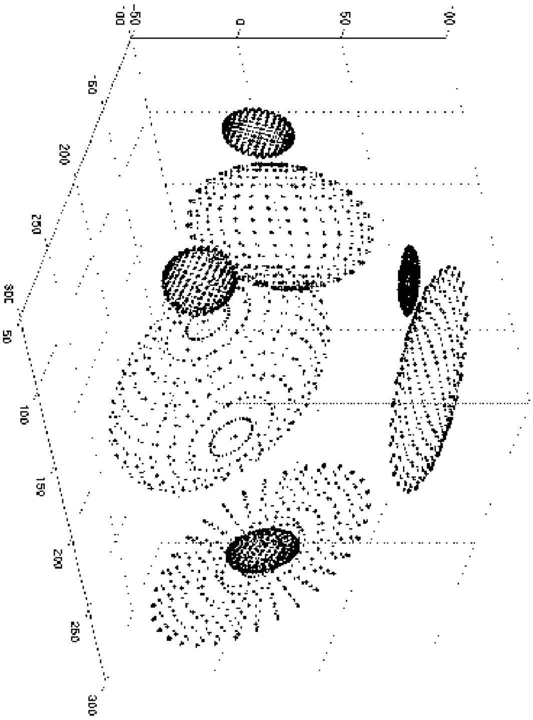


Figure 9. Example ellipsoids from front view (top) and side view (bottom)

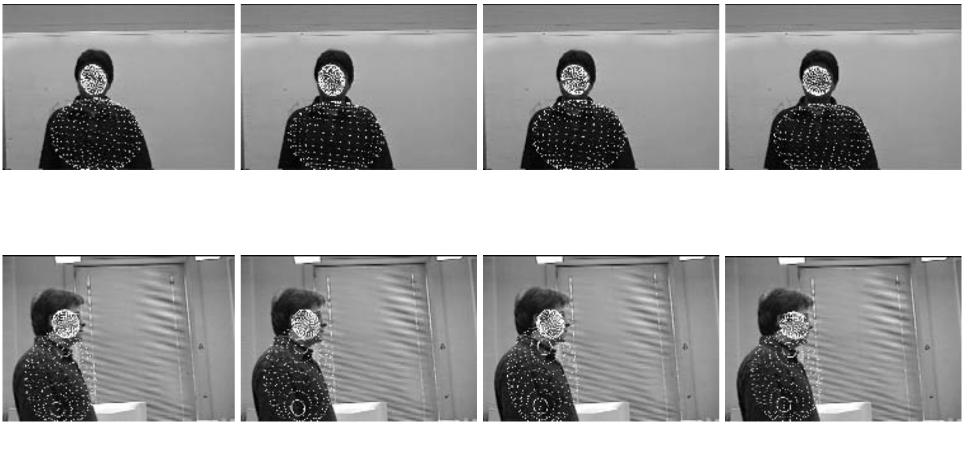


Figure 10. Example ellipsoids from front view: Hand left-right.

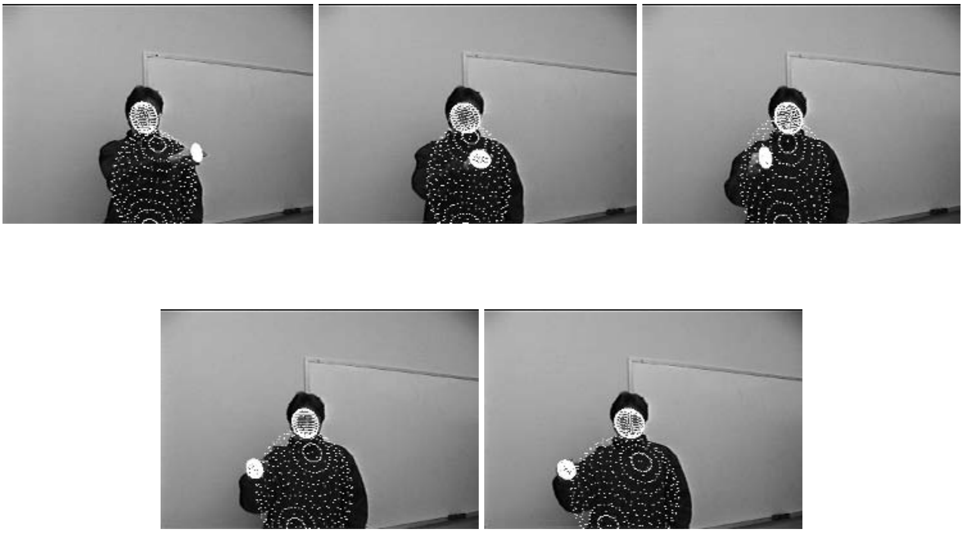
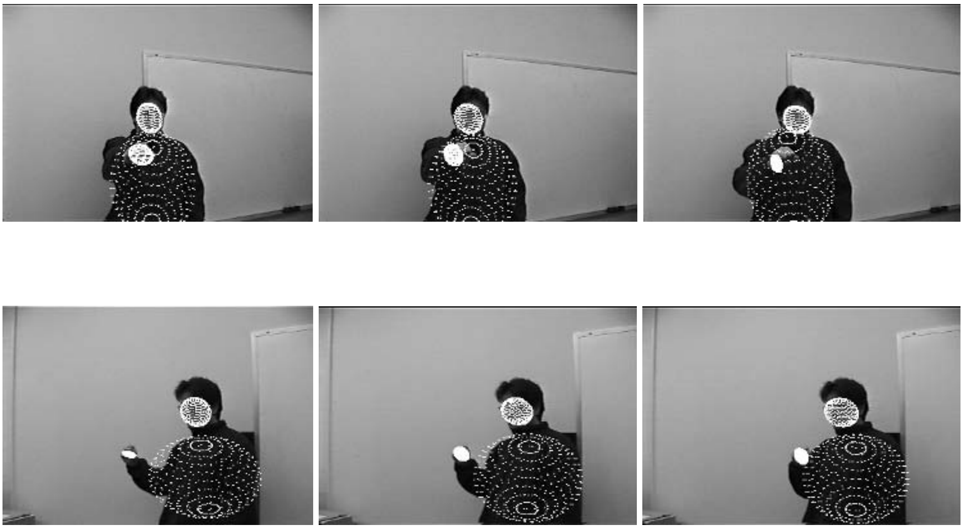


Figure 11. Example ellipsoids for "pointing camera1" activity from front and side view.



camera view cannot find the activities such as pointing towards the camera, e.g., area change with time is not reliable for small body parts. However, the proposed system combines the activity directions and body pose information from multiple views and recognizes the correct activity in real-time.

Figure 12. Recognized activity: Moving left.

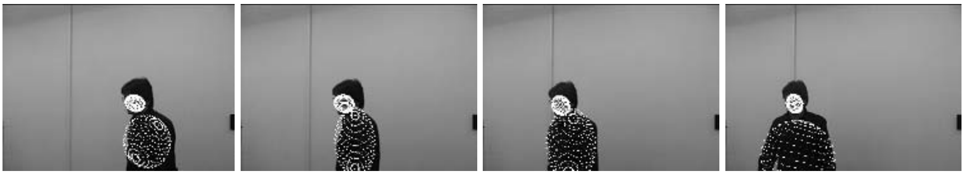


Figure 13. Moving down and moving up activities.

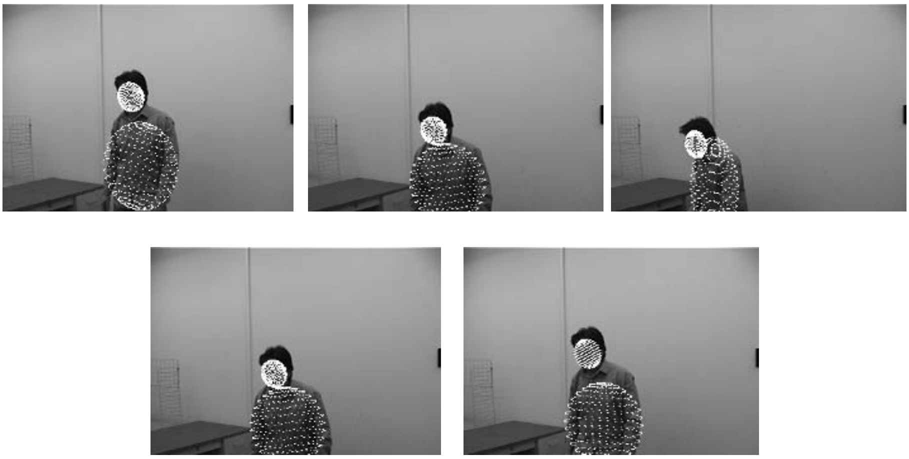
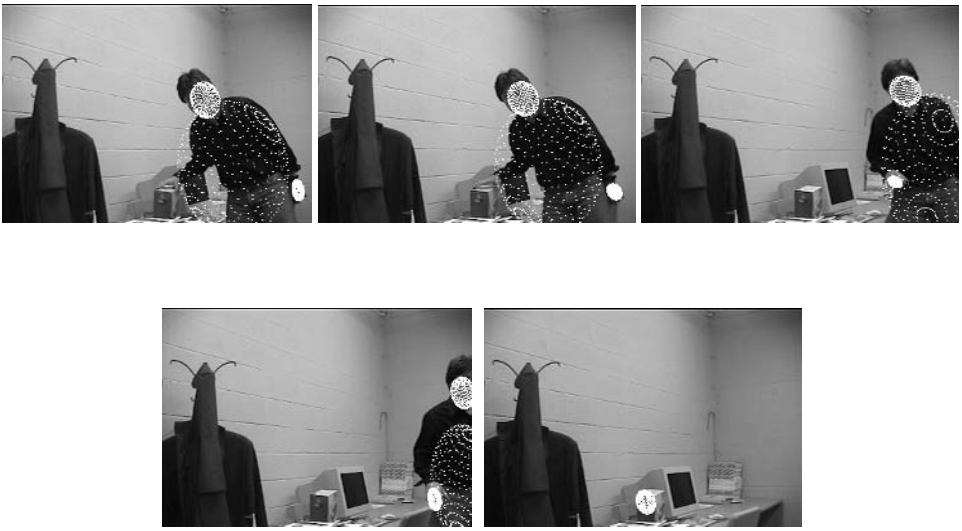


Figure 12 shows recognized moving left activity. Figure 13, first row, shows moving down activity where the person turns from the first camera to the second one during the activity period. The second row is another correctly recognized moving down/up activity for a different person.

Figure 14 is an example for unattended object detection. The person enters the scene with an object in his hand and leaves the object on the table. After a predefined time, the system can detect the body parts and the object left on the table correctly. An alarm can be generated and sent to the security personnel for the unattended object.

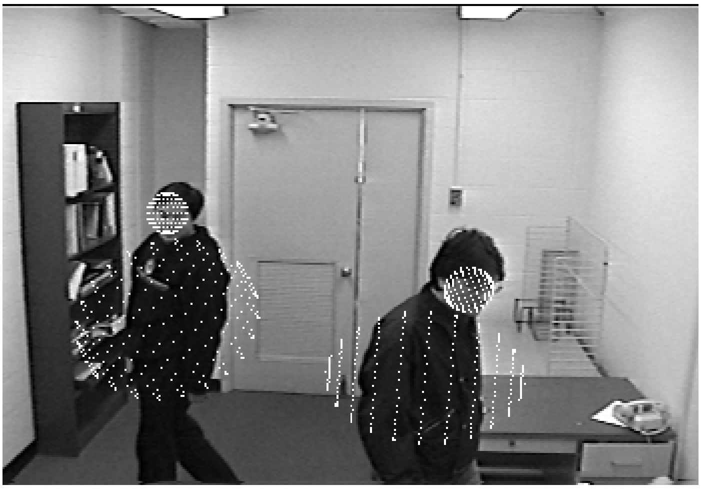
For applications such as smart rooms where devices are controlled by people, the deployment of cameras can be adjusted for optimum capture of the motion. However, for less structured motion estimation applications such as surveillance, self-occlusion may occur. In this case, the corresponding parts are modeled less accurately for the duration of the occlusion. One of our future works includes a feedback scheme that uses temporal dependency. Furthermore, more cameras

Figure 14: Unattended object detection.



would improve the performance by alleviating the self-occlusion problem. Another challenge is detecting multiple persons and recognizing their activities. Figure 15 displays an example frame from security cameras in a test room. Note that the algorithm can generate the 3D model of each body part, unless there is occlusion because of the other person or because of another body-part. As it is mentioned before, occlusion problems can be overcome by using multiple cameras (more than two) and using a feedback scheme.

Figure 15: Multiple persons.



Note that the most time-consuming parts of the algorithm are computed in parallel by different video processors and the integration to reconstruct the 3D model is based on the processing of parameters as opposed to pixel processing. This feature of the algorithm makes the integration of multiple cameras more attractive. A detailed description of multi-camera architecture is presented in the next section.

Architectures for 3D Video Processing

In this section, we present parallel architectures for a multi-camera system. We analyze the available data independencies for the previously mentioned 2D example, and discuss the potential architectures to exploit the parallelism that resulted from these independencies. Three architectures — VLIW, symmetric parallel architecture and macro-pipeline architectures are discussed. After this, we extend our discussion to 3D systems.

The following discussion from a hardware perspective can be applied to both standard hardware, such as PC platform, and to application specific hardware. For real-time video applications, the demand on computation capability can be a rather heavy burden on general processors, or even exceed their capability. As a result, real-time video applications usually need support from application hardware such as DSPs on video card, video capturing device, etc. For this reason, we focus our discussion primarily on application specific hardware, although part of our conclusion can be extended to standard computer systems.

Instruction Level Parallelism and VLIW Architecture

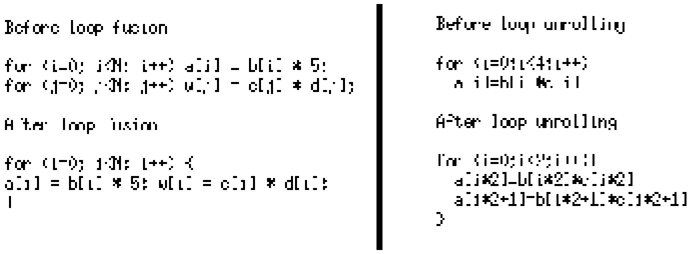
In pixel-level processing stages, such as background elimination and skin area detection stages, the operations on different pixels are independent. This independence can be converted into different forms of parallelism such as instruction-level parallelism, thread-level parallelism, process-level parallelism, as well as spatial parallelism, which can be utilized by array processors. Instruction-level parallelism takes advantages of the fact that instructions in the execution path can be issued simultaneously under certain conditions. Since the granularity of instructions is small, instruction-level parallelism is usually associated with fine-grained parallelism existing in a program. Thread and process-level parallelisms are explicitly exhibited in the program as it will have more than one execution path. Thread and process-level parallelism are associated with the large cost of initializing and terminating threads/processes. Since in our case the input frame size is not large enough to make those additional costs ignorable, we convert this intra-frame data independency into instruction-level parallelism, which can be explored by VLIW or superscalar architecture processors. The instruction-level parallelism can be explicitly expressed in an executable file, since the parallelism is available during the compile-time. Both VLIW and superscalar processors can exploit static instruction-level parallelism. Superscalar processors use hardware schemes to discover instruction parallelism in a program, so a superscalar processor can provide backward compatibility for old generation processors. For this reason, most of general processors are superscalar processors. On the other hand, a VLIW processor can achieve a similar performance on a program with explicit parallelism by using significantly less hardware effort with dedicated compiler support. We use the VLIW processor to exploit the instruction-level parallelism that resulted from the intra-frame data independency, since such parallelism can be explicitly expressed at compile time. In the following, we will introduce our process of converting intra-frame data independency to instruction-level parallelism. Although the target is a VLIW processor, most parts of this process can benefit from superscalar processors, as well.

The first step is to use loop fusion, a way of combining two similar, adjacent loops for reducing the overhead, and loop unrolling, which partitions the loops to discover loop-carried dependencies that may let several iterations be executed at the same time, which increases the basic block size and thus increases available instruction parallelism. Figure 16 shows examples of loop fusion and unrolling.

When a loop is executed, there might be dependencies between trips. The instructions that need to be executed in different trips cannot be executed simultaneously. The essential idea behind loop fusion and loop unrolling is to decrease the total number of trips needed to be executed by putting more tasks in each trip. Loop fusion merges loops together without changing the result of the executed program. In Figure 16, two loops are merged into one loop. This change will increase the number of instructions in each trip. Loop unrolling merges consecutive trips together to reduce the total trip count. In this example, the trip count is reduced from four to two as loop unrolling is performed. These source code transformations do not change the execution results, but increase the number of instructions located in each loop trip and thus increase the number of instructions that can be executed simultaneously.

Both loop fusion and loop unrolling increase basic block size by merging several basic blocks together. While loop fusion merges basic blocks in code-domain, in that different code segments are merged, loop unrolling merges basic blocks in time-domain, in that different loop iterations are merged. This step increases the code size for each loop trip. However, we do not observe significant basic block

Figure 16. Loop fusion and unrolling.



```
Before loop fusion

for (i=0; i<N; i++) a[i] = b[i] * 5;
for (j=0; j<N; j++) w[j] = c[j] * d[j];

After loop fusion

for (i=0; i<N; i++) {
a[i] = b[i] * 5; w[i] = c[i] * d[i];
}
```

```
Before loop unrolling

for (i=0;i<4;i++)
  a[i]=b[i]*c[i]

After loop unrolling

for (i=0;i<2;i++){
  a[i*2]=b[i*2]*c[i*2]
  a[i*2+1]=b[i*2+1]*c[i*2+1]
}
```

size changes. The conditional operations, such as absolute value calculation and threshold inside loop, block the increase of basic block size.

In the second step, we sought two methods to reduce the branches, which limit the basic block size in loops. A solution for this is to use conditional execution instructions, which requires hardware support. The TriMedia processors offer such instructions, such as IABS, that calculate the absolute value in a single instruction. This optimization provides a significant performance improvement. Another technique we used is to convert control flow dependency to data dependency by using look-up tables. In our algorithm, contour following, the instruction level parallelism is limited by the control flow dependency. The critical control flow structure is shown on the left-hand side of Figure 17. Although if-conversion is a general method to remove branches caused by if-else statements, the if-conversion does not help much for such a control flow dependency. To increase the available parallelism, we convert the control

Figure 17. Branch reduction.

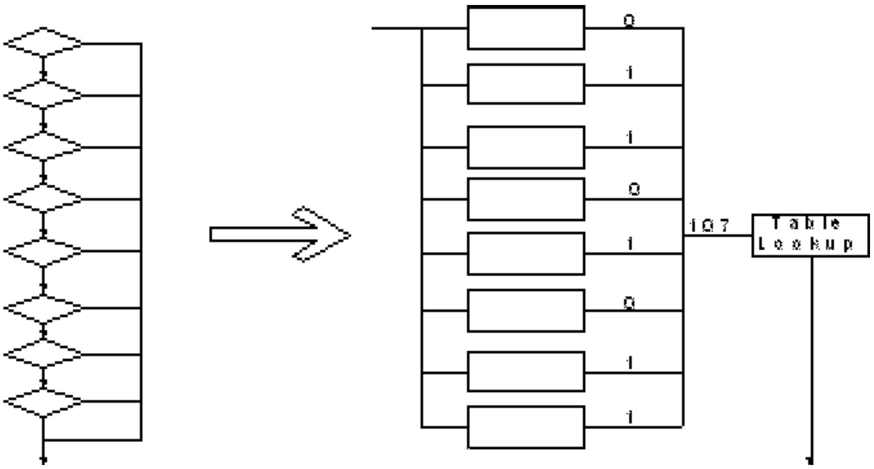
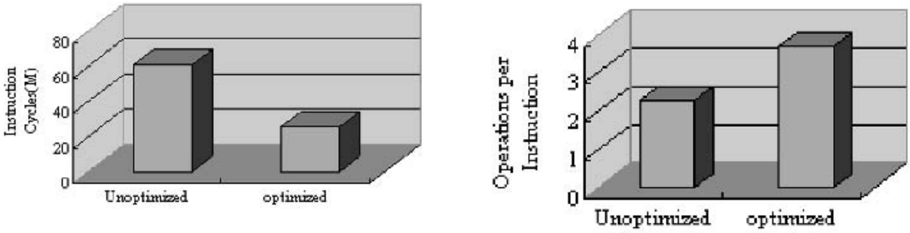


Figure 18. Instruction cycles for 10 frames (left), and available parallelism (right).



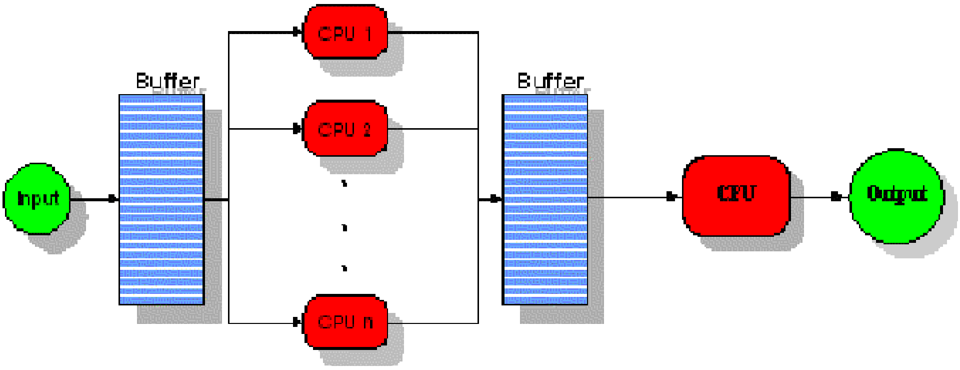
dependency to data dependency by evaluating the condition in each if-statement. The results are put together into one single byte, where each bit represents the result from each evaluation. We build into the algorithm a table that provides the result for any given evaluated byte. By using this method, the branches are eliminated and the instruction-level parallelism in the contour following algorithm block is increased.

The results of these optimizations are shown in Figure 18. Here, operations-per-instruction is used as a measurement for instruction-level parallelism. While optimization towards higher instruction-level parallelism can significantly improve system performance, there are still limitations. The instruction-level parallelism is a fine-grained parallelism, which limits its ability to exploit coarse-grained data independencies, such as inter-frame independency. From a hardware point of view, the increasing global interconnection delay will prevent processor designers from building a large amount of functional units into one single processor, which also limits the exploration of instructional parallelism. In addition, the recent trends show that both application specific computer systems and general computers are starting to incorporate multiple processors. This will provide hardware support for exploiting coarse-grained parallelisms. Considering this, we are starting to explore alternative methods.

Inter-Frame-Level Parallelism and Symmetric Architecture

A different level of data independency in our smart camera system is the inter-frame data independency. Since this independency lies between different input frames, it is a coarse-grained data independency. The corresponding parallel-

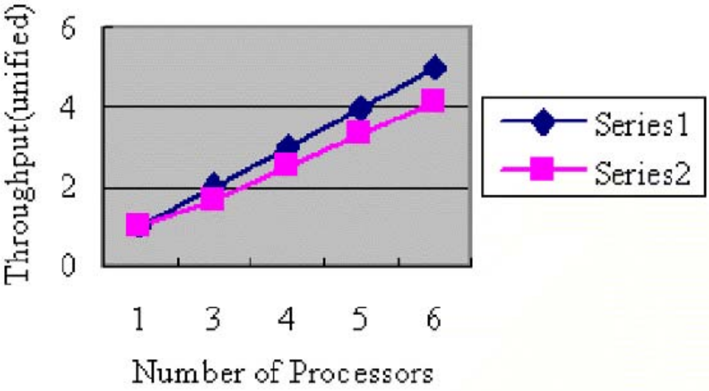
Figure 19. Symmetric parallel architecture.



isms are thread or process-level parallelisms. SMT (Simultaneous Multithreading) and CMP (single chip multi-processor) architectures can exploit process-level parallelism. However, the SMT architecture does not seem to be a good choice for this parallelism, since the almost identical threads will content the same resource and do not increase the functional unit utilization over the single thread model. Thus, we propose using CMP architecture, or even separate chip processors, to exploit such inter-frame parallelism. A proposed architecture is shown in Figure 19.

Figure 20 shows the projected performance change on such parallel architectures, where series1 is the performance under the assumption that communication cost is negligible, while series2 is the performance change where the communication cost is 20.

Figure 20. Performance of symmetric architecture.



Inter-Stage-Level Parallelism and Pipeline Architecture

The above discussions are about the available data independencies. There is another parallelism resulting from the data flow structure. The algorithm stages of the low-level processing part form a pipelined process. A corresponding architecture is a pipelined multi-processor architecture (Figure 21). Figure 22 shows the projected performance of such architecture. Series 1 shows the throughput when communication cost is zero, while in series2 the communication cost is 20% of the computation cost. The additional benefit of such architecture over other parallel architecture is that the processor can be tailored to the requirement of the stage. For example, the CPU used to process background elimination does not have to carry a floating-point unit. The limiting factor of such architecture is the granularity of the stages. When a stage counts more than 50% of the overall computation time, the speed-up is limited.

Figure 21. Macro-pipeline architecture.

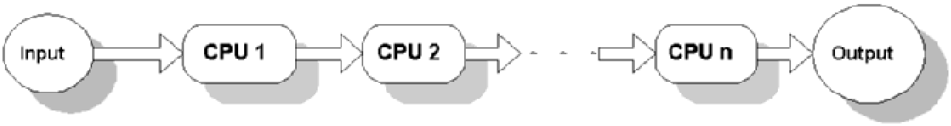
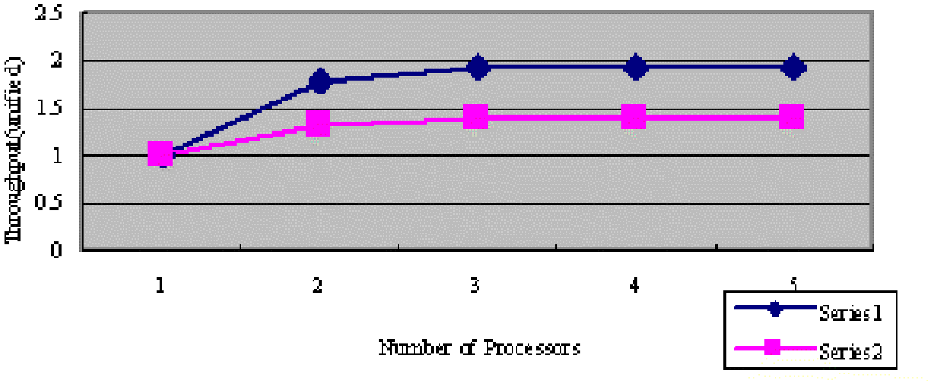


Figure 22. Throughput of pipelined architecture.



Comparison between Architectures and Data Independency

Table 2 summarizes comparison results between different architectures discussed. Among all the architectures, the symmetric parallel architecture can provide the better speed-up, while the pipelining architecture will be able to reduce hardware effort on processors. As we can see through our discussion, those different architectures do not mutually exclude each other. Thus, we would expect a better solution by combining them together.

Table 2. Performance Comparison

Independency	Architecture	Dedicated Architecture	Performance
Intra-frame Independency	VLIW/Superscalar	VLIW-Trimedia1300 Processor	3.7 x
Inter-frame independency	CMP/SMT	Symmetric Parallel Architecture	5 x
Inter-stage independency	CMP/SMT	Macro-Pipelined Architecture	1.3 x

Discussion on Multiple Camera Systems

In this subsection, we examine the parallelism architecture aspects of the 3D camera system. Figure 23 shows the algorithm stages in the system. Figure 24 shows the processing time for each algorithm stage.

Figure 23. Algorithm stages.

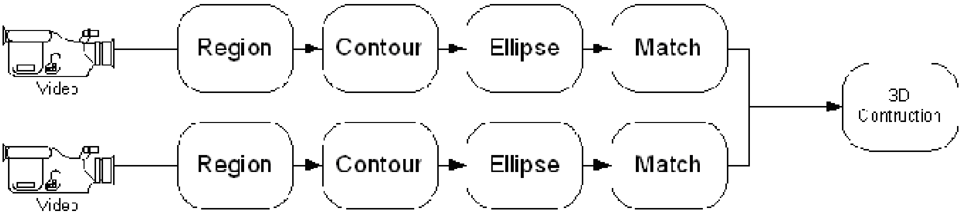
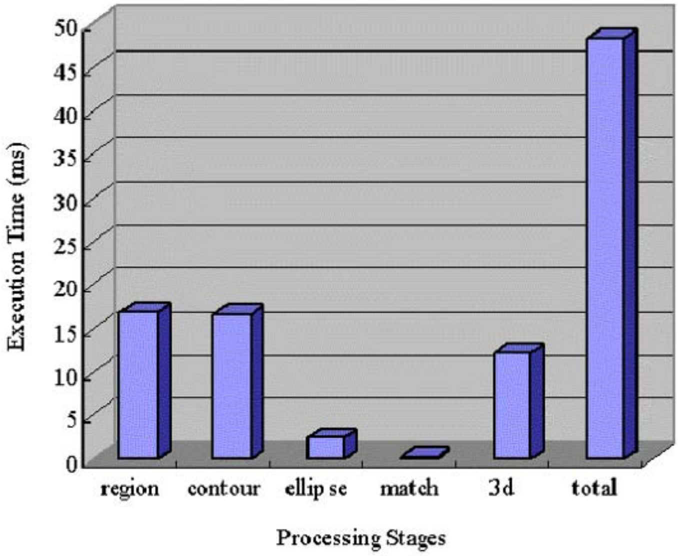


Figure 24. Processing times.



Suppose we need a two-processor architecture. Since, except for the 3d stage, all the stages have two duplicated copies, we can evenly distribute them to the two processors and then put the 3d stage into another (Figure 25). However, by scheduling these tasks, we can find that if we put the 3d stage into a processor, while putting all the ellipse and match stages into another processor, the workload would be more balanced (Figure 26). While such distribution gives the best performance result, when area is more important, we can allocate all the floating point related algorithm stages (ellipse, match and 3d) into one processor and trim off the floating point unit on the other processor (Figure 27).

Figure 25. Intuitive partition.

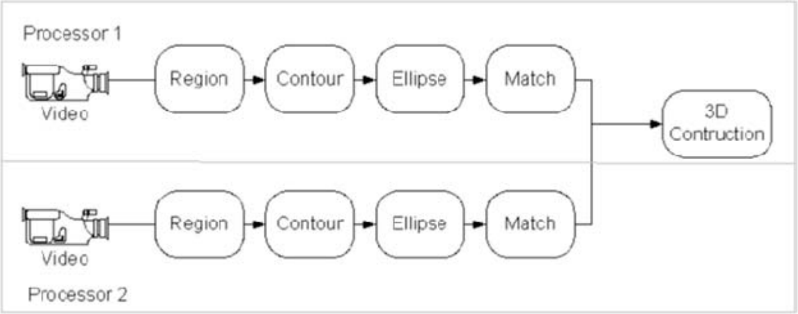


Figure 26. Balanced partition.

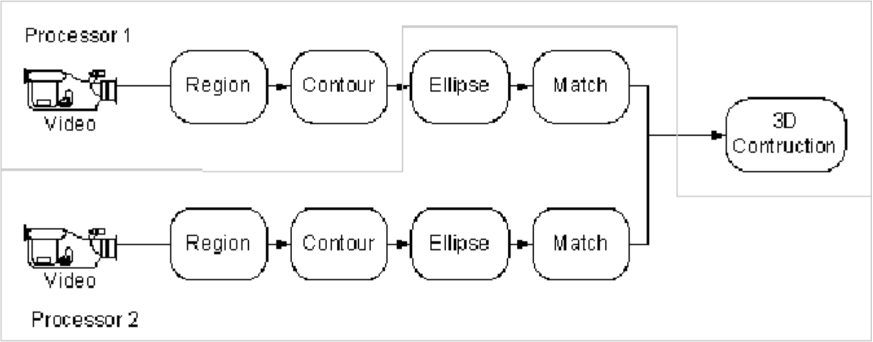
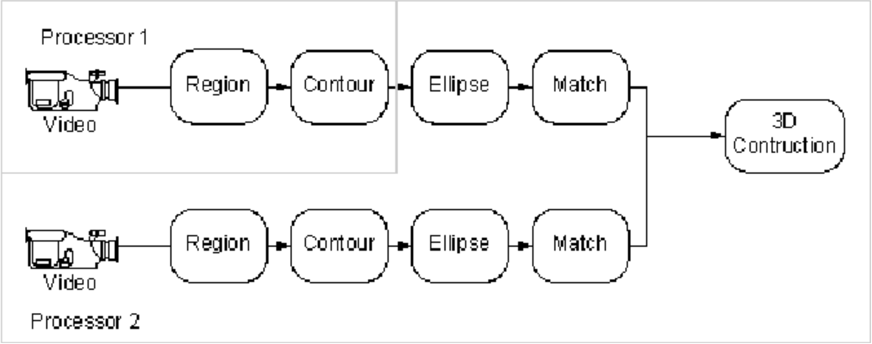


Figure 27. Hardware efficient partition.



Another factor that needs to be considered is communication cost. The amount of data that needs to be transferred between the video camera interface and region, and between the region stage and contour, is significantly larger than the data size exchanged among another stages. Therefore, we would prefer to allocate a set consisting of video interface, region stage, and contour stage into one processor. In the above partitioning, we comply with this rule. The above discussion is limited to inter-stage parallelism. In the following, we will show how the inter-frame parallelism can be taken into consideration. At the first, we duplicate every processing stage. For example, if we want to process two frames in parallel, we will have two copies of each processing stage. After this, we can perform scheduling and get the corresponding architecture. Figure 28 shows a five-processor partition with two consecutive frames processed in parallel. Figure 29 shows the corresponding architecture for the partition.

Figure 28. Partitioning with inter-frame parallelism.

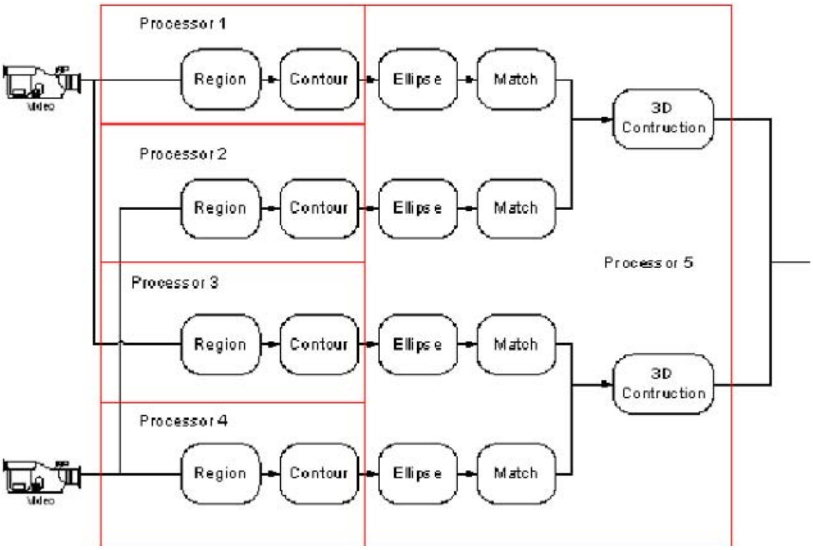
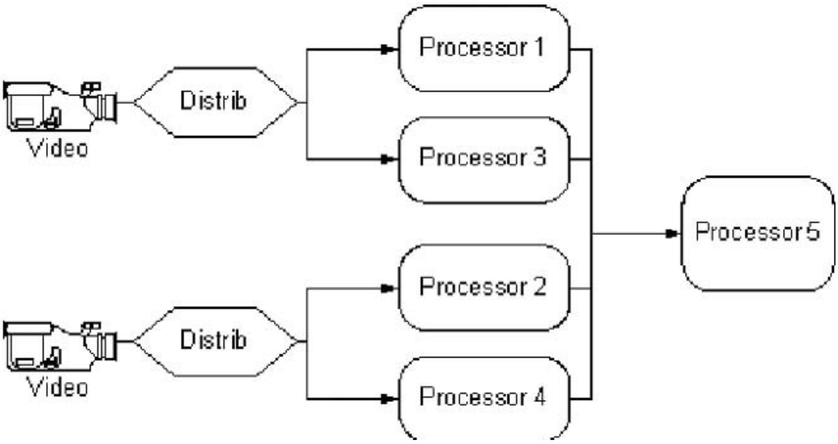


Figure 29. A five-processor architecture.



Conclusions

In this chapter, we review previous 3D methods on real-time processing of multiple views for human detection and activity recognition algorithms. We discuss the advantages and drawbacks of these algorithms with respect to algorithmic and architectural issues. Furthermore, we present our multiple-camera system to investigate the relationship between the activity recognition algorithms and the architectures required to perform these tasks in real-time. The chapter describes the proposed activity recognition method that consists of a distributed algorithm and a data fusion scheme for two and three-dimensional visual analysis, respectively. Furthermore, we analyze the available data independencies for our new algorithm, and discuss the potential architectures to exploit the parallelism resulting from these independencies. Three architectures, i.e., VLIW, symmetric parallel, and macro-pipelined architectures are presented and compared in the chapter.

References

Aggarwal, J. K. & Cai, Q. (1999). Human Motion Analysis: A Review. *Computer Vision and Image Understanding,* 73(3), 428-440.

Bregler, C. & Malik, J. (1998). Tracking people with twists and exponential maps. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 8-15.

Cheung, G. K. M., Kanade, T., Bouguet, J. Y. & Holler, M. (2000). A real time system for robust 3d voxel reconstruction of human motions. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 714-720.

Cohen, I. & Lee, M. W. (2002). 3D Body Reconstruction for Immersive Interaction. *Proceedings of the International Workshop on Articulated Motion and Deformable Objects,* 119-130.

Comaniciu, D., Ramesh, V. & Meer, P. (2000). Real-time Tracking of Non-rigid Objects using Mean Shift. *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition,* 142-149.

Crockett, T. W. (1997). An Introduction to Parallel Rendering. *Parallel Computing,* 23(7), 819-843.

Davis, L. S., Borovikov, E., Cutler, R. & Horprasert, T. (1999). Multi-perspective Analysis of Human Action. *Proceedings of the International Workshop on Cooperative Distributed Vision.*

Delamarre, Q. & Faugeras, O. (2001). 3D Articulated Models and Multi-view Tracking with Physical Forces. *Computer Vision and Image Understanding,* 81(3), 328-357.

Deutscher, J., North, B., Bascle, B. & Blake, A. (1999). Tracking through singularities and discontinuities by random sampling. *Proceedings of the IEEE International Conference on Computer Vision*, 1144-1149.

DiFranco, D., Cham, T. & Rehg, J. (2001), Reconstruction of 3d figure motion from 2d correspondences. *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 307-314.

Dockstader, S. L. & Tekalp, A. M. (2001). Multiple Camera Tracking of Interacting and Occluded Human Motion. *Proceedings of the IEEE*, (89)10, 1441-1455.

England, N. (1986). A graphics system architecture for interactive application-specific display functions. *IEEE Computer Graphics and Applications*, 6(1), 60-70.

Faugeras, O. (1993). *Real time correlation based stereo: Algorithm, implementations and applications*. Research Report 2013, INRIA Sophia-Antipolis.

Focken. D. & Stiefelhagen R. (2002). Towards Vision-based 3-D People Tracking in a Smart Room. *Proceedings of the International Conference on Multimodal Interfaces,* 400-405.

Fritts, J., Wolf, W. & Liu, B. (1999). Understanding Multimedia Application Characteristics for Designing Programmable Media Processors. *Proceedings of the SPIE Photonics West, Media Processors,* 2-13.

Fuchs, H. et al. (1989). Pixel-planes 5: A heterogeneous multi-processor graphics system using processor enhanced memories. *Proceedings of the Siggraph International Conference on Computer Graphics and Interactive Techniques,* 79-88.

Gavrila D. M. (1999). The Visual Analysis of Human Movement: A Survey. *Computer Vision and Image Understanding,* 73(1), 82-98.

Gavrila, D. M. & Davis, L. (1996). 3D model based tracking of humans in action: a multi-view approach. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 73-80.

Goddard, N. (1994). Incremental model-based discrimination of articulated movement direct from motion features. *Proceedings of the IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, 89-94.

Goldlücke, B., Magnor, M. A. & Wilburn, B. (2002). Hardware-accelerated Dynamic Light Field Rendering. *Proceedings of the Conference on Vision, Modeling and Visualization,* 455-462.

Guo, Y., Xu, G. & Tsuji, S. (1994). Understanding human motion patterns. *Proceedings of the International Conference on Pattern Recognition,* 325-329 (B).

Hammond, L., Nayfeh, B. & Olukotun, K. (1997). A Single-Chip Multiprocessor. *IEEE Computer*, 30(9), 79-85.

Haritaoglu, I., Harwood, D. & Davis, L. (1998). W4: A Real Time system for Detecting and Tracking People. *Proceedings of the International Conference on Automatic Face and Gesture Recognition,* 222-227.

Hartley, R. & Zisserman, A. (2000). *Multiple View Geometry.* Cambridge University Press.

Hogg, D. (1983). Model-based vision: a program to see a walking person. *Image Vision Computing,* 1(1), 5-20.

Howe, N., Leventon, M. & Freeman, B. (2000). Bayesian reconstruction of 3d human motion from single camera video. *Advances in Neural Information Processing Systems*, 12, 820-826.

Huang, X. D., Ariki, Y. & Jack, M. A. (1990). *Hidden Markov Models for Speech Recognition.* Edinburgh University Press.

Isard, M. & MacCormick, J. (2001). A Bayesian Multiple-blob Tracker. *Proceedings of the IEEE International Conference on Computer Vision*, 34-41.

Kakadiaris, I. & Metaxas, D. (1995). 3D human body model acquisition from multiple views. *Proceedings of the IEEE International Conference on Computer Vision,* 618-623.

Kakadiaris, I. & Metaxas, D. (1996). Model based estimation of 3D human motion with occlusion based on active multi-viewpoint selection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 81-87.

Kakadiaris, I. A. & Metaxas, D. (1998). Three-dimensional human body model acquisition from multiple views. *International Journal of Computer Vision*, 30(3), 227-230.

Kanade, T., Rander, P. & Narayanan, P. J. (1997). Virtualized Reality: Constructing Virtual Worlds from Real Scenes. *IEEE Multimedia*, 4(1), 34-47.

Kanade, T., Yoshida, A., Oda, K., Kano, H. & Tanaka, M. (1996). A Stereo Machine for Video Rate Dense Depth Mapping and its New Applications. *Proceedings of the Conference on Computer Vision and Pattern Recognition,* 196-202.

Khailany, B. et al. (2001). Imagine: Media Processing with Streams. *IEEE Micro*, 21(2), 35-46.

Kohler, M. & Schroter, S. (1998). *A Survey of Video-based Gesture Recognition - Stereo and Mono Systems*. Research Report No. 693/1998, Fachbereich Informatik, University of Dortmund.

Konolige, K. (1997). Small Vision Systems: Hardware and Implementation. *Proceedings of the International Symposium on Robotics Research,* 203-212.

Koschan, A. & and Rodehorst, V. (1995). Towards real-time stereo employing parallel algorithms for edge-based and dense stereo matching. *Proceedings of the IEEE Workshop Computer Architectures for Machine Perception*, 234-241.

Kuch, J. & Huang, T. (1995). Vision based hand modeling and tracking for virtual teleconferencing and telecollaboration. *Proceedings of the IEEE International Conference on Computer Vision,* 666-671.

Kunimatsu, A. et al. (2000). Vector unit architecture for emotion synthesis. *IEEE Micro*, 20(2), 40-47.

LaViola, J. J. (1999). *A survey of hand postures and gesture recognition techniques and technology*. Technical Report of Brown University Providence: CS-99-11.

Levinthal, A. & Porter, T. (1984). CHAP: A SIMD graphics processor. *Proceedings of the Siggraph International Conference on Computer Graphics and Interactive Techniques*, 77-82.

Li, M., Magnor, M. & Seidel, H. P. (2003). Hardware-Accelerated Visual Hull Reconstruction and Rendering. *Proceedings of the Conference on Graphics Interface.*

Li, Y., Hilton, A. & Illingworth, J. (2001). Towards Reliable Real-Time Multiview Tracking. *Proceedings of the IEEE Workshop on Multi-Object Tracking,* 43-50.

Luck, J. P., Debrunner, C., Hoff, W., He, Q. & Small, D. E. (2002). Development and Analysis of a Real-time Human Motion Tracking System. *Proceedings of the IEEE Workshop on Applications of Computer Vision*, 196-202.

Matthies, L. H. (1992). Stereo vision for planetary rovers: Stochastic modeling to near real time implementation. *International Journal of Computer Vision,* 8(1), 71-91.

Matusik, W., Buehler, C. & McMillan, L. (2001). Polyhedral Visual Hulls for Real-time Rendering. *Proceedings of the Eurographics Workshop on Rendering,* 115-125.

Matusik, W., Buehler, C., Raskar, R., Gortler, S. & McMillan, L. (2000). Image-Based Visual Hulls. *Proceedings of the Siggraph International Conference on Computer Graphics and Interactive Techniques,* 369-374.

Moeslund, T. B. & Granum, E. (2001). A Survey of Computer Vision-Based Human Motion Capture. *Computer Vision and Image Understanding,* 81, 231-268.

Mulligan, J., Isler, V. & Daniilidis, K. (2001). Trinocular Stereo: a Real-Time Algorithm and its Evaluation. *Proceedings of the IEEE Workshop on Stereo and Multi-Baseline Vision,* 1-8.

Munkelt, O., Ridder, C., Hansel, D. & Hafner, W. (1998). A Model Driven 3D Image Interpretation System Applied to Person Detection in Video Images. *Proceedings of the International Conference on Pattern Recognition,* 70-73.

Narayanan, P. J., Rander, P. W. & Kanade, T. (1998). Constructing Virtual Worlds Using Dense Stereo. *Proceedings of the International Conference on Computer Vision,* 3-10.

Nishihara, H. K. (1990). *Real-time implementation of a sign-correlation algorithm for image-matching.* Technical Report 90-2, Teleos Research.

O'Rourke, J. & Badler, N. I. (1980). Model-based image analysis of human motion using constraint propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 2(6), 522-536.

Oren, M., Papageorigiou, C., Sinha, P., Osuna, E. & Poggio, T. (1997). Pedestrian Detection Using Wavelet Templates. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 193-199.

Owens, W. H. (1984). The calculation of a best-fit ellipsoid from elliptical sections on arbitrarily oriented planes. *Journal of Structural Geology,* 6, 571-578.

Ozer, I. B. & Wolf, W. (2001). Human Detection in Compressed Domain. *Proceedings of the IEEE International Conference on Image Processing,* 247-277.

Ozer, I. B. & Wolf, W. (2002a). Real-time Posture and Activity Recognition. *Proceedings of the IEEE Workshop on Motion and Video Computing,* 133-138.

Ozer, I. B. & Wolf, W. (2002b). A Hierarchical Human Detection System in (Un)compressed Domains. *IEEE Transactions on Multimedia. Special Issues on Multimedia Databases,* 4(2), 283-300.

Ozer, I. B., Wolf, W. & Akansu, A. N. (2000). Relational Graph Matching for Human Detection and Posture Recognition. *Proceedings of the SPIE, Photonic East Internet Multimedia Management Systems.*

Papageorgiu, C. & Poggio, T. (1999). Trainable Pedestrian Detection. *Proceedings of the International Conference on Image Processing,* 25-28.

Pollefeys, M., Koch, R., Vergauwen, M. & Gool, L. V. (1999). Hand-Held Acquisition of 3D Models with a Video Camera. *Proceedings of the International Conference on 3D Digital Imaging and Modeling*, 14-23.

Rehg, J. M. & Kanade, T. (1995). Model-based tracking of self-occluding articulated objects. *Proceedings of the IEEE International Conference on Computer Vision,* 612-617.

Rose, R. C. (1992). Discriminant Wordspotting Techniques for Rejection Non-Vocabulary Utterances in Unconstrained Speech. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing,* II, 105-108.

Schardt, T. & Yuan, C. (2002). A Dynamic Communication Model for Loosely Coupled Hybrid Tracking Systems. *Proceedings of the International Conference on Information Fusion.*

Schreer, O., Brandenburg, N. & Kauff, P. (2001). Real-Time Disparity Analysis for Applications in Immersive Tele-Conference Scenarios - A Comparative Study. *Proceedings of the International Conference on Image Analysis and Processing.*

Sim, B. K. & Kim, J. H. (1997). Ligature Modeling for Online Cursive Script Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6), 623-633.

Sminchisescu, C. & Triggs, B. (2001), Covariance scaled sampling for monocular 3d body tracking. *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 447-454.

Starner, T. & Pentland, A. (1995). *Real-Time American Sign Language Recognition from Video Using Hidden Markov Models.* Technical Report TR-375, MIT's Media Lab.

Thompson, C. J., Hahn, S. & Oskin, M. (2002) Using Modern Graphics Architectures for General-Purpose Computing: A Framework and Analysis. *Proceedings of the ACM/IEEE International Symposium on Microarchitecture,* 306-317.

Tullsen, D. M., Eggers, S. J. & Levy, H. M. (1995). Simultaneous Multithreading: A Platform for Next-Generation Processors. *Proceedings of the International Symposium on Computer Architecture*, 392-403.

Wachter, S. & Nagel, H. H. (1999). Tracking persons in monocular image sequences. *Computer Vision and Image Understanding*, 74(3), 174-192.

Watlington, J. A. & Bove, V. M. (1997). A System for Parallel Media Processing. *Parallel Computing*, 23(12), 1793-1809.

Webb, J. (1993). Implementation and performance of fast parallel multi-baseline stereo vision. *Proceedings of the Image Understanding Workshop,*1005-1012.

Wilson, A. D. & Bobick, A. F. (1999). Parametric Hidden Markov Models for Gesture Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(9), 884-900.

Wolf, W., Ozer, I. B. & Lv, T. (2002). Smart Cameras as Embedded Systems. *IEEE Computer*, 35(9), 48-53.

Wren, C. R., Azarbayejani, A., Darrell, T. & Pentland, A. (1999). Pfinder: real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7), 780-785.

Wren, C. R., Clarkson, B. P. & Pentland, A. P. (2000). Understanding Purposeful Human Motion. *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, 378-383.

Wu, Y. & Huang, T. S. (1999). Vision-based gesture recognition: A review. *Proceedings of the Gesture Workshop*, 103-115.

Yang, R., Welch, G. & Bishop, G. (2002). Real-Time Consensus-Based Scene Reconstruction Using Commodity Graphics Hardware. *Proceedings of the Pacific Conference on Computer Graphics and Applications*, 225-234.

Zisserman, A., Fitzgibbon, A. & Cross, G. (1999). VHS to VRML: 3D Graphical Models from Video Sequences. *Proceedings of the International Conference on Multimedia Systems*, 51-57.

Chapter V

Facial Expression and Gesture Analysis for Emotionally-Rich Man-Machine Interaction

Kostas Karpouzis, Amaryllis Raouzaiou, Athanasios Drosopoulos,
Spiros Ioannou, Themis Balomenos, Nicolas Tsapatsoulis, and
Stefanos Kollias
National Technical University of Athens, Greece

Abstract

This chapter presents a holistic approach to emotion modeling and analysis and their applications in Man-Machine Interaction applications. Beginning from a symbolic representation of human emotions found in this context, based on their expression via facial expressions and hand gestures, we show that it is possible to transform quantitative feature information from video sequences to an estimation of a user's emotional state. While these features can be used for simple representation purposes, in our approach they are utilized to provide feedback on the users' emotional state, hoping to provide next-generation interfaces that are able to recognize the emotional states of their users.

Introduction

Current information processing and visualization systems are capable of offering advanced and intuitive means of receiving input from and communicating output to their users. As a result, Man-Machine Interaction (MMI) systems that utilize multimodal information about their users' current emotional state are presently at the forefront of interest of the computer vision and artificial intelligence communities. Such interfaces give the opportunity to less technology-aware individuals, as well as handicapped people, to use computers more efficiently and, thus, overcome related fears and preconceptions. Besides this, most emotion-related facial and body gestures are considered universal, in the sense that they are recognized among different cultures. Therefore, the introduction of an "emotional dictionary" that includes descriptions and perceived meanings of facial expressions and body gestures, so as to help infer the likely emotional state of a specific user, can enhance the affective nature of MMI applications (Picard, 2000).

Despite the progress in related research, our intuition of what a human expression or emotion actually represents is still based on trying to mimic the way the human mind works while making an effort to recognize such an emotion. This means that even though image or video input are necessary to this task, this process cannot come to robust results without taking into account features like speech, hand gestures or body pose. These features provide the means to convey messages in a much more expressive and definite manner than wording, which can be misleading or ambiguous. While a lot of effort has been invested in individually examining these aspects of human expression, recent research (Cowie et al., 2001) has shown that even this approach can benefit from taking into account multimodal information. Consider a situation where the user sits in front of a camera-equipped computer and responds verbally to written or spoken messages from the computer: speech analysis can indicate periods of silence from the part of the user, thus informing the visual analysis module that it can use related data from the mouth region, which is essentially ineffective when the user speaks. Hand gestures and body pose provide another powerful means of communication. Sometimes, a simple hand action, such as placing one's hands over their ears, can pass on the message that they've had enough of what they are hearing more expressively than any spoken phrase.

In this chapter, we present a systematic approach to analyzing emotional cues from user facial expressions and hand gestures. In the Section "Affective analysis in MMI," we provide an overview of affective analysis of facial expressions and gestures, supported by psychological studies describing emotions as discrete points or areas of an "emotional space." The sections "Facial expression analysis" and "Gesture analysis" provide algorithms and experimental results from the analysis of facial expressions and hand gestures in video sequences. In the case of facial expressions, the motion of tracked feature points is translated to MPEG-4 FAPs, which describe their observed motion in a high-level manner. Regarding hand gestures, hand segments are located in a video sequence via color segmentation and motion estimation algorithms. The position of these segments is tracked to provide the hand's position over time and fed into a HMM architecture to provide affective gesture estimation.

In most cases, a single expression or gesture cannot help the system deduce a positive decision about the users' observed emotion. As a result, a fuzzy architecture is employed that uses the symbolic representation of the tracked features as input. This concept is described in the section "Multimodal affective analysis." The decision of the fuzzy system is based on rules obtained from the extracted features of actual video sequences showing emotional human discourse, as well as feature-based description of common knowledge of what everyday expressions and gestures mean. Results of the multimodal affective analysis system are provided here, while conclusions and future work concepts are included in the final section "Conclusions – Future work."

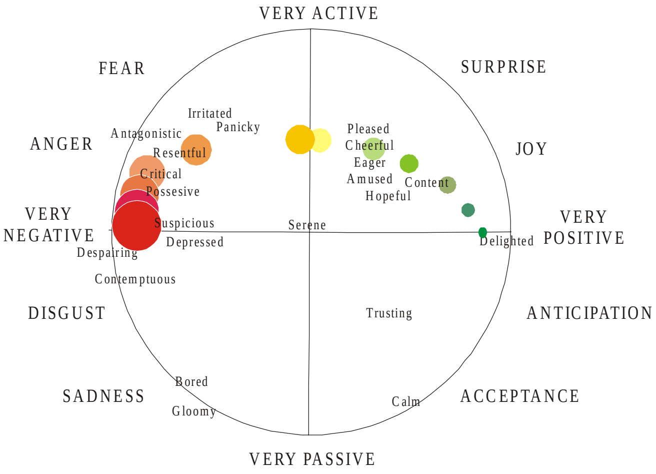
Affective Analysis in MMI

Representation of Emotion

The obvious goal for emotion analysis applications is to assign category labels that identify emotional states. However, labels as such are very poor descriptions, especially since humans use a daunting number of labels to describe emotion. Therefore, we need to incorporate a more transparent, as well as continuous, representation that more closely matches our conception of what emotions are or, at least, how they are expressed and perceived.

Activation-emotion space (Whissel, 1989) is a representation that is both simple and capable of capturing a wide range of significant issues in emotion (Cowie et al., 2001). Perceived full-blown emotions are not evenly distributed in this space; instead, they tend to form a roughly circular pattern. From that and related evidence, Plutchik (1980) shows that there is a circular structure inherent in emotionality. In this framework, emotional strength can be measured as the distance from the origin to a given point in activation-evaluation space. The concept of a full-blown emotion can then be translated roughly as a state where emotional strength has passed a certain limit. A related extension is to think of primary or basic emotions as cardinal points on the periphery of an emotion

Figure 1. The Activation-emotion space.



circle. Plutchik has offered a useful formulation of that idea, the "emotion wheel" (see Figure 1).

Activation-evaluation space is a surprisingly powerful device, which is increasingly being used in computationally oriented research. However, it has to be noted that such representations depend on collapsing the structured, high-dimensional space of possible emotional states into a homogeneous space of two dimensions. There is inevitably loss of information. Worse still, there are different ways of making the collapse lead to substantially different results. That is well illustrated in the fact that fear and anger are at opposite extremes in Plutchik's emotion wheel, but close together in Whissell's activation/emotion space. Thus, extreme care is needed to ensure that collapsed representations are used consistently.

MPEG-4 Based Representation

In the framework of MPEG-4 standard, parameters have been specified for Face and Body Animation (FBA) by defining specific Face and Body nodes in the scene graph. MPEG-4 specifies 84 feature points on the neutral face, which provide spatial reference for FAPs definition. The FAP set contains two high-level parameters, visemes and expressions. Most of the techniques for facial animation are based on a well-known system for describing "all visually distinguishable facial movements" called the Facial Action Coding System (FACS) (Ekman & Friesen, 1978). FACS is an anatomically oriented coding system, based on the definition of "Action Units" (AU) of a face that cause facial movements. An Action Unit could combine the movement of two muscles or work in the reverse way, i.e., split into several muscle movements. The FACS model has inspired the derivation of facial animation and definition parameters in the framework of MPEG-4 (Tekalp & Ostermann, 2000). In particular, the Facial Definition Parameter (FDP) and the Facial Animation Parameter (FAP) set were designed to allow the definition of a facial shape and texture. These sets eliminate the need for specifying the topology of the underlying geometry, through FDPs, and the animation of faces reproducing expressions, emotions and speech pronunciation, through FAPs.

Affective Facial Expression Analysis

There is a long history of interest in the problem of recognizing emotion from facial expressions (Ekman & Friesen, 1978), and extensive studies on face perception during the last 20 years (Davis & College, 1975). The salient issues in emotion recognition from faces are parallel in some respects to the issues associated with voices, but divergent in others.

In the context of faces, the task has almost always been to classify examples of archetypal emotions. That may well reflect the influence of Ekman and his colleagues, who have argued robustly that the facial expression of emotion is inherently categorical. More recently, morphing techniques have been used to probe states that are intermediate between archetypal expressions. They do reveal effects that are consistent with a degree of categorical structure in the domain of facial expression, but they are not particularly large, and there may be alternative ways of explaining them — notably by considering how category terms and facial parameters map onto activation-evaluation space (Karpouzis, Tsapatsoulis & Kollias, 2000).

Analysis of the emotional expression of a human face requires a number of pre-processing steps which attempt to detect or track the face, to locate characteristic facial regions such as eyes, mouth and nose, to extract and follow the movement of facial features, such as characteristic points in these regions or model facial gestures using anatomic information about the face.

Facial features can be viewed (Ekman & Friesen, 1975) as static (such as skin color), slowly varying (such as permanent wrinkles), or rapidly varying (such as raising the eyebrows) with respect to time evolution. Detection of the position and shape of the mouth, eyes and eyelids and extraction of related features are the targets of techniques applied to still images of humans. It has, however, been shown (Bassili, 1979) that facial expressions can be more accurately recognized from image sequences, than from single still images. Bassili's experiments used point-light conditions, i.e., subjects viewed image sequences in which only white dots on a darkened surface of the face were visible. Expressions were recognized at above chance levels when based on image sequences, whereas only happiness and sadness were recognized when based on still images.

Affective Gesture Analysis

The detection and interpretation of hand gestures has become an important part of human computer interaction (MMI) in recent years (Wu & Huang, 2001). Sometimes, a simple hand action, such as placing a person's hands over his ears, can pass on the message that he has had enough of what he is hearing. This is conveyed more expressively than with any other spoken phrase.

Gesture tracking and recognition

In general, human hand motion consists of the global hand motion and local finger motion. Hand motion capturing deals with finding the global and local motion of hand movements. Two types of cues are often used in the localization process: color cues (Kjeldsen & Kender, 1996) and motion cues (Freeman & Weissman, 1995). Alternatively, the fusion of color, motion and other cues, like speech or gaze, is used (Sharma, Huang & Pavlovic, 1996).

Hand localization is locating hand regions in image sequences. Skin color offers an effective and efficient way to fulfill this goal. According to the representation of color distribution in certain color spaces, current techniques of skin detection can be classified into two general approaches: nonparametric (Kjeldsen & Kender, 1996) and parametric (Wren, Azarbayejani, Darrel & Pentland, 1997).

To capture articulate hand motion in full degree of freedom, both global hand motion and local finger motion should be determined from video sequences. Different methods have been taken to approach this problem. One possible method is the appearance-based approach, in which 2-D deformable hand-shape templates are used to track a moving hand in 2-D (Darrell, Essa & Pentland, 1996). Another possible way is the 3-D model-based approach, which takes the advantages of *a priori* knowledge built in the 3-D models.

Meaningful gestures could be represented by both temporal hand movements and static hand postures. Hand postures express certain concepts through hand configurations, while temporal hand gestures represent certain actions by hand movements. Sometimes, hand postures act as special transition states in temporal gestures and supply a cue to segment and recognize temporal hand gestures.

In certain applications, continuous gesture recognition is required and, as a result, the temporal aspect of gestures must be investigated. Some temporal gestures are specific or simple and could be captured by low-detail dynamic models. However, many high detail activities have to be represented by more complex gesture semantics, so modeling the low-level dynamics is insufficient. The HMM (Hidden Markov Model) technique (Bregler, 1997) and its variations (Darrell & Pentland, 1996) are often employed in modeling, learning, and recognition of temporal signals. Because many temporal gestures involve motion trajectories and hand postures, they are more complex than speech signals. Finding a suitable approach to model hand gestures is still an open research problem.

Facial Expression Analysis

Facial Features Relevant to Expression Analysis

Facial analysis includes a number of processing steps that attempt to detect or track the face, to locate characteristic facial regions such as eyes, mouth and nose, to extract and follow the movement of facial features, such as characteristic points in these regions or model facial gestures using anatomic information about the face.

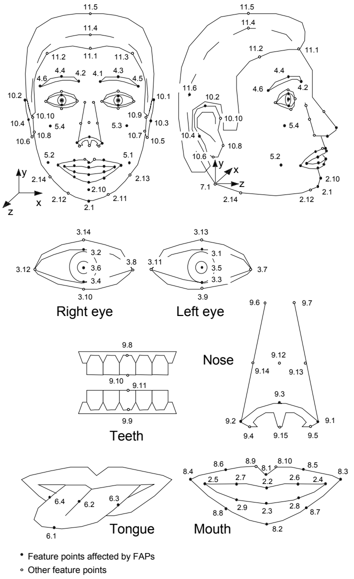
Although FAPs provide all the necessary elements for MPEG-4 compatible animation, they cannot be directly used for the analysis of expressions from video sequences, due to the absence of a clear quantitative definition framework. In order to measure FAPs in real image sequences, we have to define a mapping between them and the movement of specific FDP feature points (FPs), which correspond to salient points on the human face.

Table 1 provides the quantitative modeling of FAPs that we have implemented using the features labeled as f_i (*i=1..15*) (Karpouzis, Tsapatsoulis & Kollias, 2000). This feature set employs feature points that lie in the facial area and can be automatically detected and tracked. It consists of distances, noted as *s(x,y)*, between protuberant points, *x* and *y*, corresponding to the Feature Points shown in Figure 2. Some of these points are constant during expressions and can be used as reference points. Distances between these points are used for normalization purposes (Raouzaiou, Tsapatsoulis, Karpouzis & Kollias, 2002).

Table 1. Quantitative FAP modeling: (1) s(x,y) is the Euclidean distance between the FPs; (2) $D_{i\text{-}NEUTRAL}$ refers to the distance D_i when the face is in its neutral position.

FAP name	Feature for the description	Utilized feature
squeeze_l_eyebrow (F_{37})	D_1=s(4.5,3.11)	$f_{1=} D_{1\text{-}NEUTRAL} - D_1$
squeeze_r_eyebrow (F_{38})	D_2=s(4.6,3.8)	$f_{2=} D_{2\text{-}NEUTRAL} - D_2$
lower_t_midlip (F_4)	D_3=s(9.3,8.1)	$f_{3=} D_3 - D_{3\text{-}NEUTRAL}$
raise_b_midlip (F_5)	D_4=s(9.3,8.2)	$f_{4=} D_{4\text{-}NEUTRAL} - D_4$
raise_l_I_eyebrow (F_{31})	D_5=s(4.1,3.11)	$f_{5=} D_5 - D_{5\text{-}NEUTRAL}$
raise_r_I_eyebrow (F_{32})	D_6=s(4.2,3.8)	$f_{6=} D_6 - D_{6\text{-}NEUTRAL}$
raise_l_o_eyebrow (F_{35})	D_7=s(4.5,3.7)	$f_{7=} D_7 - D_{7\text{-}NEUTRAL}$
raise_r_o_eyebrow (F_{36})	D_8=s(4.6,3.12)	$f_{8=} D_8 - D_{8\text{-}NEUTRAL}$
raise_l_m_eyebrow (F_{33})	D_9=s(4.3,3.7)	$f_{9=} D_9 - D_{9\text{-}NEUTRAL}$
raise_r_m_eyebrow (F_{34})	D_{10}=s(4.4,3.12)	$f_{10=} D_{10} - D_{10\text{-}NEUTRAL}$
open_jaw (F_3)	D_{11}=s(8.1,8.2)	$f_{11=} D_{11} - D_{11\text{-}NEUTRAL}$
close_t_l_eyelid (F_{19}) – close_b_l_eyelid (F_{21})	D_{12}=s(3.1,3.3)	$f_{12=} D_{12} - D_{12\text{-}NEUTRAL}$
close_t_r_eyelid (F_{20}) – close_b_r_eyelid (F_{22})	D_{13}=s(3.2,3.4)	$f_{13=} D_{13} - D_{13\text{-}NEUTRAL}$
stretch_l_cornerlip (F_6) (stretch_l_cornerlip_o)(F_{53}) – stretch_r_cornerlip (F_7) (stretch_r_cornerlip_o)(F_{54})	D_{14}=s(8.4,8.3)	$f_{14=} D_{14} - D_{14\text{-}NEUTRAL}$
squeeze_l_eyebrow (F_{37}) AND squeeze_r_eyebrow (F_{38})	D_{15}=s(4.6,4.5)	$f_{15=} D_{15\text{-}NEUTRAL} - D_{15}$

Figure 2. FDP feature points (adapted from (Tekalp & Ostermann, 2000)).



Facial Feature Extraction

The facial feature extraction scheme used in the system proposed in this chapter is based on a hierarchical, robust scheme, coping with large variations in the appearance of diverse subjects, as well as the same subject in various instances within real video sequences (Votsis, Drosopoulos & Kollias, 2003). Soft *a priori* assumptions are made on the pose of the face or the general location of the features in it. Gradual revelation of information concerning the face is supported under the scope of optimization in each step of the hierarchical scheme, producing *a posteriori* knowledge about it and leading to a step-by-step visualization of the features in search.

Face detection is performed first through detection of skin segments or blobs, merging them based on the probability of their belonging to a facial area, and identification of the most salient skin color blob or segment. Following this, primary facial features, such as eyes, mouth and nose, are dealt with as major discontinuities on the segmented, arbitrarily rotated face. In the first step of the method, the system performs an optimized segmentation procedure. The initial estimates of the segments, also called seeds, are approximated through min-max analysis and refined through the maximization of a conditional likelihood function. Enhancement is needed so that closed objects will occur and part of the artifacts will be removed. Seed growing is achieved through expansion, utilizing chromatic and value information of the input image. The enhanced seeds form an object set, which reveals the in-plane facial rotation through the use of active contours applied on all objects of the set, which is restricted to a finer set, where the features and feature points are finally labeled according to an error minimization criterion.

Experimental Results

Figure 3 shows a characteristic frame from the “hands over the head” sequence. After skin detection and segmentation, the primary facial features are shown in Figure 4. Figure 5 shows the initial detected blobs, which include face and mouth. Figure 6 shows the estimates of the eyebrow and nose positions. Figure 7 shows the initial neutral image used to calculate the FP distances. In Figure 8 the horizontal axis indicates the FAP number, while the vertical axis shows the corresponding FAP values estimated through the features stated in the second column of Table 1.

Figure 3. A frame from the original sequence.

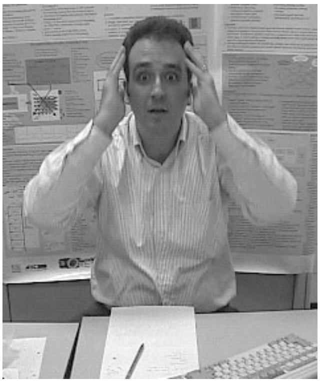


Figure 4. Detected primary facial features.

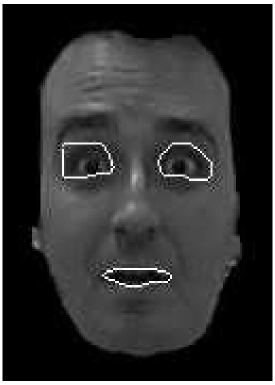


Figure 5. The apex of an expression.

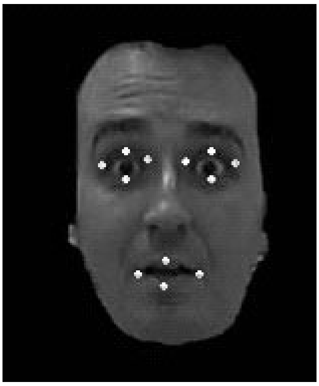


Figure 6. Detected facial features.

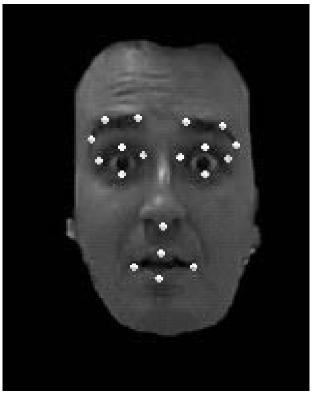


Figure 7. A neutral expression.

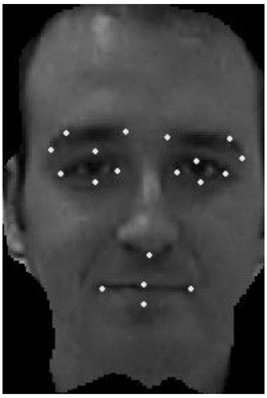
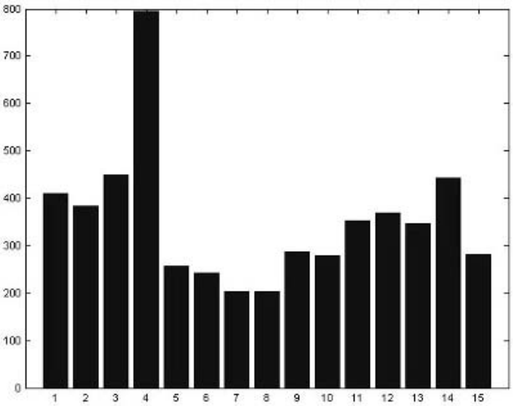


Figure 8. Estimated FAP values for Figure 6.



Gesture Analysis

Hand Detection and Tracking

In order to extract emotion-related features through hand movement, we implemented a hand-tracking system. Emphasis was on implementing a near real-time, yet robust enough system for our purposes. The general process involves the creation of *moving skin masks*, namely skin color areas that are tracked between subsequent frames. By tracking the centroid of those skin masks, we produce an estimate of the user's movements.

In order to implement a computationally light system, our architecture (Figure 9) takes into account *a priori* knowledge related to the expected characteristics of the input image. Since the context is MMI applications, we expect to locate the head in the middle area of the upper half of the frame and the hand segments near the respective lower corners. In addition to this, we concentrate on the motion of hand segments, given that they are the end effectors of the hand and arm chain and, thus, the most expressive object in tactile operations.

For each frame, as in the face detection process, a skin color probability matrix is computed by calculating the joint probability of the Cr/Cb image values (Figure 10). The skin color mask is then obtained from the skin probability matrix using thresholding (Figure 11). Possible moving areas are found by thresholding the difference pixels between the current frame and the next, resulting in the possible-motion mask (Figure 18). This mask does not contain information about the direction or the magnitude of the movement, but is only indicative of the motion and is used to accelerate the algorithm by concentrating tracking only in moving image areas. Both color (Figure 11) and motion (Figure 18) masks

Figure 9. Abstract architecture of the hand tracking module.

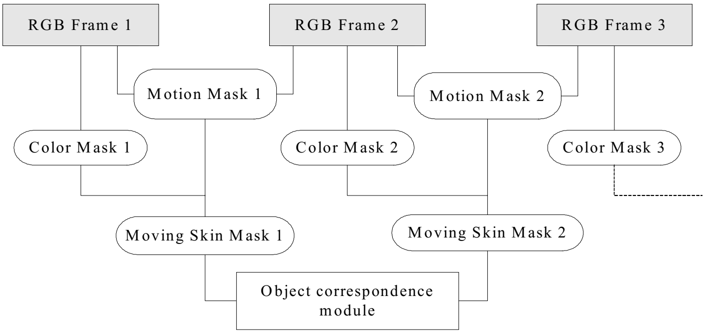


Figure 10. Skin Probability.

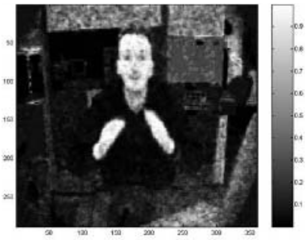


Figure 11. Thresholded skin probability (p> 0.8).

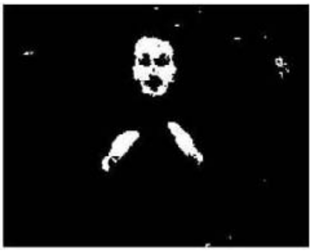


Figure 12. Distance transform of Figure 11.

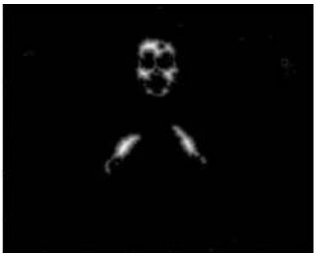


Figure 13. Markers extracted from Figure 12 (area smaller than 2% of the image).

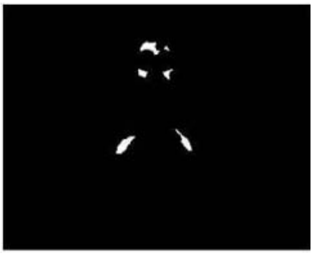


Figure 14. Reconstruction of Figure 11 using Figure 13.

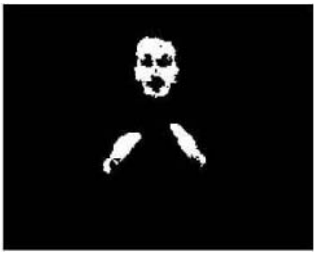
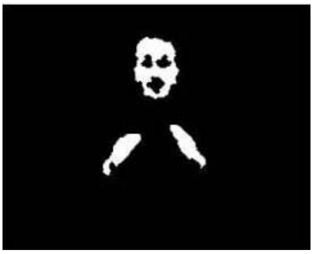


Figure 15. Closing of Figure 14, final color mask.



contain a large number of small objects due to the presence of noise and objects with color similar to the skin. To overcome this, morphological filtering is employed on both masks to remove small objects. All described morphological operations are carried out with a disk-structuring element with a radius of 1% of the image width. The distance transform of the color mask is first calculated (Figure 12) and only objects above the desired size are retained (Figure 13). These objects are used as markers for the morphological reconstruction of the initial color mask. The color mask is then closed to provide better centroid calculation.

The moving skin mask (msm) is then created by fusing the processed skin and motion masks (sm, mm), through the morphological reconstruction of the color mask using the motion mask as marker. The result of this process, after excluding the head object, is shown in Figure 19. The moving skin mask consists of many large connected areas. For the next frame, a new moving skin mask is created, and a one-to-one object correspondence is performed. Object correspondence

Figure 16. Skin color probability for the input image.

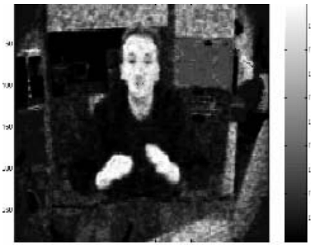


Figure 17. Initial color mask created with skin detection.

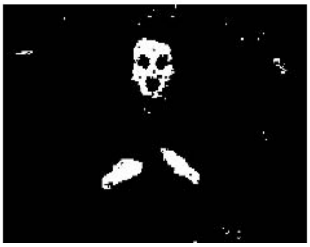


Figure 18: Initial motion mask (after pixel difference thresholded to 10% of max.).

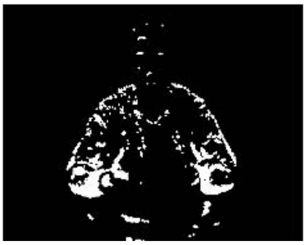


Figure 19. Moving hand segments after morphological reconstruction.

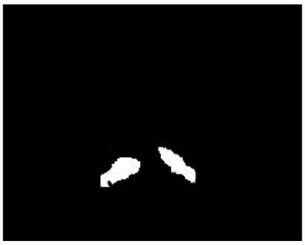


Figure 20. Tracking of one hand object in the "lift of the hand" sequence.

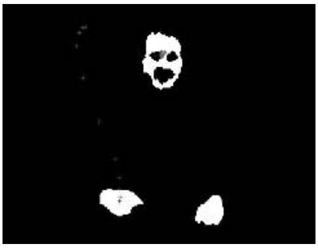
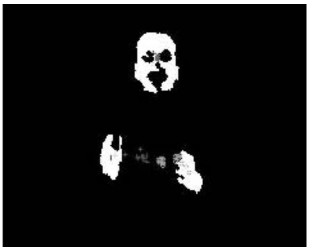


Figure 21. Tracking of both hand objects in the "clapping" sequence.



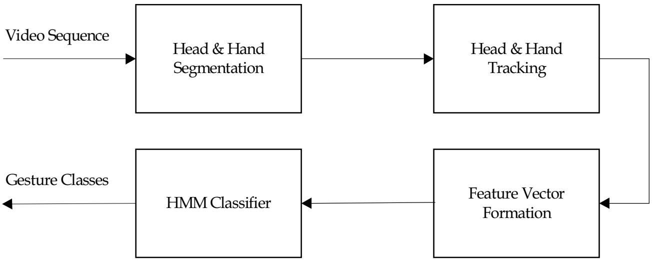
between two frames is performed on the color mask and is based on object centroid distance for objects of similar (at least 50%) area (Figure 20). In these figures, crosses represent the position of the centroid of the detected right hand of the user, while circles correspond to the left hand. In the case of hand object merging and splitting, e.g., in the case of clapping, we establish a new matching of the left-most candidate object to the user's right hand and the right-most object to the left hand (Figure 21).

Following object matching in the subsequent moving skin masks, the mask flow is computed, i.e., a vector for each frame depicting the motion direction and magnitude of the frame's objects. The described algorithm is lightweight, allowing a rate of around 12 fps on a usual PC during our experiments, which is enough for continuous gesture tracking. The object correspondence heuristic makes it possible to individually track the hand segments correctly, at least during usual meaningful gesture sequences. In addition, the fusion of color and motion information eliminates any background noise or artifacts, thus reinforcing the robustness of the proposed approach.

Gesture Classification Using HMMs

Figure 22 shows the architecture of the gesture classification subsystem. Head and hand segmentation and tracking have been described in previous sections, while the remaining blocks of this architecture are described in the following paragraphs.

Figure 22. A general framework for gesture classification through HMMs.



The HMM classifier

In Table 2 we present the utilized features that feed (as sequences of vectors) the HMM classifier, as well as the output classes of the HMM classifier.

Table 2: a) Features (inputs to HMM) and b) Gesture classes (outputs of HMM).

Features	X_{lh} - X_{rh}, X_f -X_{rh}, X_f -$X_{lh,}$ Y_{lh} - $Y_{rh,}$ Y_f -$Y_{rh,}$ Y_f -Y_{lh} where C_f=(X_f, Y_f) are the coordinates of the head centroid, C_{rh}=(X_{rh}, Y_{rh}) and C_{lh}=(X_{lh}, Y_{lh}) are the coordinates of the right and left hand centroids respectively
Gesture Classes	hand clapping – high frequency, hand clapping – low frequency lift of the hand – low speed, lift of the hand – high speed hands over the head – gesture, hands over the head – posture italianate gestures

A general diagram of the HMM classifier is shown in Figure 23. The recognizer consists of *M* different HMMs corresponding to the modeled gesture classes. In our case, *M=7* as it can be seen in Table 2. We use first order left-to-right models consisting of a varying number (for each one of the HMMs) of internal states $G_{k,j}$ that have been identified through the learning process. For example, the third HMM, which recognizes low speed on *hand lift*, consists of only three states $G_{3,1}$, $G_{3,2}$ and $G_{3,3}$. More complex gesture classes, like the *hand clapping*, require as much as eight states to be efficiently modeled by the corresponding HMM. Some characteristics of our HMM implementation are presented below.

- The output probability for any state $G_{k,\,j}$ (*k* corresponds to the *id* of the HMM while *j* refers to the *id* of the state within a particular HMM) is obtained by a continuous probability density function (pdf). This choice has been made in order to decrease the amount of training data. In the discrete case, the size of the code book should be large enough to reduce quantization error and, therefore, a large amount of training data is needed to estimate the output probability. One problem with the continuous pdf is the proper selection of the initial values of density's parameters so as to avoid convergence in a local minimum.
- The output pdf of state $G_{k,\,j}$ is approximated using a multivariate normal distibution model, i.e.:

$$b_{k,j}(\mathbf{O}_i) = \frac{\exp\{-\frac{1}{2}(\mathbf{O}_i - \boldsymbol{\mu}_{k,j})^T \mathbf{C}_{k,j}{}^{-1}(\mathbf{O}_i - \boldsymbol{\mu}_{k,j})\}}{(2\pi)^{\frac{K}{2}} \cdot \left|\mathbf{C}_{k,j}\right|^{\frac{1}{2}}} \tag{1}$$

  where $\mathbf{O}_i$ is *i-th* observation (input feature vector), $\boldsymbol{\mu}_{k,\,j}$ is the mean vector of state $G_{k,\,j}$, $\mathbf{C}_{k,\,j}$ is the respective covariance matrix and *K* is the number of components in $\mathbf{O}_i$ (in our case *K*=6). Initial values for $\boldsymbol{\mu}_{k,\,j}$ and $\mathbf{C}_{k,\,j}$ were obtained off-line by using statistical means. Re-estimation is executed using a variant of the Baum-Welch procedure to account for vectors (such as $\boldsymbol{\mu}_{k,\,j}$) and matrices (such as $\mathbf{C}_{k,\,j}$).
- Transition probabilities $a_{k,mn}$ between states $G_{k,\,m}$ and $G_{k,\,n}$ are computed by using the cumulative probability of $b_{k,m}$ ($\mathbf{O}_i$) gives the estimation of the transition probability, i.e., $a_{k,mn} = 1 - \Phi_{k,m}(\mathbf{O}_i)$. Note that, since the HMM is assumed to operate in a left-to-right mode, $a_{k,mn}$=0, $n<m$, $a_{k,mn}$=1-$a_{k,mn}$ at all times.

- The match score of feature vector sequence $\mathbf{O} = \mathbf{O}_1\mathbf{O}_2 ... \mathbf{O}_T$ given the model $\lambda_m(\mathbf{A}_m, \mathbf{B}_m, \boldsymbol{\pi}_m)$ (m=1,2,…,M) is calculated as follows:
  - o We compute the best state sequence $\mathbf{Q}^*$ given the observation sequence $\mathbf{O}$, using Viterbi's algorithm, i.e.:

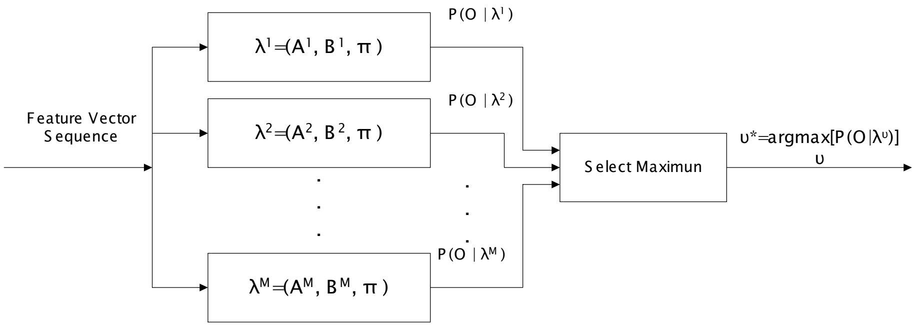
$$\mathbf{Q}^* = \arg\max_{\mathbf{Q}} \{P(\mathbf{Q} / \mathbf{O}, \lambda_m)\} \qquad (2)$$

  - o The match score of observation sequence $\mathbf{O}$ given the state sequence $\mathbf{Q}^*$ is the following quantity:

$$P^* = P(\mathbf{O} / \mathbf{Q}^*, \lambda_m) \qquad (3)$$

It should be mentioned here that the final block of the architecture corresponds to a hard decision system, i.e., it selects the best-matched gesture class. However, when gesture classification is used to support the facial expression analysis process, the probabilities of the distinct HMMs should be used instead (soft decision system). In this case, since the HMMs work independently, their outputs do not sum up to one.

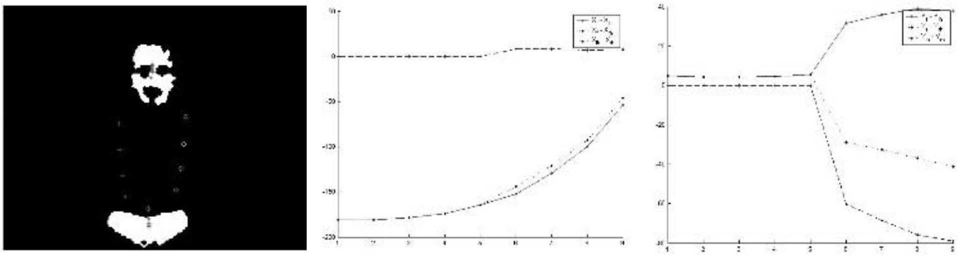
Figure 23. Block diagram of the HMM classifier.



Experimental Results

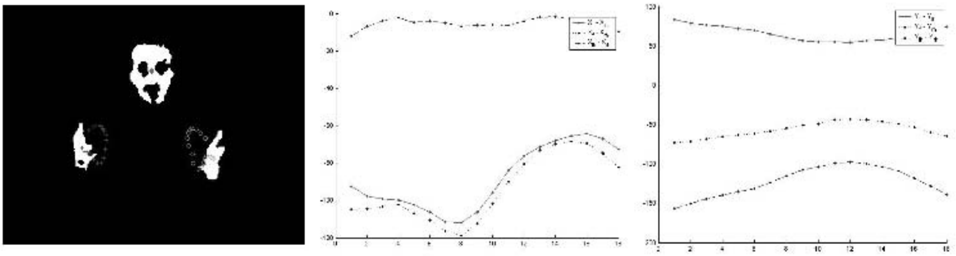
In the first part of our experiments, the efficiency of the features used to discriminate the various gesture classes is illustrated (Figure 24 to Figure 27). The first column shows a characteristic frame of each sequence and the tracked centroids of the head and left and right hand, while the remaining two columns show the evolution of the features described in the first row of Table 2, i.e., the difference of the horizontal and vertical coordinates of the head and hand segments. In the case of the first sequence, the gesture is easily discriminated since the vertical position of the hand segments almost matches that of the head, while in the closing frame of the sequence the three objects overlap. Overlapping is crucial to indicate that two objects are in contact during some point of the gesture, in order to separate this sequence from, e.g., the "lift of the hand" gesture. Likewise, during clapping, the distance between the two hand segments is zeroed periodically, with the length of the in-between time segments providing a measure of frequency, while during the "italianate" gesture the horizontal distance of the two hands follows a repetitive, sinusoidal pattern.

Figure 24. Hands over the head.



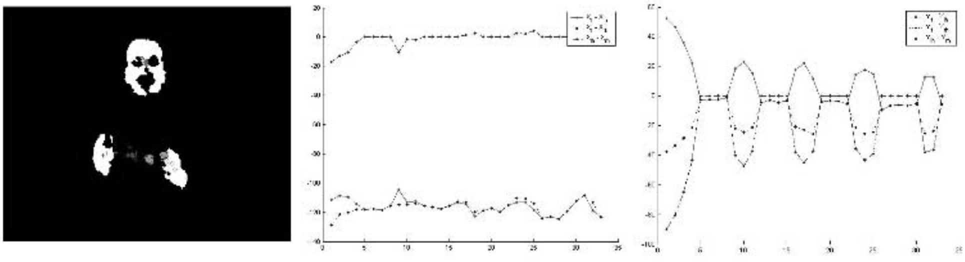
(a) (b) (c)

Figure 25. Italianate gesture.



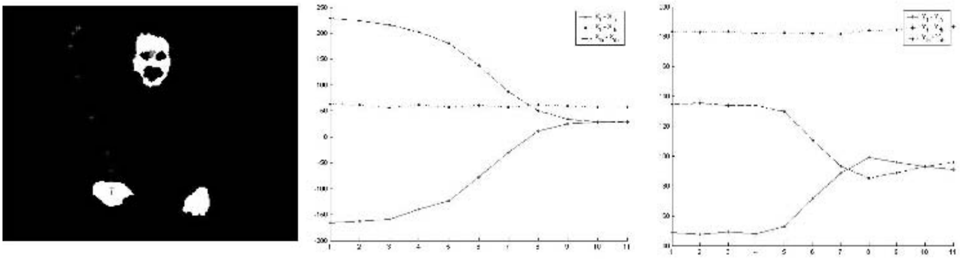
(a) (b) (c)

Figure 26. Hand clapping.



(a) *(b)* *(c)*

Figure 27. Lift of the hand.



(a) *(b)* *(c)*

Object centroids *crosses: left hand, circles: right hand, points: head*

Vertical object distances *dashes: X_{lh} - X_{rh},points: X_f - X_{rh}, line: X_f - X_{lh}Horizontal axis: framesVertical axis: pixels*

Horizontal object distances *dashes: Y_{lh}- Y_{rh},points: Y_f -Y_{rh}, line: Y_f -Y_{lh} Horizontal axis: frames Vertical axis: pixels*

Experiments for testing the recognizing performance of the proposed algorithm were also carried out. Gesture sequences of three male subjects, with maximum duration of three seconds, were captured by a typical web-camera at a rate of 10 frames per second. For each one of the gesture classes, 15 sequences were acquired: three were used for the initialization of the HMM parameters, seven for training and parameter re-estimation and five for testing. Each one of the training sequences consisted of approximately 15 frames. The selection of these frames was performed off-line so as to create characteristic examples of the gesture classes. Testing sequences were sub-sampled at a rate of five frames per second so as to enable substantial motion to occur. An overall recognition

Table 3. Gesture classification results.

Gesture Class	HC-LF	HC-HF	LH-LS	LH-HS	HoH-G	HoH-P	IG
Hand Clapping- Low Frequency (HC-LF)	5	0	0	0	0	0	0
Hand Clapping- High Frequency (HC-HF)	0	4	0	0	0	0	1
Lift of the Hand-Low Speed (LH-LS)	0	0	5	0	0	0	0
Lift of the Hand- High Speed (LH-HS)	0	0	0	5	0	0	0
Hands over the Head – Gesture (HoH-G)	0	0	0	0	5	0	0
Hands over the Head – Posture (HoH-P)	0	0	0	0	0	5	0
Italianate Gestures (IG)	0	1	0	0	0	0	4
Classification Rate (%)	100	80	100	100	100	100	80

rate of 94.3% was achieved. The experimental results are shown in the confusion matrix (Table 3).

From the results summarized in Table 3, we observe a mutual misclassification between “Italianate Gestures” (IG) and “Hand Clapping – High Frequency” (HC - HF). This is mainly due to the variations on “Italianate Gestures” across different individuals. Thus, training the HMM classifier on a personalized basis is anticipated to improve the discrimination between these two classes.

Multimodal Affective Analysis

Facial Expression Analysis Subsystem

The facial expression analysis subsystem is the main part of the presented system. Gestures are utilized to support the outcome of this subsystem.

Let us consider as input to the emotion analysis sub-system a 15-element length feature vector $\underline{f}$ that corresponds to the 15 features f_i shown in Table 1. The particular values of $\underline{f}$ can be rendered to FAP values as shown in the same table resulting in an input vector $\underline{G}$. The elements of $\underline{G}$ express the observed values of the correspondingly involved FAPs.

Expression profiles are also used to capture variations of FAPs (Raouzaiou, Tsapatsoulis, Karpouzis & Kollias, 2002). For example, the range of variations of FAPs for the expression "surprise" is shown in Table 4.

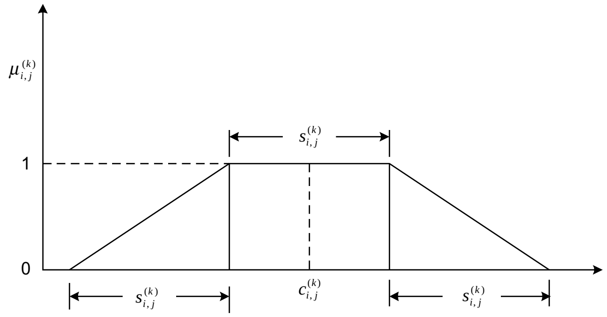
Let $X_{i,j}^{(k)}$ be the range of variation of FAP F_j involved in the *k-th* profile $P_i^{(k)}$ of emotion *i*. If $c_{i,j}^{(k)}$ and $s_{i,j}^{(k)}$ are the middle point and length of interval $X_{i,j}^{(k)}$ respectively, then we describe a fuzzy class $A_{i,j}^{(k)}$ for F_j, using the membership function $\mu_{i,j}^{(k)}$ shown in Figure 28. Let also $\Delta_{i,j}^{(k)}$ be the set of classes $A_{i,j}^{(k)}$ that correspond to profile $P_i^{(k)}$; the beliefs $p_i^{(k)}$ and b_i that an observed, through the vector $\underline{G}$, facial state corresponds to profile $P_i^{(k)}$ and emotion *i* respectively, are computed through the following equations:

$$p_i^{(k)} = \prod_{A_{i,j}^{(k)} \in \Delta_{i,j}^{(k)}} r_{i,j}^{(k)} \quad \text{and} \quad b_i = \max_k (p_i^{(k)}), \tag{4}$$

Table 4. Profiles for the archetypal emotion surprise.

Surprise ($P_{Su}^{(0)}$)	$F_3 \in [569,1201]$, $F_5 \in [340,746]$, $F_6 \in [-121,-43]$, $F_7 \in [-121,-43]$, $F_{19} \in [170,337]$, $F_{20} \in [171,333]$, $F_{21} \in [170,337]$, $F_{22} \in [171,333]$, $F_{31} \in [121,327]$, $F_{32} \in [114,308]$, $F_{33} \in [80,208]$, $F_{34} \in [80,204]$, $F_{35} \in [23,85]$, $F_{36} \in [23,85]$, $F_{53} \in [-121,-43]$, $F_{54} \in [-121,-43]$
$P_{Su}^{(1)}$	$F_3 \in [1150,1252]$, $F_5 \in [-792,-700]$, $F_6 \in [-141,-101]$, $F_7 \in [-141,-101]$, $F_{10} \in [-530,-470]$, $F_{11} \in [-530,-470]$, $F_{19} \in [-350,-324]$, $F_{20} \in [-346,-320]$, $F_{21} \in [-350,-324]$, $F_{22} \in [-346,-320]$, $F_{31} \in [314,340]$, $F_{32} \in [295,321]$, $F_{33} \in [195,221]$, $F_{34} \in [191,217]$, $F_{35} \in [72,98]$, $F_{36} \in [73,99]$, $F_{53} \in [-141,-101]$, $F_{54} \in [-141,-101]$
$P_{Su}^{(2)}$	$F_3 \in [834,936]$, $F_5 \in [-589,-497]$, $F_6 \in [-102,-62]$, $F_7 \in [-102,-62]$, $F_{10} \in [-380,-320]$, $F_{11} \in [-380,-320]$, $F_{19} \in [-267,-241]$, $F_{20} \in [-265,-239]$, $F_{21} \in [-267,-241]$, $F_{22} \in [-265,-239]$, $F_{31} \in [211,237]$, $F_{32} \in [198,224]$, $F_{33} \in [131,157]$, $F_{34} \in [129,155]$, $F_{35} \in [41,67]$, $F_{36} \in [42,68]$
$P_{Su}^{(3)}$	$F_3 \in [523,615]$, $F_5 \in [-386,-294]$, $F_6 \in [-63,-23]$, $F_7 \in [-63,-23]$, $F_{10} \in [-230,-170]$, $F_{11} \in [-230,-170]$, $F_{19} \in [-158,-184]$, $F_{20} \in [-158,-184]$, $F_{21} \in [-158,-184]$, $F_{22} \in [-158,-184]$, $F_{31} \in [108,134]$, $F_{32} \in [101,127]$, $F_{33} \in [67,93]$, $F_{34} \in [67,93]$, $F_{35} \in [10,36]$, $F_{36} \in [11,37]$

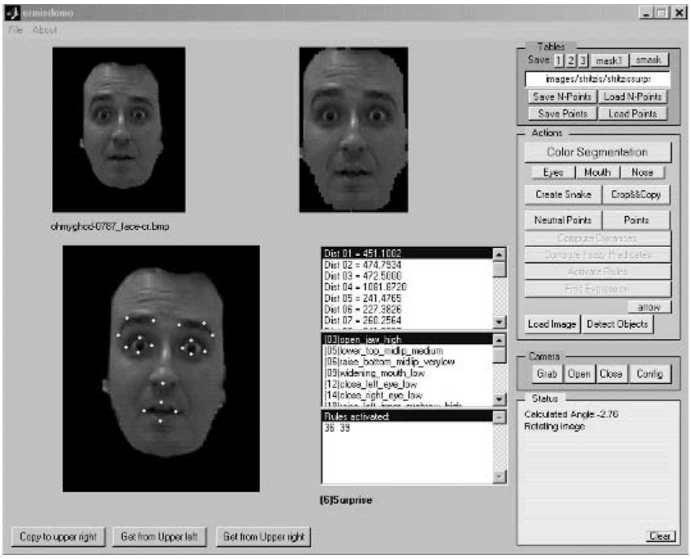
Figure 28. The form of membership functions.



where $r_{i,j}^{(k)} = \max\{g_i \cap A_{i,j}^{(k)}\}$ expresses the *relevance* $r_{i,j}^{(k)}$ of the *i*-th element of the input feature vector with respect to class $A_{i,j}^{(k)}$. Actually $\underline{g} = A'(\underline{G}) = \{g_1, g_2, ...\}$ is the fuzzified input vector resulting from a *singleton* fuzzification procedure (Klir & Yuan, 1995).

The various emotion profiles correspond to the fuzzy intersection of several sets and are implemented through a τ*-norm* of the form $t(a,b)=a\cdot b$. Similarly the belief that an observed feature vector corresponds to a particular emotion results from a fuzzy union of several sets through an σ*-norm* which is implemented as $u(a,b)=\max(a,b)$.

Figure 29. Facial expression analysis interface.



images/stritzis/stritzissurpr

ohmyghod-0787_face-cr.bmp

```
Dist 01 = 451.1002
Dist 02 = 474.7934
Dist 03 = 472.5000
Dist 04 = 1081.8720
Dist 05 = 241.4765
Dist 06 = 227.3826
Dist 07 = 260.2564
```

```
[03]open_jaw_high
[05]lower_top_midlip_medium
[06]raise_bottom_midlip_verylow
[09]widening_mouth_low
[12]close_left_eye_low
[14]close_right_eye_low
```

```
Rules activated:
36 39
```

[6]Surprise

Calculated Angle:-2.76
Rotating Image

An efficient implementation of the emotion analysis system has been developed in the framework of the IST ERMIS project (www.image.ntua.gr/ermis). In the system interface shown in Figure 29, one can observe an example of the calculated FP distances, the profiles selected by the facial expression analysis subsystem and the recognized emotion ("surprise").

Affective Gesture Analysis Subsystem

Gestures are utilized to support the outcome of the facial expression analysis subsystem, since in most cases they are too ambiguous to indicate a particular emotion. However, in a given context of interaction, some gestures are obviously associated with a particular expression — e.g., *hand clapping* of high frequency expresses *joy*, *satisfaction* — while others can provide indications for the kind of the emotion expressed by the user. In particular, quantitative features derived from hand tracking, like speed and amplitude of motion, fortify the position of an observed emotion; for example, *satisfaction* turns to *joy* or even to *exhilaration*, as the speed and amplitude of clapping increases.

As was mentioned in the section "Gesture analysis," the position of the centroids of the head and the hands over time forms the feature vector sequence that feeds an HMM classifier whose outputs corresponds to a particular gesture class. Table 5 below shows the correlation between some detectable gestures with the six archetypal expressions.

Given a particular context of interaction, gesture classes corresponding to the same emotional are combined in a "logical OR" form. Table 5 shows that a particular gesture may correspond to more than one gesture class carrying

Table 5. Correlation between gestures and emotional states.

Emotion	**Gesture Class**
Joy	*Hand clapping-high frequency*
Sadness	*Hands over the head-posture*
Anger	*Lift of the hand- high speed, italianate gestures*
Fear	*Hands over the head-gesture, italianate gestures*
Disgust	*Lift of the hand- low speed, hand clapping-low frequency*
Surprise	*Hands over the head-gesture*

different affective meaning. For example, if the examined gesture is *clapping*, detection of high frequency indicates *joy*, but a *clapping* of low frequency may express irony and can reinforce a possible detection of the facial expression *disgust*.

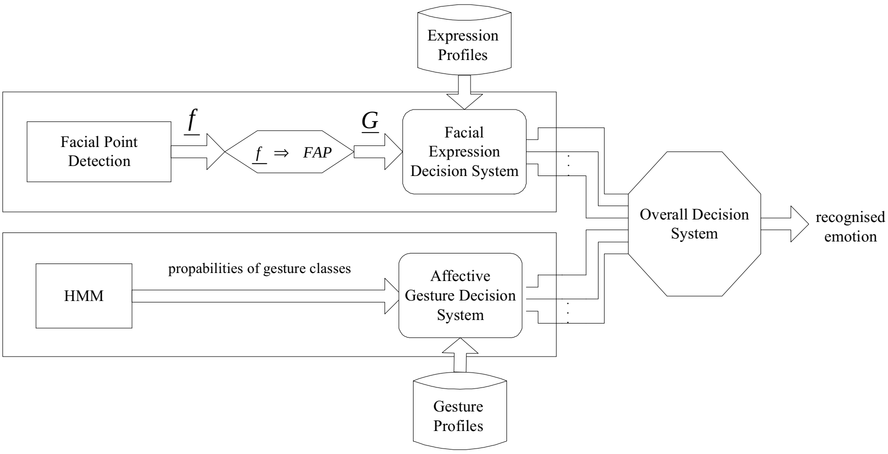
In practice, the gesture class probabilities derived by the HMM classifier are transformed to emotional state indicators by using the information of Table 5. Let EI_k be the emotional indicator of emotional state k (k ∈ {1,2,3,4,5,6} corresponds to one of the emotional states presented in Table 5 in the order of appearance, i.e., 1->Joy, 6->Surprise), $GCS = \{gc_1, gc_2, \ldots, gc_N\}$ be the set of gesture classes recognized by the HMM Classifier (N=7), $GCS^k \subseteq GCS$ be the set of gesture classes related with the emotional state k, and $p(gc_i)$ be the probability of gesture class gc_i obtained from the HMM Classifier. The *EI(k)* is computed using the following equation:

$$EI_k = \max_{gc_i \in GC^K} \{gc_i\} \tag{5}$$

The Overall Decision System

In the final step of the proposed system, the facial expression analysis subsystem and the affective gesture analysis subsystem are integrated, as shown in Figure 30, into a system which provides as a result the possible emotions of the user, each accompanied by a degree of belief.

Figure 30. Block diagram of the proposed scheme.



Although face is considered the main “demonstrator” of user’s emotion (Ekman & Friesen, 1975), the recognition of the accompanying gesture increases the confidence of the result of the facial expression subsystem. In the current implementation, the two subsystems are combined as a weighted sum: Let b_k be the degree of belief that the observed sequence presents the *k-th* emotional state, obtained from the facial expression analysis subsystem, and EI_k be the corresponding emotional state indicator, obtained from the affective gesture analysis subsystem, then the overall degree of belief d_k is given by:

$$d_k = w_1 \cdot b_k + w_2 \cdot EI_k \qquad (6)$$

where the weights w_1 and w_2 are used to account for the reliability of the two subsystems as far as the emotional state estimation is concerned. In this implementation we use w_1 =0.75 and w_2 =0.25. These values enable the affective gesture analysis subsystem to be important in cases where the facial expression analysis subsystem produces ambiguous results, while at the same time leave the latter subsystem to be the main contributing part in the overall decision system.

For the input sequence shown in Figure 3, the affective gesture analysis subsystem consistently provided a “surprise” selection. This was used to fortify the output of the facial analysis subsystem, which was around 85%.

Conclusions – Future Work

In this chapter, we described a holistic approach to emotion modeling and analysis and their applications in MMI applications. Beginning from a symbolic representation of human emotions found in this context, based on their expression via facial expressions and hand gestures, we show that it is possible to transform quantitative feature information from video sequences to an estimation of a user’s emotional state. This transformation is based on a fuzzy rules architecture that takes into account knowledge of emotion representation and the intrinsic characteristics of human expression. Input to these rules consists of features extracted and tracked from the input data, i.e., facial features and hand movement. While these features can be used for simple representation purposes, e.g., animation or task-based interfacing, our approach is closer to the target of affective computing. Thus, they are utilized to provide feedback on the user’s emotional state while in front of a computer.

Future work in the affective modeling area includes the enrichment of the gesture vocabulary with more affective gestures and feature-based descriptions. With respect to the recognition part, more sophisticated methods of combination of detected expressions and gestures, mainly through a rule-based system, are currently under investigation, along with algorithms that take into account general body posture information.

References

Bassili, J. N. (1979). Emotion recognition: The role of facial movement and the relative importance of upper and lower areas of the face. *Journal of Personality and Social Psychology*, 37, 2049-2059.

Bregler, C. (1997). Learning and recognition human dynamics in video sequences. *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 568-574.

Cowie, R. et al. (2001). Emotion Recognition in Human-Computer Interaction. *IEEE Signal Processing Magazine,* 1, 32-80.

Darrell, T. & Pentland, A. (1996). Active gesture recognition using partially observable Markov decision processes. In *Proc. IEEE Int. Conf. Pattern Recognition,* 3, 984-988.

Darrell, T., Essa, I. & Pentland, A. (1996). Task-Specific Gesture Analysis in Real-Time Using Interpolated Views. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(12), 1,236-1,242.

Davis, M. & College, H. (1975). *Recognition of Facial Expressions*. New York: Arno Press.

Ekman, P. & Friesen, W. (1975). *Unmasking the Face*. New York: Prentice-Hall.

Ekman, P. & Friesen, W. (1978). *The Facial Action Coding System*. San Francisco, CA: Consulting Psychologists Press.

Faigin, G. (1990). *The Artist's Complete Guide to Facial Expressions*. New York: Watson-Guptill.

Freeman, W. T. & Weissman, C. D. (1995). Television Control by Hand Gestures. *Proc. Int'l Workshop on Automatic Face and Gesture Recognition*, Switzerland, 179-183.

Karpouzis, K., Tsapatsoulis N. & Kollias, S. (2000). Moving to Continuous Facial Expression Space using the MPEG-4 Facial Definition Parameter (FDP) Set. *Proc. of SPIE Electronic Imaging 2000,* San Jose, CA, USA.

Kjeldsen, R. & Kender, J. (1996). Finding skin in color images. *Proc. 2nd Int. Conf. Automatic Face and Gesture Recognition*, 312-317.

Klir, G. & Yuan, B. (1995). *Fuzzy Sets and Fuzzy Logic, Theory and Application.* New Jersey: Prentice-Hall.

Picard, R. W. (2000). *Affective Computing.* Cambridge, MA: MIT Press.

Plutchik, R. (1980). *Emotion: A psychoevolutionary synthesis.* New York: Harper and Row.

Raouzaiou, A., Tsapatsoulis, N., Karpouzis, K. & Kollias, S. (2002). Parameterized facial expression synthesis based on MPEG-4. *EURASIP Journal on Applied Signal Processing*, 10, 1021-1038.

Sharma, R., Huang, T. S. & Pavlovic, V. I. (1996). A Multimodal Framework for Interacting with Virtual Environments. In C. A. Ntuen and E. H. Park (Eds), *Human Interaction With Complex Systems.* Kluwer Academic Publishers.

Tekalp, M. & Ostermann, J. (2000). Face and 2-D mesh animation in MPEG-4. *Image Communication Journal*, 15(4-5), 387-421.

Votsis, G., Drosopoulos, A. & Kollias, S. (2003). A modular approach to facial feature segmentation on real sequences. *Signal Processing, Image Communication,* 18, 67-89.

Whissel, C. M. (1989). The dictionary of affect in language. In R. Plutchnik & H. Kellerman (Eds), *Emotion: Theory, research and experience: volume 4, The measurement of emotions*. New York: Academic Press.

Wren, C., Azarbayejani, A., Darrel, T. & Pentland, A. (1997). Pfinder: Real-time tracking of the human body. *IEEE Trans. Pattern Anal. Machine Intell.*, 9(7), 780-785.

Wu, Y. & Huang, T. S. (2001). Hand modeling, analysis, and recognition for vision-based human computer interaction. *IEEE Signal Processing Magazine*, 18(3), 51-60.

Chapter VI

Techniques for Face Motion & Expression Analysis on Monocular Images

Ana C. Andrés del Valle
Institut Eurécom, France

Jean-Luc Dugelay
Institut Eurécom, France

Abstract

This chapter presents a state-of-the-art compilation on facial motion and expression analysis. The core of the chapter includes the description and comparison of methods currently being developed and tested to generate face animation from monocular static images and/or video sequences. These methods are categorized into three major groups: "those that retrieve emotion information," "those that obtain parameters related to the Face Animation synthesis used," and "those that use explicit face synthesis during the image analysis." A general overview about the processing fundamentals involved in facial analysis is also provided. Readers will have a clear understanding of the ongoing research performed in the field *of facial expression and motion analysis on monocular images by easily finding the right references to the detailed description of all mentioned methods.*

Introduction

Researchers from the Computer Vision, Computer Graphics and Image Processing communities have been studying the problems associated with the analysis and synthesis of faces in motion for more than 20 years. The analysis and synthesis techniques being developed can be useful for the definition of low-rate bit image compression algorithms (model-based coding), new cinema technologies, as well as for the deployment of virtual reality applications, videoconferencing, etc. As computers evolve towards becoming more human-oriented machines, human-computer interfaces, behavior-learning robots and disable-adapted computer environments will use face expression analysis to be able to react to human action. The *analysis of motion and expression from monocular (single) images* is widely investigated because non-stereoscopic static images and videos are the most affordable and extensively used visual media (i.e., webcams).

This chapter reviews current techniques for the analysis of single images to derive face animation. These methods can be classified based upon different criteria:

1. the nature of the analysis: global versus feature-based, real-time oriented;
2. the complexity of the information retrieved: general expression generation versus specific face motion;
3. the tools utilized during the analysis: for instance, the cooperation of a 3D head model;
4. the degree of realism obtained from the Face Animation (FA) synthesis; and
5. the environmental conditions during the analysis: controlled or uniform lighting, head-pose dependence or not.

Table 1 depicts a rough evaluation of the techniques that we review in this chapter by comparing these criteria, considering the data provided by the referenced articles, books and other bibliographical material, as well as the judgment of the authors.

The analysis algorithms presented include those most related to face motion and expression understanding. Specific image processing can also be used to locate faces on images, for face recognition intended for biometrics, for general head tracking and pose deduction, as well as for face animation synthesis. For those readers acquainted mainly with 3D and graphics, we provide a brief overview of the most common image processing methods and mathematical tools involved, pointing to some sources for the algorithmic details that will not be explained or will be assumed to be known during the description of the state-of-the–art approaches.

The core of the chapter includes the description of the methods currently being developed and tested to generate face animation from real face images. The techniques herein discussed analyze static images and/or video sequences to obtain general face expressions or explicit face motion parameters. We have categorized these methods in three groups: "those that retrieve emotion information," "those that obtain parameters related to the Face Animation synthesis used," and "those that use explicit face synthesis during image analysis."

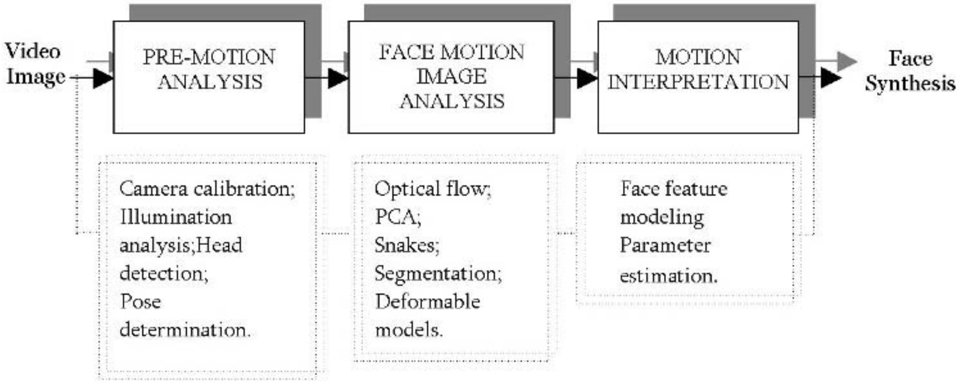
Background

Many video encoders do motion analysis over video sequences to search for motion information that will help compression. The concept of motion vectors, first conceived at the time of the development of the first video coding techniques, is intimately related to motion analysis. These first analysis techniques help to regenerate video sequences as the exact or approximate reproduction of the original frames by using motion compensation from neighboring pictures. They are able to compensate for, but not to understand the actions of the objects moving on the video and, therefore, they cannot restore the object's movements from a different orientation. Faces play an essential role in human communication. Consequently, they have been the first objects whose motion has been studied in order to recreate animation on synthesized models or to interpret motion for *a posteriori* use.

Synthetic faces are classified into two major groups: avatars and clones. Generally, avatars are a rough and symbolic representation of the person, and their animation is speaker independent because it follows generic rules disregarding the individual that they personify. Clones are more realistic and their animation takes into account the nature of the person and his real movements. Whether we want to animate avatars or clones, we face a great challenge: the automatic generation of face animation data. Manually generated animation has long been used to create completely virtual characters and has also been applied to animate avatars. Nevertheless, many computer applications require real-time and easy-to-use face animation parameter generation, which means that the first solutions developed using motion capture equipment prove to be too tedious for many practical purposes. Most applications utilizing Talking Heads aim at telecommunication uses. In such a context, real-time capabilities and low computing cost for both analysis and synthesis are required. Current trends in research tend to use speech analysis or synthesized speech from text as a source of real-time animation data. Although these techniques are strong enough to generate parameters to be used by avatars, they cannot provide realistic data for face animation.

To obtain realistic and natural 3D Face Animation (FA), we need to study and understand the complete human face behavior and those image-based methods that are cost-flexible techniques for face movement understanding. In this chapter we present the latest and most effective systems to analyze face expression over monocular images to generate facial animation to restitute speaker-dependent face motion on 3D face models. Figure 1 represents the basic flowchart for systems dedicated to facial expression and motion analysis on monocular images. Video or still images are first analyzed to detect, control and deduce the face location on the image and the environmental conditions under which the analysis will be made (head pose, lighting conditions, face occlusions, etc.). Then, some image motion and expression analysis algorithms extract specific data, which is finally interpreted to generate face motion synthesis.

Figure 1. Image input is analyzed in the search for the face general characteristics: global motion, lighting, etc. At this point, some image processing is performed to obtain useful data that can be interpreted afterwards to obtain face animation synthesis.


Video Image
PRE-MOTION ANALYSIS
FACE MOTION IMAGE ANALYSIS
MOTION INTERPRETATION
Face Synthesis
Camera calibration; Illumination analysis;Head detection; Pose determination.
Optical flow; PCA; Snakes; Segmentation; Deformable models.
Face feature modeling Parameter estimation.

Each of the modules may be more or less complex depending on the purpose of the analysis (i.e., from the understanding of general behavior to exact 3D-motion extraction). If the analysis is intended for later face expression animation, the type of FA synthesis often determines the methodology used during expression analysis. Some systems may not go through either the first or the last stages or some others may blend these stages in the main *motion & expression image analysis*. Systems lacking the *pre-motion analysis* step are most likely to be limited by environmental constraints, like special lighting conditions or pre-determined head pose. Those systems that do not perform *motion interpretation* do not focus on delivering any information to perform face animation synthesis afterwards. A system that is thought to analyze video to generate face animation data in a robust and efficient way needs to develop all three modules. The approaches currently under research and that will be exposed in this chapter clearly perform the *facial motion & expression image analysis* and to some extent the *motion interpretation* to be able to animate 3D models. Nevertheless, many of them fail to have a strong *pre-motion analysis* step to ensure some robustness during the subsequent analysis.

Processing Fundamentals

Pre-Processing Techniques

The conditions under which the user may be recorded are susceptible to change from one determined moment to the next one. Some changes may come from the hardware equipment used, for instance, the camera, the lighting environment, etc. Furthermore, although only one camera is used, we cannot presuppose that the speaker's head will remain motionless and looking straight into the camera at any time. Therefore, pre-processing techniques must help to homogenize the analysis conditions before studying non-rigid face motion.

Camera calibration

Accurate motion retrieval is highly dependent on the precision of the image data we analyze. Images recorded by a camera undergo different visual deformations due to the nature of the acquisition material. Camera calibration can be seen as the starting point of a precise analysis.

If we want to express motion in real space, we must relate the motion measured in terms of pixel coordinates to the real/virtual world coordinates. That is, we need to relate the world reference frame to the image reference frame. Simply knowing the pixel separation in an image does not allow us to determine the distance of those points in the real world. We must derive some equations to link the world reference frame to the image reference frame in order to find the relationship between the coordinates of points in 3D-space and the coordinates of the points in the image. We introduce the camera reference frame because there is no direct relation between the previously mentioned reference frames. Then, we can find an equation linking the camera reference frame with the image reference frame (LinkI), and another equation linking the world reference frame with the camera reference frame (LinkE). Identifying LinkI and LinkE is equivalent to finding the camera's characteristics, also known as the camera's extrinsic and intrinsic parameters.

Many calibration techniques exist that have been reported in the past two decades. The developed methods can be roughly classified into two groups: photogrammetic calibration and self-calibration. We refer the reader to Zhang (2000) and Luong and Faugeras (1997) to obtain examples and more details about these approaches.

Illumination analysis and compensation

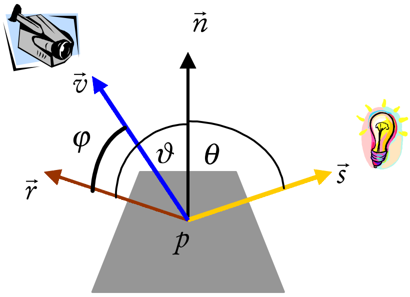
Other unknown parameters during face analysis are the lighting characteristics of the environment in which the user is being filmed. The number, origin, nature and intensity of the light sources of the scene can significantly transform the appearance of a face. Face reflectance is not uniform all over the face and, thus, is very difficult to model.

There are two major categories of reflected light:

1. Diffuse Reradiation (scattering): this occurs when the incident light penetrates the surface and is reflected equally in all directions.
2. Specular Reflection: light does not penetrate the object, but it is instead directly reflected from its outer surface.

The intensity of the pixels that we get from the image of the face is the result of the light from the recorded scene (i.e., the face) scattered towards the camera lens. The nature of the reflection phenomenon requires the knowledge of some vector magnitudes (Figure 2):

Figure 2. The reflected light that reaches the camera lens depends on the direction of the normal to the surface ($\vec{n}$), the vector from the studied point to the light source ($\vec{s}$) and the vector from the point to the camera lens ($\vec{v}$). $\vartheta = \theta$ for perfectly specular reflections. φ is the angular difference between the reflected beam and the camera perspective towards the object.



- the normal $\vec{n}$ to the surface at the point p being studied;
- the vector $\vec{v}$ from p to the camera; and
- the vector $\vec{s}$ from p to the light source.

Due to the difficulty of deducing the great number of parameters and variables, one common hypothesis usually taken is to consider faces as lambertian surfaces (only reflecting diffuse light), so as to reduce the complexity of the illumination model. Luong, Fua and Leclerc (2002) studied the light conditions of faces to be able to obtain texture images for realistic head synthesis from video sequences under this hypothesis. Other reflectance models are also used (Debevec et al., 2000), although they focus more on reproducing natural lighting on synthetic surfaces than on understanding the consequences of the lighting on the surface, itself. In most cases, the analysis of motion and expressions on faces is more concerned with the effect of illumination on the facial surface studied than with the overall understanding of the lighting characteristics. A fairly extended approach to appreciate the result of lighting on faces is to analyze illumination by trying to synthetically reproduce it on the realistic 3D-model of the user's head. Phong's reflection model is the 3D shading model most heavily used to assign shades to each individual pixel of the synthetic face. It is characterized by simplifying second-order reflections, introducing an ambient reflection term that simulates the sparse (diffuse) reflection coming from sources whose light has been so dispersed that it is very difficult to determine its origin. Whether the lighting synthesis is used to compensate the image input (Eisert & Girod, 2002) or to lighten the synthesized model used to help the analysis (Valente & Dugelay, 2001), it proves to be reasonable to control how the lighting modifies the aspect of the face on the image.

Head detection and pose determination

If we intend to perform robust expression and face motion analysis, it is important to control the location of the face on the image plane. It is also crucial to know which orientation the face has with regard to the camera. The find-a-face problem is generally reduced to the detection of its skin on the image. The most generalized methods for skin detection use a probabilistic approach where the colorimetric characteristics of human skin are taken into account. First, a probabilistic density function — $P(rgb|skin)$ — is usually generated for a given space color (RGB, YUV, HSV, or others). $P(rgb|skin)$ indicates which is the probability of belonging to the skin surface. It is difficult to create this function, as well as to decide which will be the threshold to use to determine if the studied pixel belongs to the skin or not. In some approaches (Jones & Rehg, 1999), researchers study in detail the color models used and also give a probability function for the pixels that do not belong to the skin — $P(rgb|skin)$. Others, like the one presented by Sahbi, Geman and Boujemaa (2002), perform their detection in different stages, giving more refinement at each step of the process. More complex algorithms (Garcia & Tziritas, 1999) allow regions with non-homogeneous skin color characteristics to be found.

Determining the exact orientation of the head becomes a more complicated task. In general, we find two different ways to derive the head pose: either using static methods or using dynamic approaches. Static methods search for specific features of the face (eyes, lip corners, nostrils, etc.) on a frame-by-frame basis, and determine the user's head orientation by finding the correspondences between the projected coordinates of these features and the real world coordinates. They may use template-matching techniques to find the specific features, as Nikolaidis and Pitas (2000) do. This method works fine, although it requires very accurate spotting of the relevant features. Unfortunately, this action has to be redone at each frame and it is somewhat tedious and imprecise. Another possibility is to use 3D-data, for instance, from a generic 3D-head model, to accurately determine the pose of the head on the image. This is the solution given by Shimizu, Zhang, Akamatsu and Deguchi (1998).

To introduce time considerations by taking advantage of previous results, dynamic methods have been developed. These methods perform face tracking by analyzing video sequences as a more or less smooth sequence of frames.

They use the pose information retrieved from one frame to analyze and derive the pose information on the next one. One of the most extended techniques involves the use of Kalman filters to predict analytical data, as well as the pose parameters themselves. We refer the reader to other research (Ström, Jebara, Basu & Pentland, 1999; Valente & Dugelay, 2001; Cordea, E. M. Petriu, Georganas, D. C. Petriu & Whalen, 2001) to find related algorithmic details.

Image Processing Algorithms

The complexity of expression analysis is usually simplified by trying to understand either the shape of some parts of the face, the location of very specific points or the change in magnitude of some characteristic of the area analyzed, for example, its color. In order to do this, several image-processing techniques are used and tuned to work on human faces. In this section, we try to summarize the basics of the most common techniques utilized.

Optical flow

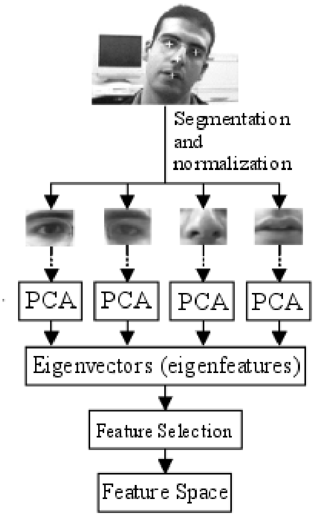
The field of displacement vectors of the objects that compose a scene cannot be computed directly: we can just find the apparent local motion, also called optical flow, between two images.

There are two major methods to estimate the optical flow: either we match objects with no ambiguity from image to image, or we calculate the image gradients between frames. In the first case, the main goal consists in determining in one of the studied images the group of points that can be related to their homologues in the second image, thus giving out the displacement vectors. The most difficult part of this approach is the selection of the points, or regions, to be matched. In general, the biggest disadvantage of this kind of method is that it determines motion in a discrete manner and motion information is only precise for some of the pixels on the image.

The second technique, the gradient-descent method, generates a more dense optical flow map, providing information at the pixel level. It is based on the supposition that the intensity of a pixel $I(x, y, t)$ is constant on two consequent frames, and that its displacement is relatively small. In these circumstances we verify:

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0, \qquad (1)$$

Figure 3. For the creation of the eigenfeature database, several images of the studied features are segmented, then normalized and finally analyzed using Principal Component techniques. Diagram courtesy of the Instituto de Matemática e Estatística at the Universidade de São Paulo.



where $u = \frac{\partial x}{\partial t}$ and $v = \frac{\partial y}{\partial t}$ are the pixel displacements between two images. Each point on the image has one equation with two unknowns, *u* and *v*, which implies that motion cannot be directly computed. There exist different methods that try to solve (1) iteratively.

A complete bibliographical compilation of different optical flow methods can be found in Wiskott (2001).

Principal component analysis — Eigen-decomposition

Optical flow methods are extensively used in shape recognition, but they do not perform well in the presence of noise. If we want to identify a more general class of objects, it is convenient to take into account the probabilistic nature of the object appearance and, thus, to work with the class distribution in a parametric and compact way.

The Karhunen-Loève Transform meets the requirements needed to do so. Its base functions are the eigenvectors of the covariance matrix of the class being modeled:

$$\Lambda = \Phi^T \Sigma \Phi , \tag{2}$$

being Σ the covariance matrix, Λ the diagonal matrix of eigenvalues and Φ the matrix of eigenvectors. The vector base obtained is optimal in terms of compactness (we can easily isolate vectors of low energy) and parametric (each eigenvector is orthogonal to the others, creating a parametric eigenspace).

Elements of one class, that is, a vector whose dimension is M, can be represented by the linear combination of the M *eigenvectors* obtained for this class. The Principal Component Analysis (PCA) technique states that the same object can be reconstructed by only combining the N<M eigenvectors of greatest energy, also called principal components. It also says that we will minimize the error difference when performing the approximation if the linear coefficients for the combination are obtained from projecting the class vector onto the sub-space of principal components.

This theory is only applicable to objects that can be represented by vectors. Images have this property, therefore, this theory is easily extended to image processing and generally used to model the variability of 2D objects on images like, for example, faces.

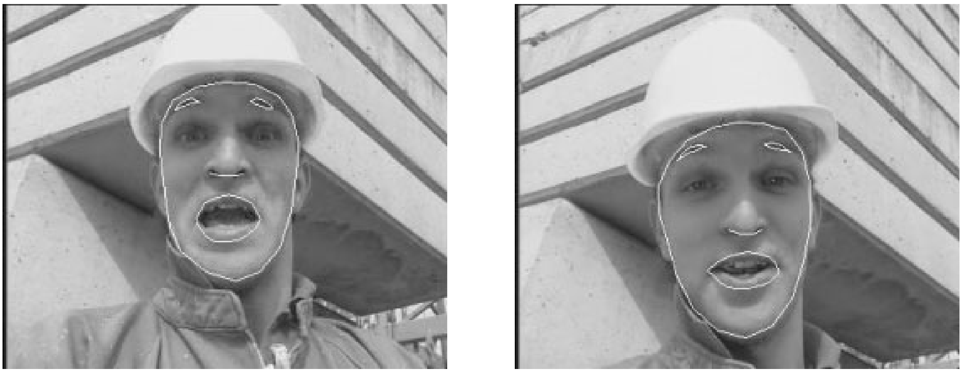
Very often PCA is utilized to analyze and identify features of the face. It introduces some restrictions. One of them is the need for one training stage previous to the analysis, during which the base of principal component vectors, in this case images, must be generated. It also forces all images being analyzed to be the same size. Using PCA in face analysis has lead to the appearance of concepts like Eigenfaces (Turk & Pentland, 1991), utilized for face recognition, or Eigenfeatures (Pentland, Mohaddam & Starner, 1994) used to study more concrete areas of faces robustly.

The book *Face Image Analysis by Unsupervised Learning* (Bartlett, 2001) is a complete study of the strengths and weaknesses of methods based on Independent Component Analysis (ICA) in contrast with PCA. It also includes a full explanation of concepts like Eigenactions and describes recent approaches in facial image analysis.

Active contour models — Snakes

Active contour models, generally called snakes, are geometric curves that approximate the contours of an image by minimizing an energy function. Snakes are used to track moving contours within video sequences because they have the property of deforming themselves to stick onto a contour that evolves along the time.

Figure 4. By using snakes, face and feature contours are tracked on each frame of the sequence. Images courtesy of the Image Processing Group at the Universitat Politècnica de Catalunya.



In general, the energy function can be decomposed into two terms, an internal energy and an external energy:

$$E_{total} = E_{\text{int}} + E_{ext} \,. \qquad (3)$$

The role of the external energy is to attract the point of the snake towards the image contours. The internal energy tries to ensure certain regularity on the snake while E_{ext} acts, from a spatial as well as from a temporal perspective. Once the energy function is defined, we use an iterative process to find its minimum. We can understand the minimum energy point as the equilibrium position of a dynamic system submitted to the forces derived from the energy functions.

Mathematical morphology — Edge detection & segmentation

When analyzing images of faces under unconstrained conditions, classical image filtering techniques may not be robust enough to extract all the information from them.

Mathematical morphology appeared as an alternative mathematical tool to process an image from a visual perspective, instead of from a numerical one. The techniques for mathematical morphology are based on set-theoretic concepts and non-linear superposition of signals and images. Morphological operations have been applied successfully to a wide range of problems including image processing, analysis tasks, noise suppression, feature extraction, pattern recognition, etc. In Serra (1982, 1988), the authors explain in depth how to take advantage of these techniques for the processing of images. This set of tools gives the means to develop algorithms to efficiently detect edges and specific areas of the face.

Deformable models

A deformable model is a group of parametric curves with which we try to approximate the contours of an image and the behavior of the objects present on it. The advantages of a deformable template are its computational simplicity and the few number of parameters needed to describe different shapes. Unfortunately, since a template is generally made specifically for a given shape, we need to redefine the rules of parameter variation so that the model follows the right contours. Since they have a difficult adaptation to unexpected shapes, their biggest disadvantage is dealing with noisy images. The diversification of solutions is well seen in the literature, where we can find as many different models as articles treating the subject (Yuille, 1991). Some of the most common models are:

- Elliptical: circles and ellipsoids can model the eyes (Holbert & Dugelay, 1995).
- Quadratic: parabolic curves are often used to model the lips (Leroy & Herlin, 1995).
- Splines: to develop more complex models, splines are an option. They have already been used to characterize mouth expressions (Moses, Reynard & Blake, 1995).

Post-Processing Techniques and Their Related Mathematical Tools

To recreate motion on synthesized 3D-models, it is necessary to relate the analyzed information to the Facial Action Units (AUs) or Facial Animation Parameters (FAPs). If motion is not derived heuristically from the image processing results themselves, the derivation of motion is sometimes helped by the iterative feedback synthesis of the motion actions on the model. As explained by Eisert and Girod (1998), we must find some mathematical solution to tie analysis to synthesis.

Motion modeling of facial features

To extract motion information from specific features of the face (eyes, eyebrows, lips, etc.), we must know the animation semantics of the FA system that will synthesize the motion. Deformable models, such as snakes, deliver information about the feature in the form of the magnitudes of the parameters that control the analysis. It is also necessary to relate these parameters to the actions that we must apply to the 3D-model to recreate motion and expressions. If there are many different image-processing techniques to analyze face features, there are at least as many corresponding feature motion models. These motion models translate the results into face animation parameters.

Malciu and Prêteux (2001) track face features using snakes. Their snakes are at the same time deformable models containing the Facial Definition Parameters (FDPs) defined on the MPEG-4 standard (MPEG-4, 2000). Their technique is capable of tracking FDPs very efficiently, but it does not give out the FAPs that would animate the model to generate the observed feature motion. Chou, Chang and Chen (2001) go one step further. They present an analysis technique that searches for the points belonging to the projection of a simple 3D-model of the lips, also containing the FDPs. From the projected location they derive the FAPs that operate on them to generate the studied motion. Since one FAP may act on more than one point belonging to their lip model, they use a least-square solution to solve for the magnitudes of the FAPs involved. Goto, Kshirsagar and Magnenat-Thalmann (1999) use a simpler approach where image processing is reduced to the search of edges and the mapping of the obtained data is done in terms of motion interpretation: open mouth, close mouth, half-opened mouth, etc. The magnitude of the motion is related to the location of the edges. They extend this technique to eyes, developing their own eye motion model. Similarly, eyebrows are tracked on the image and associated to model actions.

Estimators

Once facial expressions are visually modeled by some image processing technique, we obtain a set of parameters. The mapping of these parameters onto the corresponding face animation parameters is done by solving for the estimator that relates face motion parameters to analysis parameters. To establish the mapping relationship there must be a training process. Among others we find the following estimators: linear, neural networks and RBF networks. We will describe the first two in detail.

Linear

Let us call $\vec{\lambda}$ the vector of parameters obtained from the image analysis and $\vec{\mu}$ the vector of FA parameters for the synthesis observed by $\vec{\lambda}$. The usual way to construct the linear estimator *L*, which best satisfies $\vec{\mu} = L \cdot \vec{\lambda}$ on the training database, is to find a solution in the least square sense. We verify that this linear estimator is given by

$$L = \mathrm{M}\Lambda^T (\Lambda\Lambda^T)^{-1} \tag{4}$$

where $\mathrm{M} = [\vec{\mu}_1 | \ldots | \vec{\mu}_d]$ and $\Lambda = [\vec{\lambda}_1 | \ldots | \vec{\lambda}_d]$ are the matrices obtained by concatenating all $\vec{\mu}$ and $\vec{\lambda}$ vectors from the training set.

Valente, Andrés del Valle and Dugelay (2001) compare the use of a linear estimator against an RBF (Radial Basis Functions) network estimator. In their experiments, $\vec{\lambda}$ are the set of the coefficients obtained from projecting an image of the feature being analyzed (*imagette*) onto a PCA imagette database of the feature recorded making different expressions under different lighting conditions. $\vec{\mu}$ contains the actions to apply on the model, in form of AUs, to generate these different expressions. RBF networks find the relationship between a pair of examples (input and output) of different dimensions, through the combination of functions of simple variables whose main characteristic is that they are continuous in $\mathfrak{R}^+$ and radial (Poggio & Girosi, 1990).

Neural networks

Neural networks are algorithms inspired on the processing structures of the brain. They allow computers to learn a task from examples. Neural networks are typically organized in layers. Layers are made up of a number of interconnected "nodes," which contain an "activation function." (See Figure 5a.)

Most artificial neural networks, or ANNs, contain some form of *learning rule* that modifies the weights of the connections according to the input patterns that it is presented with. The most extensively used rule is the *delta rule*. It is utilized in the most common class of ANNs called *backpropagational neural networks* (BPNNs). Backpropagation is an abbreviation for the backwards propagation of error.

ANNs complement image-processing techniques that need to *understand* images and in analysis scenarios where some previous training is permitted. In Tian, Kanade and Cohn (2001), we find one fine example of the help neural

Figure 5a. Patterns are presented to the network via the "input layer," which communicates to one or more "hidden layers" where the actual processing is done via a system of weighted "connections." The hidden layers then link to an "output layer" where the answer is output, as shown in the graphic below.

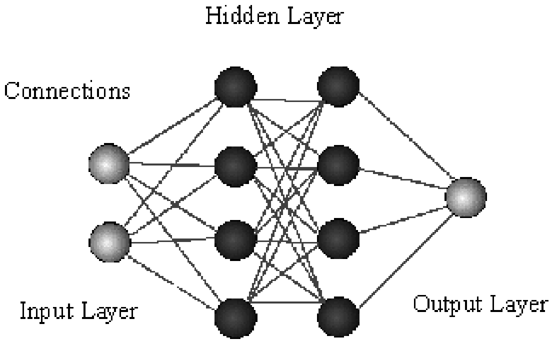
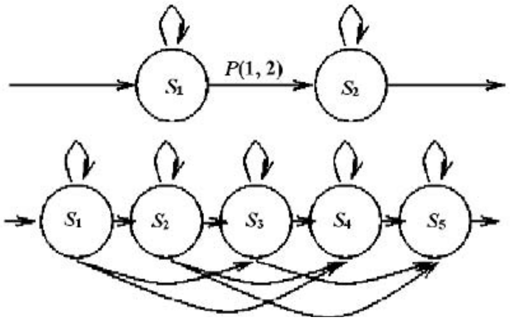


Figure 5b. Top: A typical illustration of a two state HMM. Circles represent states with associated observation probabilities, and arrows represent non-zero transition arcs, with associated probability. Bottom: This is an illustration of a five state HMM. The arcs under the state circles model the possibility that some states may be skipped.



networks can provide. In this article, Tian et al. explain how they have developed the Automatic Face Analysis to analyze facial expressions. Their system takes as input the detailed parametric description of the face features they analyze. They use neural networks to convert these data into AUs following the motion semantics of the Facial Action Coding System (FACS). A similar approach, aimed at analyzing spontaneous facial behavior, is taken by Bartlett et al. (2001). Their system also uses neural networks to describe face expressions in terms of

AUs. These two approaches differ in the image processing techniques and parameters they use to describe the image characteristics introduced as input to the neural network.

A model for motion — Hidden Markov models

By collecting data from real human motion, we can model behavior patterns as statistical densities over configuration space. Different configurations have different observation probabilities. One very simple behavior model is the Gaussian Mixture Model (GMM), in which the probability distribution is modeled as a collection of Gaussians. In this case the composite density is described by:

$$\sum_{k=1}^{N} P_k \cdot \Pr(O|\lambda = k) \tag{5}$$

where P_k is the observed prior probability of sub-model k. The mixture model represents a clustering of data into regions within the observation space. Since human motion evolves over time, in a complex way, it is advantageous to explicitly model temporal dependence and internal states. A hidden Markov model is one way to do this, and has been shown to perform quite well recognizing human motion. Figure 5b illustrates their graphical representation.

Hidden Markov models (HMM) are a powerful modern statistical technique. A Markov process not only involves probability, but also depends on the “memory” of the system being modeled. An HMM consists of several states. In the formulation of HMMs, each state is referred to individually, and thus practical and feasible examples of these models have a small number of states. In an HMM, a system has a number of states $S_1 \dots S_n$. The probability that the system passes from state i to state j is called $P(i, j)$. The states of the system are not known, but the system does have one observable parameter on output, which has m possible values from 1 to m. For the system in state i, the probability that output value v will be produced is called $O(i, v)$. We must point out that it is required that the transition probabilities depend on the state, not the output.

We refer the reader to the tutorial on HMMs by Rabiner (1989), where theoretical bases are further discussed and examples of the most common applications can be found. In Metaxas (1999), the author presents a framework to estimate human motion (including facial movements) where the traditional use of HMMs is modified to ensure reliable recognition of gesture. More specifically, Pardàs and Bonafonte (2002) use an HMM to deduce the expression of faces on video sequences. In their work, they introduce the concept of high-level/low-level analysis. In their approach, the high-level analysis structure takes as input the FAP produced by the low-level analysis tool and, by means of an HMM classifier, detects the facial expression on the frame.

Fuzzy systems

Fuzzy systems are an alternative to traditional notions of set membership and logic. The notion central to fuzzy systems is that true values (in fuzzy logic) or membership values (in fuzzy sets) are indicated by a value on the range [0.0, 1.0], with 0.0 representing the absolute Falseness and 1.0 representing absolute Truth. This is a new approach to the binary set 0 (False) — 1 (True) used by classical logic. Fuzzy systems try to gather mathematical tools to represent natural language, where the concepts of True and False are too extreme and intermediate or more vague interpretations are needed.

Apart from the basic operations among sets, fuzzy systems permit the definition of "hedges," or modifiers of fuzzy values. These operations are provided in an effort to maintain close ties to natural language, and to allow for the generation of fuzzy statements through mathematical calculations. As such, the initial definition of hedges and operations upon them is quite a subjective process and may vary from one application to another. Hedges mathematically model concepts such as "very," "somewhat," "sort of," and so on.

In many applications fuzzy systems appear as a complement to the image processing involved; they help in the decision-making process needed to evaluate results from analyzed images. Huntsberger, Rose and Ramaka (1998) have developed a face processing system called Fuzzy-Face that combines wavelet pre-processing of input with a fuzzy self-organizing feature map algorithm. The wavelet-derived face space is partitioned into fuzzy sets, which are characterized by face exemplars and memberships values to those exemplars. The most interesting properties for face motion analysis which this system presents are that it improves the training stage because it uses relatively few training epochs and that it generalizes to face images that are acquired under different lighting conditions. Fellenz et al. (2000) propose a framework for the processing of face image sequences and speech, using different dynamic techniques to extract appropriate features for emotion recognition. The features are used by a hybrid classification procedure, employing neural network techniques and fuzzy logic, to accumulate the evidence for the presence of an emotional facial expression and the speaker's voice.

Expression Analysis Frameworks for Facial Motion Understanding

Systems analyzing faces from monocular images are designed to give motion information with the most suitable level of detail, depending on their final application. Some of the most significant differences among the techniques found in the literature come from the animation semantics they utilize to describe face actions. Some systems may aim at providing very high level face motion and expression data in the form of emotion semantics, for instance, detecting joy, fear or happiness on faces. Some others may provide generic motion data determining what the action of the facial features is, for example, detecting open/closed eyes. And others could even estimate more or less accurately the 3D-motion of the overall face, giving out very low-level face animation parameters.

In an analysis-synthesis scheme for generating face animation, both analysis and synthesis parts must share the same level of semantics. The more specific the motion information given by the analysis is, the fewer free-style interpretations the FA system will have to make. To replicate the exact motion of the person being analyzed, it is necessary to generate very detailed action information. Otherwise, if we only generate rough data about the face actions, we will only be able to get customized face motion if the person's expression behavior has previously been studied and the FA already has the specific details of the individual.

It is quite difficult to classify face motion and expression analysis methods due to the common processing characteristics that many of them share. Despite this fact, we have tried to group them based on the precision of the motion information generated and the importance of the role that the synthesis plays during the analysis.

Methods that Retrieve Emotion Information

Humans detect and interpret faces and facial expressions in a scene with little or no effort. The systems we discuss in this section accomplish this task automatically. The main concern of these techniques is to classify the observed facial expressions in terms of generic facial actions or in terms of emotion categories and not to attempt to understand the face animation that could be involved to synthetically reproduce them.

Yacoob has explored the use of local parameterized models of image motion for recognizing the non-rigid and articulated motion of human faces. These models provide a description of the motion in terms of a small number of parameters that are related intuitively to the motion of some facial features under the influence of expressions. The expression description is obtained after analyzing the spatial distribution of the motion direction field obtained from the optical flow analysis computed at points of high gradient values of the image of the face. This technique gives fairly good results, although the use of optical flow needs very stable lighting conditions and very smooth movement of head motion during the analysis. Computationally, it is also quite heavy. From the starting research (Yacoob & Davis, 1994) to the last published results about the performance of the system (Black & Yacoob, 1997), improvements in the tuning of the processing have been added to make it more robust to head rotations.

Huang and Huang (1997) introduce a system developed in two parts: facial feature extraction (for the training-learning of expressions) and facial expression recognition. The system applies a point distribution model and a gray-level model to find the facial features. Then, the position variations are described by ten Action Parameters (APs). During the training phase, given 90 different expressions, the system classifies the principal components of the APs into six different clusters. In the recognition phase, given a facial image sequence, it identifies the facial expressions by extracting the ten APs, analyzes the principal components, and finally calculates the AP profile correlation for a higher recognition rate. To perform the image analysis, deformable models of the face features are fitted onto the images. The system is only trained for faces on a frontal view. Apparently it seems more robust to illumination conditions than the previous approach, but they do not discuss the image processing techniques, making this point hard to evaluate.

Pantic and Rothkrantz (2000) describe another approach, which is the core of the Integrated System for Facial Expression Recognition (ISFER). The system finds the contour of the features with several methods suited to each feature: snakes, binarization, deformable models, etc., making it more efficient under uncontrolled conditions: irregular lighting, glasses, facial hair, etc. An NN architecture of fuzzy classifiers is designed to analyze the complex mouth movements. In their article, they do not present a robust solution to the non-frontal view positions.

To some extent, all systems discussed have based their description of face actions on the Facial Action Coding System (FACS) proposed by Ekman and Friesen (1978). The importance granted to FACS is such that two research teams, one at the University of California, San Diego (UCSD) and the Salk Institute, and another at the University of Pittsburgh and Carnegie Mellon University (CMU), were challenged to develop prototype systems for automatic recognition of spontaneous facial expressions.

The system developed by the UCSD team, described in Bartlett et al. (2001), analyzes face features after having determined the pose of the individual in front of the camera, although tests of their expression analysis system are only performed on frontal view faces. Features are studied using Gabor filters and afterwards classified using a previously trained HMM. The HMM is applied in two ways:

- taking Gabor representations as inputs, and
- taking support vector machine (SVM) outputs as inputs.

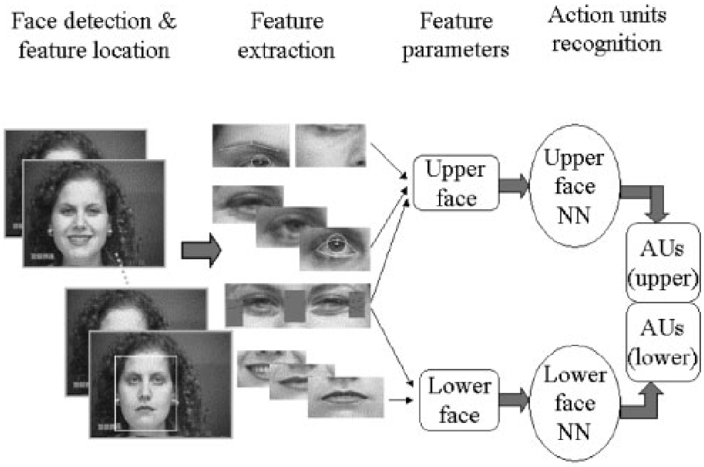
SVMs are used as classifiers. They are a way to achieve good generalization rates when compared to other classifiers because they focus on maximally informative exemplars, the support vectors. To match face features, they first convolve them with a set of kernels (out of the Gabor analysis) to make a jet. Then, that jet is compared with a collection of jets taken from training images, and the similarity value for the closest one is taken. In their study, Bartlett et al. claim an AU detection accuracy from 80% for eyebrow motion to around 98% for eye blinks.

CMU has opted for another approach, where face features are modeled in multi-state facial components of analysis. They use neural networks to derive the AUs associated with the motion observed. They have developed the facial models for lips, eyes, brows, cheeks and furrows. In their article, Tian et al. (2001) describe this technique, giving details about the models and the double use of NN, one for the upper part of the face and a different one for the lower part. (See Figure 6.) They do not discuss the image processing involved in the derivation of the feature model from the images. Tests are performed over a database of faces recorded under controlled light conditions. Their system allows the analysis of faces that are not completely in a frontal position, although most tests were performed only on frontal view faces. The average recognition rates achieved are around 95.4% for upper face AUs and 95.6% for lower face AUs.

Piat and Tsapatsoulis (2000) take the challenge of deducing face expression out of images from another perspective, no longer based on FACS. Their technique finds first the action parameters (MPEG-4 FAPs) related to the expression being analyzed and then they formulate this expression with high-level semantics. To do so, they have related the intensity of the most used expressions to their associated FAPs. Other approaches (Chen & Huang, 2000) complement the image analysis with the study of the human voice to extract more emotional information. These studies are oriented to develop the means to create a Human-Computer Interface (HCI) in a completely bimodal way.

The reader can find in Pantic and Rothkrantz (2000) overviews and comparative studies of many techniques, including some those just discussed, analyzed from the HCI perspective.

Figure 6. Face features (eyes, mouth, brows, …) are extracted from the input image; then, after analyzing them, the parameters of their deformable models are introduced into the NNs which finally generate the AUs corresponding to the face expression. Image courtesy of The Robotics Institute at Carnegie Mellon University.



Methods that Obtain Parameters Related to the Face Animation Synthesis Used

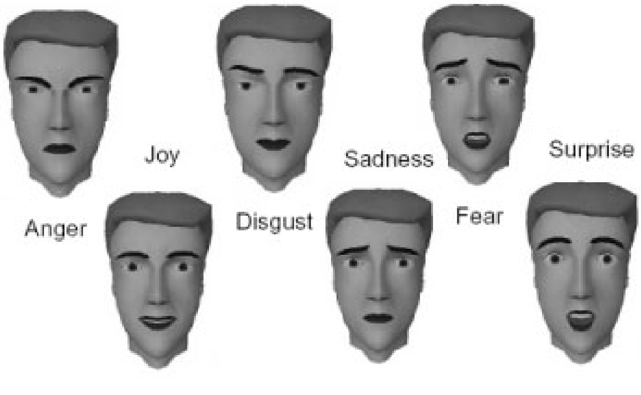
Some face animation systems need action parameters as input that specify how to open the mouth, the position of the eyelids, the orientation of the eyes, etc., in terms of parameter magnitudes associated to physical displacements. The analysis methods studied in this section try to measure displacements and feature magnitudes over the images to derive the actions to be performed over the head models. These methods do not evaluate the expression on the person's face, but extract those measurements that will permit the synthesis of it on a model from the image, as shown in Figure 7.

Terzopoulos and Waters (1993) developed one of the first solutions of this nature. Their method tracks linear facial features to estimate corresponding parameters of a three-dimensional, wireframe face model, allowing them to reproduce facial expressions. A significant limitation of this system is that it requires facial features to be highlighted with make-up for successful tracking. Although active contour models are used, the system is still passive. The tracked contour features passively shape the facial structure without any active control based on observations.

Based on a similar animation system as that of Waters', that is, developed on anatomical-based muscle actions that animate a 3D face wireframe, Essa and Petland define a suitable set of control parameters using vision-based observations. They call their solution FACS+ because it is an extension of the traditional FAC system. They use optical flow analysis along the time of sequences of frontal view faces to get the velocity vectors on 2D and then they are mapped to the parameters. They point out in Essa, Basun, Darrel and Pentland (1996) that driving the physical system with the inputs from noisy motion estimates can result in divergence or a chaotic physical response. This is why they use a continuous time Kalman filter (CTKF) to better estimate uncorrupted state vectors. In their work they develop the concept of motion templates, which are the "corrected" or "noise-free" 2D motion field that is associated with each facial expression. These templates are used to improve the optical flow analysis.

Morishima has been developing a system that succeeds in animating a generic parametric muscle model after having been customized to take the shape and texture of the person the model represents. By means of optical flow image analysis, complemented with speech processing, motion data is generated. These data are translated into motion parameters after passing through a previously trained neural network. In Morishima (2001), he explains the basis of this system, as well as how to generate very realistic animation from electrical captors on the face. Data obtained from this hardware-based study permits a perfect training for coupling the audio processing.

Figure 7. Primary face expressions synthesized on a face avatar. Images courtesy of Joern Ostermann, AT&T Labs - Research.



To control the optical flow data generated from the analysis continuous frames, Tang and Huang (1994) project the head model wireframe vertices onto the images and search for the 2D motion vectors only around these vertices. The model they animate is very simple and the 2D motion vectors are directly translated into 2D vertex motion. No 3D action is generated.

Almost the same procedure is used by Sarris and Strintzis (2001, 2002) in their system for video-phoning for the hearing impaired. The rigid head motion (pose) is obtained by fitting the projection of a 3D wireframe onto the image being analyzed. Then, non-rigid face movements (expressions) are estimated thanks to a feature-based approach adapted from the Kanade, Lucas and Tomasi algorithm. The KLT algorithm is based on minimizing the sum of squared intensity differences between a past and a current feature window, which is performed using a Newton-Raphson minimization method. The features to track are some of the projected points of the wireframe, the MPEG-4 FDPs. To derive MPEG-4 FAPs from this system, they add to the KLT algorithm the information about the degrees of freedom of motion (one or several directions) that the combination of the possible FAPs allows on the studied feature FDPs.

Ahlberg (2002) also exposes in his work a wireframe fitting technique to obtain the rigid head motion. He uses the new parameterized variant of the face model CANDIDE, named CANDIDE-3, which is MPEG-4 compliant. The image analysis techniques include PCA on eigentextures that permits the analysis of more specific features that control the model deformation parameters.

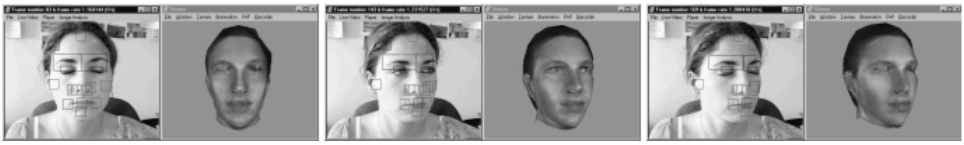
More detailed feature point tracking is developed by Chou et al. (2001). They track the projected points belonging to the mouth, eyes and nostrils provided. These models are also based on the physical vertex distribution of MPEG-4's FDPs and they are able to obtain the combination of FAPs that regenerate the expression and motion of the analyzed face. Their complete system also deals with audio input, analyzing it and complementing the animation data for the lips. The main goal of their approach is to achieve real time analysis to employ these techniques in teleconferencing applications. They do not directly obtain the pose parameters to also synthetically reproduce the pose of the head, but they experiment on how to extend their analysis to head poses other than a frontal view face, by roughly estimating the head pose from the image analysis and rectifying the original input image.

The MIRALab research team at the University of Geneva (Switzerland) has developed a complete system to animate avatars in a realistic way, in order to use them for telecommunications. In Goto et al. (2001), they review the entire process to generate customized realistic animation. The goal of their system is to clone face behavior. The first step in the overall process is to physically adapt a generic head mesh model (already susceptible to being animated) to the shape of the person to be represented. In essence, they follow the same procedure that

Morishima presents in his work. Goto et al. do this by using just a frontal and side view picture of the individual, whereas Morishima also includes other views to recover texture on self occlusions. Models are animated using MPEG-4 FAPs to allow for compatibility with other telecom systems. Animation parameters are extracted from video input of the frontal view face of the speaker and then synthesized, either on the cloned head model or on a different one. Speech processing is also utilized to generate more accurate mouth shapes. An interesting post-processing step is added. If the analysis results do not reflect coherent anatomical motion, they are rejected and the system searches in a probability database for the most probable motion solution to the incoherence. In Goto, Escher and Magnenat-Thalmann (1999), the authors give a more detailed explanation about the image processing involved. Feature motion models for eyes, eyebrows, and mouth allow them to extract image parameters in the form of 2D point displacements. These displacements represent the change of the feature from the neutral position to the instant of the analysis and are easily converted into FAPs. Although the system presents possibilities to achieve face cloning, the current level of animation analysis only permits instant motion replication with little precision. We consider that face cloning is not guaranteed even if realistic animation is.

Also aiming at telecom applications, Andrés del Valle and Dugelay (2002) have developed a system that takes advantage of robust face feature analysis techniques, as well as the synthesis of the realistic clone of the individual being analyzed. We can consider their approach a hybrid between the methods discussed in this section and those that will be presented in the next one. They use a Kalman filter to recover the head global position and orientation. The data predicted by the filter allows them to synthesize a highly realistic 3D model of the speaker with the same scale, position and orientation of the individual being recorded. These data are also useful to complement and adapt feature analysis

Figure 8. In the approach proposed by Andrés del Valle and Dugelay, the avatar does not only reproduce the common techniques that non-rigid, action feature-based analysis would permit, but also synthesizes the rigid motion, thanks to the use of Kalman filtering during pose prediction. Images courtesy of the Image Group at the Institut Eurécom.



algorithms initially designed to work for a frontal point of view under any other head pose. The analysis algorithm parameters and variables are no longer defined over the image plane in 2D, but over the realistic 3D head-model. This solution controls face feature analysis during the change of the speaker's pose. Although the system utilizes the clone of the speaker to analyze, the obtained parameters are general enough to be synthesized on other models or avatars. (See Figure 8.)

Methods that Use Explicit Face Synthesis During the Image Analysis

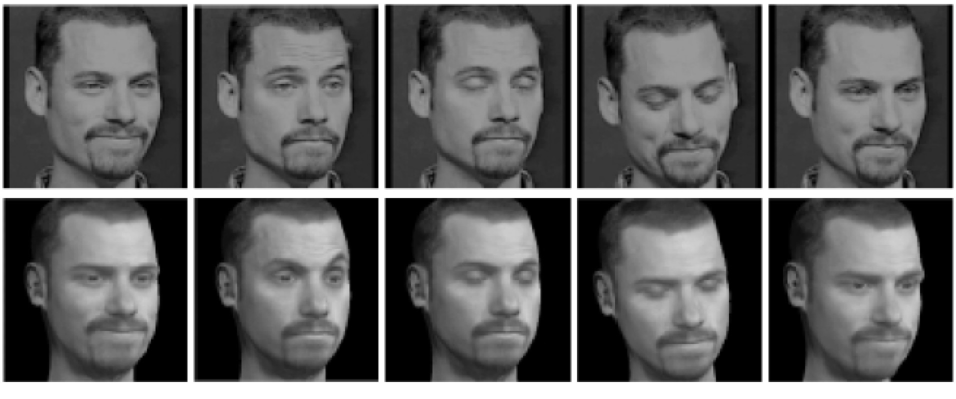
Some face motion analysis techniques use the synthesized image of the head model to control or to refine the analysis procedure. In general, the systems that use synthesized feedback in their analysis need a very realistic head model of the speaker, a high control of the synthesis and a knowledge of the conditions of the face being recorded.

Li, Roivainen and Forchheimer (1993) presented one of the first works to use resynthesized feedback. Using a 3D model — Candide — their approach is characterized by a feedback loop connecting computer vision and computer graphics. They prove that embedding synthesis techniques into the analysis phase greatly improves the performance of motion estimation. A slightly different solution is given by Ezzat and Poggio (1996a, 1996b). In their articles, they describe image-based modeling techniques that make possible the creation of photo-realistic computer models of real human faces. The model they use is built using example views of the face, bypassing the need of any 3D computer graphics. To generate the motion for this model, they use an analysis-by-synthesis algorithm, which is capable of extracting a set of high-level parameters from an image sequence involving facial movement using embedded image-based models. The parameters of the models are perturbed in a local and independent manner for each image until a correspondence-based error metric is minimized. Their system is restricted to understand a limited number of expressions.

More recent research works are able to develop much more realistic results with three-dimensional models. Eisert and Girod (1998), for instance, present a system that estimates 3D motion from image sequences showing head and shoulder scenes for video telephone and teleconferencing applications. They use a very realistic 3D head model of the person in the video. The model constrains the motion and deformation in the face to a set of FAPs defined by the MPEG-4 standard. Using the model, they obtain a description of both global (head pose) and local 3D head motion as a function of unknown facial parameters. Combining the 3D information with the optical flow constraint leads to a linear algorithm that estimates the facial animation parameters. Each synthesized image reproducing face motion from frame *t* is utilized to analyze the image of frame *t*+1. Since natural and synthetic frames are compared at the image level, it is necessary for the lighting conditions of the video scene to be under control. This implies, for example, standard, well distributed light.

Pighin, Szeliski and Salesin (1999) maximize this approach by customizing animation and analysis on a person-by-person basis. They use new techniques to automatically recover the face position and the facial expression from each frame in a video sequence. For the construction of the model, several views of the person are used. For the animation, studying how to linearly combine 3D face models, each corresponding to a particular facial expression of the individual, ensures realism. Their mesh morphing approach is detailed in Pighin, Hecker, Lischinski, Szeliski and Salesin (1998). Their face motion and expression analysis system fits the 3D model on each frame using a continuous optimization technique. During the fitting process, the parameters are tuned to achieve the most accurate model shape. Video image and synthesis are compared to find the degree of similarity of the animated model. They have developed an optimization method whose goal is to compute the model parameters yielding a rendering of the model that best resembles the target image. Although a very slow procedure, the animated results are very impressive because they are highly realistic and very close to what we would expect from face cloning. (See Figure 9.)

Figure 9. Tracking example of Pighin's system. The bottom row shows the result of fitting their model to the target images on the top row. Images courtesy of the Computer Science Department at the University of Washington.



Conclusions and Future Trends

The importance granted to Talking Heads has increased in such a dramatic way during the last decade, that analysis and synthesis methods developed to generate face animation are under continuous change to meet the new application requirements. Following this trend, the analysis of monocular images to extract facial motion to be rendered on synthetic 3D-head models has appeared as a way to simplify and adapt facial animation to current video media. The effort of this research aims at making facial animation technologies and methods available to the general public and permitting the study and representation of already stored image data.

Although the chapter has only covered techniques related to the analysis of face motion, it is important to remark that there exists a tight relationship between the methodology used for the analysis and the way the head model is synthesized. Both analysis and synthesis must share the same *semantics* and the same *syntax* in their motion description. In semantics we include the concept of extracting the same set of possible actions or analyzed movements that the model can actually render. By syntax we imply the way this motion is described. Given certain parameters and magnitudes involved in a specific movement, we should be able to express and make them represent the same action in the analysis module, which has generated them, and the synthesis module, which will reproduce them. Although accomplishing these requirements apparently seems a trivial task, in current research there is still little implication as to how the motion semantics and syntax determine the way analysis techniques should be designed. Therefore, solutions proposed to achieve the same goal, face motion analysis, are rare and lead to the development of algorithms used only in very specific environments. There exists a trade-off between the degree of motion detail extracted from the images and the level of semantic understanding desired. Very precise analysis techniques that are able to generate information to accurately animate face models often cannot provide meaningful information about the general facial expression. Current research trends try to satisfy both needs: accurate motion analysis and expression understanding, so as to generate better facial motion synthesis. As a result, the different research perspectives of the scientific communities involved in the field of facial animation (analysis and synthesis) are starting to converge.

To be able to synthesize facial motion extracted from media that represent reality so as to replicate human face behavior in real-time in such a way that we could no longer distinguish natural from synthetic media can be considered the ultimate objective of research in facial animation. Every step taken towards this target allows emerging new technology domains to use Talking Heads in daily/common-use applications. Telecommunications appears as one of these recent

Table 1. Comparative Study of Some Analysis Techniques Reviewed in the Chapter. The techniques are grouped by methodology. In the first column, we give a summary of the main image processing algorithms used. In the second and the third one, we provide the reference to the work and the researchers involved. The rest of the columns depict the method characteristics.

			Training?	Controlled lighting?	Does it allow rotations? (pose understanding)	Markers?	Potential real-time?	Dees it use a 3D face model?	Possible synthesis in other head poses?	Realistic reproduction?	Time-line (video) analysis?
Methods that obtain emotion information											
Optical flow/parametric model of image motion	[BY97]	J. Black & Y. Yacoob	N	Y	Y	N	N	N	N.A.	N.A.	Y
Deformable models / PCA	[HH97]	C. H. Huang & Y. M. Huang	Y	Y ♣	N	N	N	N	N.A.	N.A.	Y
Feature modeling/neural networks	[TKC01]	Y. Tian et al.	Y	Y	N	N	N	N	N.A.	N.A.	N
NN/Fuzzy logic/deformable models	[PR00]	M. Pantic & L.J.M. Rothkrantz	Y	N ♣	N	N	N	N	N.A.	N.A.	N
HMM/optical flow/Gabor filters/PCA/ICA	[BBL^{+}01]	M. S. Bartlett et al.	Y	Y	Y	Y♦	N	N	Y	Y	Y
Methods that obtain parameters related to the Face Animation synthesis afterwards used											
Snakes	[TW93]	D. Terzopoulos & K. Waters	N	N	N	Y	N	N	Y	Y	Y
Optical flow/motion templates	[EBD^{+}96]	I Essa et al.	Y	Y	N	N	N	N	Y	Y	Y
Optical flow/neural networks	[Mor01]	S. Morishima	Y	Y	N	N/Y	N	N	Y	Y	Y
Model fitting/feature point tracking	[SS01]	N. Sarris & M.G. Strintzis	N	N	~	N	Y	Y	Y	N/Y	Y
Model fitting/PCA/active model/*eifentextures*	[Ahl02]	J. Ahlberg	Y	N	~	N	Y	Y	Y	N/Y	Y
Optical flow	[TH94]	Li-an Tang & T. S. Huang	N	Y	N	N	N	Y	N	N	Y
Feature models	[CCC01]	J. C. Chou, Y.-J. Chang & Y.-C. Chen	N	N	~	N	Y	N	Y	N/Y	Y
Kalman filtering/feature motion models	[VAD01] [AD02]	J.-L. Dugelay, S. Valente & A. C. Andrés	N	N	Y	N	Y	Y	Y	Y	Y
Feature motion models	[GKMT01] [GEZ^{+}99]	Goto et al.	N	N	N	N	Y	N	Y	Y	Y
Methods that use explicit synthesis during the analysis											
Image-based techniques	[EP96] [EP96^{2}]	T. Ezzat & T. Poggio	Y	N	N	N	Y	N	N	Y	Y
Optical flow/spline-based 3D face model	[EG98]	P. Eisert & B. Girod	N	Y	Y	N	N	Y	Y	Y	Y
3D model fitting/image difference minimization	[PSS99]	F. Pinghin et al.	Y	Y	Y	N	N	Y	Y	Y	Y

♣ Author's comment. ♦ For the face tracking, which is based on point tracking. ~ Slight rotations are permitted although there is no direct use of the pose data during image processing

fields. A proof of this interest is how the new standard for the coding of hybrid natural-synthetic media, MPEG-4, has given special importance to facial animation (Ostermann, 2002). The standard specifies common syntax to describe face behavior, thus permitting interoperability amongst different face animation systems. At this point of the evolution and deployment of applications compliant with MPEG-4, several concerns have appeared: Has the standard given a global solution that all specific face animation systems can adopt? Or, does the syntax restrict the semantics of the possible achievable motion too much?

No matter the answer, the existence of all these doubts shows that there is still a long way to go to master face animation and, more concretely, the automatic generation of realistic human-like face motion. All the analysis techniques covered in this chapter are of great help in the study of facial motion because image analysis intrudes the least in the observed scenario, thus permitting the study of real and completely natural behavior.

References

Ahlberg, J. (2002). An active model for facial feature tracking. *Eurasip Journal on Applied Signal Processing*, 6, 566-571.

Andrés del Valle, A. C. & Dugelay, J. L. (2002). Facial expression analysis robust to 3D head pose motion. *Proceedings of the International Conference on Multimedia and Expo*.

Bartlett, M. S. (2001). *Face image analysis by unsupervised learning*. Boston, MA: Kluwer Academic Publishers.

Bartlett et al. (2001). *Automatic Analysis of spontaneous facial behavior: A final project report*. (Tech. Rep. No. 2001.08). San Diego, CA: University of California, San Diego, MPLab.

Black, M. J. & Yacoob, Y. (1997). Recognizing facial expressions in image sequences using local parameterized models of image motion. I*nternational Journal of Computer Vision*, 25(1), 23-48.

Chen, L. S. & Huang, T. S. (2000). Emotional expressions in audiovisual human computer interaction. *Proceedings of the International Conference on Multimedia and Expo*.

Chou, J. C., Chang, Y. J. & Chen, Y. C. (2001). Facial feature point tracking and expression analysis for virtual conferencing systems. *Proceedings of the International Conference on Multimedia and Expo*.

Cordea, M. D., Petriu, E. M., Georganas, N. D., Petriu, D. C. & Whalen, T. E. (2001). 3D head pose recovery for interactive virtual reality avatars.

Proceedings of the IEEE Instrumentation and Measurement Technology Conference.

Debevec, P., Hawkins, T., Tchou, C., Duiker, H. P., Sarokin, W. & Sagar, M. (2000). Acquiring the reflectance field of a human face. *Proceedings of SIGGRAPH 2000*, 145-156. ACM Press/ACM SIGGRAPH/Addison Wesley Longman.

Eisert, P. & Girod, B. (1998). Analyzing facial expression for virtual conferencing. *Proceedings of the IEEE Computer Graphics & Applications*, 70-78.

Eisert, P. & Girod, B. (2002). Model-based enhancement of lighting conditions in image sequences. *Proceedings of the Visual Communications and Image Processing*.

Ekman, P. & Friesen, W. V. (1978). *The facial action coding system*. Palo Alto, CA.: Consulting Psychologists Press.

Essa, I., Basu, S., Darrel, T. & Pentland, A. (1996). Modeling, tracking and interactive animation of faces and heads using input from video. *Proceedings of Computer Animation*.

Ezzat, T. & Poggio, T. (1996a). Facial analysis and synthesis using image based models. *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*.

Ezzat, T. & Poggio, T. (1996b). Facial analysis and synthesis using image based models. *Proceedings of the Workshop on the Algorithm Foundations of Robotics*.

Fellenz et al. (2000). On emotion recognition and of speech using neural networks, fuzzy logic and the ASSESS system. *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*.

Garcia, C. & Tziritas, G. (1999, September). Face detection using quantized skin color regions merging and wavelet packet analysis. *IEEE Transactions on Multimedia*. 1(3), 264-277.

Goto, T., Escher, M., Zanardi, C. & Magnenat-Thalmann, N. (1999). MPEG-4 based animation with face feature tracking. *In Computer Animation and Simulation*.

Goto, T., Kshirsagar, S. & Magnenat-Thalmann, N. (2001, May). Automatic face cloning and animation. *IEEE Signal Processing Magazine*. 17-25.

Holbert, S. & Dugelay, J. L. (1995). Active contours for lip-reading: Combining snakes with templates. *Quinzième colloque GRETSI*, 717-720.

Huang, C. L. & Huang, Y. M. (1997, September). Facial expression recognition using model-based feature extraction and action parameters classification. *Journal of Visual Communication and Image Representation*, 8(3), 278-290.

Huang, F. J. & Chen, T. (2000). Tracking of multiple faces for human-computer interfaces and virtual environments. *Proceedings of the International Conference and Multimedia Expo*.

Huntsberger, T. L., Rose, J. & Ramaka, A. (1998). Fuzzy-Face: A hybrid wavelet/fuzzy self-organizing feature map system for face processing. *Journal of Biological Systems*.

Jones, M. J. & Rehg, J. M. (1999). Statistical color models with application to skin detection. *Proceedings of the Computer Vision and Pattern Recognition,*274-280.

Leroy, B. & Herlin, I. L. (1995). Un modèle deformable paramétrique pour la reconnaissance de visages et le suivi du movement des lèvres [A parametric deformable model for face recognition and lip motion tracking]. *Quinzième colloque GRETSI*, 701-704.

Li, H., Roivainen, P. & Forchheimer, R. (1993). 3-D motion estimation in model-based facial image coding. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 15(6), 545-555

Luong, Q. T. & Faugeras, O. D. (1997). Self-calibration of a moving camera from point correspondences and fundamental matrices. *International Journal of Computing Vision,* 22(3), 261-289.

Luong, Q. T., Fua, P. & Leclerc, Y. (2002, January). The radiometry of multiple images. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 24(1), 19-33.

Malciu, M. & Prêteux, F. (2001). MPEG-4 compliant tracking of facial features in video sequences. *Proceedings of the International Conference on Augmented Virtual Environments and 3-D Imaging*, 108-111.

Metaxas, D. (1999). Deformable model and HMM-based tracking, analysis and recognition of gestures and faces. *Proceedings of the International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems.*

Morishima, S. (2001, May). Face analysis and synthesis for duplication expression and impression. *IEEE Signal Processing Magazine,* 26-34.

Moses, Y., Reynard, D. & Blake, A. (1995). Determining facial expressions in real time. *Proceedings of the International Workshop on Automatic Face and Gesture Recognition*, 332-337.

MPEG-4. (2000, January) *Signal Processing: Image Communication*. Tutorial Issue on the MPEG-4 Standard, 15(4-5).

Nikolaidis, A. & Pitas, I. (2000). Facial feature extraction and pose determination. *The Journal of the Pattern Recognition Society,* 33, 1783-1791.

Ostermann, J. (2002). Face animation in MPEG-4. In I. S. Pandzinc & R. Forchheimer (Eds.), *MPEG-4 facial animation: The standard, implementation and applications*. England, UK: John Wiley & Sons Ltd.

Pantic, M. & Rothkrantz, L. J. M. (2000, December). Automatic analysis of facial expression: the state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12), 1424-1445.

Pardàs, M. & Bonafonte, A. (2002). Facial animation parameters extraction and expression recognition using Hidden Markov Models. *Eurasip Signal Processing: Image Communication*, 17(9), 675-688.

Pentland, A., Moghaddam, B. & Starner, T. (1994). View-based and modular eigenspaces for face recognition. *Proceedings of the International Conference on Computer Vision and Pattern Recognition*.

Piat, F. & Tsapatsoulis, N. (2000). Exploring the time course of facial expressions with a fuzzy system. *Proceedings of the International Conference on Multimedia and Expo*.

Pighin, F., Hecker, J., Lischinski, D., Szeliski, R. & Salesin, S. (1998). Synthesizing realistic facial expressions from photographs. *Proceedings of ACM SIGGRAPH* 98, 75-84.

Pighin, F., Szeliski, R. & Salesin, D. H. (1999). Resynthesizing facial animation through 3D model-based tracking. *Proceedings of the International Conference on Computer Vision*.

Poggio, T. & Girosi, F. (1990). Networks for approximation and learning. *Proceedings of IEEE*, 78(9), 1481-1497.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), 257-285.

Sahbi, H., Geman, D. & Boujemaa, N. (2002). Face detection using coarse-to-fine support vector classifiers. *Proceedings of the International Conference on Image Processing.*

Sarris, N. & Strintzis, M. G. (2001, July-September). Constructing a video phone for the hearing impaired using MPEG-4 tools. *IEEE Multimedia*, 8(3).

Sarris, N, Grammaidis, N. & Strintzis, M. G. (2002). FAP extraction using three-dimensional motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(10), 865-876.

Serra, J. (Ed.). (1982). *Image Analysis and Mathematical Morphology*. London: Academic Press.

Serra, J. (Ed.). (1988). *Image Analysis and Mathematical Morphology*. Volume 2: Theoretical Advances. London: Academic Press.

Shimizu, I., Zhang, Z., Akamatsu, S. & Deguchi, K. (1998). Head pose determination from one image using a generic model. *Proceedings of the Third International Conference on Automatic Face and Gesture Recognition*, 100-105.

Ström, J., Jebara, T., Basu, S. & Pentland, A. (1999). Real time tracking and modeling of faces: An EKF-based analysis by synthesis approach. *Proceedings of the Modelling People Workshop at ICCV'99*.

Tang, L. & Huang, T. S. (1994). Analysis-based facial expression synthesis. *Proceedings of the International Conference on Image Processing*, 98-102.

Terzopoulos, D. & Waters, K. (1993, June). Analysis and synthesis of facial image sequences using physical and anatomical models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6).

Tian, Y., Kanade, T. & Cohn, J. F. (2001, February). Recognizing action units for facial expression analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 23(2), 97-115.

Turk, M. & Pentland, A. (1991). Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1).

Valente, S. & Dugelay, J. L. (2001). A visual analysis/synthesis feedback loop for accurate face tracking. *Signal Processing: Image Communication*, 16(6), 585-608.

Valente, S., Andrés del Valle, A. C. & Dugelay, J. L. (2001). Analysis and reproduction of facial expressions for realistic communicating clones. *Journal of VLSI and Signal Processing*, 29, 41-49.

Wiskott, L. (2001, July). Optical Flow Estimation. Retrieved September, 26, 2002, from the World Wide Web: http://www.cnl.salk.edu/~wiskott/Bibliographies/FlowEstimation.html.

Yacoob, Y. & Davis, L. (1994). Computing spatio-temporal representations of human faces. *Proceedings of Computer Vision and Pattern Recognition Conference*, 70-75.

Yuille, A. L. (1991). Deformable templates for face recognition. *Journal of Cognitive Neuroscience*, 3(1), 59-70.

Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11), 1330-1334.

Chapter VII

Analysis and Synthesis of Facial Expressions

Peter Eisert
Fraunhofer Institute for Telecommunications, Germany

Abstract

In this chapter, the state-of-the-art in facial animation and expression analysis is reviewed and new techniques for the estimation of 3-D human motion, deformation, and facial expressions from monocular video sequences are presented. Since illumination has a considerable influence on the appearance of objects in a scene, methods for the derivation of photometric scene properties from images are also addressed. For a particular implementation, the potential of these analysis techniques is illustrated for applications like character animation and model-based video coding. Experiments have shown that the usage of 3-D computer models allows video transmissions at bit-rates of a few kbit/s, enabling a wide variety of new applications.

Introduction

Facial expression analysis and synthesis techniques have received increasing interest in recent years. Numerous new applications in the areas of low-bit-rate communication, user-friendly computer interfaces, the film industry, or medicine are becoming more available with today's computers. In this chapter, the state-of-the-art in facial animation and analysis is reviewed and new techniques for the estimation of 3-D human motion, deformation, and facial expressions from monocular video sequences are presented. The chapter starts with an overview of existing methods for representing human heads and facial expressions three-dimensionally in a computer. Algorithms for the determination of facial expressions from images and image sequences are reviewed, focusing on feature-based and optical-flow based methods. For natural video capture conditions, scene lighting often varies over time. This illumination variability has a considerable influence not only on the visual appearance of the objects in the scene, but also on the performance of the estimation algorithms. Therefore, methods for determining lighting changes in the scene are discussed for the purpose of robust facial analysis under uncontrolled illumination settings. After this overview, an example of a hierarchical, gradient-based method for the robust estimation of MPEG-4 facial animation parameters is given, illustrating the potential of model-based coding. This method is able to simultaneously determine both global and local motion in the face in a linear, low-complexity framework. In order to improve the robustness against lighting changes in the scene, a new technique for the estimation of photometric properties based on *Eigen light maps* is added to the system. The performance of the presented methods is evaluated in some experiments given in the application section. First, the concept of model-based coding is described, where head-and-shoulder image sequences are represented by computer graphics models that are animated according to the facial motion and deformation extracted from real video sequences. Experiments validate that such sequences can be encoded at less than 1 kbit/s enabling a wide range of new applications. Given an object-based representation of the current scene, changes can easily be made by modifying the 3-D object models. In that context, we will show how facial expression analysis can be used to synthesize new video sequences of arbitrary people, who act exactly in the same way as the person in a reference sequence, which, e.g., enables applications in facial animation for film productions.

Review of Facial Analysis and Synthesis Techniques

Facial Animation

Modeling the human face is a challenging task because of its familiarity. Already early in life, we are confronted with faces and learn how to interpret them. We are able to recognize individuals from a large number of similar faces and to detect very subtle changes in facial expressions. Therefore, the general acceptability of synthetic face images strongly depends on the 3-D head model used for rendering. As a result, significant effort has been spent on the accurate modeling of a person's appearance and his or her facial expressions (Parke et al., 1996). Both problems are addressed in the following two sections.

3-D head models

In principle, most head models used for animation are based on triangle meshes (Rydfalk, 1978; Parke, 1982). Texture mapping is applied to obtain a photorealistic appearance of the person (Waters, 1987; Terzopoulos et al., 1993; Choi et al., 1994; Aizawa et al., 95; and Lee et al., 1995). With extensive use of today's computer graphics techniques, highly realistic head models can be realized (Pighin et al., 1998).

Modeling the shape of a human head with polygonal meshes results in a representation consisting of a large number of triangles and vertices which have to be moved and deformed to show facial expressions. The face of a person, however, has a smooth surface and facial expressions result in smooth movements of surface points due to the anatomical properties of tissue and muscles. These restrictions on curvature and motion can be exploited by splines which satisfy certain continuity constraints. As a result, the surface can be represented by a set of spline control points that is much smaller than the original set of vertices in a triangle mesh. This has been exploited by Hoch et al. (1994) where B-splines with about 200 control points are used to model the shape of human heads. In Ip et al. (1996), non-uniform rational B-splines (NURBS) represent the facial surfaces. Both types of splines are defined on a rectangular topology and, therefore, do not allow a local patch refinement in areas that are highly curved. To overcome this restriction, hierarchical splines have been proposed for the head modeling (Forsey et al., 1988) to allow a recursive subdivision of the rectangular patches in more complex areas.

Face, eyes, teeth, and the interior of the mouth can be modeled similarly with textured polygonal meshes, but a realistic representation of hair is still not available. A lot of work has been done in this field to model the fuzzy shape and reflection properties of the hair. For example, single hair strands have been modeled with polygonal meshes (Watanabe et al., 1992) and the hair dynamics have been incorporated to model moving hair (Anjyo et al., 1992). However, these algorithms are computationally expensive and are not feasible for real-time applications in the near future. Image-based rendering techniques (Gortler et al., 1996; Levoy et al., 1996) might provide new opportunities for solving this problem.

Facial expression modeling

Once a 3-D head model is available, new views can be generated by rotating and translating the 3-D object. However, for the synthesis of facial expressions, the model can no longer be static. In general, two different classes of facial expression modeling can be distinguished in model-based coding applications: the clip-and-paste method and algorithms based on the deformation the 3-D surfaces.

For the *clip-and-paste method* (Aizawa et al., 1989; Welsh et al., 1990; and Chao et al., 1994), templates of facial features like eyes and the mouth are extracted from previous frames and mapped onto the 3-D shape model. The model is not deformed according to the facial expression, but remains rigid and is used only to compensate for the global motion given by head rotation and translation. All local variations in the face must, therefore, be described by texture changes of the model. During encoding of a video sequence, a codebook containing templates for different facial expressions is built. A new expression can then be synthesized by combining several feature templates that are specified by their position on the model and their template index from the codebook. As a result, a discrete set of facial expressions can be synthesized. However, the transmission of the template codebook to the decoder consumes a large number of bits, which makes the scheme unsuitable for coding purposes (Welsh et al., 1990). Beyond that, the localization of the facial features in the frames is a difficult problem. Pasting of templates extracted at slightly inaccurate positions leads to an unpleasant “jitter” in the resulting synthetic sequence.

The *deformation method* avoids these problems by using the same 3-D model for all facial expressions. The texture remains basically constant and facial expressions are generated by deforming the 3-D surface (Noh et al., 2001). In order to avoid the transmission of all vertex positions in the triangle mesh, the facial expressions are compactly represented using high-level expression parameters. Deformation rules associated with the 3-D head model describe how certain areas in the face are deformed if a parameter value changes. The superposition of many of these local deformations is then expected to lead to the desired facial expression. Due to the advantages of the deformation method over the clip-and-paste method (Welsh et al., 1990), it is used in most current approaches for representing facial expressions. The algorithms proposed in this chapter are also based on this technique and, therefore, the following review of related work focuses on the deformation method for facial expression modeling.

One of the first systems of facial expression parameterization was proposed by Hjortsjö (1970) and later extended by the psychologists Ekman and Friesen (1978). Their *facial action coding system* (FACS) is widely used today for the description of facial expressions in combination with 3-D head models (Aizawa et al., 1989; Li, 1993; Choi et al., 1994; and Hoch et al., 1994). According to that scheme, any facial expression results from the combined action of the 268 muscles in the face. Ekman and Friesen discovered that the human face performs only 46 possible basic actions. Each of these basic actions is affected by a set of muscles that cannot be controlled independently. To obtain the deformation of the facial skin that is caused by a change of an action unit, the motion of the muscles and their influence on the facial tissue can be simulated using soft tissue models (Terzopoulos et al., 1993; Lee et al., 1995). Due to the high computational complexity of muscle-based tissue simulation, many applications model the surface deformation directly (Aizawa et al., 1989; Choi et al., 1994) using heuristic transforms between action units and surface motion.

Very similar to the FACS is the parameterization in the *synthetic and natural hybrid coding* (SNHC) part of the MPEG-4 video coding standard (MPEG, 1999). Rather than specifying groups of muscles that can be controlled independently and that sometimes lead to deformations in larger areas of the face, the single parameters in this system directly correspond to locally limited deformations of the facial surface. There are 66 different facial animation parameters (FAPs) that control both global and local motion.

Instead of using facial expression descriptions that are designed with a relation to particular muscles or facial areas, data-driven approaches are also used for the modeling. By linearly interpolating 3-D models in a database of people showing different facial expressions, new expressions can be created (Vetter et al., 1998; Blanz et al., 1999). Ortho-normalizing this *face-space* using a KLT leads to a compact description that allows the representation of facial expressions with a small set of parameters (Hölzer, 1999; Kalberer et al., 2001).

Facial Expression Analysis

Synthesizing realistic head-and-shoulder sequences is only possible if the facial animation parameters are appropriately controlled. An accurate estimation of these parameters is, therefore, essential. In the following sections, different methods are reviewed for the estimation of 3-D motion and deformation from monoscopic image sequences. Two different groups of algorithms are distinguished: feature-based approaches, which track distinct features in the images and optical flow based methods that exploit the entire image for estimation.

Feature-based estimation

One common way for determining the motion and deformation in the face between two frames of a video sequence is the use of feature points (Kaneko et al., 1991; Terzopoulos et al., 1993; Gee et al., 1994; Huang et al., 1994; Lopez et al., 1995; and Pei, 1998). Highly discriminant areas with large spatial variations, such as areas containing the eyes, nostrils, or mouth corners, are identified and tracked from frame to frame. If corresponding features are found in two frames, the change in position determines the displacement.

How the features are searched depends on properties such as color, size, and shape. For facial features, extensive research has been performed, especially in the area of face recognition (Chellappa et al., 1995). Templates (Brunelli et al., 1993), often used for finding facial features, are small reference images of typical features. They are compared at all positions in the frame to find a good match between the template and the current image content (Thomas et al., 1987). The best match is said to be the corresponding feature in the second frame. Problems with templates arise from the wide variability of captured images due to illumination changes or different viewing positions. To compensate for these effects, eigen-features (Moghaddam et al., 1997; Donato et al., 1999), which span a space of possible feature variations or deformable templates (Yuille, 1991) and reduce the features to parameterized contours, can be utilized.

Instead of estimating single feature points, the whole contour of features can also be tracked (Huang et al., 1991; Pearson, 1995) using *snakes*. Snakes (Kas et al., 1987) are parameterized active contour models that are composed of internal and external energy terms. Internal energy terms account for the shape of the feature and smoothness of the contour, while the external energy attracts the snake towards feature contours in the image.

All feature-based algorithms have in common that single features, like the eyes, can be found quite robustly. Dependent on the image content, however, only a small number of feature correspondences can typically be determined. As a result, the estimation of 3-D motion and deformation parameters from the displacements lacks the desired accuracy if a feature is erroneously associated with a different feature in the second frame.

Optical flow based estimation

Approaches based on optical flow information utilize the entire image information for the parameter estimation, leading to a large number of point correspondences. The individual correspondences are not as reliable as the ones obtained with feature-based methods, but due to the large number of equations, some mismatches are not critical. In addition, possible outliers (Black et al., 1996) can generously be removed without obtaining an underdetermined system of equations for the determination of 3-D motion.

One way of estimating 3-D motion is the explicit computation of an optical flow field (Horn et al., 1981; Barron et al., 1994; and Dufaux et al., 1995), which is followed by the derivation of motion parameters from the resulting dense displacement field (Netravali et al., 1984, Essa et al., 1994; and Bartlett et al., 1995). Since the computation of the flow field from the optical flow constraint equation (Horn et al., 1981), which relates image gradient information (Simoncelli, 1994) to 2-D image displacements, is an underdetermined problem, additional smoothness constraints have to be added (Horn, 1986; Barron et al., 1994). A non-linear cost function (Barron et al., 1994) is obtained that is numerically minimized. The use of hierarchical frameworks (Enkelmann, 1988; Singh, 1990; and Sezan et al., 1993) can reduce the computational complexity of the optimization in this high-dimensional parameter space. However, even if the global minimum is found, the heuristical smoothness constraints may lead to deviations from the correct flow field, especially at object boundaries and depth discontinuities.

In model-based motion estimation, the heuristical smoothness constraints are, therefore, often replaced by explicit motion constraints derived from the 3-D object models. For rigid body motion estimation (Kappei, 1988; Koch, 1993), the 3-D motion model, specified by three rotational and three translational degrees of freedom, restricts the possible flow fields in the image plane. Under the assumption of perspective projection, known object shape, and small motion between two successive video frames, an explicit displacement field can be derived that is linear in the six unknown degrees of freedom (Longuet, 1984; Netravali et al., 1984; and Waxman et al., 1987). This displacement field can easily be combined with the optical flow constraint to obtain a robust estimator for the six motion parameters. Iterative estimation in an analysis-synthesis framework (Li et al., 1993) removes remaining errors caused by the linearization of image intensity and the motion model.

For facial expression analysis, the rigid body assumption can no longer be maintained. Surface deformations due to facial expressions have to be considered additionally. Most approaches found in the literature (Ostermann, 1994; Choi et al., 1994; Black et al., 1995; Pei, 1998; and Li et al., 1998) separate this problem into two steps. First, global head motion is estimated under the assumption of rigid body motion. Local motion caused by facial expressions is regarded as noise (Li et al., 1994b) and, therefore, the textured areas around the mouth and the eyes are often excluded from the estimation (Black et al., 1995; and Li et al., 1994b). Given head position and orientation, the remaining residuals of the motion-compensated frame are used to estimate local deformations and facial expressions. In (Black et al., 1995; Black et al., 1997), several 2-D motion models with six (affine) or eight parameters are used to model local facial deformations. By combining these models with the optical flow constraint, the unknown parameters are estimated in a similar way as in the rigid body case.

High-level facial animation parameters can finally be derived from the estimated set of 2-D motion parameters. Even higher robustness can be expected by directly estimating the facial animation parameters using more sophisticated motion models. In Choi et al. (1994), a system is described that utilizes an explicit 3-D head model. This head model directly relates changes of facial animation parameters to surface deformations. Orthographic projection of the motion constraints and combination with optical flow information result in a linear estimator for the unknown parameters. The accuracy problem of separate global and local motion estimation is here relaxed by an iterative framework that alternately estimates the parameters for global and local motion.

The joint estimation of global head motion together with facial expressions is rarely addressed in the literature. In Li et al. (1993; 1994), a system for the combined estimation of global and local motion is presented that stimulated the approaches presented in the next section. A 3-D head model based on the Candide (Rydfalk, 1978) model is used for image synthesis and provides explicit 3-D motion and deformation constraints. The affine motion model describes the image displacements as a linear function of the six global motion parameters and the facial action units from the FACS system, which are simultaneously estimated in an analysis-synthesis framework. Another approach that allows a joint motion and deformation estimation has been proposed by DeCarlo et al. (1996, 1998). A deformable head model is employed that consists of ten separate face components that are connected by spring-like forces incorporating anthropometric constraints (DeCarlo et al., 1998b; Farkas, 1995). Thus, the head shape can be adjusted similar to the estimation of local deformations. For the determination of motion and deformation, again a 3-D motion model is combined with the optical flow constraint. The 3-D model also includes a dynamic, Lagrangian description for the parameter changes similar to the work of Essa (Essa et al., 1994; Essa et al., 1997). Since the head model lacks any color information, no synthetic frames can be rendered which makes it impossible to use an analysis-synthesis loop. Therefore, additional edge forces are added to avoid an error accumulation in the estimation.

Illumination Analysis

In order to estimate the motion of objects between two images, most algorithms make use of the *brightness constancy assumption* (Horn, 1986). This assumption, which is an inherent part of all optical flow-based and many template-based methods, implies that corresponding object points in two frames show the same brightness. However, if the lighting in the scene changes, the brightness of corresponding points might differ significantly. But, also, if the orientation of the object surface relative to a light source changes due to object motion, brightness is in general not constant (Verri et al., 1989). On the contrary, intensity changes due to varying illumination conditions can dominate the effects caused by object motion (Pentland, 1991; Horn, 1986; and Tarr, 1998). For accurate and robust extraction of motion information, lighting effects must be taken into account.

In spite of the relevance of illumination effects, they are rarely addressed in the area of 3-D motion estimation. In order to allow the use of the optical flow constraint for varying brightness, higher order differentials (Treves et al., 1994) or pre-filtering of the images (Moloney, 1991) have been applied. Similarly, *lightness algorithms* (Land et al., 1971; Ono et al., 1993; and Blohm, 1997) make use of the different spectral distributions of texture and intensity changes due to shading, in order to separate irradiance from reflectance. If the influence of illumination cannot be suppressed sufficiently by filtering as, e.g., in image regions depicting highlights caused by specular reflections, the corresponding parts are often detected (Klinker et al., 1990; Stauder, 1994; and Schluens et al., 1995) and classified as outliers for the estimation.

Rather than removing the disturbing effects, explicit information about the illumination changes can be estimated. This not only improves the motion estimation but also allows the manipulation and visual enhancement of the illumination situation in an image afterwards (Blohm, 1997). Under controlled conditions with, e.g., known object shape, light source position (Sato et al., 1997; Sato et al., 1996; and Baribeau et al., 1992), and homogeneous non-colored surface properties (Ikeuchi et al., 1991; Tominaga et al., 2000), parameters of sophisticated reflection models like the Torrance-Sparrow model (Torrance et al., 1967; Nayar et al., 1991; and Schlick, 1994) which also includes specular reflection, can be estimated from camera views. Since the difficulty of parameter estimation increases significantly with model complexity, the analysis of global illumination scenarios (Heckbert, 1992) with, e.g., inter-reflections (Forsyth et al., 1991) is only addressed for very restricted applications (Wada et al., 1995).

In the context of motion estimation, where the exact position and shape of an object are often not available, mostly simpler models are used that account for the dominant lighting effects in the scene. The simplest scenario is the assumption of pure ambient illumination (Foley et al., 1990). Other approaches (Gennert et al., 1987; Moloney et al., 1991; and Negahdaripour et al., 1993) extend the optical flow constraint is extended by a two-parameter function to allow for global intensity scaling and global intensity shifts between the two frames. Local shading effects can be modeled using additional directional light sources (Foley et al., 1990). For the estimation of the illuminant direction, surface-normal information is required. If this information is not available as, e.g., for the large class of *shape-from-shading* algorithms (Horn et al., 1989; Lee et al., 1989), assumptions about the surface-normal distribution are exploited to derive the direction of the incident light (Pentland, 1982; Lee et al., 1989; Zheng et al., 1991; and Bozdagi et al., 1994).

If explicit 3-D models and with that surface-normal information are available, more accurate estimates of the illumination parameters are obtainable (Stauder, 1995; Deshpande et al., 1996; Brunelli, 1997; and Eisert et al., 1997). In these approaches, Lambertian reflection is assumed in combination with directional and ambient light. Given the surface normals, the illumination parameters are estimated using neural networks (Brunelli, 1997), linear (Deshpande et al., 1996; Eisert et al., 1997), or non-linear (Stauder, 1995) optimization.

Rather than using explicit light source and reflection models to describe illumination effects, multiple images captured from the same viewing position, but under varying illumination can also be exploited. Hallinan et al. showed (Hallinan et al., 1994; Epstein et al., 1995) that five eigen images computed from a set of differently illuminated facial images are sufficient to approximate arbitrary lighting conditions by linearly blending between the eigen images. An analytic method for the derivation of the eigen components can be found in Ramamoorthi (2002). This low-dimensional space of face appearances can be represented as an illumination cone as shown by Belhumeur et al. (1998). In Ramamoorthi et al. (2001), the reflection of light was theoretically described by convolution in a signal-processing framework. Illumination analysis or inverse rendering can then be considered as deconvolution. Beside the creation of arbitrarily illuminated face images, the use of multiple input images also allows the estimation of facial shape and thus a change of head pose in 2-D images (Georgiades et al., 1999). Using eigen light maps of explicit 3-D models (Eisert et al., 2002) instead of blending between eigen images, also extends the applicability of the approach to locally deforming objects like human faces in image sequences.

For the special application of 3-D model-based motion estimation, relatively few approaches have been proposed that incorporate photometric effects. In Bozdagi et al. (1994), the illuminant direction is estimated according to Zheng et al. (1991) first without exploiting the 3-D model. Given the illumination parameters, the optical flow constraint is extended to explicitly consider intensity changes caused by object motion. For that purpose, surface normals are required which are derived from the 3-D head model. The approach proposed in Stauder (1995 and 1998) makes explicit use of normal information for both illumination estimation and compensation. Rather than determining the illuminant direction from a single frame, the changes of surface shading between two successive frames are exploited to estimate the parameters. The intensity of both ambient and directional light, as well as the direction of the incident light, is determined by minimizing a non-linear cost function. Experiments performed for both approaches show that the consideration of photometric effects can significantly improve the accuracy of estimated motion parameters and the reconstruction quality of the motion-compensated frames (Bozdagi et al.; 1994, Stauder, 1995).

Hierarchical Model-Based Facial Expression Analysis

The most challenging part of facial expression analysis is the estimation of 3-D facial motion and deformation from two-dimensional images. Due to the loss of one dimension caused by the projection of the real world onto the image plane, this task can only be solved by exploiting additional knowledge of the objects in the scene. In particular, the way the objects move can often be restricted to a low number of degrees of freedom that can be described by a limited set of parameters. In this section, an example of a new 3-D model-based method for the estimation of facial expressions is presented that makes use of an explicit parameterized 3-D human head model describing shape, color, and motion constraints of an individual person (Eisert, 2000). This model information is jointly exploited with spatial and temporal intensity gradients of the images. Thus, the entire area of the image showing the object of interest is used, instead of dealing with discrete feature points, resulting in a robust and highly accurate system. A linear and computationally efficient algorithm is derived for different scenarios. The scheme is embedded in a hierarchical analysis-synthesis framework to avoid error accumulation in the long-term estimation.

Optical-Flow Based Analysis

In contrast to feature-based methods, gradient-based algorithms utilize the optical flow constraint equation:

$$\frac{\partial I(X,Y)}{\partial X} d_x + \frac{\partial I(X,Y)}{\partial Y} d_y = I(X,Y) - I'(X,Y), \qquad (1)$$

where $\frac{\partial I}{\partial X}$ and $\frac{\partial I}{\partial Y}$ are the spatial derivatives of the image intensity at pixel position [X, Y]. I′-I denotes the temporal change of the intensity between two time instants Δt=t′-t corresponding to two successive frames in an image sequence. This equation, obtained by Taylor series expansion up to first order of the image intensity, can be set up anywhere in the image. It relates the unknown 2-D motion displacement **d**=[$d_{x,}$ d_y] with the spatial and temporal derivatives of the images.

The solution of this problem is under-determined since each equation has two new unknowns for the displacement coordinates. For the determination of the optical flow or motion field, additional constraints are required. Instead of using heuristical smoothness constraints, explicit knowledge about the shape and motion characteristics of the object is exploited. Any 2-D motion model can be used as an additional motion constraint in order to reduce the number of unknowns to the number of motion parameters of the corresponding model. In that case, it is assumed that the motion model is valid for the complete object. An over-determined system of equations is obtained that can be solved robustly for the unknown motion and deformation parameters in a least-squares sense.

In the case of facial expression analysis, the motion and deformation model can be taken from the shape and the motion characteristics of the head model description. In this context, a triangular B-spline model (Eisert et al., 1998a) is used to represent the face of a person. For rendering purposes, the continuous spline surface is discretized and approximated by a triangle mesh as shown in Figure 6. The surface can be deformed by moving the spline's control points and thus affecting the shape of the underlying mesh. A set of facial animation parameters (FAPs) according to the MPEG-4 standard (MPEG, 1999) characterizes the current facial expression and has to be estimated from the image sequence. By concatenating all transformations in the head model deformation and using knowledge from the perspective camera model, a relation between image displacements and FAPs can be analytically derived

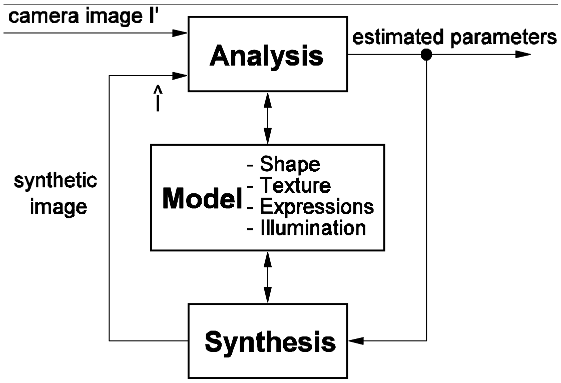
$$\mathbf{d} = f\left(FAP_0, FAP_1, \ldots, FAP_{N-1}\right). \qquad (2)$$

Combining this motion constraint with the optical flow constraint (1) leads to a linear system of equations for the unknown FAPs. Solving this linear system in a least squares sense, results in a set of facial animation parameters that determines the current facial expression of the person in the image sequence.

Hierarchical Framework

Since the optical flow constraint equation (1) is derived assuming the image intensity to be linear, it is only valid for small motion displacements between two successive frames. To overcome this limitation, a hierarchical framework can be used (Eisert et al., 1998a). First, a rough estimate of the facial motion and deformation parameters is determined from sub-sampled and low-pass filtered images, where the linear intensity assumption is valid over a wider range. The 3-D model is motion compensated and the remaining motion parameter errors are reduced on frames having higher resolutions.

Figure 1. Analysis-synthesis loop of the model-based estimator.



The hierarchical estimation can be embedded into an analysis-synthesis loop as shown in Figure 1. In the analysis part, the algorithm estimates the parameter changes between the previous synthetic frame Î and the current frame I′ from the video sequence. The synthetic frame Î is obtained by rendering the 3-D model (synthesis part) with the previously determined parameters. This approximative solution is used to compensate the differences between the two frames by rendering the deformed 3-D model at the new position. The synthetic frame now approximates the camera frame much better. The remaining linearization errors are reduced by iterating through different levels of resolution. By estimating the parameter changes with a synthetic frame that corresponds to the 3-D model, an error accumulation over time is avoided.

Linear Illumination Analysis

For natural video capture conditions, scene lighting often varies over time. This illumination variability has a considerable influence not only on the visual appearance of the objects in the scene, but also on the performance of computer vision algorithms or video-coding methods. The efficiency and robustness of these algorithms can be significantly improved by removing the undesired effects of changing illumination. In this section, we introduce a 3-D model-based technique for estimating and manipulating the lighting in an image sequence (Eisert et al., 2002). The current scene lighting is estimated for each frame exploiting 3-D model information and by synthetic re-lighting of the original video frames. To provide the estimator with surface-normal information, the objects in the scene are represented by 3-D shape models and their motion and deformation are tracked over time using a model-based estimation method. Given the normal information, the current lighting is estimated with a linear algorithm of low computational complexity using an orthogonal set of light maps.

Light Maps

Instead of explicitly modeling light sources and surface reflection properties in the computer graphics scene and calculating shading effects during the rendering process as it is done in by some researchers (Bozdagi et al., 1994; Stauder, 1995; and Eisert et al., 1998b), the shading and shadowing effects are here described by a linear superposition of several light maps which are attached to the object surface. Light maps are, similar to texture maps, two-dimensional images that are wrapped around the object containing shading, instead of color information. During rendering, the unshaded texture map $I_{tex}^{C}(\mathbf{u})$ with $C \in \{R, G, B\}$ representing the three color components and the light map L(**u**) are multiplied according to

$$I^{C}(\mathbf{u}) = I_{tex}^{C}(\mathbf{u}) \cdot L(\mathbf{u}) \tag{3}$$

in order to obtain a shaded texture map $I^{C}(\mathbf{u})$. The two-dimensional coordinate **u** specifies the position in both texture map and light map that are assumed to have the same mapping to the surface. For a static scene and viewpoint-independent surface reflections, the light map can be computed off-line which allows the use of more sophisticated shading methods as, e.g., radiosity algorithms (Goral et al., 1984), without slowing down the final rendering. This approach, however, can only be used if both object and light sources do not move. To overcome this limitation, we use a linear combination of scaled light maps instead of a single one

$$I^{C}(\mathbf{u}) = I_{tex}^{C}(\mathbf{u}) \cdot \sum_{i=0}^{N-1} \alpha_i^C L_i(\mathbf{u}). \tag{4}$$

By varying the scaling parameter α_i^C and thus blending between different light maps L$_i$, different lighting scenarios can be created. Moreover, the light map approach can also model wrinkles and creases which are difficult to describe by

3-D geometry (Pighin et al., 1998; Liu et al., 2001). The N light maps $L_i(\mathbf{u})$ can be computed off-line with the same surface normal information $\mathbf{n}(\mathbf{u})$, but with different light source configurations. In our experiments, we use one constant light map L_0 representing ambient illumination while the other light maps are calculated assuming Lambert reflection and point-light sources located at infinity having illuminant direction $\mathbf{l}_i$

$$\begin{aligned} L_0(\mathbf{u}) &= 1 \\ L_i(\mathbf{u}) &= \max\{-\mathbf{n}(\mathbf{u})\cdot\mathbf{l}_i, 0\}, \quad 1 \le i \le N-1. \end{aligned} \qquad (5)$$

This configuration can be interpreted as an array of point-light sources whose intensities and colors can be individually controlled by the parameters α_i^C. Figure 2 shows an example of such an array with the illuminant direction varying between -60° and 60° in longitudinal and latitudinal direction, respectively.

Figure 2: Array of light maps for a configuration with 7 by 7 light sources.

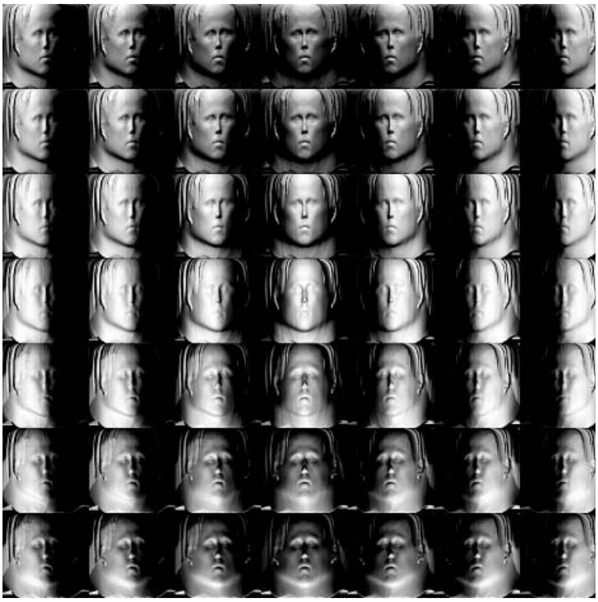
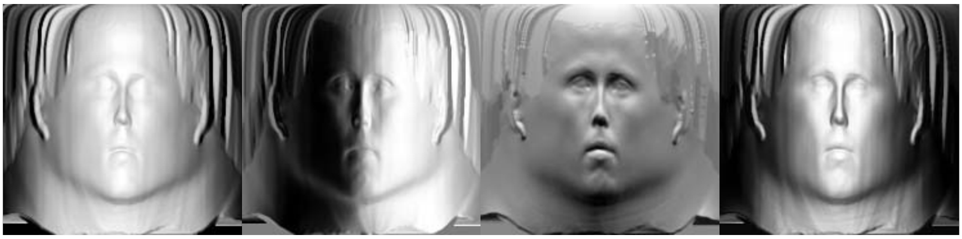


Figure 3: First four eigen light maps representing the dominant shading effects.



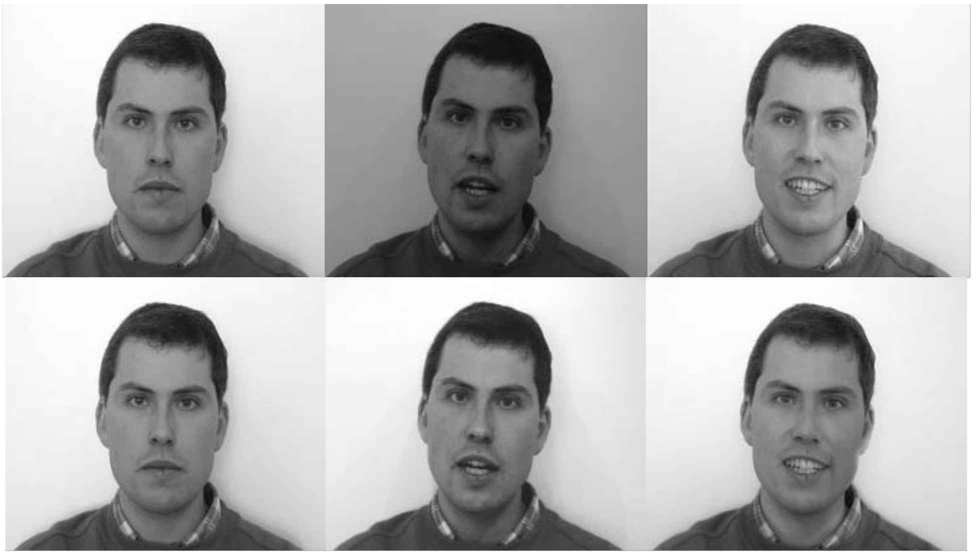
Eigen Light Maps

In order to reduce the number of unknowns α_i^C that have to be estimated, a smaller orthogonal set of light maps is used rather than the original one. A Karhunen-Loève transformation (KLT) (Turk et al., 1991) is applied to the set of light maps L_i with $1 \leq i \leq N-1$ creating *eigen light maps* which concentrate most energy in the first representations. Hence, the number of degrees of freedom can be reduced without significantly increasing the mean squared error when reconstructing the original set. Figure 3 shows the first four *eigen light maps* computed from a set of 50 different light maps. The mapping between the light maps and the 3-D head model is here defined by cylindrical projection onto the object surface.

Estimation of Lighting Properties

For the lighting analysis of an image sequence, the parameters α_i^C have to be estimated for each frame. This is achieved by tracking motion and deformation of the objects in the scene as described above and rendering a synthetic motion-compensated model frame using the unshaded texture map I_{tex}^C. From the pixel intensity differences between the camera frame $I_{shaded}^C(\mathbf{x})$ with $\mathbf{x}$ being the pixel position and the model frame $I_{unshaded}^C(\mathbf{x})$, the unknown parameters α_i^C are derived. For each pixel $\mathbf{x}$, the corresponding texture coordinate $\mathbf{u}$ is determined and the linear equation

Figure 4: Upper row: Original video frames; Lower row: Corresponding frames of illumination-compensated sequence with constant lighting.



$$I^{C}_{shaded}(\mathbf{x}) = I^{C}_{unshaded}(\mathbf{x}) \cdot \sum_{i=0}^{N-1} \alpha_i^C L_i(\mathbf{u}(\mathbf{x})). \qquad (6)$$

is set up. Since each pixel **x** being part of the object contributes one equation, a highly over-determined linear system of equations is obtained that is solved for the unknown α_i^C 's in a least-squares sense. Rendering the 3-D object model with the shaded texture map using the estimated parameters ,α_i^C leads to a model frame which approximates the lighting of the original frame. In the same way, the inverse this formula can be used to remove the lighting variations in real video sequences as it is shown in Figure 4.

Applications

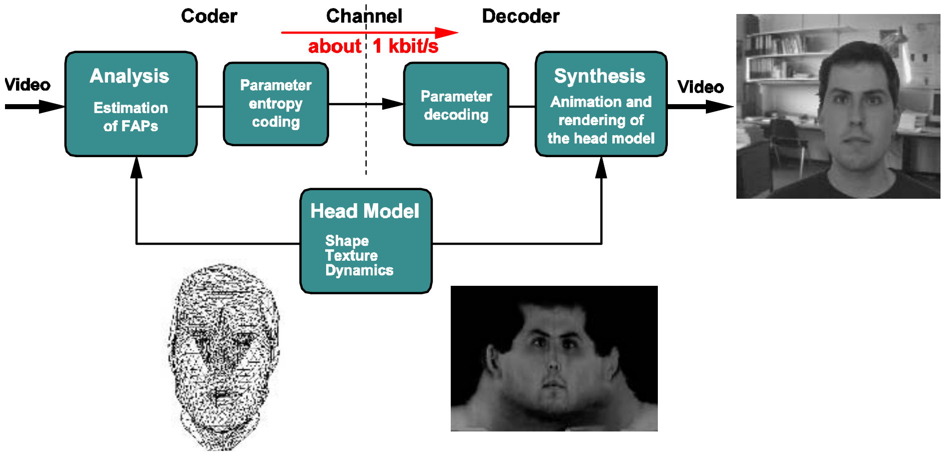
In this section, two applications, model-based coding and facial animation, are addressed which make use of the aforementioned methods for facial expression analysis and synthesis. Experimental results from the approach in Eisert (2000) are provided in order to illustrate the applicability of model-based techniques to these applications.

Model-Based Coding

In recent years, several video coding standards, such as H.261/3 and MPEG-1/2/4 have been introduced to address the compression of digital video for storage and communication services. These standards describe a hybrid video coding scheme, which consists of block-based motion-compensated prediction (MCP) and DCT-based quantification of the prediction error. The recently determined H.264 standard also follows the same video coding approach. These waveform-based schemes utilize the statistics of the video signal without knowledge of the semantic content of the frames and achieve compression ratios of several hundreds-to-one at a reasonable quality.

If semantic information about a scene is suitably incorporated, higher coding efficiency can be achieved by employing more sophisticated source models. Model-based video codecs, e.g., use 3-D models for representing the scene content. Figure 5 shows the structure of a model-based codec for the application of video telephony.

Figure 5: Structure of a model-based codec.



A video camera captures images of the head-and-shoulder part of a person. The encoder analyzes the frames and estimates 3-D motion and facial expressions of the person using a 3-D head model. A set of facial animation parameters (FAPs) is obtained that describes — together with the 3-D model — the current appearance of the person. Only a few parameters have to be encoded and transmitted, resulting in very low bit-rates. The head model has to be transmitted only once if it has not already been stored at the decoder in a previous session. At the decoder, the parameters are used to deform the head model according to the person's facial expressions. The original video frame is finally approximated by rendering the 3-D model at the new position.

The use of model-based coding techniques in communication scenarios leads to extremely low bit-rates of only a few kbit/s for the transmission of head-and-shoulder image sequences. This also enables video streaming over low-bandwidth channels for mobile devices like PDAs or smart phones. The rendering complexity is comparable to that of a hybrid video codec and, in experiments, frame rates of 30 Hz have been achieved on an iPAQ PDA. On the other hand, the intensive exploitation of a-priori knowledge restricts the applicability to special scenes that can be described by 3-D models available at the decoder. In a video-phone scenario, e.g., other objects like a hand in front of the face simply do not show up unless explicitly modeled in the virtual scene. In order to come up with a codec that is able to encode arbitrary scenes, hybrid coding techniques can be incorporated, increasing bit-rate but assuring generality to unknown objects. The model-aided codec is an example of such an approach (Eisert et al., 2000). Model-based coding techniques, however, also offer additional features besides low bit-rates, enabling many new applications that cannot be achieved with traditional hybrid coding methods. In immersive video-conferencing (Kauff et al., 2002), multiple participants who are located at different places can be seated at a joint virtual table. Due to the 3-D representation of the objects, pose modification for correct seating positions can easily be accomplished, as well as view-point corrections according to the user's motion. By replacing the 3-D model of one person by a different one, other people can be animated with the expressions of an actor as shown in the next section. Similarly, avatars can be driven to create user-friendly man-machine interfaces, where a human-like character interacts with the user. Analyzing the user with a web cam also gives the computer feedback about the user's emotions and intentions (Picard, 1997). Other cues in the face can assist the computer-aided diagnosis and treatment of patients in medical applications. For example, asymmetry in facial expressions caused by facial palsy can be measured three-dimensionally (Frey et al., 1999) or craniofacial syndromes can be detected by the 3-D analysis of facial feature positions (Hammond et al., 2001). These examples indicate the wide variety of applications for model-based facial analysis and synthesis techniques.

Model-Based View Synthesis

In this section, experimental results of a model-based video coding scheme using facial expression analysis are presented.

Figure 6 shows a head-and-shoulder video sequence recorded with a camera in CIF resolution at 25 Hz. A generic head model is roughly adjusted in shape to the person in the sequence and the first frame is projected onto the 3-D model. Non-visible areas of the texture map are extrapolated. The model is encoded and transmitted to the decoder and neither changed nor updated during the video sequence. Only facial animation parameters and lighting changes are streamed over the channel. In this experiment, 18 facial animation parameters are estimated, quantified, encoded, and transmitted. The frames in the middle row of Figure 6 are synthesized from the deformed 3-D model, which is illustrated in the lower row of Figure 6 by means of a wireframe. The bit-rate needed to encode these parameters is below 1 kbit/s at a quality of 34.6 dB PSNR. The PSNR between synthesized and original frames is here measured only in the facial area

Figure 6: Upper row: Original video sequence; Middle row: Synthesized sequence; Lower row: Hidden line representation.

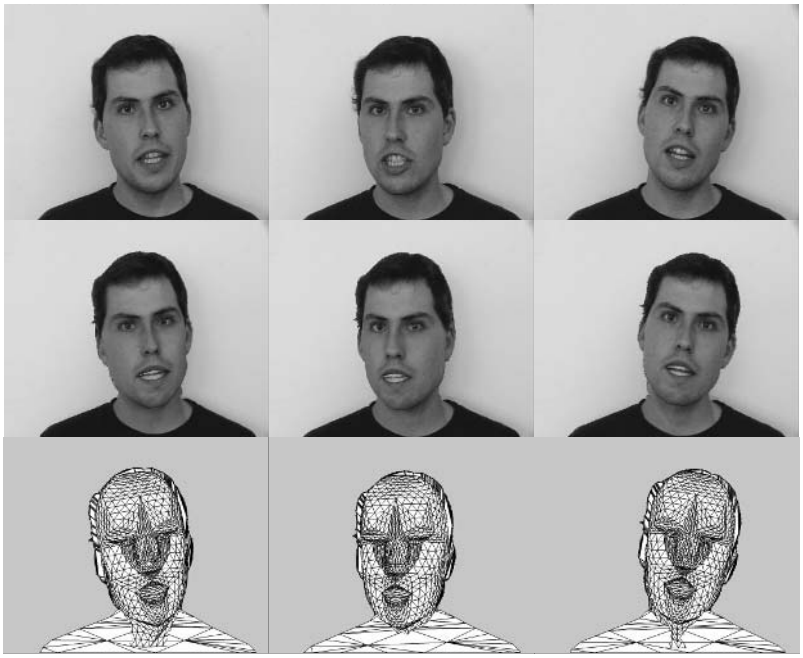
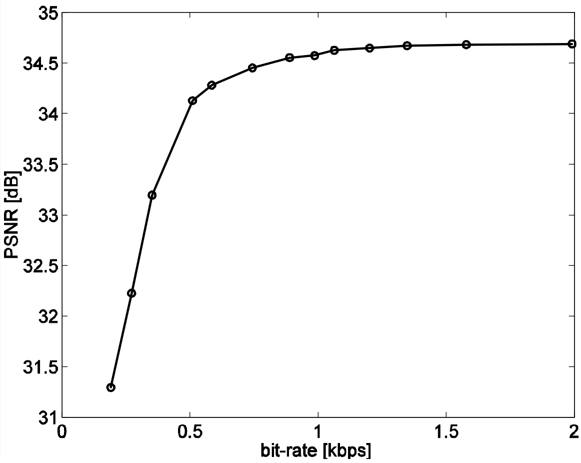


Figure 7: Reconstruction quality in PSNR over bit-rate needed for encoding the animation parameters.

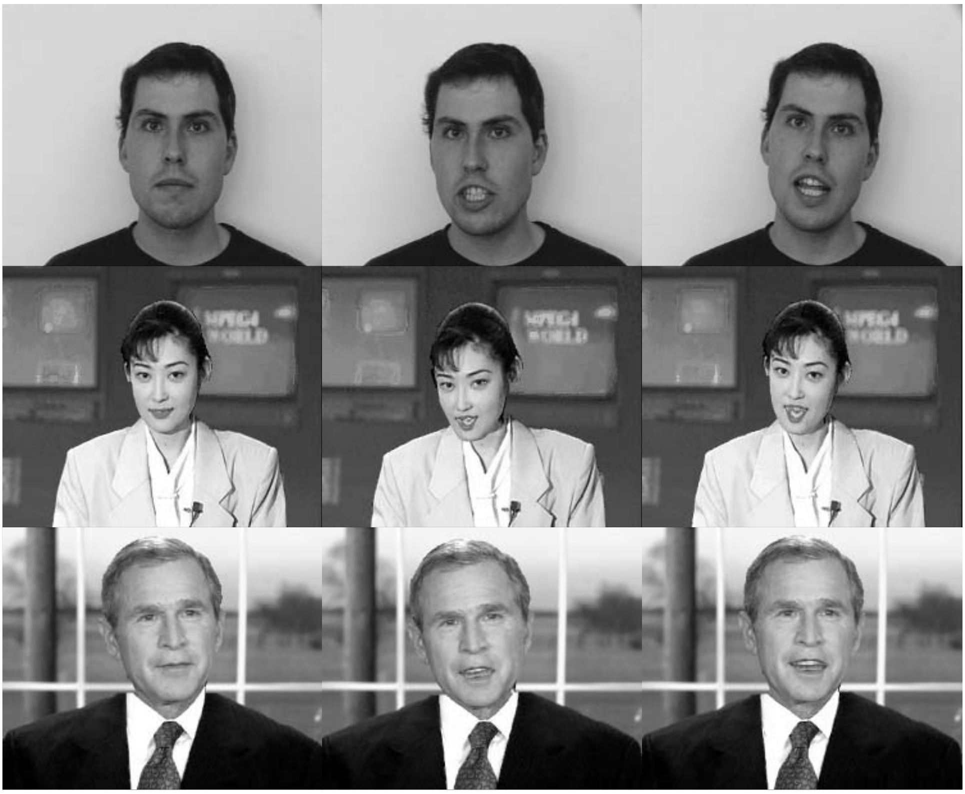


to exclude effects from the background, which is not explicitly modeled. The trade-off between bit-rate, which can be controlled by changing the quantifying values for the FAPs, and reconstruction quality is shown in Figure 7.

Facial Animation

The use of different head models for analysis and synthesis of head-and-shoulder sequences is also interesting in the field of character animation in film productions or web applications. The facial play of an actor sitting in front of a camera is analyzed and the resulting FAPs are used to control arbitrary 3-D models. This way, different people, animals, or fictitious creatures can be animated realistically. The exchange of the head model to animate other people is shown in Figure 8. The upper row depicts some frames of the original sequence used for facial expression analysis. Instead of rendering the sequence with the same 3-D head model used for the FAP estimation, and thus reconstructing the original sequence, the head model is exchanged for image synthesis leading to new sequences with different people that move according to the original sequence. Examples of this character animation are shown in the lower two rows of Figure 8. In these experiments, the 3-D head models for *Akiyo* and *Bush* are derived from a single image. A generic head model whose shape is controlled by a set of parameters is roughly adjusted to the outline of the face and the position of eyes and mouth. Then, the image is projected onto the 3-D model and used as

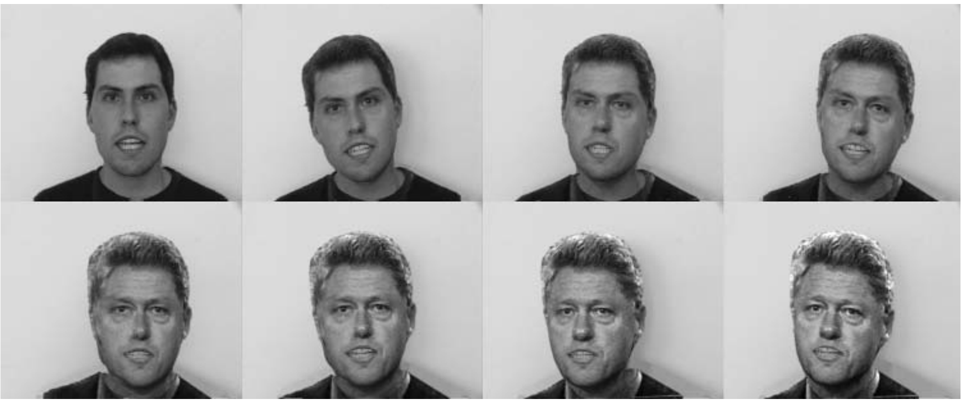
Figure 8: Animation of different people using facial expressions from a reference sequence. Upper row: Reference sequence; Middle and lower row: Synthesized new sequences.



a texture map. Since the topology of the mesh is identical for all models, the surface deformation description need not be changed and facial expressions can easily be applied to different people.

Since the same generic model is used for all people, point correspondences between surface points and texture coordinates are inherently established. This enables the morphing between different characters by linearly blending between the texture map and the position of the vertices. In contrast to 2-D approaches (Liu et al., 2001), this might be done during a video sequence due to use of a 3-D model. Local deformations caused by facial expressions are not affected by this morphing. Figure 9 shows an example of a view of the morphing process between two different people.

Figure 9: Motion-compensated 3-D morph between two people.



Conclusions

Methods for facial expression analysis and synthesis have received increasing interest in recent years. The computational power of current computers and handheld devices like PDAs already allow a real-time rendering of 3-D facial models, which is the basis for many new applications in the near future. Especially for handheld devices that are connected to the Internet via a wireless channel, bit-rates for streaming video is limited. Transmitting only facial expression parameters drastically reduces the bandwidth requirements to a few kbit/s. In the same way, face animations or new human-computer interfaces can be realized with low demands on storage capacities. On the high quality end, film productions may get new impacts for animation, realistic facial expression, and motion capture without the use of numerous sensors that interfere with the actor. Last, but not least, information about motion and symmetry of facial features can be exploited in medical diagnosis and therapy.

All these applications have in common that accurate information about 3-D motion deformation and facial expressions is required. In this chapter, the state-of-the-art in facial expression analysis and synthesis has been reviewed and a new method for determining FAPs from monocular images sequences has been presented. In a hierarchical framework, the parameters are robustly found using optical flow information together with explicit knowledge about shape and motion constraints of the objects. The robustness can further be increased by incorporating photometric properties in the estimation. For this purpose, a computationally efficient algorithm for the determination of lighting effects was given. Finally, experiments have shown that video transmission of head-and-shoulder scenes can be realized at data rates of a few kbit/s, even with today's technologies, enabling a wide variety of new applications.

References

Aizawa, K. & Huang, T. S. (1995). Model-based image coding: Advanced video coding techniques for very low bit-rate applications. *Proc. IEEE*, 83(2), 259-271.

Aizawa, K., Harashima, H. & Saito, T. (1989). Model-based analysis synthesis image coding (MBASIC) system for a person's face. *Sig. Proc.: Image Comm.*, 1(2), 139-152.

Anjyo, K., Usami, Y. & Kurihara, T. (1992). A simple method for extracting the natural beauty of hair. *SIGGRAPH*, 26, 111-120.

Baribeau, R., Rioux, M. & Godin, G. (1992). Color reflectance modeling using a polychromatic laser range sensor. *IEEE Tr. PAMI*, 14(2), 263-269.

Barron, J. L., Fleet, D. J. & Beauchemin, S. S. (1994). Systems and experiment: Performance of optical flow techniques. *International Journal of Comp. Vision*, 12(1), 43-77.

Bartlett, M. et al. (1995). Classifying facial action. *Advances in Neural Inf. Proc. Systems 8*, MIT Press, 823-829.

Belhumeur, P. N. & Kriegman, D. J. (1998). What is the set of images of an object under all possible illumination conditions. *International Journal of Comp. Vision*, 28(3), 245-260.

Black, M. J. & Anandan, P. (1996). The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1), 75-104.

Black, M. J. & Yacoob, Y. (1995). Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. *Proc. ICCV*, 374-381.

Black, M. J., Yacoob, Y. & Ju, S. X. (1997). Recognizing human motion using parameterized models of optical flow. *Motion-Based Recognition*, 245-269. Kluwer Academic Publishers.

Blanz, V. & Vetter, T. (1999). A morphable model for the synthesis of 3D faces. *SIGGRAPH*, 187-194.

Blohm, W. (1997). Lightness determination at curved surfaces with apps to dynamic range compression and model-based coding of facial images. *IEEE Tr. Image Proc.*, 6(8), 1129-1138.

Bozdagi G., Tekalp A. M. & Onural L. (1994). 3-D motion estimation and wireframe adaption including photometric effects for model-based coding of facial image sequences. *IEEE Tr. CSVT*, 4(3), 246-256.

Brunelli, R. (1997). Estimation of pose and illuminant direction for face processing. *Image-and-Vision-Computing*, 15(10), 741-748.

Brunelli, R. & Poggio, T. (1993). Face recognition: Features versus templates. *IEEE Tr. PAMI*, 15(10), 1042-1052.

Chao, S. & Robinson, J. (1994). Model-based analysis/synthesis image coding with eye and mouth patch codebooks. *Proc. of Vision Interface*, 104-109.

Chellappa, R., Wilson, C. L. & Sirohey, S. (1995). Human and machine recognition of faces: A survey. *Proc. IEEE*, 83(5), 705-740.

Choi, C., Aizawa, K., Harashima, H. & Takebe, T. (1994). Analysis and synthesis of facial image sequences in model-based image coding. *IEEE Tr. CSVT*, 4(3), 257-275.

DeCarlo, D. & Metaxas, D. (1996). The integration of optical flow and deformable models with applications to human face shape and motion estimation. *Proc. CVPR*, 231-238.

DeCarlo, D. & Metaxas, D. (1998). Deformable model-based shape and motion analysis from images using motion residual error. *Proc. ICCV*, 113-119.

DeCarlo, D., Metaxas, D. & Stone, M. (1998). An anthropometric face model using variational techniques. *SIGGRAPH*, 67-74.

Deshpande, S. G. & Chaudhuri, S. (1996). Recursive estimation of illuminant motion from flow field. *Proc. ICIP*, 3, 771-774.

Donato, G., Bartlett, M. S., Hager, J. C., Ekman, P. & Sejnowski, T. (1999). Classifying facial actions. *IEEE Tr. PAMI*, 21(10), 974-989.

Dufaux, F. & Moscheni, F. (1995). Motion estimation techniques for digital TV: A review and a new contribution. Proc. IEEE, 83(6), 858-876.

Eisert, P. (2000). *Very Low Bit-Rate Video Coding Using 3-D Models*. Ph.D. thesis, University of Erlangen, Shaker Verlag, Aachen, Germany.

Eisert, P. & Girod, B. (1997). Model-based 3D motion estimation with illumination compensation. *Proc. International Conference on Image Proc. and Its Applications*, 1, 194-198.

Eisert, P. & Girod, B. (1998). Analyzing facial expressions for virtual conferencing. *IEEE Computer Graphics and Applications*, 18(5), 70-78.

Eisert, P. & Girod, B. (1998b). Model-based coding of facial image sequences at varying illumination conditions. *Proc. 10th IMDSP. Workshop 98*, 119-122.

Eisert, P. & Girod, B. (2002). Model-based enhancement of lighting conditions in image sequences. *Proc. SPIE VCIP, VCIP-02.*

Eisert, P., Wiegand, T. & Girod, B. (2000). Model-aided coding: A new approach to incorporate facial animation into motion-compensated video coding. *IEEE Tr. CSVT,* 10(3), 344-358.

Ekman, P. & Friesen, W. V. (1978). *Facial Action Coding System*. Palo Alto, CA: Consulting Psychologists Press.

Enkelmann, W. (1988). Investigations of multigrid algorithms for estimation of optical flow fields in image sequences. *Comp. Vision, Graphics and Image Proc.,* 43(2), 150-177.

Epstein, R., Hallinan, P. & Yuille, A. (1995). 5 plus or minus 2 eigenimages suffice: An empirical investigation of low-dimensional lighting models. *Proc. IEEE Workshop on Physics-based Modeling in Comp. Vision.*

Essa, I. A. & Pentland, A. P. (1994). A vision system for observing and extracting facial action parameters. *Proc. CVPR*, 76-83.

Essa, I. A. & Pentland, A. P. (1997). Coding, analysis, interpretation, and recognition of facial expressions. *IEEE Tr. PAMI,* 19(7), 757-763.

Farkas, L. G. (1995). *Anthropometry of the Head and Face.* Raven Press.

Foley, J. D., van Dam, A., Feiner, S. K. & Hughes, J. F. (1990). *Computer Graphics, Principles and Practice.* Addison-Wesley.

Forsey, D. R. & Bartels, R. H. (1988). Hierarchical B-spline refinement. *SIGGRAPH,* 22, 205-212.

Forsyth, D. & Zisserman, A. (1991). Reflections on shading. *IEEE Tr. PAMI,* 13(7), 671-679.

Frey, M., Giovanoli, P., Gerber, H., Slameczka, M. & Stüssi, E. (1999). Three-dimensional video analysis of facial movements: A new method to assess the quantity and quality of the smile. *Plastic and Reconstructive Surgery,* 104(7), 2032-2039.

Gee, A. & Cipolla, R. (1994). Determining the gaze of faces in images. *Image and Vision Computing,* 12(10), 639-647.

Georghiades, A. S., Belhumeur, P. N. & Kriegman, D. J. (1999). Illumination-based image synthesis: Creating novel images of human faces under different pose and lighting. *Proc. IEEE Work. on Multi-View Modeling and Analysis of Visual Scenes.*

Gennert, M. A. & Negahdaripour, S. (1987). *Relaxing the brightness constancy assumption in computing optical flow.* Technical report, MIT AI Lab Memo No. 975.

Goral, C. M., Torrance, K. E., Greenberg, D. P. & Battaile, B. (1984). Modeling the interaction of light between diffuse surfaces. *SIGGRAPH,* 18, 213-222.

Gortler, S. J., Grzeszczuk, R., Szeliski, R. & Cohen, M. F. (1996). The Lumigraph. *SIGGRAPH,* 43-54.

Hallinan, P. W. (1994). A low-dimensional representation of human faces for arbitrary lighting conditions. *Proc. CVPR.*

Hammond, P., Hutton, T. J., Patton, M. A. & Allanson, J. E. (2001). Delineation and visualisation of congenital abnormality using 3D facial images. *Intell. Data Analysis in Medicine and Pharm.*

Heckbert, P. S. (1992). Introduction to global illumination. *Global Illumination Course, SIGGRAPH.*

Hjortsjö, C. H. (1970). *Man's face and mimic language.* Student literature, Lund, Sweden.

Hoch, M., Fleischmann, G. & Girod, B. (1994). Modeling and animation of facial expressions based on B-splines. *Visual Computer,* 11, 87-95.

Horn, B. K. P. (1986). *Robot Vision.* Cambridge, MA: MIT Press.

Horn, B. K. P. & Brooks, M. J. (1989). *Shape from Shading*. Cambridge, MA: MIT Press.

Horn, B. K. P. & Schunck, B. G. (1981). Determining optical flow. *Artificial Intelligence,* 17(1-3), 185-203.

Hölzer, A. (1999). *Optimierung eines dreidimensionalen Modells menschlicher Gesichtsausdrücke für die Codierung von Videosequenzen.* Diploma thesis, University of Erlangen-Nuremberg.

Huang, T. S. & Netravali, A. N. (1994). Motion and structure from feature correspondences: A review. *Proc. IEEE*, 82(2), 252-268.

Huang, T. S., Reddy, S. & Aizawa, K. (1991). Human facial motion analysis and synthesis for video compression. *Proc. SPIE VCIP*, 234-241.

Ikeuchi, K. & Sato, K. (1991). Determining reflectance properties of an object using range and brightness images. *IEEE Tr. PAMI*, 13(11), 1139-1153.

Ip, H. H. S. & Chan, C. S. (1996). Script-based facial gesture and speech animation using NURBS based face model. *Computer and Graphics*, 20(6), 881-891.

ISO/IEC FDIS 14496-2. (1999). Generic Coding of audio-visual objects: (MPEG-4 video), Final Draft International Standard, Document N2502.

Kalberer, G. A. & Van Gool, L. (2001). Lip animation based on observed 3D speech dynamics. *Proc. SPIE VCIP,* 16-25.

Kaneko, M., Koike, A. & Hatori, Y. (1991). Coding of facial image sequence based on a 3-D model of the head and motion detection. *Journal of Visual Communication and Image Representation*, 2(1), 39-54.

Kappei, F. (1988). *Modellierung und Rekonstruktion bewegter dreidimensionaler Objekte in einer Fernsehbildfolge*. Ph.D. thesis, University Hannover.

Kass, M., Witkin, A. & Terzopoulos, D. (1987). Snakes: Active contour models. *International Journal of Computer Vision*, 1(4), 321-331.

Kauff, P. & Schreer, O. (2002). Virtual team user environments: A step from tele-cubicles towards distributed tele-collaboration in mediated workspaces. *Proc. ICME.*

Klinker, G. J., Shafer, S. A. & Kanade, T. (1990). A physical approach to color image understanding. *International Journal of Computer Vision*, 4, 7-38.

Koch, R. (1993). Dynamic 3-D scene analysis through synthesis feedback control. *IEEE Tr. PAMI,* 15(6), 556-568.

Land, E. H. & McCann, J. J. (1971). Lightness and retinex theory. *Journal of the Optometric Society of America*, 61, 1-11.

Lee, C. H. & Rosenfeld, A. (1989). *Shape from Shading.* In Improved Methods of Estimating Shape from Shading using the Light Source Coordinate System. Cambridge, MA: MIT Press.

Lee, Y., Terzopoulos, D. & Waters, K. (1995). Realistic modeling for facial animation. *SIGGRAPH,* 55-61.

Levoy, M. & Hanrahan, P. (1996). Light field rendering. *SIGGRAPH*, 31-42.

Li, H. (1993). *Low Bitrate Image Sequence Coding*. Ph.D. thesis, Linköping University. Linköping Studies in Science and Technology, No. 318.

Li, H. & Forchheimer, R. (1994). Two-view facial movement estimation. *IEEE Tr. CSVT*, 4(3), 276-287.

Li, H., Lundmark, A. & Forchheimer, R. (1994). Image sequence coding at very low bitrates: A review. *IEEE Tr. Image Proc.*, 3(5), 589-609.

Li, H., Roivainen, P. & Forchheimer, R. (1993). 3-D motion estimation in model-based facial image coding. *IEEE Tr. PAMI*, 15(6), 545-555.

Li, Y. & Chen, Y. (1998). A hybrid model-based image coding system for very low bit-rate coding. *IEEE Journal on Selected Areas in Communications*, 16(1), 28-41.

Liu, Z., Shan, Y. & Zhang, Z. (2001). Expressive expression mapping with ratio images. *SIGGRAPH.*

Longuet-Higgins, H. C. (1984). The visual ambiguity of a moving plane. *Proc. of the Royal Society of London*, B 223, 165-175.

Lopez, R. & Huang, T. S. (1995). 3D head pose computation from 2D images: Template versus features. *Proc. ICIP*, 599-602.

Moghaddam, B. & Pentland, A. (1997). Probabilistic visual learning for object representation. *IEEE Tr. PAMI*, 19(7), 696-710.

Moloney, C. R. (1991). Methods for illumination-independent processing of digital images. *IEEE Pacific Rim Conference on Communication, Computers and Sig. Proc.*, 2, 811-814.

Moloney, C. R. & Dubois, E. (1991). Estimation of motion fields from image sequences with illumination variation. *Proc. ICASSP*, 4, 2425-2428.

Nayar, S. K., Ikeuchi, K. & Kanade, T. (1991). Surface reflection: Physical and geometrical perspectives. *IEEE Tr. PAMI*, 13(7), 611-634.

Negahdaripour, S. & Yu, C. H. (1993). A generalized brightness change model for computing optical flow. *Proc. ICCV*, 2-11.

Netravali, A. N. & Salz, J. (1985). Algorithms for estimation of three-dimensional motion. *AT&T Technical Journal*, 64(2), 335-346.

Noh, J. Y. & Neumann, U. (2001). Expression cloning. *SIGGRAPH*.

Ono, E., Morishima, S. & Harashima, H. (1993). A model based shade estimation and reproduction schemes for rotational face. *Proc. PCS, 2.2*.

Ostermann, J. (1994). Object-based analysis-synthesis coding (OBASC) based on the source model of moving flexible 3D-objects. *IEEE Tr. Image Proc.*, 705-711.

Parke, F. I. (1982). Parameterized models for facial animation. *IEEE Computer Graphics and Applications*, 2(9), 61-68.

Parke, F. I. & Waters, K. (1996). *Computer Facial Animation*. Cambridge, MA: A. K. Peters.

Pearson, D. E. (1995). Developments in model-based video coding. *Proc. IEEE*, 83(6), 892-906.

Pei, S., Ko, C. & Su, M. (1998). Global motion estimation in model-based image coding by tracking three-dimensional contour feature points. *IEEE Tr. CSVT*, 8(2), 181-190.

Pentland, A. (1982). Finding the illuminant direction. *Journal of the Optometric Society of America*, 72(4), 170-187.

Pentland, A. (1991). Photometric motion. *IEEE Tr. PAMI*, 13(9), 879-890.

Picard, R. W. (1997). *Affective Computing*. Cambridge, MA: MIT Press.

Pighin, F., Hecker, J., Lischinski, D., Szeliski, R. & Salesin, H. D. (1998). Synthesizing realistic facial expressions from photographs. *SIGGRAPH*, 75-84.

Ramamoorthi, R. (2002). Analytic PCA construction for theoretical analysis of lighting variability in images of a Lambertian object. *IEEE Tr. PAMI*.

Ramamoorthi, R. & Hanrahan, P. (2001). A signal-processing framework for inverse rendering. *SIGGRAPH*.

Rydfalk, M. (1978). *CANDIDE: A Parameterized Face*. Ph.D. thesis, Linköping University, LiTH-ISY-I-0866.

Sato, Y. & Ikeuchi, K. (1996). Reflectance analysis for 3D computer graphics model generation. *Graphical Models and Image Processing*, 58(5), 437-451.

Sato, Y., Wheeler, M. D. & Ikeuchi, K. (1997). Object shape and reflectance modeling from observation. *SIGGRAPH*, 379-387.

Schlick, C. (1994). A survey of shading and reflectance models. *Computer Graphics Forum*, 13(2), 121-132.

Schluens, K. & Teschner, M. (1995). Analysis of 2D color spaces for highlight elimination in 3D shape reconstruction. *Proc. ACCV*, 2, 801-805.

Sezan, M. I. & Lagendijk, R. L. (1993). *Motion Analysis and Image Sequence Processing*, chapter Hierarchical Model-Based Motion Estimation, 1-22. Kluwer Academic Publishers.

Simoncelli, E. P. (1994). Design of multi-dimensional derivative filters. *Proc. ICIP*, 790-794.

Singh, A. (1990). An estimation theoretic framework for image-flow computation. *Proc. ICCV*, 168-177.

Stauder, J. (1994). Detection of highlights for object-based analysis-synthesis coding. Proc. PCS, 300-303.

Stauder, J. (1995). Estimation of point light source parameters for object-based coding. *Sig. Proc.: Image Comm.*, 7(4-6), 355-379.

Stauder, J. (1998). Illumination analysis for synthetic/natural hybrid image sequence generation. *Comp. Graphics Intern. (CGI 98)*, 506-511.

Tarr, M. J. (1998). Why the visual recognition system might encode the effects of illumination. *Vision Research*, 38, 2259-2275.

Terzopoulos, D. & Waters, K. (1993). Analysis and synthesis of facial image sequences using physical and anatomical models. *IEEE Tr. PAMI*, 15(6), 569-579.

Thomas, G. A. & Hons, B. A. (1987). *Television motion measurement for DATV and other applications*. BBC Research Department Report, 1-20.

Tominaga, S. & Tanaka, N. (2000). Estimating reflection parameters from a single color image. IEEE *Computer Graphics and Applications*, 20(5), 58-66.

Torrance, K. E. & Sparrow, E. M. (1967). Theory of off-specular reflection from roughened surfaces. *Journal of the Optometric Society of America*, 1105-1114.

Treves, P. & Konrad, J. (1994). Motion estimation and compensation under varying illumination. *Proc. ICIP*, I, 373-377.

Turk, M. & Pentland, A. (1991). Eigenfaces for recognition. *Journal for Cognitive Neuroscience*, 3(1), 71-86.

Verri, A. & Poggio, T. (1989). Motion field and optical flow: Qualitative properties. *IEEE Tr. PAMI*, 11(5), 490-498.

Vetter, T. & Blanz, V. (1998). Estimating coloured 3D face models from single images: An example based approach. *Proc. ECCV*, 2, 499-513.

Wada, T., Ukida, H. & Matsuyama, T. (1995). Shape from shading with interreflections under proximal light source. *Proc. ICCV*, 66-71.

Watanabe, Y. & Suenaga, Y. (1992). A trigonal prism-based method for hair image generation. *IEEE Computer Graphics and Applications*, 12(1), 47-53.

Waters, K. (1987). A muscle model for animating three-dimensional facial expressions. *SIGGRAPH*, 21, 17-24.

Waxman, A. M., Kamgar-Parsi, B. & Subbarao, M. (1987). Closed-form solutions to image flow equations for 3D structure and motion. *International Journal of Comp. Vision*, 1, 239-258.

Welsh, W. J., Searsby, S. & Waite, J. B. (1990). Model-based image coding. *British Telecom Technology Journal*, 8(3), 94-106.

Yuille, A. L. (1991). Deformable templates for face recognition. *Journal of Cognitive Neuroscience*, 3(1), 59-70.

Zheng, Q. & Chellappa, R. (1991). Estimation of illuminant direction, albedo, and shape from shading. *IEEE Tr. PAMI*, 13(7), 680-702.

Chapter VIII

Modeling and Synthesis of Realistic Visual Speech in 3D

Gregor A. Kalberer
BIWI – Computer Vision Lab, Switzerland

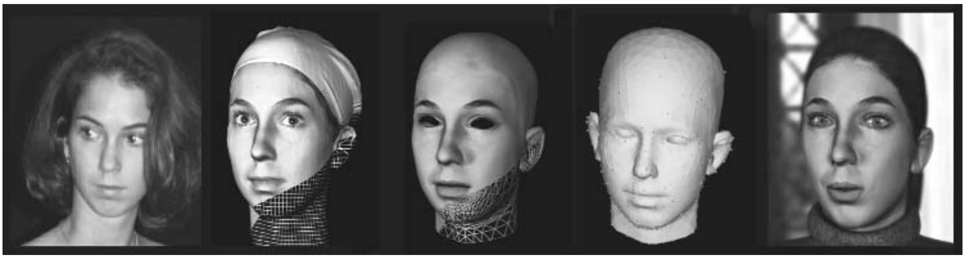
Pascal Müller
BIWI – Computer Vision Lab, Switzerland

Luc Van Gool
BIWI – Computer Vision Lab, Switzerland and VISICS, Belgium

Abstract

The problem of realistic face animation is a difficult one. This is hampering a further breakthrough of some high-tech domains, such as special effects in the movies, the use of 3D face models in communications, the use of avatars and likenesses in virtual reality, and the production of games with more subtle scenarios. This work attempts to improve on the current state-of-the-art in face animation, especially for the creation of highly realistic lip and speech-related motions. To that end, 3D models of faces are used and — based on the latest technology — speech-related 3D face motion will be learned from examples. Thus, the chapter subscribes to the surging field of image-based modeling and widens its scope to include animation. The exploitation of detailed 3D motion sequences is quite unique, thereby

Figure 1. The workflow of our system: (a) An original face is (b) captured, (c) re-meshed, (d) analyzed and integrated for (e) an animation.



narrowing the gap between modeling and animation. From measured 3D face deformations around the mouth area, typical motions are extracted for different "visemes". Visemes are the basic motion patterns observed for speech and are comparable to the phonemes of auditory speech. The visemes are studied with sufficient detail to also cover natural variations and differences between individuals. Furthermore, the transition between visemes is analyzed in terms of co-articulation effects, i.e., the visual blending of visemes as required for fluent, natural speech. The work presented in this chapter also encompasses the animation of faces for which no visemes have been observed and extracted. The "transplantation" of visemes to novel faces for which no viseme data have been recorded and for which only a static 3D model is available allows for the animation of faces without an extensive learning procedure for each individual.

Introduction

Realistic face animation for speech still poses a number of challenges, especially when we want to automate it to a large degree. Faces are the focus of attention for an audience, and the slightest deviation from normal faces and face dynamics is noticed.

There are several factors that make facial animation so elusive. First, the human face is an extremely complex geometric form. Secondly, the face exhibits countless tiny creases and wrinkles, as well as subtle variations in color and texture, all of which are crucial for our comprehension and appreciation of facial expressions. As difficult as the face is to model, it is even more problematic to animate. Facial deformations are a product of the underlying skeletal and muscular forms, as well as the mechanical properties of the skin and subcutaneous layers, which vary in thickness and composition in different parts of the face. The mouth area is particularly demanding, because there are additional movements of the mandible and intra-oral air pressures, which influence the visible morphology of this area. All of these problems are enormously magnified by the fact that we as humans have an uncanny ability to read expressions and lips — an ability that is not merely a learned skill, but part of our deep-rooted instincts. For facial expressions, the slightest deviation from reality is something any person will immediately detect. This said, people would find it difficult to put their finger on what exactly it is that was wrong. We have to deal with subtle effects that leave strong impressions.

Face animation research dates back to the early 70s (Parke, 1972). Since then, the level of sophistication has increased dramatically. For example, the human head models used in Pixar's *Toy Story* had several thousand control points each (Eben, 1997). More recent examples, such as *Final Fantasy* and *Lord of the Rings,* demonstrate that now a level of realism can be achieved that allows "virtual humans" to play a lead part in a feature movie. Nevertheless, there is still much manual work involved.

For face animation, both 2D image-based and 3D model-based strategies have been proposed. Basically, the choice was one between photorealism and flexibility.

2D: For reaching photorealism, one of the most effective approaches has been to reorder short video sequences (Bregler et al., 1997) or to 2D morph between photographic images (Beier et al., 1992; Bregler et al., 1995; and Ezzat et al., 2000). A problem with such techniques is that they do not allow much freedom in face orientation, relighting or compositing with other 3D objects.

3D: A 3D approach typically yields such flexibility. Here, a distinction can be made between appearance-based and physics-based approaches. The former is typically based on scans or multi-view reconstructions of the face exterior. Animation takes the form of 3D morphs between several, static expressions (Chen et al., 1995; Blanz et al., 1999; and Pighin et al., 1998) or a more detailed replay of observed face dynamics (Guenter et al., 1998; Lin et al., 2001). Physics-based approaches model the underlying anatomy in detail, as a skull with layers of muscles and skin (Waters et al., 1995; Pelachaud et al., 1996; Eben, 1997; and Kähler et al., 2002). The activation of the virtual muscles drives the animation. Again, excellent results have been demonstrated. Emphasis has often been on the animation of emotions.

So far, what seems to be lacking is highly realistic animation of *speech for novel character*s.

2.5D: Cosatto (Cosatto et al., 2000; Cosatto, 2002) developed a 2.5D talking head as a clever combination of 2D and 3D techniques. Image sequences are mapped onto a crude head model composed of different 3D parts. Photorealism is combined with maximal head rotations of ± 15 degrees.

We present a system for realistic face animation focused on speech — a system that can help to automate the process further, while not sacrificing too much realism. The approach is purely 3D. Since people can clearly tell good animations from bad ones without any knowledge about facial anatomy, we go for the relative simplicity of the appearance-based school. Realism comes through the extensive use of detailed motion-capture data. The system also supports the animation of novel characters based on their static head model, but with dynamics, which, nevertheless, are adapted to their physiognomy.

Viseme Selection

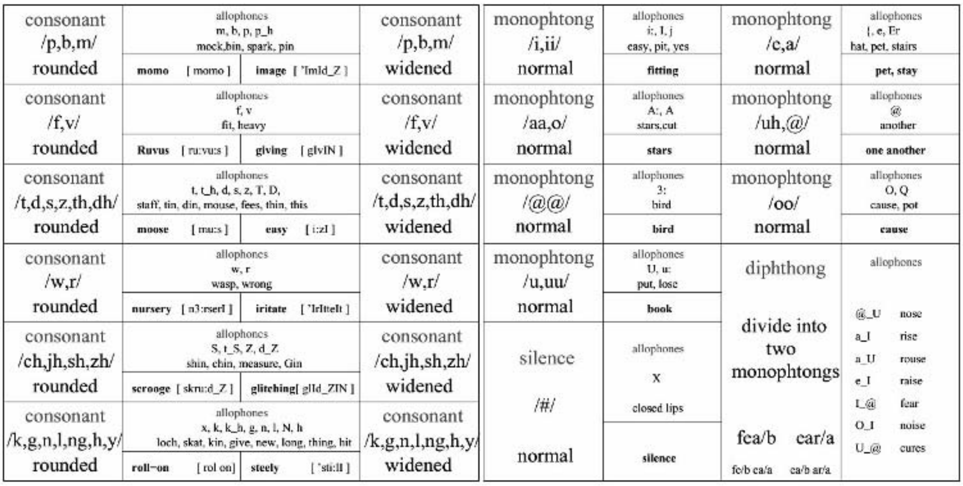
Animation of speech has much in common with speech synthesis. Rather than composing a sequence of phonemes according to the laws of co-articulation to get the transitions between the phonemes right, the animation generates sequences of visemes. Visemes correspond to the basic, visual mouth expressions that are observed in speech. Whereas there is a reasonably strong consensus about the set of phonemes, there is less unanimity about the selection of visemes. Approaches aimed at realistic animation of speech have used any number, from as few as 16 (Ezzat et al., 2000) up to about 50 visemes (Scott et al., 1994). This number is by no means the only parameter in assessing the level of sophistication of different schemes. Much also depends on the addition of co-articulation effects. There certainly is no simple one-to-one relation between the 52 phonemes and the visemes, as different sounds may look the same and, therefore, this mapping is rather many-to-one. For instance **/b/** and **/p/** are two bilabial stops which differ only in the fact that the former is voiced, while the latter is voiceless. Visually, there is hardly any difference in fluent speech.

We based our selection of visemes on the work of Owens (Owens et al., 1985) for consonants. We use his consonant groups, except for two of them, which we combine into a single **/k,g,n,l,ng,h,y/** viseme. The groups are considered as single visemes because they yield the same visual impression when uttered. We do not consider all the possible instances of different, neighboring vocals that

Owens distinguishes, however. In fact, we only consider two cases for each cluster: rounded and widened, that represent the instances farthest from the neutral expression. For instance, the viseme associated with **/m/** differs depending on whether the speaker is uttering the sequence **omo** or **umu** vs. the sequence **eme** or **imi**. In the former case, the **/m/** viseme assumes a rounded shape, while the latter assumes a more widened shape. Therefore, each consonant was assigned to these two types of visemes. For the visemes that correspond to vocals, we used those proposed by Montgomery et al. (1985).

As shown in Figure 2, the selection contains a total of 20 visemes: 12 representing the consonants (boxes with "consonant" title), seven representing the monophtongs (boxes with title "monophtong") and one representing the neutral pose (box with title "silence"). Diphtongs (box with title "diphtong") are divided into two, separate monophtongs and their mutual influence is taken care of as a co-articulation effect. The boxes with the smaller title "allophones" can be discarded by the reader for the moment. The table also contains examples of words producing the visemes when they are pronounced. This viseme selection differs from others proposed earlier. It contains more consonant visemes than most, mainly because the distinction between the rounded and widened shapes is made systematically. For the sake of comparison, Ezzat and Poggio (Ezzat et al., 2000) used six (only one for each of Owens' consonant groups, while also combining two of them), Bregler et al. (1997) used ten (same clusters, but they subdivided the cluster **/t,d,s,z,th,dh/** into **/th,dh/** and the rest, and **/k,g,n,l, ng,h,y/** into **/ng/**, **/h/**, **/y/**, and the rest, what boils down to making an even more precise subdivision for this cluster), and Massaro (1998) used nine (but this

Figure 2. Overview of the visemes used.



consonant /p,b,m/ rounded	allophones m, b, p, p_h mock,bin, spark, pin	**momo** [momo]	**image** ['Imld_Z]	consonant /p,b,m/ widened
consonant /f,v/ rounded	allophones f, v fit, heavy	**Ruvus** [ru:vu:s]	**giving** [glvIN]	consonant /f,v/ widened
consonant /t,d,s,z,th,dh/ rounded	allophones t, t_h, d, s, z, T, D, staff, tin, din, mouse, fees, thin, this	**moose** [mu:s]	**easy** [i:zl]	consonant /t,d,s,z,th,dh/ widened
consonant /w,r/ rounded	allophones w, r wasp, wrong	**nursery** [n3:rserl]	**iritate** ['lrltelt]	consonant /w,r/ widened
consonant /ch,jh,sh,zh/ rounded	allophones S, t_S, Z, d_Z shin, chin, measure, Gin	**scrooge** [skru:d_Z]	**glitching**[glId_ZIN]	consonant /ch,jh,sh,zh/ widened
consonant /k,g,n,l,ng,h,y/ rounded	allophones x, k, k_h, g, n, l, N, h loch, skat, kin, give, new, long, thing, hit	**roll−on** [rol on]	**steely** ['sti:ll]	consonant /k,g,n,l,ng,h,y/ widened

monophtong /i,ii/ normal	allophones i:, I, j easy, pit, yes	**fitting**	monophtong /e,a/ normal	allophones {, e, Er hat, pet, stairs	**pet, stay**
monophtong /aa,o/ normal	allophones A:, A stars,cut	**stars**	monophtong /uh,@/ normal	allophones @ another	**one another**
monophtong /@@/ normal	allophones 3: bird	**bird**	monophtong /oo/ normal	allophones O, Q cause, pot	**cause**
monophtong /u,uu/ normal	allophones U, u: put, lose	**book**	diphthong divide into two monophtongs fea/b car/a fe/b ca/a ca/b ar/a	allophones @_U nose a_I rise a_U rouse e_I raise I_@ fear O_I noise U_@ cures	
silence /#/ normal	allophones X closed lips	**silence**			

animation was restricted to cartoon-like figures, which do not show the same complexity as real faces). Our selection came out to be a good compromise between the number of visemes needed in the animation and the realism that is obtained.

It is important to note that our speech model combines the visemes with additional co-articulation effects. Further increases in realism are also obtained by adapting the viseme deformations to the shape of the face. These aspects are described in the section, *Face Animation*.

Learning Viseme Expressions

The deformations that come with the different visemes had to be analyzed carefully. The point of departure in developing the animation system has, therefore, been to extract detailed, 3D deformations during speech for ten example faces. These faces differed in age, race, and gender. A first issue was the actual part of the face that had to be acquired. The results of Munhall and Vatikiotis-Bateson (Munhall et al., 1998) provide evidence that lip and jaw motions affect the entire facial structure below the eyes. Therefore, we extracted 3D data for a complete face, but with emphasis on the area between the eyes and the chin. The extraction of the 3D visemes follows a number of steps, which were repeated for the different example faces:

1. a 3D reconstruction is produced for all instances of all visemes
2. a generic head model is fitted to these 3D visemes
3. prototypes of the visemes are defined

These steps are now described in more detail.

Raw Viseme Extraction

The first step in learning realistic, 3D face deformations for the different visemes was to extract real deformations from talking faces. Before the data were extracted, it had to be decided what the test person would say during the acquisition. It was important that all relevant visemes would be observed at least once. The subjects were asked to read a short text that contained multiple instances of the visemes in Figure 2.

For the 3D shape extraction of the talking face, we have used a 3D acquisition system that uses structured light (Eyetronics, 1999). It projects a grid onto the face, and extracts the 3D shape and texture from a single image. By using a video camera, a quick succession of 3D snapshots can be gathered. We are especially interested in frames that represent the different visemes. These are the frames where the lips reach their extremal positions for that sound (Ezzat and Poggio (Ezzat et al., 2000) followed the same approach in 2D). The acquisition system yields the 3D coordinates of several thousand points for every frame. The output is a triangulated, textured surface. The problem is that the 3D points correspond to projected grid intersections, not corresponding, physical points of the face. Hence, the points for which 3D coordinates are given change from frame to frame. The next steps have to solve for the physical correspondences.

Fitting of the Generic Head Model

Our animation approach assumes a specific topology for the face mesh. This is a triangulated surface with 2'268 vertices for the skin, supplemented with separate meshes for the eyes, teeth, and tongue (another 8'848, mainly for the teeth). Figure 3 shows the generic head and its topology.

The first step in this fitting procedure deforms the generic head by a simple rotation, translation, and anisotropic scaling operation, to crudely align it with the neutral shape of the example face. This transformation minimizes the average distance between a number of special points on the example face and the model

Figure 3. The generic head model that is fitted to the scanned 3D data of the example face. Left: Shaded version; Right: Underlying mesh.

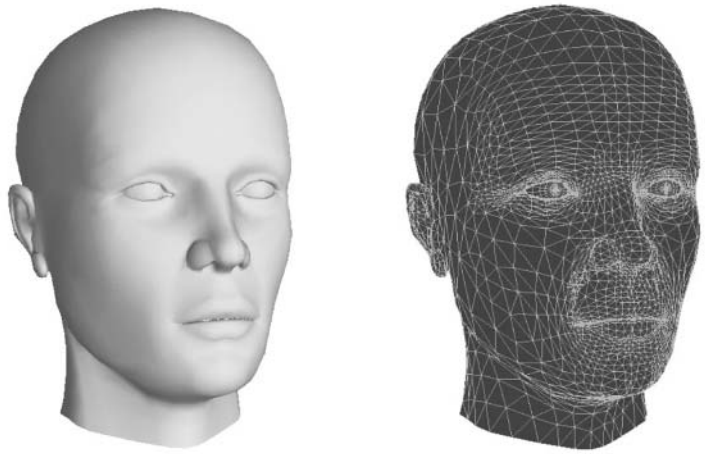
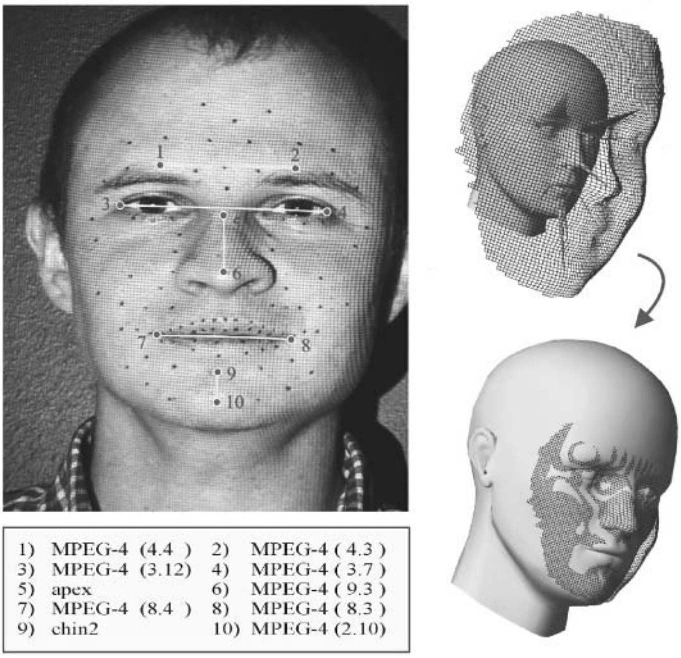


Figure 4. A first step in the deformation of the generic head to make it fit a captured 3D face is to globally align the two. This is done using 10 feature points indicated in the left part of the figure. The right part shows the effect: Patch and head model are brought into coarse correspondence.



1)	MPEG-4 (4.4)	2)	MPEG-4 (4.3)
3)	MPEG-4 (3.12)	4)	MPEG-4 (3.7)
5)	apex	6)	MPEG-4 (9.3)
7)	MPEG-4 (8.4)	8)	MPEG-4 (8.3)
9)	chin2	10)	MPEG-4 (2.10)

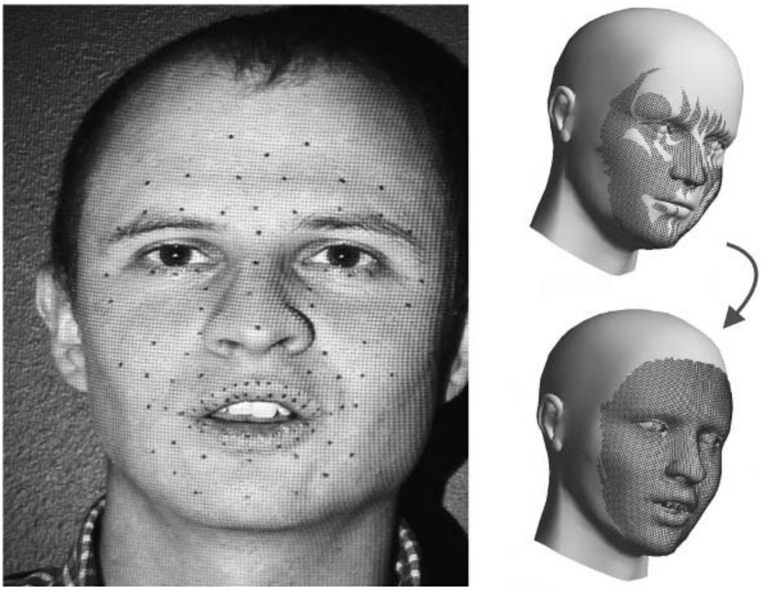
(these 10 points are indicated in Figure 4). These have been indicated manually on the example faces, but could be extracted automatically (Noh et al., 2001). After this initial transformation, the salient features may not be aligned well, yet. The eyes could, e.g., be at a different height from the nose tip.

In order to correct for such flaws, a piecewise constant vertical stretch is applied. The face is vertically divided into five intervals, ranging from top-of-head to eyebrows, from eyebrows to eye corners, from eye corners to nose tip, from nose tip to mouth corners, and from mouth corners to bottom of the chin. Each part of the transformed model is vertically scaled in order to bring the border points of these intervals into good correspondence with the example data, beginning from the top of the head. A final adaptation of the model consists of the separation of the upper and lower lip, in order to allow the mouth to open. The dividing line is defined by the midpoints of the upper and lower edges of the mouth outline.

This first step fixes the overall shape of the head and is carried out only once (for the neutral example face). The result of such process is shown in the right column of Figure 4: starting from the 3D patch for the neutral face and the generic model that are shown at the top, the alignment at the bottom is obtained. As can be seen, the generic model has not yet been adapted to the precise shape of the head at that point. The second step starts with the transformed model of the first step and performs a local morphing. This morphing maps the topology of the generic model head precisely onto the given shape. This process starts from the correspondences for a few salient points. This set includes the ten points of the previous step, but is also extended to 106 additional points, all indicated in black in Figure 5.

After the crude matching of the previous step, most of these points on the example face will already be close to the corresponding points on the deformed generic model. Typically, the initial frame of the video sequence corresponds to the neutral expression. This makes a manual drag and drop operation for the 116

Figure 5. To make the generic head model fit the captured face data precisely, a morphing step is applied using the 116 anchor points (black dots) and the corresponding Radial Basis Functions for guiding the remainder of the vertices. The right part of the figure shows a result.



points rather easy. At that point all 116 points are in good correspondence. Further snapshots of the example face are no longer handled manually. From the initial frame, the points are tracked automatically throughout the video. The tracker looks for point candidates in a neighborhood around their previous position. A dark blob is looked for and its midpoint is taken. As data are sampled at video rate, the motions between frames are small and this very simple tracking procedure only required manual help at a dozen or so frames for the set of example data. The main reason was two candidate points falling into the search region. Using this tracker, correspondences for all points and for all frames could be established with limited manual input.

In order to find the deformations for the visemes, the corresponding frames were selected from the video and their 3D reconstructions were made. The 3D positions of the 116 points served as anchor points, to map all vertices of the generic model to the data. The result is a model with the shape and expression of the example face and with 2'268 vertices at their correct positions. This mapping was achieved with the help of Radial Basis Functions.

Radial Basis Functions (RBFs) have become quite popular for face model fitting (Pighin et al., 1998; Noh et al., 2001). They offer an effective method to interpolate between a network of known correspondences. RBFs describe the influence that each of the 116 known (anchor) correspondences have on the nearby points in between in this interpolation process.

Consider the following equations,

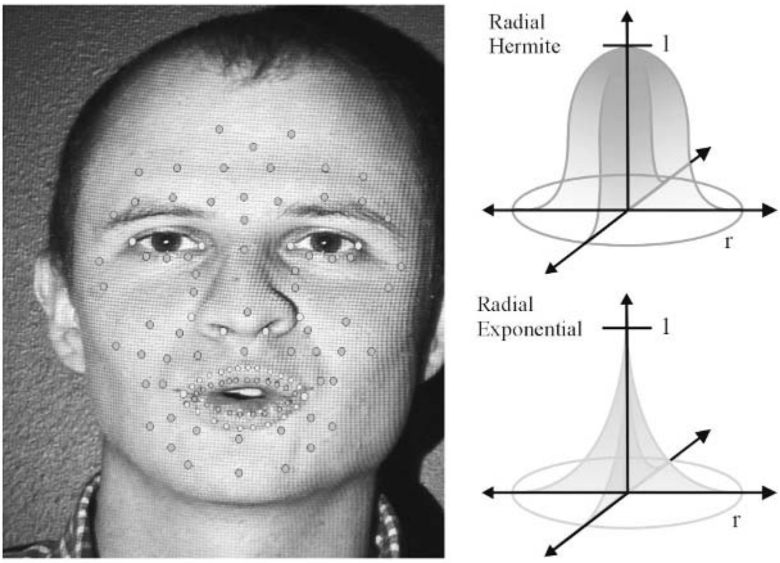
$$\mathbf{y}_{i_{new}} = \mathbf{y}_i + \sum_{j=1}^{n} \omega_j \mathbf{d}_j \tag{1}$$

which specify how the positions $\mathbf{y}_i$ of the intermediate points are changed into $\mathbf{y}_{i_{new}}$ under the influence of the n vertices $\mathbf{m}_j$ of the known network (the 116 vertices in our case). The shift is determined by the weights ω_j and the virtual displacements $\mathbf{d}_j$ that are attributed to the vertices of the known network of correspondences. More about these displacements is to follow. The weights depend on the distance of the intermediate point to the known vertices:

$$\omega_j = h(s_j / r) \qquad s_j = \|\mathbf{y}_i - \mathbf{m}_j\| \tag{2}$$

For $s_j \leq r$, where r is a cut-off value for the distance beyond which h is put to zero, and where in the interval $[0, r]$ the function $h(x)$ is of one of two types:

Figure 6. In the morphing step, two types of Radial Basis Functions are applied. (1) The hermite type is shown in the top-right part of the figure and is applied to all dark grey points on the face. (2) The exponential type is shown in the bottom-right part and is applied to the light grey points.



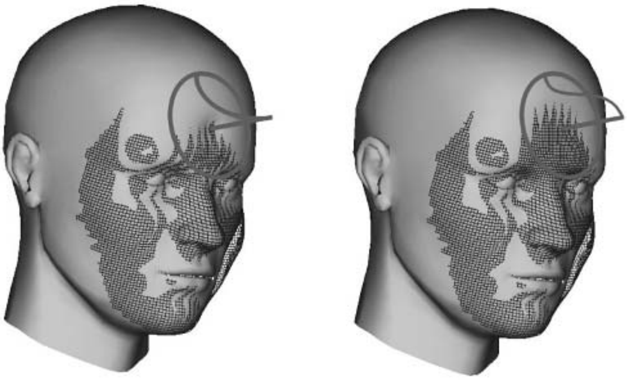
$$h_1 = 1 - x^{\log(b)/\log(0.5)} \qquad b \approx 5 \tag{3}$$

$$h_2 = 2x^3 - 3x^2 + 1 \tag{4}$$

Figure 6 shows these two functions.

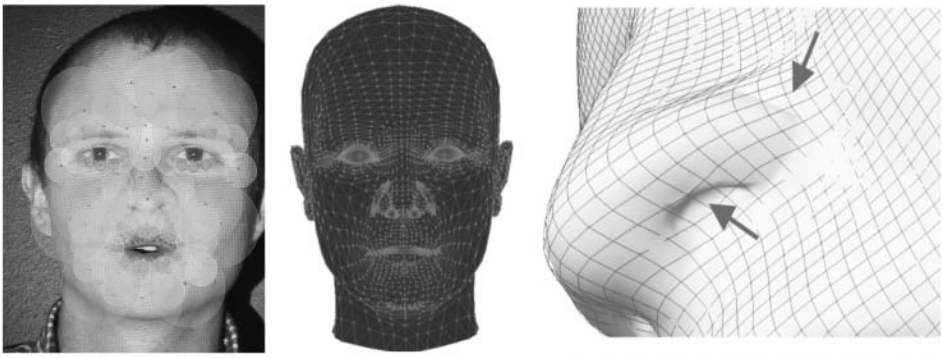
The first type is an exponential function yielding weights that decrease rapidly when moving away from the vertex, whereas the second type — a hermite basis function — shows more like a plateau in its neighborhood. The exponential type is used at vertices with high curvature, limiting the spatial extent of their influence, whereas the hermite type is used for vertices in a region of low surface curvature, where the influence of the vertex should reach out quite far. The vertices indicated in bright grey on the face are given exponential functions, the dark grey ones hermite functions. Figure 7 illustrates the result of changing a point on the forehead from exponential to hermite. The smaller influence region results in a dip.

Figure 7. The selection of RBF type is adapted to the local geometry. The figure shows the improvement that results from switching from exponential to hermite for the central point on the forehead.



Similarly, there are places where an exponential is much more effective than a hermite RBF. If the generic head, which is of a rather Caucasian type, has to be mapped onto the head of an Asian person, hermite functions will tend to copy the shape of the mesh around the eyes, whereas one wants local control in order to narrow the eyes and keep the corners sharp. The size of the region of influence is also determined by the scale *r*. Three such scales were used (for both RBF types). These scales and their spatial distribution over the face are shown in Figure 8(1). As can be seen, they vary with the scale of the local facial structures.

Figure 8. (1) The RBF sizes are also adapted to local geometry. There are three sizes, where the largest is applied to those parts that are the least curved. (2,3) For a small subset of points lying in a cavity the cylindrical mapping is not carried out, to preserve geometrical detail at places where captured data quality deteriorates.



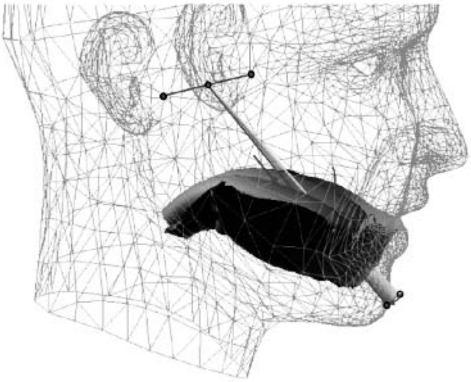
The virtual displacements $\mathbf{d}_j$ of equation (1) are determined by demanding that the 116 vertices undergo the (known) motions that map them from the generic model onto the 3D face patch. This condition leads to a system of equations for the *X*, *Y*, and *Z* components of the 116 vertex motions, which are combined into three column vectors $\mathbf{c}_{X,Y,Z}$ respectively:

$$\mathbf{Ad}_{X,Y,Z} = \mathbf{c}_{X,Y,Z} . \tag{5}$$

In these equations, the vectors $\mathbf{d}_{X,Y,Z}$ represent the column vectors containing all the *X*, *Y*, or *Z* components of the virtual displacement vectors $\mathbf{d}_j$. The influence matrix **A** contains the weights that the vertices of the known network apply to each other. After solving these systems for $\mathbf{c}_{X,Y,Z}$, the interpolation is ready to be applied. It is important to note that vertices on different sides of the dividing line of the mouth are decoupled in these calculations.

A third step in the processing projects the interpolated points onto the extracted 3D surface. This is achieved via a cylindrical mapping. This mapping is not carried out for a small subset of points which lie in a cavity, however. The reason is that the acquisition system does not always produce good data in these cavities. The position of these points should be determined fully by the deformed head model, and not subject to being degraded under the influence of the acquired data. They are shown on the right side of Figure 8. On Figure 8(3), this is illustrated for the nostril. The extracted 3D grid is too smooth there and does not follow the sharp dip that the nose takes. The generic model dominates the fitting procedure and caters for the desired, high curvatures, as can be seen.

Figure 9. The jaw and lower teeth rotate around the midpoint of the places where the jaw is attached to the skull, and translated (see text).



An extreme example where the model takes absolute preference is the mouth cavity. The interior of the mouth is part of the model, which, e.g., contains the skin connecting the teeth and the interior parts of the lips. Typically, scarcely any 3D data will be captured for this region, and those that are captured tend to be of low quality. The upper row of teeth is fixed rigidly to the model and has already received their position through the first step (the global transformation of the model, possibly with a further adjustment by the user). The lower teeth follow the jaw motion, which is determined as a rotation about the midpoint between the points where the jaw is attached to the skull and a translation. The motion itself is quantified by observing the motion of a point on the chin, standardized as MPEG-4 point 2.10. These points have also been defined on the generic model, as can be seen in Figure 9, and can be located automatically after the morph.

It has to be mentioned at this point that all the settings, like type and size of RBFs, as well as whether vertices have to be cylindrically mapped or not, are defined only once in the generic model as attributes of its vertices.

Viseme Prototype Extraction

The previous subsection described how a generic head model was deformed to fit 3D snapshots. Not all frames were reconstructed, but only those that represent the visemes (i.e., the most extreme mouth positions for the different cases of Figure 2). About 80 frames were selected from the sequence for each of the example faces. For the representation of the corresponding visemes, the 3D reconstructions, themselves, were not taken (the adapted generic heads), but the difference of these heads with respect to the neutral one for the same person. These deformation fields of all the different subjects still contain a lot of redundancy. This was investigated by applying a Principal Component Analysis. Over 98.5% of the variance in the deformation fields was found in the space spanned by the 16 most dominant components. We have used this statistical method not only to obtain a very compact description of the different shapes, but also to get rid of small acquisition inaccuracies. The different instances of the same viseme for the different subjects cluster in this space. The centroids of the clusters were taken as the prototype visemes used to animate these faces later on.

Face Animation

The section, *Learning Viseme Expressions,* describes an approach to extract a set of visemes from a face that could be observed in 3D, while talking. This process is quite time consuming and one would not want to repeat it for every single face that has to be animated. This section describes how novel faces can be animated, using visemes which could not be observed beforehand.

Such animation requires a number of steps:

1. *Personalizing the visemes*

   The shape or "physiognomy" of the novel face is taken into account by determining the face's relative position with respect to the neutral example faces in a Face Space. This information is used to generate a set of visemes specific for the novel face.

2. *Automatic, audio-based animation*

   From fluent speech a file is generated that contains visemes and their timing. This file is automatically transformed into an animation of the face by producing a sequence of viseme expressions combined with intermediate co-articulation effects.

3. *Possible modifications by the animator*

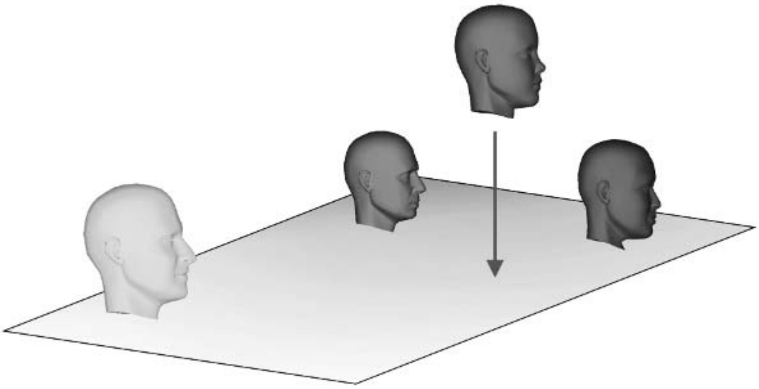
   As the animator should remain in control, tools are provided that allow the animator to modify the result as desired. A "Viseme Space" can be roamed using its independent components.

Personalizing the Visemes

A good animation requires visemes that are adapted to the shape or "physiognomy" of the face at hand. Hence, one cannot simply copy or "clone" the deformations that have been extracted from one of the example faces to a novel face. Although it is not precisely known at this point how the viseme deformations depend on the physiognomy, visual improvements were observed by adapting the visemes in a simple way described in this section.

Faces can be efficiently represented as points in a so-called "Face Space" (Blanz et al., 1999). These points actually represent their deviation from the average face. This can be done for the neutral faces from which the example visemes have been extracted using the procedure described in the section, *Learning Viseme Expressions*, as well as for a neutral, novel face. The example faces span a hyper-plane in Face Space. By orthogonally projecting the novel face onto this plane, a linear combination of the example faces is found that comes closest to the projected novel face. This procedure is illustrated in Figure 10. Suppose we put the Face Space coordinates of the face that corresponds to this projection into a single column vector $\tilde{\mathbf{F}}_{nov}$ and, similarly, the coordinates of

Figure 10. Orthogonal projection of a novel face onto the hyper-plane formed by the neutral example faces.



the example face *i* into the vector $\mathbf{F}_i$. If the coordinates of the projected, novel face $\tilde{\mathbf{F}}_{nov}$ are given by

$$\tilde{\mathbf{F}}_{nov} = \sum_{i=1}^{n} \omega_i \mathbf{F}_i \tag{6}$$

the same weights ω_i are applied to the visemes of the example faces, to yield a personalized set of visemes for the novel face. The effect is that a rounded face will get visemes that are closer to those of the more rounded example faces, for instance.

This step in the creation of personalized visemes is schematically represented in Figure 11.

Figure 11. A novel face can be approximated as a linear combination of example faces. The same combination of the example faces' visemes yields a first version of the novel face's viseme set.

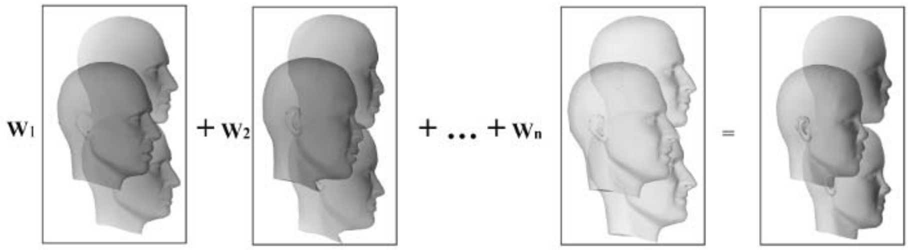
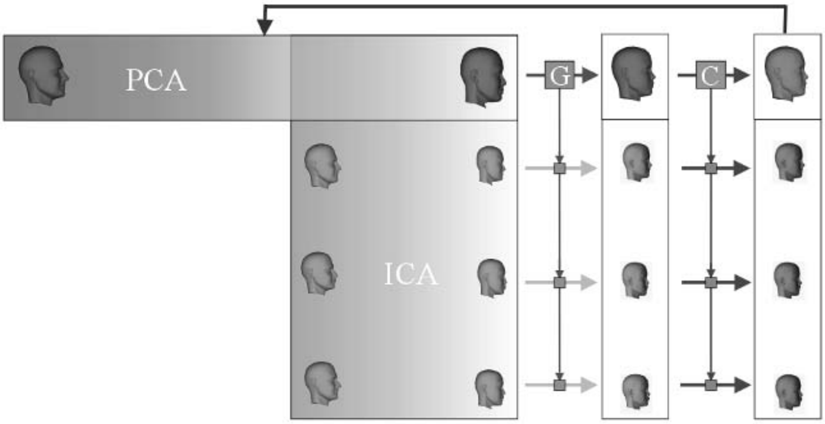


Figure 12. The personalization of visemes follows two steps symbolized by the two horizontal transitions: 1) The linear combination of the example visemes as described in the text, and 2) A residual adaptation, following the cloning technique described in Noh et al. (2001).



As the face $\tilde{\mathbf{F}}_{nov}$ is still a bit different from the original novel face $\mathbf{F}_{nov}$, expression cloning is applied as a last step to the visemes found from projection (Noh et al., 2001). We have found that the direct application of viseme cloning from an example face to other faces yields results that are less convincing. This is certainly the case for faces that differ substantially. According to the proposed strategy the complete set of examples is exploited, and cloning only has to deal with a small residue. The process of personalizing visemes is summarized in Figure 12.

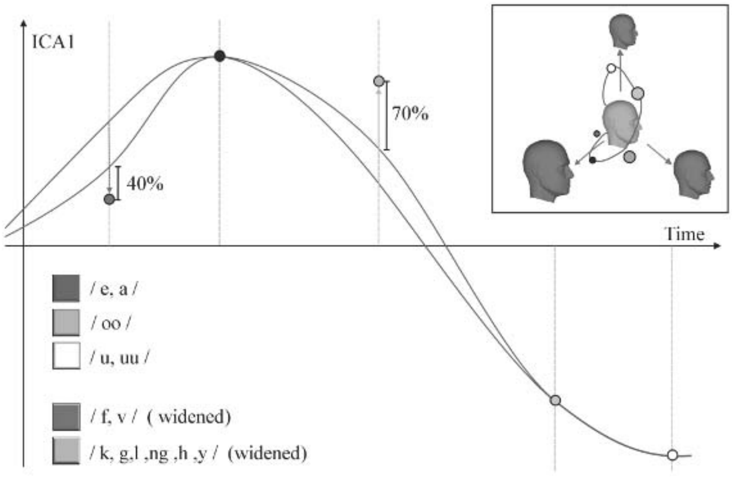
Automatic, Audio-Based Animation

Once the visemes for a face have been determined, animation can be achieved as a concatenation of visemes. The visemes, which have to be visited, the order in which this should happen, and the time intervals in between are generated automatically from an audio track containing speech. First a file is generated that contains the ordered list of allophones and their timing. "Allophones" correspond to a finer subdivision of phonemes. This transcription has not been our work and we have used an existing tool, described in Traber (1995). The allophones are then translated into visemes (the list of visemes is provided in Figure 2). The vocals and silence are directly mapped to the corresponding visemes. For the consonants, the context plays a stronger role. If they immediately follow a vocal among **/o/**, **/u/**, and **/@@/** (this is the vocal as in "bird"), then the allophone is mapped onto a rounded consonant. If the vocal is among **/i/**, **/a/**, and **/e/** then the allophone is mapped onto a widened consonant. When the consonant is not preceded immediately by a vocal, but the subsequent allophone is one, then a similar decision is made. If the consonant is flanked by two other consonants, the preceding vocal decides.

From these data — the ordered list of visemes and their timing — the system automatically generates an animation. The concatenation of the selected visemes can be achieved elegantly as a navigation through a "Viseme Space," similar to a Face Space. The Viseme Space is obtained by applying an Independent Component Analysis to all extracted, example visemes. It came out that the variation can be captured well with as few as 16 Independent Components. (This underlying dimensionality is determined as the PCA step that is part of our ICA implementation (Hyvärinen, 1997).) Every personalized viseme can be represented as one point in this 16D Viseme Space. Animation boils down to subsequently applying the deformations represented by the points along a trajectory that leads from viseme to viseme, and that is influenced by co-articulation effects. An important advantage of animating in Viseme Space is that all visited deformations remain realistic.

Performing animation as navigation through a Viseme Space of some sort is not new *per se*. Such approach was already demonstrated by Kalberer and Van Gool

Figure 13. Fitting splines in the "Viseme Space" yields good co-articulation effects, after attraction forces exerted by the individual nodes (visemes) were learned from ground-truth data.



(Kalberer et al., 2001; Kalberer et al., 2002a) and by Kshirsagar (2001), but for fewer points on the face. Moreover, their Viseme Spaces where based on PCA (Principal Component Analysis), not ICA. A justification for using ICA rather than PCA is to follow later.

Straightforward point-to-point navigation as a way of concatenating visemes would yield jerky motions. Moreover, when generating the temporal samples, these may not precisely coincide with the pace at which visemes change. Both problems are solved by fitting splines to the Viseme Space coordinates of the visemes. This yields smoother changes and allows us to interpolate in order to get the facial expressions needed at the fixed times of subsequent frames. We used NURBS curves of order three.

A word on the implementation of co-articulation effects is in order here. A distinction is made between vocals and labial consonants on the one hand, and the remainder of the visemes on the other. The former impose their deformations much more strictly onto the animation than the latter, which can be pronounced with a lot of visual variation. In terms of the spline fitting, this means that the animation trajectory will move precisely through the former visemes and will only be attracted towards the latter. Figure 13 illustrates this for one Viseme Space coordinate.

Initially a spline is fitted through the values of the corresponding component for the visemes of the former category. Then, its course is modified by bending it towards the coordinate values of the visemes in the latter category. This second category is subdivided into three subcategories: (1) somewhat labial consonants like those corresponding to the **/ch,jh,sh,zh/** viseme pull stronger than (2) the viseme **/f,v/** , which in turn pulls stronger than (3) the remaining visemes of the second category. In all three cases the same influence is given to the rounded and widened versions of these visemes. The distance between the current spline (determined by vocals and labial consonants) and its position if it had to go through these visemes is reduced to (1) 20%, (2) 40%, and (3) 70%, respectively. These are also shown in Figure 13. These percentages have been set by comparing animations against 3D ground-truth. If an example face is animated with the same audio track used for training, such comparison can be easily made and deviations could be minimized by optimizing these parameters. Only distances between lip positions were taken account of so far.

Modifications by the Animator

A tool that automatically generates a face animation which the animator then has to take or leave is a source of frustration, rather than a help. The computer cannot replace the creative component that the human expert brings to the animation process. The animation tool described in this paper only proposes the speech-based face animation as a point of departure. The animator can thereafter still change the different visemes and their influences, as well as the complete splines that define the trajectory in “Viseme Space.”

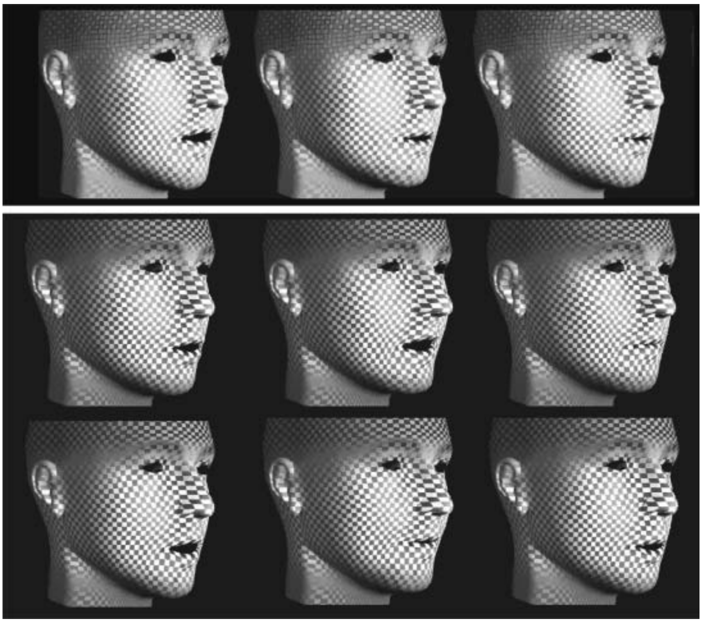
In terms of the space of possible deformations, PCA and ICA basically yield the same result. As already mentioned, PCA is part of the ICA algorithm, and determines the degrees of freedom to be kept. The importance of ICA mainly lies in the more intuitive, manual changes that the animator can make afterwards. A face contains many muscles, and several will be active together to produce the different visemes. In as far as their joint effect can be modeled as a linear combination of their individual effects, ICA is *the* way to decouple the net effect again (Kalberer et al., 2002b). Of course, this model is a bit naive but, nevertheless, one would hope that ICA is able to yield a reasonable decomposition of face deformations into components that themselves are more strongly correlated with the facial anatomy than the principal components. This hope has proved not to be in vain.

From a mathematical point of view, there also is a good indication that ICA may be more appropriate than PCA to deliver the basis of a Viseme Space. The distribution of the extracted visemes comes out to have a shape that is quite non-Gaussian, which can clearly be observed from χ^2 plots.

Independent Component Analysis tries to explain data as a linear combination of maximally independent basis signals, the “Independent Components.” Basically, these independent components are found as the linear combinations of principal components that have, in terms of information theory, minimal mutual information between each pair of input. This is mathematically related to finding combinations with distributions that are maximally non-Gaussian. As the central limit theorem makes clear, distributions of composed signals will tend to be more Gaussian than those of the underlying, original signals. For these reasons, ICA often is successful in retrieving a set of original signals that can only be observed together, e.g., to split mixed audio signals into their different components. These separate, original components typically correspond to the maximally non-Gaussian directions of the distribution that represents the joint probabilities of the observed signal values. If the original signals have Gaussian distributions, ICA will fail. The fact that the composed distributions in our case are already non-Gaussian is an indication the ICA can make sense.

The advantage that independent components have over principal components doesn’t lie in their respective numbers, as, in fact, these are the same. Indeed, the ICs are found in the reduced space spanned by the dominant PCs and this space’s dimension determines the number of ICs that ICA extracts (our implementation of ICA follows that propounded by Hyvärinen (1997)). As already mentioned, 16 components were used, which together cover 98.5% of

Figure 15. Independent components yield more intuitive face deformations than principal components in viseme space. Top: Principal components; Middle, Bottom: Independent components.



the variation. The advantage, rather, is the more intuitive deformations that correspond to the independent components, where each stays closer to a single, anatomical action of the face.

Finally, on a more informal score, we found that only about one or two PCs could be easily described, e.g., “opening the mouth.” In the case of ICs, six or so components could be described in simple terms. Figure 15 shows a comparison between principal and independent components. In both cases, there is a component that one could describe as opening the mouth. When it comes to a simple action, like rounding the mouth, there is a single IC that corresponds to this effect. But, in the case of PCs, this rounding is never found in isolation, but is combined with the opening of the mouth or other effects. Similar observations can be made for the other ICs and PCs.

One could argue that animation can proceed directly and uniquely as a combination of basic modes (e.g., independent components) and that going via visemes is an unnecessary detour. Discussions with animators made it clear, however, that they insist on having intuitive keyframes, like visemes and basic emotions, as the primary interface. Hence, we give animators control both at the level of complete visemes and single independent components. Having the system work on the basis of the same keyframes (i.e., in a Viseme Space) helps to make the interaction with the animator more intuitive, as the animator and the animation tool "speak" the same language.

Results

As a first example, we show some frames out of an animation sequence, where the person is animated using his own visemes. This person was one of our examples. Four frames are shown in Figure 16. Although it is difficult to demonstrate the realism of animation on paper, the different face expressions look natural, and so does the corresponding video sequence.

A second example shows the transition between two faces (see Figure 17). In this case, the visemes of the man are simply cloned to get those for the boy (Noh

Figure 16. One of the example faces uses its own visemes.

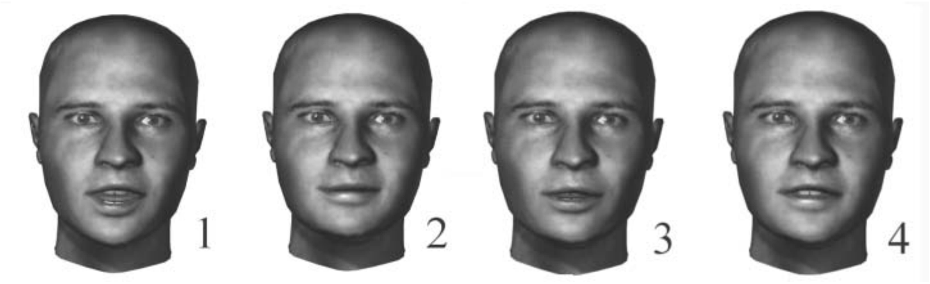


Figure 17. To see the effect of purely cloned visemes, a specific experiment was performed. The man's visemes are kept throughout the sequence and are cloned onto the mixed face.

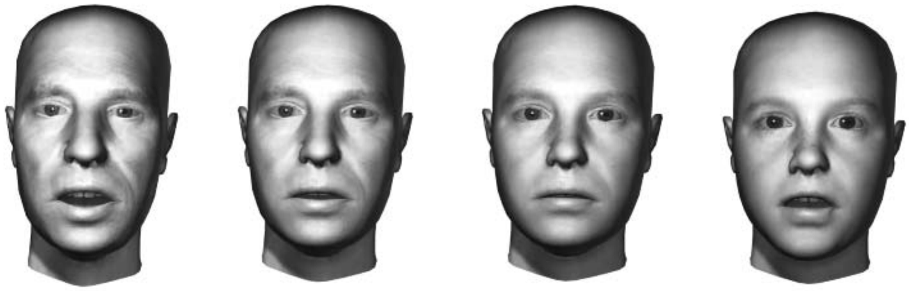
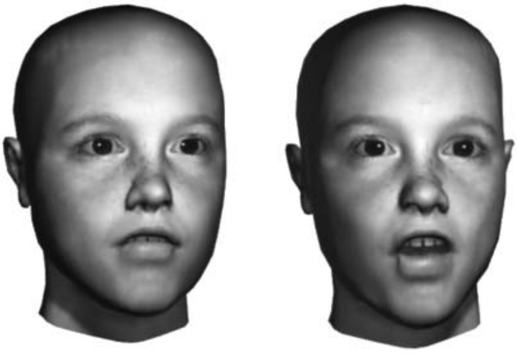


Figure 18. Two representative snapshots of purely cloned visemes exemplify that cloning (Noh et al., 2001) does not always result in convincing shapes.



et al., 2001). The animation shows a few flaws, which become stronger as the morph gets closer to the boy's face. Some of these flaws are highlighted in Figure 18 (but they are more outspoken if seen as a video). This example shows that cloning alone does not suffice to yield realistic animation of speech.

A third example shows the result of our full viseme personalization. The three faces on the left-hand side of Figure 19 are three of the 10 example faces used to create the hyper-plane. The face on the right-hand side has been animated by first projecting it onto the hyper-plane, weighing the visemes of the examples accordingly, and finally cloning these weighted visemes to the original face.

Figure 19. Combination of the same viseme (represented by the faces in the upper row) are combined and cloned onto a novel face according to its physiognomy (face in the center).

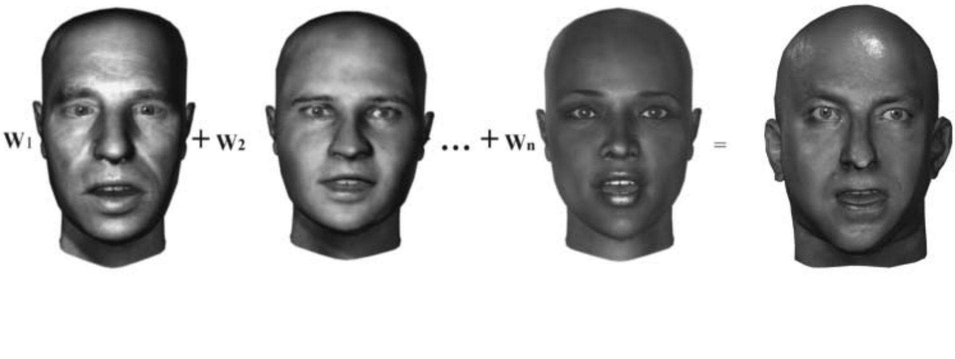
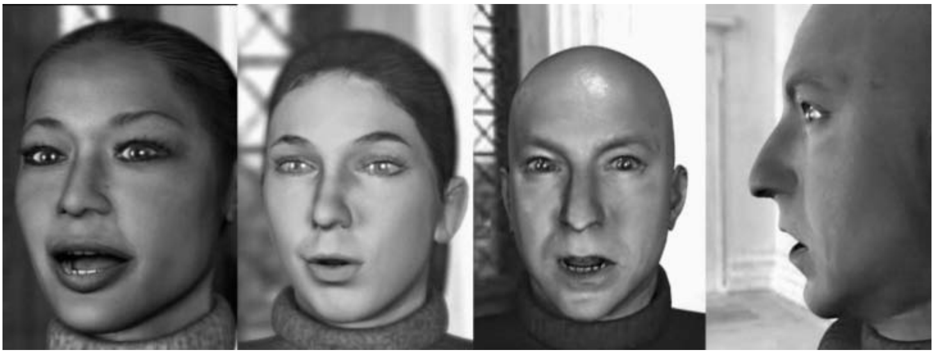


Figure 20. The final animation can be enhanced with expressions that are not related to speech, such as eye blinks and emotions.



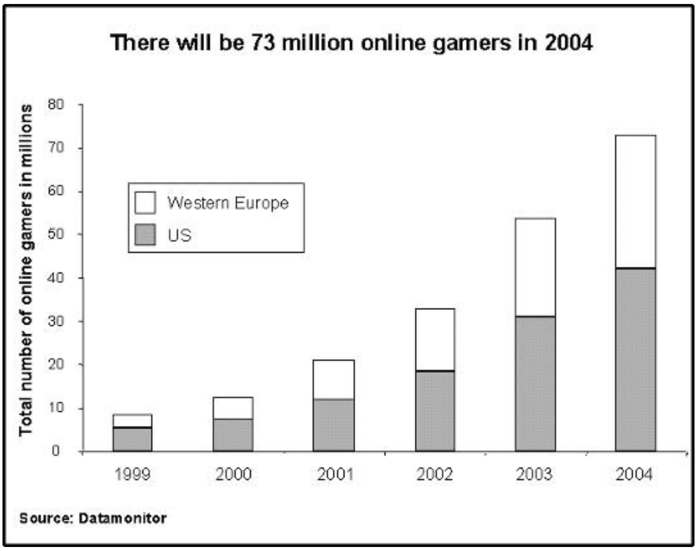
As a final note, it is interesting to mention that the system proposed here has been implemented as an Alias/Wavefront's Maya plug-in. Figure 1 gives a quick overview of the processing steps.

Furthermore, we have already experimented with the superposition of speech and emotions. Detailed displacements were measured for the six basic emotions. We found that linear addition of displacements due to visemes and emotions worked out well in these preliminary trials. An example is shown in Figure 20.

Trends

Technologically the trend in face animation is one towards a stronger 3D component in the modeling and animation pipeline. Entertainment certainly is one of the primary marketplaces for the work described in this chapter. The 3D industry has had a significant impact on this industry. With human characters as one of the central elements, 3D animation of characters and special effects have become an integral part of many blockbuster movie productions. The game industry has in the meantime eclipsed the movie industry in terms of gross revenues. Also, in this branch of industry, there is a trend towards more realistic human characters. The most notable trend in 3D digital media is the convergence of these playgrounds. Productions often target multiple markets simultaneously, with, e.g., movies coupled to games and Web sites, as well as an extensive line of gadgets.

Figure 21. The forecasts expect the European online game market to reach 43% by 2004 when there should be around 73 million online gamers, as shown on the chart.



The creation of 3D models is often financed through the acquisition of a copyright over the scanned material, as such copyright enables the holder to spin off such alternative applications. A segment expected to see a steep growth is online gaming. Online 3D gaming subscription revenue is expected to grow at an annual growth rate of 19.7% through 2007, as these sites offer unique experiences and even episodic updates to gamers (see Figure 21).

Conclusions

Realistic face animation is still a challenge. We have tried to attack this problem via the acquisition and analysis of 3D face shapes for a selection of visemes. Such data have been captured for a number of faces, which allows the system to at least apply a crude adaptation of the visemes to the physiognomy of a novel face for which no such data could be obtained. The animation is organized as a navigation through “Viseme Space,” where the influence of different visemes on the space trajectory varies. Given the necessary input in the form of an ordered sequence of visemes and their timing, a face animation can be created fully automatically. Such animation will in practice rather be a point of departure for an animator, who still has to be able to modify the results. This remains fully possible within the proposed framework.

The proposed method yields realistic results, with more detail in the underlying, 3D models than usual. The RBF morphing, described in the section, Learning Viseme Expressions, has been implemented with special care to make this possible. The way of combining the visemes to a novel face seems to be both novel and effective. By giving the animator control over independent, rather than the more usual, principal components this space can be navigated in a more intuitive way.

Although we believe that our results can already be of help to an animator, several improvements can be imagined. The following issues are a few examples. Currently, a fixed texture map is used for all the visemes, but the 3D acquisition method allows us to extract a separate texture map for every viseme. This will help to create the impression of wrinkles on rounded lips, for instance.

Another issue is the selection of the visemes. For the moment, only a rounded and widened version of the consonants has been included. In reality, an **/m/** in **ama** lies between that in **omo** and **imi**. There is a kind of gradual change from **umu**, over **omo**, **ama**, and **eme**, up to **imi**. Accordingly, more versions of the visemes can be considered. Another possible extension is the inclusion of tongue position in the viseme classification. Some of the consonant classes have to be subdivided in that case. A distinction has to be made between, e.g., /**l**/ and /**n**/ on the one hand, and /**g**/ and /**k**/ on the other.

It is also possible to take more example images, until they span Face Space very well. In that case, the final viseme cloning step in our viseme personalization can probably be left out.

Last, but not least, speech-oriented animation needs to be combined with other forms of facial deformations. Emotions are probably the most important example. It will be interesting to see what else is needed to combine, e.g., visemes and emotions and keep the overall impression natural. All these expressions call on the same facial muscles. It remains to be seen whether linear superpositions of the different deformations really suffice, as it was the case in our preliminary experiments.

Acknowledgments

This research has been supported by the ETH Research Council (Visemes Project), the KULeuven Research Council (GOA VHS+ Project), and the EC

IST project MESH (www.meshproject.com), with the assistance of our partners University Freiburg, DURAN, EPFL, EYETRONICS, and University of Geneva.

References

Beier, T. & Neely, S. (1992). Feature-based image metamorphosis. *SIGGRAPH*, ACM Press.

Blanz, V. & Vetter, T. (1999). A morphable model for the synthesis of 3d faces. *SIGGRAPH*, ACM Press.

Brand, M. (1999). Voice puppetry. *Animation SIGGRAPH*, ACM Press.

Bregler, C. & Omohundro, S. (1995). Nonlinear image interpolation using manifold learning. *NIP*S, volume 7.

Bregler, C., Covell, M. & Slaney, M. (1997). Video rewrite: driving visual speech with audio. *SIGGRAPH*, ACM Press.

Chen, D., State, A. & Banks, D. (1995). Interactive shape metamorphosis. In *Symposium on Interactive 3D Graphics*, *SIGGRAPH*.

Cosatto, E. (2000). *Sample-based talking-head synthesis*. In Ph.D. Thesis, Signal Processing Lab, Swiss Federal Institute of Techology, Lausanne, Switzerland.

Cosatto, E. & Graf, H. P. (2000). Photo-realistic talking-heads from image samples. In *IEEE Trans. on Multimedia*, 2, 152–163.

Eben, E. (1997). Personal communication. Pixar Animation Studios.

Eyetronics. (1999). Retrieved from the WWW: http://www.eyetronics.com.

Ezzat, T. & Poggio, T. (2000). Visual speech synthesis by morphing visemes. In Kluwer Academic Publishers, *International Journal of Computer Vision*, 38, 45–57.

Guenter, B., Grimm, C., Wood, D., Malvar, H. & Pighin, F. (1998). Making faces. *SIGGRAPH*, ACM Press.

Hyvärinen, A. (1997). *Independent component analysis by minimizing of mutual information*. In Technical Report A46, Helsinki University of Technology, 1997.

Kähler, K., Haber, J., Yamauchi, H. & Seidel, H. P. Head shop: Generating animated head models with anatomical structure. *SIGGRAPH Symposium on Computer Animation*, 55–63, ACM Press.

Kalberer, G. A. & Van Gool, L. (2001). Realistic face animation for speech. *Videometrics*, *Electronic Imaging*, *IS&T/SPIE*, 4309, 16–25.

Kalberer, G. A. & Van Gool, L. (2002a). Animation based on observed 3d speech dynamics. *International Journal of Visualization and Computer Animation*, 13, 97–106.

Kalberer, G. A., Mueller, P. & Van Gool, L. (2002b). Biological Motion of Speech, *Biologically Motivated Computer Vision BMCV*, 199 – 206.

Kshirsagar, S., Molet, T. & Magnenat-Thalmann, N. (2001). Principal components of expressive speech animation. *Computer Graphics International 2001*, 38–44.

Lin, I., Yeh, J. & Ouhyoung, M. (2001). Realistic 3d facial animation parameters from mirror-reflected multi-view video. *Computer Animation* 20012–11.

Massaro, D. W. (1998). *Book: Perceiving Talking Faces*. MIT Press.

Montgomery, A. & Jackson, P. (1983). Physical characteristics of the lips underlying vowel lipreading performance. In *Journal Acoust. Society America*, 73, 2134–2144.

Munhall, K. G. & Vatikiotis-Bateson, E. (1998). The moving face during speech communication. In Hearing by Eye, 2, 123–139.

Noh, J. & Neumann, U. (2001). *Expression cloning*. ACM Press, SIGGRAPH.

Owens, O. & Blazek, B. (1985). Visemes observed by hearing-impaired and normal-hearing adult viewers. In *Journal of Speech and Hearing Research*, 28, 381–393.

Parke, F. I. (1972). Computer generated animation of faces. In *ACM National Conferenc*e, 451–457.

Pelachaud, C., Badler, N. & Steedman, M. (1996). Generating facial expressions for speech. *Cognitive Science*, 20(1), 1-46.

Pighin, F., Hecker, J., Lischinski, D., Szeliski, R. & Salesin, D. H. (1998). *Synthesizing realistic facial expressions from photographs*. ACM Press, SIGGRAPH.

Reveret, L., Bailly, G. & Badin, P. (2000). Mother, a new generation of talking heads providing a flexible articulatory control for videorealistic speech animation. *ICSL2000*.

Scott, K. C. et al. (1994). Synthesis of speaker facial movement to match selected speech sequences. In *Proceedings of the Fifth Australian Conference on Speech Science and Technology*, 2, 620–625.

Tao, H. & Huang, T. S. (1999). Explanation-based facial motion tracking using a piecewise beziér volume deformation model. *IEEE Conf. CVPR1999*.

Traber, C. (1995). *SVOX: The Implementation of a Text-to-Speech System*. Ph.D. Thesis. Computer Engineering and Networks Laboratory, ETH; No. 11064.

Turk, M. A. & Pentland, A. P. (1991). Face recognition using eigenfaces. *IEEE Conf. CVPR1991*, 586-591.

Waters, K. and Frisbie, J. (1995). A coordinated muscle model for speech animation. *Graphics Interface*, 163-170.

Chapter IX

Automatic 3D Face Model Adaptation with Two Complexity Modes for Visual Communication*

Markus Kampmann
Ericsson Eurolab Deutschland GmbH, Germany

Liang Zhang
Communications Research Centre, Canada

Abstract

This chapter introduces a complete framework for automatic adaptation of a 3D face model to a human face for visual communication applications like video conferencing or video telephony. First, facial features in a facial image are estimated. Then, the 3D face model is adapted using the estimated facial features. This framework is scalable with respect to complexity. Two complexity modes, a low complexity and a high complexity mode, are introduced. For the low complexity mode, only eye and mouth features are estimated and the low complexity face model Candide *is adapted. For the* *high complexity mode, a more detailed face model is adapted, using eye and mouth features, eyebrow and nose features, and chin and cheek contours. Experimental results with natural videophone sequences show that with this framework automatic 3D face model adaptation with high accuracy is possible.*

Introduction

In the last few years, virtual humans and especially animated virtual faces (also called talking heads) have achieved more and more attention and are used in various applications. In modern computer games, virtual humans act as football players or Kung Fu fighters. In movies, highly realistic animated virtual humans are replacing real actors (e.g., in the science fiction movie “Final Fantasy”). On the Internet, animated virtual faces are acting as news announcers or sales agents. In visual communication applications, like video telephony or video conferencing, the real faces of the participants are represented by virtual face clones of themselves. If we take a closer look at the technology behind these animated faces, the underlying shape of a virtual face is often built from a 3D wireframe consisting of vertices and triangles. This wireframe is textured using textures from a real person’s facial image. Synthetic facial expressions are generated by animating the 3D wireframe. Usually, the face is animated by movement of the wireframe’s vertices. In order to produce natural looking facial movements, an underlying animation structure (providing rules for animation) is needed, simulating the behavior of a real human face.

The creation of such an animated face requires generating a well-shaped and textured 3D wire-frame of a human face, as well as providing rules for animation of this specific 3D wireframe. There are different ways to create an animated face. One possibility is that an animated face is created manually by an experienced 3D modeler or animator. However, an automatic approach is less time consuming and is required for some applications. Dependent on the specific application and its requirements, different ways for the automatic creation of an animated face exist.

For 3D modeling of the shape of the head or face, i.e., for generation of the 3D wire-frame, techniques that are common for the 3D modeling of objects in general could be used. With a 3D scanner, a laser beam is sent out and reflected by the object’s surface. Range data from the object can be obtained and used for 3D modeling. Other approaches use range data from multi-view images (Niem, 1994) obtained by multiple cameras for 3D modeling. All these techniques allow a very accurate 3D modeling of an object, i.e., a human head or face. However, the generated 3D model could not be immediately animated, since the underlying animation structure is missing.

An alternative approach is the use of a generic 3D face model with a built-in animation structure. *Action Units* from the *Facial Action Coding System* (Ekman & Friesen, 1977), *MPEG-4 facial animation parameters (FAP)* (Sarris, Grammalidis & Strintzis, 2002) or *muscle contraction parameter*s (Fischl, Miller & Robinson, 1993) from a model of facial muscles can be used as an animation structure for facial expression. A limited number of characteristic feature points on a generic face model are defined, e.g., the tip of the chin or the left corner of the mouth. At the first step of 3D modeling using a generic 3D face model, those defined feature points are detected in facial images. Then, the characteristic feature points of the generic 3D face model are adapted using the detected feature points. This process is also called *face model adaptation*. According to available input resources, 3D face model adaptation approaches can be categorized as follows: *(a) range image*: An approach using range image to adapt a generic face model with a physics-based muscular model for animation in 3D is proposed in Lee, Terzopoulos & Waters (1995). From the generic 3D face model, a planar generic mesh is created using a cylindrical projection. Based on the range image, the planar generic face mesh adaptation is iteratively performed to locate feature points in the range image by feature-based matching techniques; *(b) stereoscopic images/videos*: An approach to using stereoscopic images/videos for face model adaptation is proposed in Fua, Plaenkers & Thalman (1999). Information about the surface of the human face is recovered by using stereo matching to compute a disparity map and then by turning each valid disparity value into a 3D point. Finally, the generic face model is deformed so that it conforms to the cloud of those 3D points based on least-squares adjustment; *(c) orthogonal facial images*: Orthogonal facial images are used to adapt a generic face model in Lee & Magnenat-Thalmann (2000) and Sarris, Grammalidis & Strintzis (2001). They all require two or three cameras which must be carefully set up so that their directions are orthogonal; *(d) monocular images/videos*: For face model adaptation using monocular images/videos, facial features in the facial images are determined and the face model is adapted (Kampmann, 2002). Since no depth information is available from monocular images/videos, depth information for feature points is provided only in advance by a face model and is adapted in relation to the determined 2D feature points.

In the following, we concentrate on animated faces for applications in the field of visual communication where only monocular images are available. For visual communication applications, like video conferencing or video telephony, a virtual face clone represents the human participant in the video conference or in the videophone call. Movements and facial expressions of the human participants have to be extracted and transmitted. At the receiver side, the virtual face model is animated using the extracted information about motion and facial expressions.

Since such information can be coded with a limited amount of bits, a video conference or a videophone system with very low bit rates is possible (Musmann, 1995). To implement such a coding system, a generic 3D face model has to be adapted to the particular face of the participant involved in the monocular video telephony or video conferencing call. This adaptation must be carried out at the beginning of the video sequence. Instead of achieving the highest 3D modeling accuracy, it is the quality of the animated facial expressions in 2D images that is more important for visual communication at the receiver side.

Face model adaptation for visual communication differs from other applications in that it has to be done without human interaction and without *a priori* information about the participant's face and its facial features. It is unrealistic to assume that a participant always has a particular facial expression, such as a neutral expression with a closed mouth or a particular pose position, in a 3D world. An algorithm for 3D face model adaptation should not only adapt the face model to the shape of the real person's face. An adaptation to the initial facial expression at the beginning of the video sequence is also necessary.

Furthermore, an algorithm for 3D face model adaptation should be scalable, since a number of different devices will likely be used for visual communication in the future. On the one hand, there will be small mobile devices like a mobile phone with limited computational power for image analysis and animation. The display size is restricted, which results in less need for high quality animation. On the other hand, there will be devices without these limitations like stationary PCs. In the case of a device with limitations regarding power and display, the face model adaptation algorithm would need to switch to a mode with reduced computational complexity and less modeling accuracy. For more powerful devices, the algorithm should switch to a mode with higher computational complexity and greater modeling accuracy.

Some algorithms in the literature deal with automatic face model adaptation in visual communication. In Kampmann & Ostermann (1997), a face model is adapted only by means of eye and mouth center points. In addition, nose position, and eye and mouth corner points are also used in Essa & Pentland (1997). A 3D generic face model onto which a facial texture has previously been mapped by hand is adapted to a person's face in the scene by a steepest-gradient search method (Strub & Robinson, 1995). No rotation of the face model is allowed. In Kuo, Huang & Lin (2002), a method is proposed using anthropometric information to adapt the 3D facial model. In Reinders et al. (1995), special facial features like a closed mouth are at first estimated and the face model is then adapted to these estimated facial features. Rotation of the face model is restricted. Furthermore, no initial values for the facial animation parameters like Action Units (Reinders et al., 1995) or muscle contraction parameters (Essa & Pentland, 1997) have been determined by the adaptation algorithms. An approach for automatically adapting a generic face model to individual faces without these kinds of limitations has not been developed, yet.

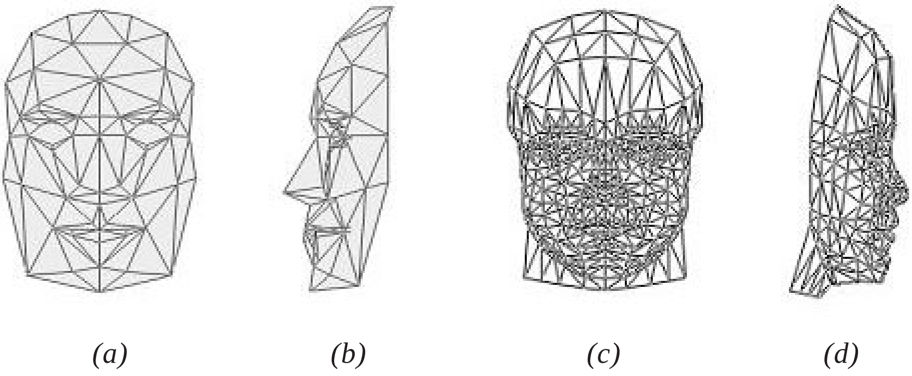
In this chapter, a complete framework for 3D face model adaptation based on monocular facial images without human interaction is addressed. Limitations like a closed mouth or neutral facial expressions do not exist for the proposed framework. In this framework, a two-step approach for face model adaptation is introduced. In the first step, facial features are estimated from the first frames of the video sequence. In the second step, the 3D face model is adapted using these estimated facial features. Furthermore, face model adaptation is done with two complexity modes. For the low complexity mode, the face model *Candide* (Rydfalk, 1987), with a small number of triangles is used, and only eye and mouth features are estimated, since these features are most important for visual impression. For facial animation in the low complexity mode, Action Units are used. For the high complexity mode, an advanced face model with a higher number of triangles is used, and other additional facial features like chin and cheek contours, eyebrow and nose features are further estimated. In the high complexity mode, a muscle-based model is imposed for facial animation.

This chapter is organized as follows. The next section presents the two face models of different complexities and their animation parameters. The section following describes algorithms for facial feature estimation. Special emphasis is given to the estimation of eye and mouth features. The fourth section presents the algorithms for 3D face model adaptation using the facial features estimated in the third section. Experimental results are presented in the final section.

3D Face Models

For visual communication like video telephony or video conferencing, a real human face can be represented by a generic 3D face model that must be adapted to the face of the individual. The shape of this 3D face model is described by a 3D wireframe. Additional scaling and facial animation parameters are aligned with the face model. Scaling parameters describe the adaptation of the face model towards the real shape of the human face, e.g., the size of the face, the width of the eyes or the thickness of the lips. Once determined, they remain fixed for the whole video telephony or video conferencing session. Facial animation parameters describe the facial expressions of the face model, e.g., local movements of the eyes or mouth. These parameters are temporally changed with the variations of the real face’s expressions. In this framework, face model adaptation is carried out in two complexity modes, with a low complexity face model and a high complex face model. These two face models are described in detail below.

Figure 1. 3D face models: (a)(b) Low complex 3D face mode Candide *(79 vertices and 132 triangles): (a) Front view, (b) Side view; (c)(d) High complex 3D face model (423 vertices and 816 triangles): (c) Front view, (d) Side view.*

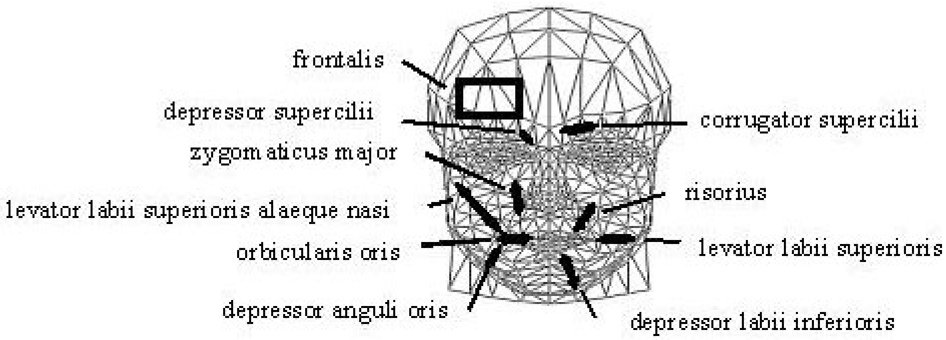


The face model *Candide* (Rydfalk, 1987) used for the low complexity mode is shown in Figures 1a and 1b. This face model consists of only 79 vertices and 132 triangles. For adaptation of *Candide* to the shape of the real face, scaling parameters are introduced. With these parameters, the global size of the face, the size of the eyes and the lip thickness could be changed.

As facial animation parameters for the face model *Candide*, six Action Units from the *Facial Action Coding System* (Ekman & Friesen, 1977) are utilized. Each Action Unit (AU) describes the local movement of a set of vertices. Two Action Units (AU_{41}, AU_7) are defined for the movements of the eyelids and the remaining four Action Units are defined for the movements of the mouth corners (AU_8, AU_{12}) and the lips (AU_{10}, AU_{16}).

For the high complexity mode, a derivative from the face model presented in Fischl, Miller & Robinson (1993) is used (ref. Figures 1c and 1d). This generic face model consists of 423 vertices and 816 triangles. Compared with the low complexity face model *Candide*, more scaling parameters are introduced. Here, the global size of the face, the size of eyes, nose and eyebrows, the lip thickness, as well as the shape of chin and cheek contours could be scaled. As facial animation parameters, facial muscle parameters as described in Fischl, Miller and Robinson (1993) are utilized. These muscle parameters describe the amount of contraction of the facial muscles within the human face. Ten different facial muscles are considered (ref. Figure 2). Since they occur on the left and on the right side of the face, respectively, 20 facial muscle parameters are used. In

Figure 2. High complexity face model: Position of facial muscles.



addition to Fischl, Miller and Robinson (1993), additional facial animation parameters are introduced. They describe the rotation of the jaw and the movements of eyelids and irises.

Facial Feature Estimation

The estimation of facial features in the 2D facial images is the first part of the face model adaptation algorithm. For the low complexity mode, eye and mouth features are estimated (described in the next subsection). For the high complexity mode, chin and cheek contours, eyebrow and nose features are additionally estimated (described in the following subsection).

Eye and Mouth Features

The eye and mouth features are estimated by means of 2D eye and mouth models. In the following, subscripts *r*, *l*, *u*, *o* stand for *right*, *left*, *upper* and *lower*, respectively.

2D eye model

The eye in a facial image is represented by a 2D eye model, shown in Figure 3, and consists of a pair of parabolic curves W_1, W_2 and a circle W_3 (Yuille, Hallinan & Cohen, 1992; Zhang, 1998). $\boldsymbol{h}$ is the pupil point and r the radius of the iris. r

Figure 3. 2D model of the eye.

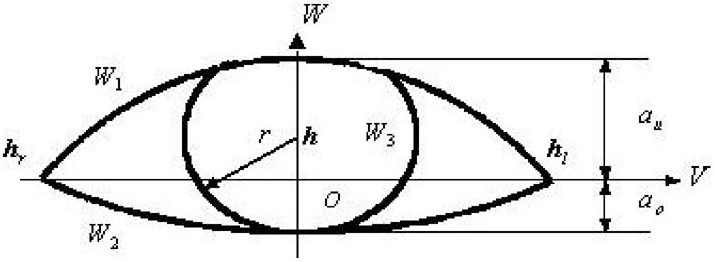
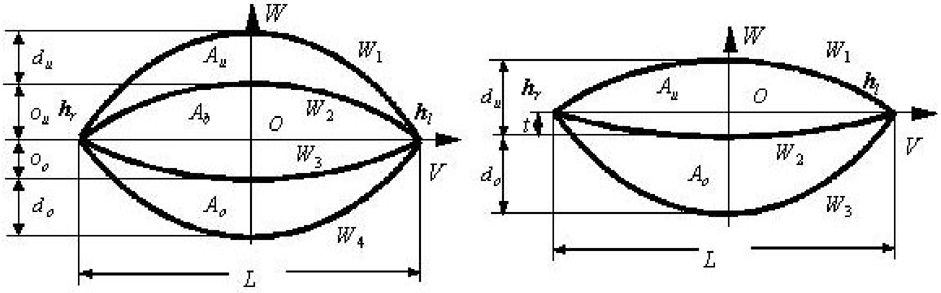


Figure 4. 2D models of the mouth.



(a) mouth-open model *(b) mouth-closed model*

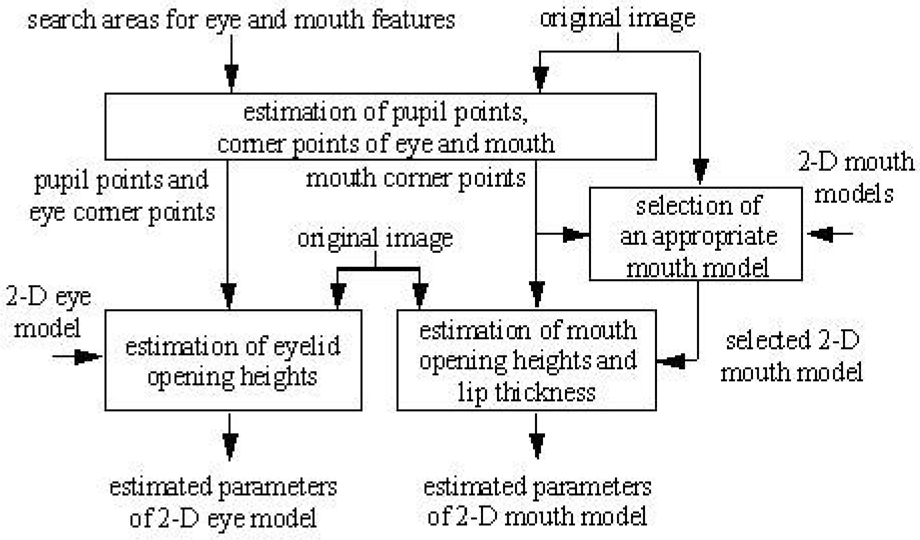
is set to a fixed fraction of the eye width that is the distance between two eye corner positions $\boldsymbol{h}_l$ and $\boldsymbol{h}_r$. The parameters of the parabolic curves (a_u, a_o) represent the opening heights of the eyelids. It is assumed that both eyes have the same opening heights. In order to represent eye features with a 2D eye model, eight parameters have to be estimated, namely: (i) four eye corner points, (ii) two pupil points, and (iii) two opening heights of the eyelids.

2D mouth models

The mouth is represented by a *mouth-open* model or a *mouth-closed* model. The *mouth-open* model (ref. Figure 4a) consists of four parabolic curves, W_i, and the *mouth-closed* model (ref. Figure 4b) of three parabolic curves, W_i (Zhang, 1998). The parameters (o_u, o_o) describe the opening heights of the lips and the parameters (d_u, d_o) stand for the thickness of the lips. The mouth width L is calculated as the distance between the left mouth corner point $\boldsymbol{h}_l$ and the right mouth corner point $\boldsymbol{h}_r$. To represent the *mouth-open* features, six parameters are needed: (i) two mouth corner points, (ii) two thickness of the lips, and (iii) two opening heights of the lips. For the representation of the *mouth-closed* features, five parameters are needed: (i) two mouth corner points, (ii) two thickness of the lips, and (iii) one parameter t which refers to the height between the level of the corners of the mouth and the contact point of the two lips.

Based on the representation of eye and mouth features using the 2D parametric models described above, the parameters of the models are separately estimated one after another (ref. Figure 5). The search areas for eye and mouth features are first determined using the algorithm proposed in Kampmann & Ostermann (1997). Within these search areas, the pupil points and the corner points of the eyes and the mouth are estimated with template matching techniques (Zhang, 1997). After that, these estimated points are utilized to fix the search areas for the lip thickness and the opening heights of the lips and the eyelids. Since two mouth models are exploited for representing the 2D mouth features, an appropriate mouth model has to be automatically selected first. The lip thickness and the opening heights of the lips will then be estimated by minimization of cost functions using an appropriate 2D mouth model. The eyelid opening heights are also estimated using the 2D eye model analogous to the estimation of the lip

Figure 5. Flowchart of eye and mouth feature estimation.


search areas for eye and mouth features
original image
estimation of pupil points, corner points of eye and mouth
pupil points and eye corner points
mouth corner points
2-D mouth models
selection of an appropriate mouth model
original image
2-D eye model
estimation of eyelid opening heights
estimation of mouth opening heights and lip thickness
selected 2-D mouth model
estimated parameters of 2-D eye model
estimated parameters of 2-D mouth model

thickness and opening heights (Kampmann & Zhang, 1998). In the following, only selection of an appropriate mouth model and estimation of mouth opening heights and lip thickness are addressed in detail.

Automatic Selection of 2D Mouth Models

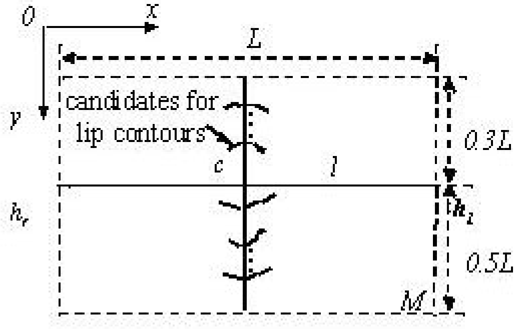
To estimate the 2D mouth features, an appropriate 2D mouth model has to be automatically selected. To do this, it must be known whether the mouth is open or not. Compared with a closed mouth, which consists of three lip contours, an open mouth has an additional fourth lip contour.

The mouth area *M* is determined by means of the mouth width *L*, the distance between the mouth corner positions (ref. Figure 6). Here, the area of the lower lip with the size of $L \times 0.5L$ is supposed to be larger than the area of the upper lip with the size of $L \times 0.3L$, because the lower lip has larger movement than the upper lip. Since a lip contour is labeled with high luminance variation, edge strength (image gradient) $g_y(x,y)$ in the mouth area is computed using a morphologic edge operator. It is further binarized with a threshold and thinned. The lines produced using these methods are the candidates for the possible lip contours. Finally, an appropriate 2D mouth model is automatically selected by comparing a number of possible positions for the lip contours above and below the line *l*, connecting both mouth corner positions.

Estimation of the Thickness and the Opening Heights of Lips

After the appropriate 2D mouth model has been selected, the parameters of the mouth model are estimated by minimization of cost functions. Since the mouth

Figure 6. Position detection of candidates for lip contours for 2D mouth model.



corner positions have already been estimated, only the thickness (d_u, d_o) and the opening heights (o_u, o_o) of the lips in the *mouth-open* case (ref. Figure 4), as well as the lip thickness (d_u, d_o) and the value of the parameter t in the *mouth-closed* case (ref. Figure 4) need to be estimated. A perpendicular bisector of the line l connecting both mouth corner positions in the mouth area M is defined as the search area for these parameters (ref. Figure 6). Different values of these parameters create different forms of the 2D mouth models. In order to determine these parameters, the similarity between the selected 2D mouth model with the geometrical form of the real mouth in the image is measured using a cost function. This cost function utilizes texture characteristics of the real mouth by means of the selected 2D mouth model.

It is observed that there are texture characteristics existing in the mouth area:

1. The areas of the lips have almost the same chrominance value. In the *mouth-open* case, an additional area, mouth-inside, exists and is strongly distinguished from the lip areas. Mouth-insides normally have different luminance values due to, e.g., teeth (white) and shadow areas (black). But, it has an almost uniform chrominance value.
2. On the lip, the luminance values of the contours strongly vary.

In the following, only the *mouth-open* case is discussed as an example. The solution to the *mouth-closed* case can be derived from the *mouth-open* case. Let $f_{open}(d_u,d_o,o_u,o_o)$ stand for the cost function used in the *mouth-open* case. Based on the texture characteristics mentioned above, the cost function $f_{open}(d_u,d_o,o_u,o_o)$ for the *mouth-open* case is defined as follows:

$$f_{open}(d_u,d_o,o_u,o_o) = c_1 \times f_{open1}(d_u,d_o,o_u,o_o) + c_2 \times f_{open2}(d_u,d_o,o_u,o_o) \quad (1)$$

where $f_{open1}(d_u,d_o,o_u,o_o)$ describes the first texture characteristic and $f_{open2}(d_u,d_o,o_u,o_o)$ describes the second texture characteristic. The coefficients c_1 and c_2 are constant weighting factors. The choice of these two coefficients is dependent on the trade-off between those two assumptions. In the experiments in the final section, the values of c_1 and c_2 are assigned to be 1.

The term $f_{open1}(d_u,d_o,o_u,o_o)$ consists of the means and variances of chrominance values U in the area of the upper lip A_u, in the area of the lower lip A_o and in the area of the mouth-inside A_b. The areas of A_u, A_o and A_b are defined using the 2D mouth model (ref. Figure 4) and are dependent on the parameters (d_u, d_o, o_u, o_o) to be estimated. The means of chrominance in the lip areas should be the same, but they should be different to the mean of chrominance in the mouth-inside. The variances are utilized to describe the uniformity of the chrominance values U within these areas. The term $f_{open2}(d_u, d_o, o_u, o_o)$ consists of the addends of edge strength (image gradient) $g_Y(x, y)$ along lip contours W_i. The run and length of the parabolic curves are defined with a 2D *mouth-open* model and are dependent on the parameters to be estimated.

The parameters (d_u, d_o, o_u, o_o) in the *mouth-open* model are determined by minimization of the cost $f_{open}(d_u, d_o, o_u, o_o)$. To reduce the computational complexity, the cost function is only evaluated at the already detected candidates for the lip contours (ref. Figure 6). From all possible combinations of the lip contour's candidates, the combination with the least cost is determined as the estimates for the lip thickness and the lip opening heights in the *mouth-open* model.

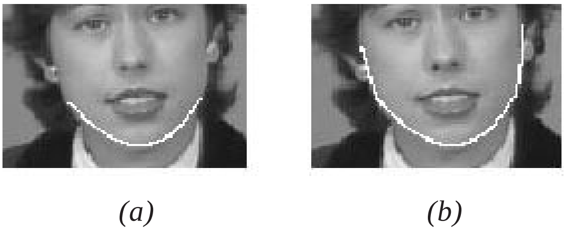
Other Facial Features

In case of the high complexity mode, other facial features besides the eyes and the mouth are estimated.

Chin and cheek contours

For the estimation of chin and cheek contours, the approach described in Kampmann (2002) is used. The chin and the cheeks are represented by a parametric 2D model. This parametric model consists of four parabola branches linked together. The two lower parabola branches represent the chin, the two upper parabola branches define the left and the right cheek. Taking into account the estimated eye and mouth middle positions, search areas for the chin and cheek contours are established. Inside each search area, the probability of the

Figure 7. Estimation of chin and cheek contours: (a) Estimated chin contour, (b) Estimated chin and cheek contours.



(a) *(b)*

occurrence of the chin and cheek contours is calculated for each pixel. This calculation takes advantage of the fact that the chin and cheek contours are more likely to be located in the middle of the search area than at the borders of the search area. An estimation rule is established using this probability and assumes high image gradient values at the position of the chin and cheek contours. By maximization, the position of the chin contour (ref. Figure 7a), as well as the positions of the cheek contours (ref. Figure 7b) are estimated.

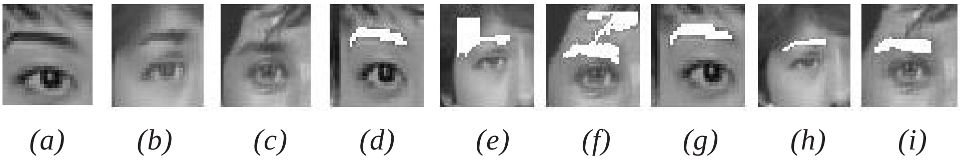
Eyebrows

For estimation of eyebrow features, some knowledge about these features is exploited (Kampmann & Zhang, 1998):

1. Eyebrows are located above the eyes.
2. Eyebrows are darker than the surrounding skin.
3. Eyebrows have a typical curvature, length and thickness.
4. Eyebrows could be covered by hair.

Using this knowledge, eyebrow features are estimated. First, using the estimated eye positions, search areas for the eyebrows are introduced above the eyes. Then, a binarization of the luminance image using a threshold is carried out. In order to determine this threshold, the upper edge of the eyebrow is identified as the maximum value of the luminance gradient. The threshold is now the mean value between the luminance value of the skin and the luminance value of the eyebrow at this upper edge. After binarization, the area with a luminance value below the threshold is checked whether it has the typical curvature, length and thickness of an eyebrow. If the answer is yes, the eyebrow is detected. If the answer is no, the eyebrow is covered by hair. For this case, using the morphological image processing operations *erosion* and *dilation,* as well as knowledge about the typical shape of an eyebrow, it is decided whether the eyebrow is completely or only partly covered by hair. In the case that the

Figure 8. Estimation of eyebrow features: (a)(b)(c) Original image, (d)(e)(f) Eyebrows after binarization, (g)(h)(i) Eyebrows after removal of hair.



eyebrow is only partly covered by hair, the eyebrow is detected by removing the covered hair. Otherwise, the eyebrow is completely covered by hair and cannot be detected. Figure 8 shows the different stages of the eyebrow estimation.

Nose

As nose features, the sides of the nose are estimated (Kampmann & Zhang, 1998). Since the mouth middle position is already determined, search areas for the sides of the nose are introduced above the mouth. Since the edges of the sides of the nose have a specific shape, parametric 2D models of the sides of the nose are introduced. Using this model, each position inside the search area is evaluated as a possible position of the side of the nose. The positions with the maximum accumulated value of the luminance gradient at the nose model's border are chosen as the sides of the nose.

3D Face Model Adaptation

After facial feature estimation is carried out, the two different face models are adapted in the second step. Using perspective projection, vertices of the 3D face models which correspond to the estimated 2D facial features are projected onto the image plane. By comparing these projections with the estimated 2D facial features in the image, the scaling and facial animation parameters of the face models are calculated and the face models are adapted.

Low Complexity 3D Face Model Adaptation

For 3D face model adaptation with low complexity, the face model *Candide* is exploited. For complexity reasons, the scaling and facial animation parameters of *Candide* will be determined with eye and mouth features only (Zhang, 1998). The scaling parameters for the face model *Candide* include the scaling parameters for the face size, for the eye size and for the lip thickness. Before determining the scaling parameters, the face model *Candide* is rotated in such a way that the head tilts of the face model and of the real face in the image match.

The face size is defined by the distance between both eye middle positions, as well as the distances between the eye and mouth middle positions of the face model. The 3D eye and mouth middle positions of the face model are projected onto the image plane. Comparison of the distances of the eyes and the mouth in the image with those of the projections of *Candide* yields the scaling factors for the face size. The eye size is defined as the distance between both eye corner positions. The scaling parameter for the eye size is determined by comparing the projections of the 3D eye corner positions of the face model onto the image plane with the estimated 2D eye corner positions. As the scaling parameter for the lip thickness, displaced vectors of the lip's vertices of the face model are introduced. The vertices that represent the inside contours of the lips are fixed during the scaling of the lip thickness. The vertices that describe the outside contours of the lips are shifted outwards for the scaling of the lip thickness. The eye opening is defined by the positions of the upper and lower eyelids. The position of the upper eyelid can be changed by AU_{41} and that of the lower eyelid by AU_7. Since the 3D face model has fully opened eyes at the beginning, the eyelids of the face model are closed down, so that the opening heights of the face model match the estimated eyelid opening heights in the image plane. The values of AU_{41} and AU_7 are calculated by determining the range of the movement of the eyelids of the face model. The movement of the mouth corners are represented by AU_8 and AU_{12}. AU_8 moves the mouth corners inward and AU_{12} moves them outward. For the determination of these two Action Units' values, the estimated mouth corner positions are utilized. The mouth opening is defined by the position and movement of the upper and lower lips. The position of the upper lip is determined by AU_{10} and that of the lower lip by AU_{16}. For the determination of these two Action Units' values, the estimated 2D opening heights of the lips are used.

High Complexity 3D Face Model Adaptation

For adaptation of the high complexity face model, all estimated facial features (eyes, mouth, eyebrows, nose, chin and cheek contours) are taken into account. Using all estimated facial features, scaling and initial facial animation parameters of the high complexity face model are calculated and used for the face model adaptation to the individual face. First, orientation, size and position of the face model are adapted (only eye and mouth middle positions and cheek contours are used here). Then, the jaw of the face model is rotated. In the next step, the chin and cheek contour of face model is adapted. Finally, scaling and facial animation parameters for the rest of the facial features (eyes, mouth, eyebrows, nose) are determined.

For orientation, the lateral head rotation is adapted first. The quotient of the distances between eye middle positions and cheek contours on both sides of the face is introduced as a measure for the lateral head rotation. For adaptation, the face model is rotated around its vertical axis as long as this quotient measured in the image plane using the estimated facial features is the same as the quotient determined by projecting the face model in the image plane. Then, the head tilt is adapted. The angle between a line through the eye middle positions and the horizontal image line is a measure for the head tilt. Using the measured angle in the image, the tilt of the face model is adapted. After that, the face size is scaled. The distance between the eye middle positions is used for scaling the face width, the distance between the center of the eye middle positions and the mouth middle position for scaling the face height. The next step of face model adaptation is the adjustment of the jaw rotation. Here, the jaw of the face model is rotated until the projection of the face model's mouth opening onto the image plane matches the estimated mouth opening in the image. For scaling of the chin and cheek contours, the chin and cheek vertices of the face model are individually shifted so that their projections match the estimated face contour in the image. In order to maintain the proportions of a human face, all other vertices of the face model are shifted, too. The amount of shift is reciprocal to the distance from the vertex to the face model's chin and cheek. Finally, scaling and facial animation parameters for the rest of the facial features (eyes, mouth, eyebrows, and nose) are calculated by comparing the estimated facial features in the image with projections of the corresponding features of the face model. For scaling, the width, thickness and position of the eyebrows, the width of the eyes, the size of the iris, the width, height and depth of the nose, as well as the lip thickness are determined. For facial animation, the rotation of the eyelids, the translation of the irises, as well as facial muscle parameters for the mouth are calculated. These scaling and facial animation parameters are then used for the adaptation of the high complexity face model.

Experimental Results

For evaluation of the proposed framework, the head and shoulder video sequences *Akiyo* and *Miss America* with a resolution corresponding to CIF and a frame rate of 10Hz are used to test its performance.

Estimation of Facial Features

Figure 9 shows some examples of the estimated eye and mouth features over the original image with the sequence *Akiyo* and *Miss America*. For accuracy evaluation, the true values are manually determined from the natural video sequences, and the standard deviation between the estimated and the true values is measured. The estimate error for the pupil positions is 1.2 pel on average and

Figure 9. Estimated 2D eye and mouth models (Left: Akiyo; Right: Miss America).

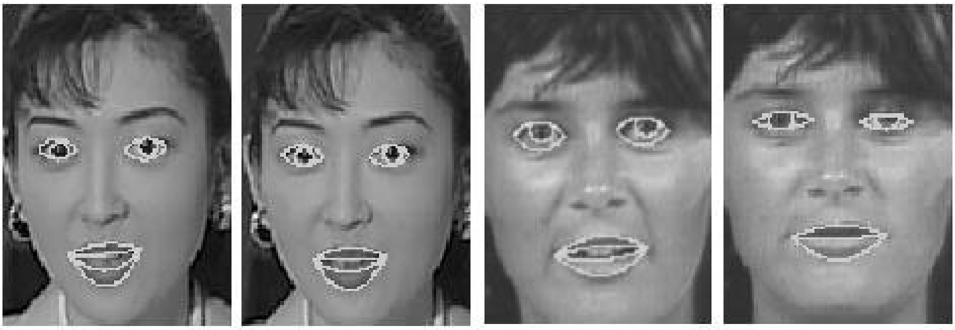
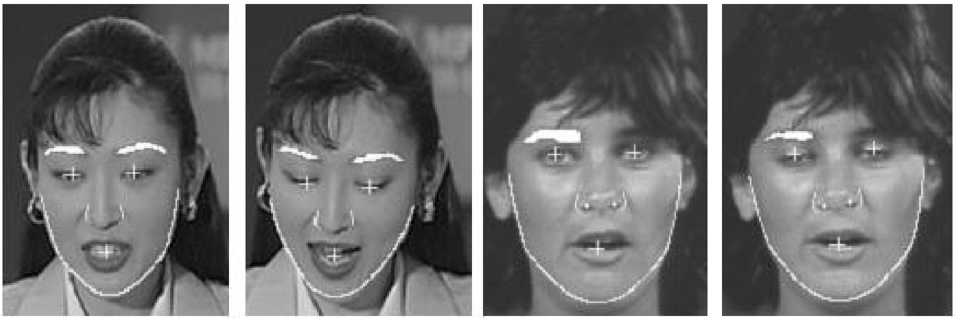


Figure 10. Estimated eyebrows, nose, chin and cheek contours (Left: Akiyo; Right: Miss America). Displayed eye and mouth middle positions are used for determining search areas for the other facial features.



for the eye and mouth corner positions is 1.8 pel. The estimate error for lip thickness and for lip opening heights is 1.5 pel on average, while the error for the eyelid opening heights amounts to 2.0 pels.

Figure 10 shows some results for the estimation of the other facial features (eyebrows, nose, chin and cheek contours) of the high complexity mode. The estimate error for the eyebrows is 2.7 pels on average and for the sides of the nose is 1.8 pel. The estimate error for chin and cheek contours is 2.7 pels on average.

Figure 11. Adapted 3D face model with low complexity over an original image based on estimated eye and mouth features (Left: Akiyo; Right: Miss America).

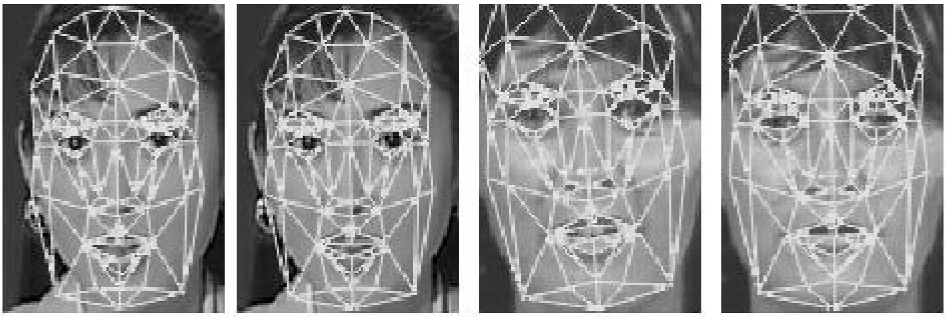
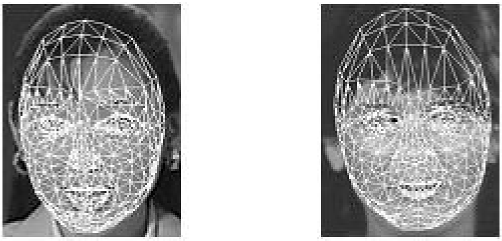


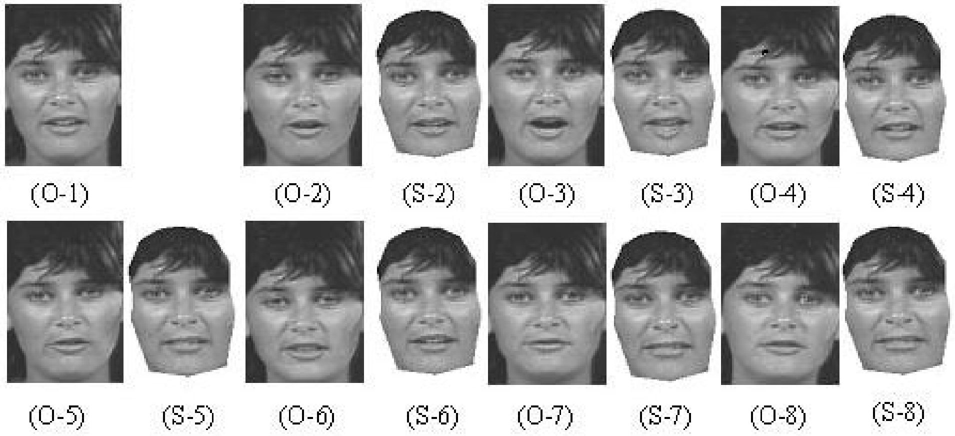
Figure 12. Adapted 3D face model with high complexity over an original image based on all estimated facial features (Left: Akiyo; Right: Miss America).



3D Face Model Adaptation

In order to evaluate the achieved accuracy of the face model adaptation framework, the proposed algorithms are tested with the video sequences *Akiyo* and *Miss America*. Figure 11 shows the adapted face models based on the estimated eye and mouth features shown in Figure 9 (low complexity mode). It can be seen that the adapted face model matches the real face in terms of the eye and mouth features very well. Figure 12 shows the adaptation results for the high complexity mode where all estimated facial features are used for adapting the face model. Here, the high complexity face model matches the original face very well, particularly for the eyebrows, the nose and the face contour.

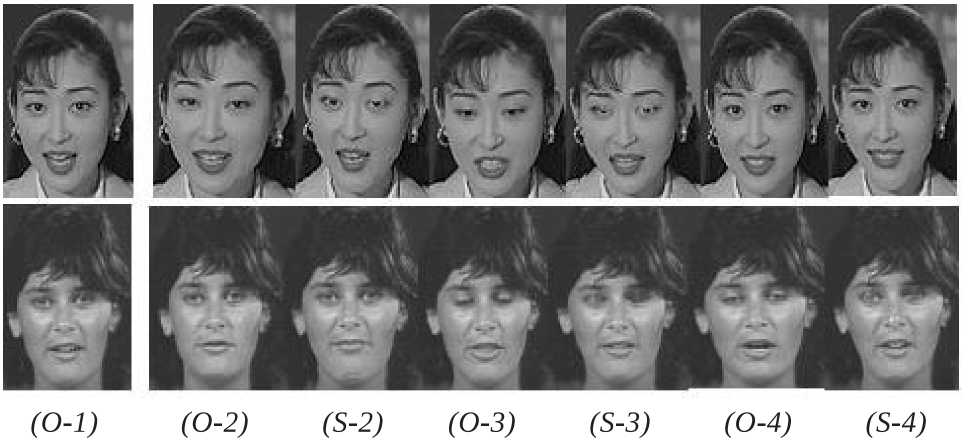
Figure 13. Facial animation using the face model of low complexity: (O-) Original images; (S-) Synthesized images.



Facial Animation

To further demonstrate the effectiveness of the proposed framework, facial animation is also carried out using the two different face models. Figure 13 shows the facial animation results with the face model of low complexity. The texture from the original image (O-1) is mapped onto the face model *Candide* that has been adapted onto this real face. According to estimated eye and mouth features, this face model is then animated. Projecting this animated textured face model

Figure 14. Facial animation using the face model of high complexity: (O-) Original images; (S-) Synthesized images.



onto the image plane creates the synthetic images (S-) which are shown in Figure 13. It can be seen that the quality of the synthetic faces is sufficient, especially for smaller changes of the facial expressions compared with the original image (O-1). For creating a higher quality synthetic face, a more detailed face model with more triangles is necessary. This high complexity face model is textured from the original images (O-1) in Figure 14. The synthetic images (S-) from Figure 14 show the results of animating the high complexity face model. It can be seen that using the high complexity face model results in a visually more impressive facial animation, although at the expense of higher processing complexity.

Conclusions

A framework for automatic 3D face model adaptation has been introduced which is qualified for applications in the field of visual communication like video telephony or video conferencing. Two complexity modes have been realized, a low complexity mode for less powerful devices like a mobile phone and a high complexity mode for more powerful devices such as PCs. This framework consists of two parts. In the first part, facial features in images are estimated. For the low complexity mode, only eye and mouth features are estimated. Here, parametric 2D models for the eyes, the open mouth and the closed mouth are introduced and the parameters of these models are estimated. For the high complexity mode, additional facial features, such as eyebrows, nose, chin and cheek contours, are estimated. In the second part of the framework, the estimated facial features from the first part are used for adapting a generic 3D face model. For the low complexity mode, the 3D face model *Candide* is used, which is adapted using the eye and mouth features only. For the high complexity mode a more detailed 3D face model is used, which is adapted by using all estimated facial features. Experiments have been carried out evaluating the different parts of the face model adaptation framework. The standard deviation of the 2D estimation error is lower than 2.0 pel for the eye and mouth features and 2.7 pel for all facial features. Tests with natural videophone sequences show that an automatic 3D face model adaptation is possible with both complexity modes. Using the high complexity mode, a better synthesis quality of the facial animation is achieved, with the disadvantage of higher computational load.

Endnote

* This work has been carried out at the Institut für Theoretische Nachrichtentechnik und Informationsverarbeitung, University of Hannover, Germany.

References

Ekman, P. & Friesen, V. W. (1977). *Facial action coding system.* Palo Alto, CA: Consulting Psychologists Press.

Essa, I. & Pentland, A. (1997). Coding, analysis, interpretation, and recognition of facial expressions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7), 757-763.

Fischl, J., Miller, B. & Robinson, J. (1993). Parameter tracking in a muscle-based analysis/synthesis coding system. *Picture Coding Symposium* (PCS'93). Lausanne, Switzerland.

Fua, P., Plaenkers, R. & Thalman, D. (1999). From synthesis to analysis: fitting human animation models to image data. *Computer Graphics Interface*, Alberta, Canada.

Kampmann, M. (2002). Automatic 3-D face model adaptation for model-based coding of videophone sequences. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(3), 172-182.

Kampmann, M. & Ostermann, J. (1997). Automatic adaptation of a face model in a layered coder with an object-based analysis-synthesis layer and a knowledge-based layer. *Signal Processing: Image Communication*, 9(3), 201-220.

Kampmann, M. & Zhang, L. (1998). Estimation of eye, eyebrow and nose features in videophone sequences, *International Workshop on Very Low Bitrate Video Coding (VLBV 98),* Urbana, 101-104.

Kuo, C., Huang, R. S. & Lin, T. G. (2002). 3-D facial model estimation from single front-view facial image. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(3), 183-192.

Lee, Y. & Magnenat-Thalmann, N. (2000). Fast head modeling for animation. *Journal Image and Vision Computing*, 18(4), 355-364.

Lee, Y., Terzopoulos, D. & Waters, K. (1995). Realistic modeling for facial animation. *Proceedings of ACM SIGGRAPH'95, Los Angeles*, 55-62.

Musmann, H. (1995). A layered coding system for very low bit rate video coding. *Signal Processing: Image Communication*, 7(4-6), 267-278.

Niem, W. (1994). Robust and fast modelling of 3D natural objects from multiple views. *Proceedings of Image and Video Processing II, San Jose*, 2182, 388-397.

Reinders, M. J. T., van Beek, P. J. L., Sankur, B. & van der Lubbe, J. C. (1995). Facial feature location and adaptation of a generic face model for model-based coding. *Singal Processing: Image Communication*, 7(1), 57-74.

Rydfalk, R. (1987). *CANDIDE*, a parameterised face. *Internal Report Lith-ISY-I-0866*, Linköping University, Linköping, Sweden.

Sarris, N., Grammalidis, N. & Strintzis, M. G. (2001). Building three-dimensional head models. *Graphical Models*, 63(5), 333-368.

Sarris, N., Grammalidis, N. & Strintzis, M. G. (2002). FAP extraction using three-dimensional motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(10), 865-876.

Strub, L. & Robinson, J. (1995). Automated facial conformation for model-based videophone coding. *IEEE International Conference on Image Processing* II, Washington, DC, 587-590.

Yuille, A., Hallinan, P. & Cohen, D. (1992). Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, 8(2), 99-111.

Zhang, L. (1997). Tracking a face for knowledge-based coding of videophone sequences. *Signal Processing: Image Communication*, 10(1-3), 93-114.

Zhang, L. (1998). Automatic adaptation of a face model using action units for semantic coding of videophone sequences. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(6), 781-795.

Chapter X

Learning 3D Face Deformation Model:

Methods and Applications

Zhen Wen
University of Illinois at Urbana Champaign, USA

Pengyu Hong
Harvard University, USA

Jilin Tu
University of Illinois at Urbana Champaign, USA

Thomas S. Huang
University of Illinois at Urbana Champaign, USA

Abstract

This chapter presents a unified *framework for* machine-learning-based facial deformation modeling, analysis *and* synthesis. *It enables flexible, robust face motion analysis and natural synthesis, based on a* compact *face motion model learned from motion capture data. This model, called* Motion Units *(MUs), captures the characteristics of real facial motion. The MU space can be used to constrain noisy low-level motion estimation for robust facial motion analysis. For synthesis, a face model can be deformed by adjusting the weights of MUs. The weights can also be used as visual features to learn* *audio-to-visual mapping using neural networks for real-time, speech-driven, 3D face animation. Moreover, the framework includes parts-based MUs because of the local facial motion and an interpolation scheme to adapt MUs to arbitrary face geometry and mesh topology. Experiments show we can achieve natural face animation and robust non-rigid face tracking in our framework.*

Introduction

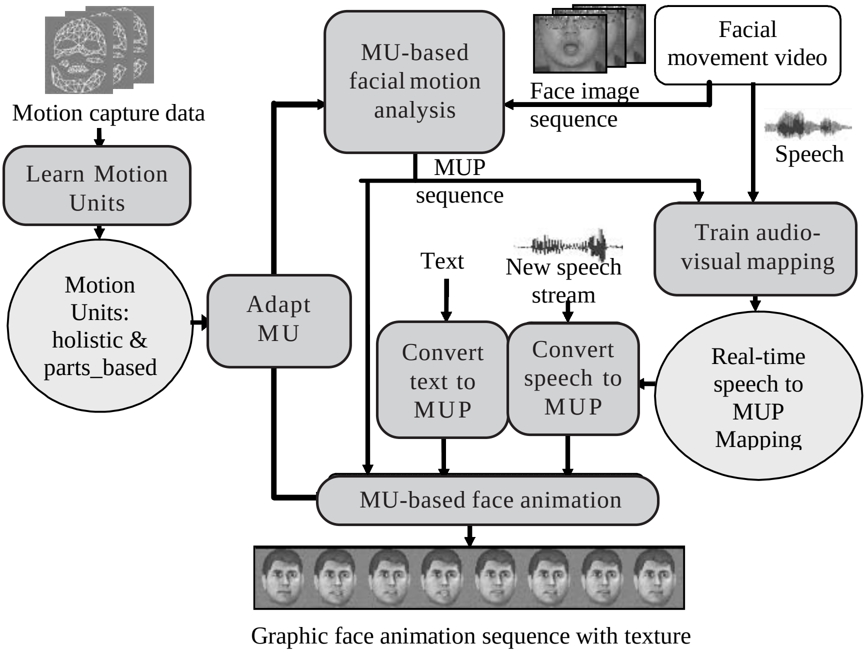
A synthetic human face provides an effective solution for delivering and visualizing information related to the human face. A realistic, talking face is useful for many applications: visual telecommunication (Aizawa & Huang, 1995), virtual environments (Leung et al., 2000), and synthetic agents (Pandzic, Ostermann & Millen, 1999).

One of the key issues of 3D face analysis (tracking and recognition) and synthesis (animation) is to model both temporal and spatial facial deformation. Traditionally, spatial face deformation is controlled by certain facial deformation control models and the dynamics of the control models define the temporal deformation. However, facial deformation is complex and often includes subtle expressional variations. Furthermore, people are very sensitive to facial appearance. Therefore, traditional models usually require extensive manual adjustment for plausible animation. Recently, the advance of motion capture techniques has sparked data-driven methods (e.g., Guenter et al., 1998). These techniques achieve realistic animation by using real face motion data to drive 3D face animation. However, the basic data-driven methods are inherently cumbersome because they require a large amount of data for producing each animation. Besides, it is difficult to use them for facial motion analysis.

More recently, machine learning techniques have been used to learn *compact* and *flexible* face deformation models from motion capture data. The learned models have been shown to be useful for realistic face motion synthesis and efficient face motion analysis. In order to allow machine-learning-based approaches to address the problems of facial deformation, analysis and synthesis in a systematic way, a unified framework is demanded. The unified framework needs to address the following problems: (1) how to learn a compact model from motion capture data for 3D face deformation; and (2) how to use the model for robust facial motion analysis and flexible animation.

In this chapter, we present a unified machine-learning-based framework on facial deformation modeling, facial motion analysis and synthesis. The framework is illustrated in Figure 1. In this framework, we first learn from extensive

Figure 1. The machine learning based framework for facial deformation modeling, facial motion analysis and synthesis.



3D facial motion capture data a *compact* set of *Motion Units* (MUs), which are chosen as the quantitative visual representation of facial deformation. Then, arbitrary facial deformation can be approximated by a linear combination of MUs, weighted by coefficients called *Motion Unit Parameters* (MUPs). Based on facial feature points and a Radial Basis Function (RBF) based interpolation, the MUs can be adapted to new face geometry and different face mesh topology. MU representation is used in both facial motion analysis and synthesis. Within the framework, face animation is done by adjusting the MUPs. For facial motion tracking, the linear space spanned by MUs is used to constrain low-level 2D motion estimation. As a result, more robust tracking can be achieved. We also utilize MUs to learn the correlation between speech and facial motion. A real-time audio-to-visual mapping is learned using an Artificial Neural Network (ANN) from an audio-visual database. Experimental results show that our framework achieved natural face animation and robust non-rigid tracking.

Background

Facial Deformation Modeling

A good survey of facial deformation modeling for animation can be found in Parke & Waters (1996). Representative 3D spatial facial deformation models include free-form interpolation models, parameterized models, physics-based models and, more recently, machine-learning-based models. Free-form interpolation models define a set of points as control points and then use the displacement of control points to interpolate the movements of any facial surface points. Popular interpolation functions include: affine functions (Hong, Wen & Huang, 2001), Splines, radial basis functions, and others. Parameterized models (such as Parke's model (Parke, 1974) and its descendants) use facial-feature-based parameters for customized interpolation functions. Physics-based muscle models (Waters, 1987) use dynamics equations to model facial muscles. The face deformation can then be determined by solving those equations. Because of the high complexity of natural facial motion, these models usually need extensive manual adjustments to achieve realistic facial deformation. To approximate the space of facial deformation using simpler units, some have proposed linear subspaces based on Facial Action Coding System (FACS) (Essa & Pentland, 1997; Tao, 1998). FACS (Ekman & Friesen, 1977) describes arbitrary facial deformation as a combination of Action Units (AUs) of a face. Because AUs are only defined qualitatively without temporal information, they are usually manually customized for computation. Recently, it is possible to collect large amounts of real human motion data. Thus, people turn to apply machine learning techniques to learn the model from the data (Kshirsagar, Molet & Thalmann, 2001; Hong, Wen & Huang, 2002; Reveret & Essa, 2001).

To model temporal facial deformation, some have used simple interpolation schemes (Waters & Levergood, 1993) or customized co-articulation functions (Pelachaud, Badler & Steedman 1991; Massaro, 1998) to model the temporal trajectory between given key shapes. Physics-based methods solve dynamics equations for these trajectories. Recently, Hidden Markov Models (HMM) trained from motion capture data are shown to be useful to capture the dynamics of natural facial deformation (Brand, 1999).

Facial Motion Analysis

Analysis of human facial motion is the key component for many applications, such as model-based, very low-bit-rate video coding for visual telecommunication (Aizawa & Huang, 1995), audio-visual speech recognition (Stork & Hennecke, 1996), and expression recognition (Cohen et al., 2002). Simple approaches only utilize low-level image features (Goto, Kshirsagar & Thalmann, 2001). However, it is not robust enough to use low-level image features alone because the error will be accumulated. High-level knowledge of facial deformation must be used to handle the error accumulation problem by imposing constraints on the possible deformed facial shapes. For 3D facial motion tracking, people have used various 3D deformable model spaces, such as a 3D parametric model (DeCarlo, 1998), MPEG-4 FAP-based B-Spline surface (Eisert, Wiegand & Girod, 2000) and FACS-based models (Tao, 1998). These models, however, are usually manually defined, which cannot capture the real motion characteristics of facial features well. Therefore, some researchers have recently proposed to train facial motion subspace models from real facial motion data (Basu, Oliver & Pentland, 1999; Reveret & Essa, 2001).

Facial Motion Synthesis

Based on spatial and temporal modeling of facial deformation, facial motion is usually synthesized according to semantic input, such as text script (Waters & Levergood, 1993), actor performance (Guenter et al., 1998), or speech (Brand, 1999; Morishima & Harashima, 1991). In this chapter, we focus on real-time speech face animation.

A synthetic talking face is useful for multi-modal human computer interaction, such as e-commerce (Pandzic, Ostermann & Millen, 1999) and computer-aided education (Cole et al., 1999). To generate facial shapes directly from audio, the core issue is the audio-to-visual mapping that converts the audio information into the visual information about facial shapes. HMM-based methods (Brand, 1999) utilize long-term contextual information to generate a smooth facial deformation trajectory. However, they can only be used in off-line scenarios. For real-time mapping, people have proposed various methods such as: Vector Quantization (VQ) (Morishima & Harashima, 1991), Gaussian mixture model (GMM) (Rao & Chen, 1996) and Artificial Neural Network (ANN) (Morishima & Harashima, 1991; Goto, Kshirsagar & Thalmann, 2001). To use short-time contextual information for a smoother result, others have proposed to use a concatenated audio feature over a short time window (Massaro et al., 1999) or to use time-delay neural network (TDNN) (Lavagetto, 1995).

Machine Learning Techniques for Facial Deformation Modeling, Analysis and Synthesis

Artificial Neural Network (ANN) is a powerful tool to approximate functions. It has been used to approximate the functional relationship between motion capture data and the parameters of pre-defined facial deformation models (Morishima, Ishikawa & Terzopoulos, 1998). This helps to automate the construction of a physics-based face muscle model. Moreover, ANN has been used to learn the correlation between facial deformation and other related signals, such as speech (Morishima & Harashima, 1991; Lavagetto, 1995; Massaro et al., 1999).

Because facial deformation is complex, yet structured, Principal Component Analysis (PCA) (Jolliffe, 1986) has been applied to learn a low-dimensional linear subspace representation of 3D face deformation (Kshirsagar, Molet & Thalmann, 2001; Reveret & Essa, 2001). Then, arbitrary complex face deformation can be approximated by a linear combination of just a few basis vectors. Moreover, the low-dimensional linear subspace can be used to constrain noisy low-level motion estimation to achieve more robust 3D facial motion analysis (Reveret & Essa, 2001).

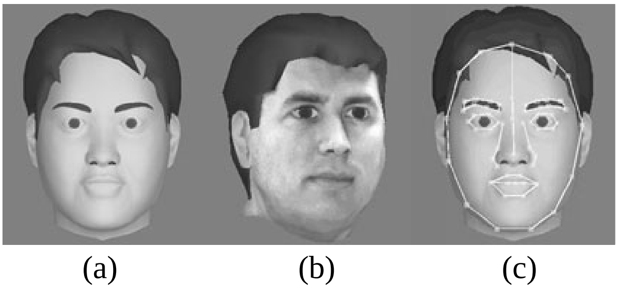
The dynamics of facial motion is complex, so it is difficult to model with analytic equations. A data-driven model, such as the Hidden Markov Model (HMM) (Rabiner, 1989), provides an effective alternative. One example is “voice puppetry” (Brand, 1999), where an HMM trained by entropy minimization is used to learn a dynamic model of facial motion during speech.

Learning 3D Face Deformation Model

In this section, we introduce the methods for a learning 3D face deformation model in our framework. 3D face deformation model describes the spatial and temporal deformation of 3D facial surface. Efficient and effective facial motion analysis and synthesis requires a compact, yet powerful, model to capture real facial motion characteristics. For this purpose, analysis of real facial motion data is needed because of the high complexity of human facial motion.

In this section, we first introduce the motion capture database we used. Then, we present our methods for learning *holistic* and *parts-based* spatial facial deformation models, respectively. Next, we describe how we adapt the learned models to arbitrary face mesh. Finally, we describe the temporal facial deformation modeling. The face models used for MU-based animation are generated by

Figure 2. (a) The generic model in iFACE; (b) A personalized face model based on the cyberware scanner data; (c) The feature points defined on generic model for MU adaptation.

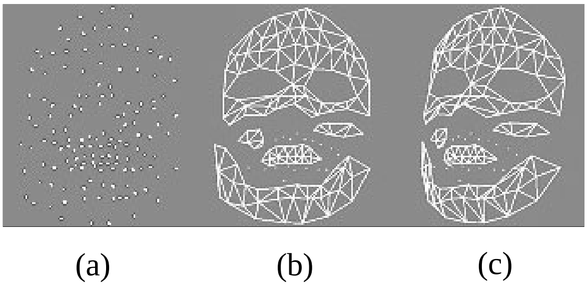


"iFACE," a face modeling and animation system developed in Hong, Wen & Huang (2001). iFACE is illustrated in Figure 2.

The Motion Capture Database

We use motion capture data from Guenter et al. (1998). The database records the 3D facial movements of talking subjects, as well as synchronous audio tracks. The facial motion is captured at the 3D positions of the markers on the faces of subjects. The motion capture data used 153 markers. Figure 3 shows an example of the markers. For the purpose of better visualization, we build a mesh based on those markers, illustrated by Figure 3 (b) and (c).

Figure 3. The markers. (a) The markers shown as small white dots; (b) and (c) The mesh is shown in two different viewpoints.



Learning Holistic Linear Subspace

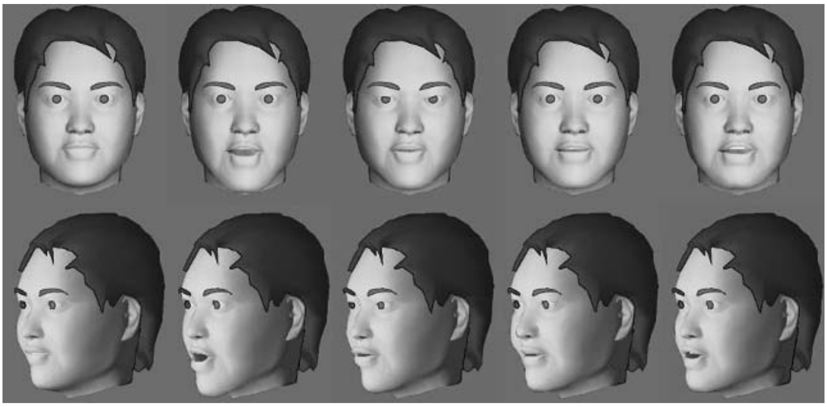
To make complex facial deformation tractable in computational models, researchers have usually assumed that any facial deformation can be approximated by a linear combination of basic deformation. In our framework, we make the same assumption and try to find optimal bases. We call these bases *Motion Units* (MUs). Using MUs, a facial shape $\vec{s}$ can be represented by

$$\vec{s} = \vec{s}_0 + (\sum_{i=1}^{M} c_i \vec{e}_i + \vec{e}_0) \qquad (1)$$

where $\vec{s}_0$ denotes the facial shape without deformation, $\vec{e}_0$ is the mean facial deformation, $\{\vec{e}_0, \vec{e}_1, ..., \vec{e}_M\}$ is the MU set, and $\{c_0, c_1, \ldots, c_M\}$ is the MU parameter (MUP) set.

PCA (Jolliffe, 1986) is applied to learning MUs from the facial deformation of the database. The mean facial deformation and the first seven eigenvectors are selected as the MUs. The MUs correspond to the largest seven eigenvalues that capture 93.2% of the facial deformation variance. The first four MUs are visualized by an animated face model in Figure 4. The top row images are the frontal views of the faces and the bottom row images are side views. The first face is the neutral face, corresponding to $\vec{s}_0$. The remaining faces are deformed by the first four MUs scaled by a constant (from left to right). The method for

Figure 4. The neutral and deformed faces corresponding to the first four MUs. The top row is the frontal view and the bottom row is the side view.



visualizing MUs is described in the subsection "MU adaptation." Any arbitrary facial deformation can be approximated by a linear combination of the MUs, weighted by the MUPs.

Learning Parts-Based Linear Subspace

It is well known that the facial motion is localized, which makes it possible to decompose the complex facial motion into parts. The decomposition helps reduce the complexity in deformation modeling and improves the analysis' robustness and the synthesis' flexibility. The decomposition can be done manually based on the prior knowledge of facial muscle distribution (Tao, 1998). However, it may not be optimal for the linear model used because of the high nonlinearity of facial motion. Parts-based learning techniques provide a way to help design *parts-based facial deformation models*, which can better approximate real, local facial motion. Recently, Non-negative Matrix Factorization (NMF) (Lee & Seung, 1999), a technique for learning localized representation of data samples, has been shown to be able to learn basis images that resemble parts of faces. In learning the basis of subspace, NMF imposes non-negativity constraints, which is compatible to the intuitive notion of combining parts to form a whole in a non-subtractive way.

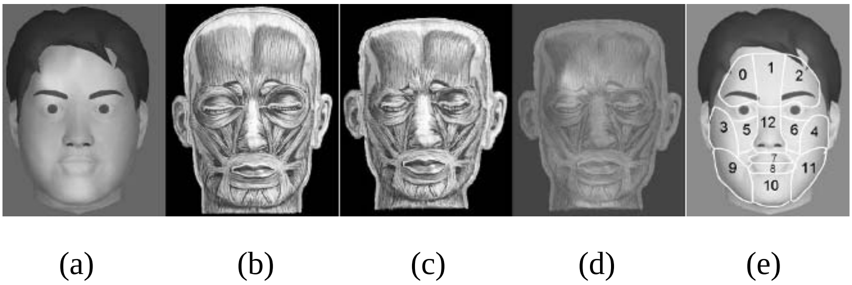
In our framework, we present a parts-based face deformation model. In the model, each part corresponds to a facial region where facial motion is mostly generated by local muscles. The motion of each part is modeled by PCA. Then, the overall facial deformation is approximated by summing up the deformation in each part:

$$\Delta\vec{s} = \sum_{j=1}^{N} \Delta\vec{s}_j = \sum_{j=1}^{N} \left(\sum_{i=1}^{M_j} c_{ij}\vec{e}_{ij} + \vec{e}_{0j}\right),$$

where $\Delta\vec{s} = \vec{s} - \vec{s}_0$ is the deformation of the facial shape. N is the number of parts. We call this representation *parts-based MU*, where the j-th part has its MU set { $\vec{e}_{0j}$, $\vec{e}_{1j}$, ..., $\vec{e}_{Mj}$ }, and MUP set { c_{0j} , c_{1j} , …, c_{Mj} }.

To decompose facial motion into parts, we propose an NMF-based method. In this method, we randomly initialize the decomposition. Then, we use NMF to reduce the linear decomposition error to a local minimum. We impose the non-negativity constraint in the linear combination of the facial motion energy. Figure 5(a) shows some parts derived by NMF. Adjacent different parts are shown in different colors that are overlayed on the face model. We then use prior knowledge about facial muscle distribution to refine the learned parts. The parts

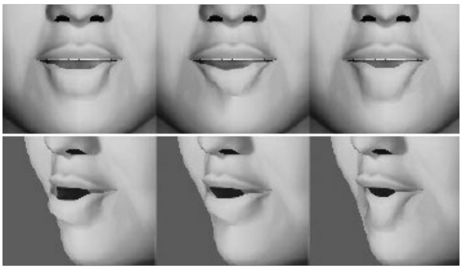
Figure 5. (a) NMF learned parts overlayed on the generic face model; (b) The facial muscle distribution; (c) The aligned facial muscle distribution; (d) The parts overlayed on muscle distribution; (e) The final parts.



can thus be: (1) more related to meaningful facial muscle distribution; and (2) less biased by individuality in the motion capture data and, thus, more easily generalized to different faces. We start with an image of human facial muscle, illustrated in Figure 5(b) (Facial muscle image, 2002). Next, we align it with our generic face model via image warping, based on facial feature points illustrated in Figure 2(c). The aligned facial muscle image is shown in Figure 5(c). Then, we overlay the learned parts on facial muscle distribution (Figure 5(d)) and interactively adjust the learned parts such that different parts correspond to different muscles. The final parts are shown in Figure 5(e).

The learned parts-based MUs give more flexibility in local facial deformation analysis and synthesis. Figure 6 shows some local deformation in the lower lips, each of which is induced by one of the learned parts-based MUs. These locally deformed shapes are difficult to approximate using holistic MUs. For each local deformation shown in Figure 6, more than 100 holistic MUs are needed to achieve 90% reconstruction accuracy. That means, although some local deformation is induced by only one parts-based MU, more than 100 holistic MUs may be needed in order to achieve good analysis and synthesis quality. Therefore, we can have more flexibility in using parts-based MUs. For example, if we are only interested in lip motion, we only need to learn parts-based MUs from lip motion data. In face animation, people often want to animate local regions separately. This task can be easily achieved by adjusting the MUPs of parts-based MUs separately. In face tracking, people may use parts-based MUs to track only the region of their interests (e.g., the lips). Furthermore, tracking using parts-based MUs is more robust because local error will not affect distant regions.

Figure 6. Three lower lip shapes deformed by three of the lower lip parts-based MUs respectively. The top row is the frontal view and the bottom row is the side view.



MU Adaptation

The learned MUs are based on the motion capture data of particular subjects. To use the MUs for other people, they need to be fitted to the new face geometry. Moreover, the MUs only sample the facial surface motion at the position of the markers. The movements at other places need to be interpolated. In our framework, we call this process "MU adaptation."

Interpolation-based techniques for re-targeting animation to new models, such as Noh & Neumann (2001), could be used for MU adaptation. Under more principled guidelines, we design our MU adaptation as a two-step process: (1) face geometry based MU-fitting; and (2) MU re-sampling. These two steps can be improved in a systematic way if enough MU sets are collected. For example, if MU statistics over a large set of different face geometries are available, one can systematically derive the geometry-to-MU mapping using machine-learning techniques. On the other hand, if multiple MU sets are available which sample different positions of the same face, it is possible to combine them to increase the spatial resolution of MU because markers in MU are usually sparser than face geometry mesh.

The first step, called "MU fitting," fits MUs to a face model with different geometry. We assume that the corresponding positions of the two faces have the same motion characteristics. Then, the "MU fitting" is done by moving the markers of the learned MUs to their corresponding positions on the new face. We interactively build the correspondence of facial feature points shown in Figure 2(c) via a GUI. Then, warping is used to interpolate the remaining correspondence.

The second step is to derive movements of facial surface points that are not sampled by markers in MUs. We use the radial basis interpolation function. The family of radial basis functions (RBF) is widely used in face animation (Guenter et al., 1998; Marschner, Guenter & Raghupathy, 2000; Noh & Neumann, 2001). Using RBF, the displacement of a certain vertex $\vec{v}_i$ is of the form

$$\Delta\vec{v}_i = \sum_{j=1}^{N} w_{ij} h(\|\vec{v}_i - \vec{p}_j\|)\Delta\vec{p}_j \tag{2}$$

where $\vec{p}_j$, $(j = 1,\ldots, N)$ is the coordinate of a marker, and $\Delta\vec{p}_j$ is its displacement. h is a radial basis kernel function, and w_{ij} are the weights. h and w_{ij} need to be carefully designed to ensure the interpolation quality. For facial deformation, the muscle influence region is local. Thus, we choose a cut-off region for each vertex. We set the weights to be zero for markers that are outside of the cut-off region, i.e., they are too far away to influence the vertex. In our current system, the local influence region for the i-th vertex is heuristically assigned as a circle, with the radius r_i as the average of the distances to its two nearest neighbors. Similar to Marschner, Guenter & Raghupathy (2000), we choose the radial basis kernel to be $h(x) = (1 + \cos(\pi \cdot x/r_i)/2$, where $x = \|\vec{v}_i - \vec{p}_j\|$. We choose w_{ij} to be a normalization factor such that $\sum_{j=1}^{N} w_{ij} h(\|\vec{v}_i - \vec{p}_j\|) = 1$. The lips and eye lids are two special cases for this RBF interpolation, because the motions of the upper parts of them are not correlated with the motions of the lower parts. To address this problem, we add "upper" or "lower" tags to vertices and markers near the mouth and eyes. Markers do not influence vertices with different tags. These RBF weights need to be computed only once for one set of marker positions. The weights are stored in a matrix, which is sparse because marker influence is local. During synthesis, the movement of mesh vertices can be computed by one multiplication of the sparse RBF matrix based on equation (2). Thus, the interpolation is fast.

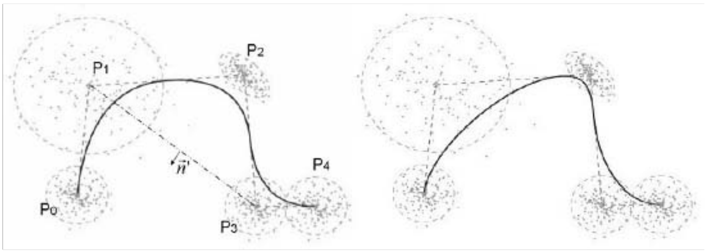
Temporal Facial Deformation

Temporal facial deformation model describes temporal variation of facial deformation given constraints (e.g., key shapes) at certain time instances. For compactness and usability, we propose to use an HMM-based model trained by a standard HMM training algorithm, which employs only a few HMM states for modeling key facial shapes. In order to get a smooth trajectory once a state sequence is found, we use NURBS (Nonuniform Rational B-splines) interpolation. The NURBS trajectory is defined as:

$$C(t) = (\sum_{i=0}^{n} N_{i,p}(t) w_i \mathbf{P}_i) \Big/ (\sum_{i=0}^{n} N_{i,p}(t) w_i),$$

where p is the order of the NURBS, $N_{i,p}$ is the basis function, $\mathbf{P}_i$ is the control point of the NURBS, and w_i is the weight of $\mathbf{P}_i$. We use $p = 2$. The HMM states (key facial shapes) are used as control points, which we assume to have Gaussian distributions. We set the weight of each control point such that the trajectory has a higher likelihood. Intuitively, it can be achieved in a way that states with small variance pull the trajectory towards them, while states with larger variance allow the trajectory to stay away from them. Therefore, we set the weights to be $w_i = 1 / (\sigma(\vec{n}_i))$, where $\vec{n}_i$ is the trajectory normal vector that also passes $\mathbf{P}_i$, $\sigma(\vec{n}_i)$ is the variance of the Gaussian distribution in $\vec{n}_i$ direction. In practice, we approximate $\vec{n}_i$ by normal vector $\vec{n}_i'$ of line segment $\overline{\mathbf{P}_{i-1}\mathbf{P}_{i+1}}$ (see $\overline{\mathbf{P}_1\mathbf{P}_3}$ in Figure 7(a)). Compared to Brand (1999), the smooth trajectory obtained is less optimal in terms of maximum likelihood. But, it is fast and robust, especially when the number of states is small. It is also a natural extension of the traditional key-frame-based spline interpolation scheme, which is easy to implement. Figure 7 shows a synthetic example comparing conventional NURBS and our statistically weighted NURBS. The green dots are samples of facial shapes. The red dashed line connects centers of the states. The blue solid line is the generated facial deformation trajectory. In Figure 7(b), the trajectory is pulled towards the states with smaller variance, thus they have a higher likelihood than trajectory in Figure 7(a).

Figure 7. (a) Conventional NURBS; (b) Statistically weighted NURBS interpolation.



(a) (b)

We use the motion capture sequence to train the model. Thirty states are used in the experiment. Each state is observed via MUPs, which are modeled using a Gaussian model. We assume the covariance matrices of the states to be diagonal in training. The centers of the trained states are used as key shapes (i.e., control points) for the NURBS interpolation scheme. The interpolation-based temporal model is used in the key-frame-based face animation, such as text-driven animation in iFACE.

Model-Based Facial Motion Analysis

In this section, we describe model-based facial motion analysis. In the existing 3D non-rigid face tracking algorithm using 3D facial deformation model, the subspace spanned by the Action Units (AUs) is used as the constraints of low-level image motion. Similar to MUs, AUs are defined in such a way that arbitrary facial deformation is approximated by a linear combination of AUs. However, the AUs are usually manually designed. For these approaches, our automatically learned MUs can be used in place of the manually designed AUs. In this way, extensive manual intervention can be avoided and natural facial deformation can be approximated better.

We choose to use the learned MUs in the 3D non-rigid face tracking system proposed in Tao (1998) because it has been shown to be robust and real-time. The facial motion observed in an image plane can be represented by

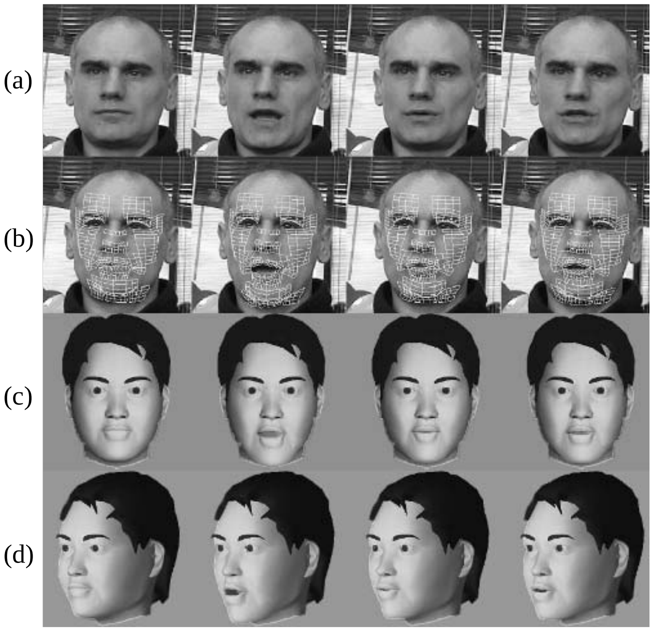
$$\mathbf{M}(\mathbf{R}(\vec{V}_0 + \mathbf{L}\vec{P}) + \vec{T}) \tag{3}$$

where $\mathbf{M}$ is the projection matrix, $\vec{V}_0$ is the neutral face, $\mathbf{L}\vec{P}$ defines the non-rigid deformation, $\mathbf{R}$ is the 3D rotation decided by three rotation angles $[w_x, w_y, w_z]^t = \vec{W}$, and $\vec{T}$ stands for 3D translation. $\mathbf{L}$ is an $N \times M$ matrix that contains M AUs, each of which is an M dimensional vector. $\vec{P} = [p_1, ..., p_M]^t$ is the coefficients of the AUs. To estimate facial motion parameters $\{\vec{T}, \vec{W}, \vec{P}\}$ from 2D inter-frame motion $\mathrm{d}\vec{V}_{2D}$, the derivative of equation (3) is taken with respect to $\{\vec{T}, \vec{W}, \vec{P}\}$. Then, a linear equation between $\mathrm{d}\vec{V}_{2D}$ and $\{\mathrm{d}\vec{T}, \mathrm{d}\vec{W}, \mathrm{d}\vec{P}\}$

can be derived (see details in Tao (1998)). The system estimates $d\hat{V}_{2D}$ using template-matching-based optical flow. The linear system is solved using the least squares method. A multi-resolution framework is used for efficiency and robustness.

In the original system, is manually designed using Bezier volume and represented by the displacements of vertices of face surface mesh. To derive **L** from the learned MUs, the "MU adaptation" process described earlier is used. In the current system, we use the holistic MUs. Parts-based MUs could be used if a certain local region is the focus of interest, such as the lips in lip-reading. The system is implemented to run on a 2.2 GHz *Pentium* 4 processor with 2GB memory. The image size of the input video is 640 × 480. The system works at 14 Hz for non-rigid face tracking. The tracking results, i.e., the coefficients of MUs, **R** and $\vec{T}$ can be directly used to animated face models. Figure 8 shows some typical frames that were tracked, along with the animated face model to

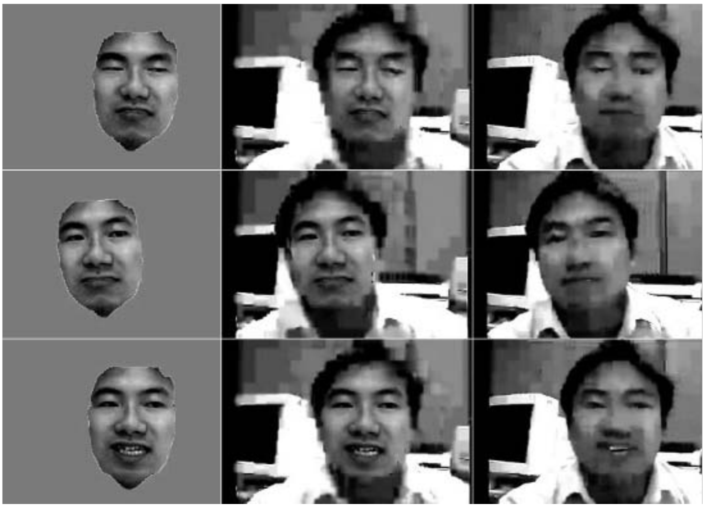
Figure 8. Typical tracked frames and corresponding animated face models. (a) The input frames; (b) The tracking results visualized by yellow mesh; (c) The front views of the face model animated using tracking results; (d) The side views of the face model animated using tracking results. In each row, the first image corresponds to neutral face.



visualize the results. It can be observed that compared with the neutral face (the first column images), the mouth opening (the second column), subtle mouth rounding and mouth protruding (the third and fourth columns) are captured in the tracking results visualized by the animated face model. Because the motion units are learned from real facial motion data, the facial animation synthesized using tracking results looks more natural than that using handcrafted action units in Tao (1998).

The tracking algorithm can be used in model-based face video coding (Tu et al., 2003). We track and encode the face area using model-based coding. To encode the residual in the face area and the background for which *a priori* knowledge is not generally available, we use traditional waveform-based coding method H.26L. This hybrid approach improves the robustness of the model-based method at the expense of increasing bit-rate. Eisert, Wiegand & Girod (2000) proposed a similar hybrid coding technique using a different model-based 3D facial motion tracking approach. We capture and code videos of 352 × 240 at 30Hz. At the same low bit-rate (18 kbits/s), we compare this hybrid coding with H.26L JM 4.2 reference software. Figure 9 shows three snapshots of a video with 147 frames. The PSNR around the facial area for hybrid coding is 2dB higher than H.26L. Moreover, the hybrid coding results have much higher visual quality. Because our tracking system works in real-time, it could be used in a real-time low-bit-rate video-phone application. Furthermore, the tracking results

Figure 9. (a) The synthesized face motion; (b) The reconstructed video frame with synthesized face motion; (c) The reconstructed video frame using H.26L codec.



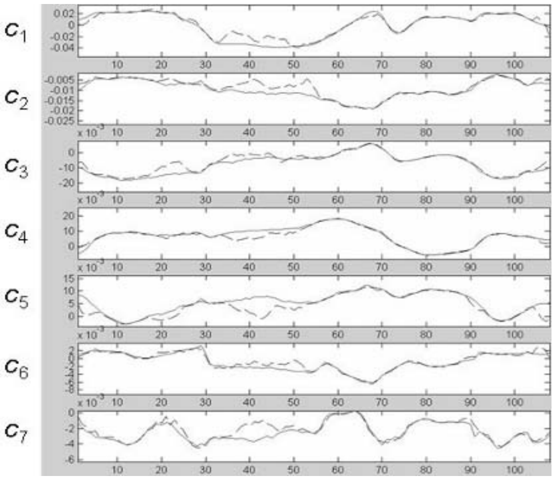
can used to extract visual features for audio-visual speech recognition and emotion recognition (Cohen et al., 2002). In medical applications related to facial motion disorder, such as facial paralysis, visual cues are important for both diagnosis and treatment. Therefore, the facial motion analysis method can be used as a diagnostic tool such as in Wachtman et al. (2001). Compared to other 3D non-rigid facial motion tracking approaches using a single camera, the features of our tracking system include: (1) the deformation space is learned automatically from data such that it avoids manual adjustments; (2) it is real-time so that it can be used in real-time applications; and (3) it is able to recover from temporary loss of tracking by incorporating a template-matching-based face detection module.

Real-Time Speech-Driven 3D Face Animation

In this section, we present the real-time speech-driven 3D face animation algorithm in our 3D face analysis and synthesis framework. We use the facial motion capture database used for learning MUs along with its audio track for learning audio-to-visual mapping. For each 33 ms window, we calculate the holistic MUPs as the visual features and 12 Mel-frequency cepstrum coefficients (MFCCs) (Rabiner & Juang, 1993) as the audio features. To include contextual information, the audio feature vectors of frames *t-3, t-2, t-1, t, t+1, t+2*, and *t+3,* are concatenated as the final audio feature vector of frame *t*.

The training audio-visual data is divided into 21 groups based on the audio feature of each data sample. The number 21 is decided heuristically based on audio feature distribution of the training database. One of the groups corresponds to silence. The other 20 groups are automatic generated using the k-means algorithm. Then, the audio features of each group are modeled by a Gaussian model. After that, a three-layer perceptron is trained to map the audio features to the visual features using each audio-visual data group. At the estimation phase, we first classify an audio vector into one of the audio feature groups whose Gaussian model gives the highest score for the audio feature vector. We then select the corresponding neural network to map the audio feature vector to MUPs, which can be used in equation (1) to synthesize the facial shape. A method using triangular average window is used to smooth the jerky mapping results. For each group, 80% of the data is randomly selected for training and 20% for testing. The maximum and minimum number of the hidden neurons is 10 and 4, respectively. A typical estimation result is shown in Figure 10. The horizontal axes in the figure represent time. The vertical axes represent the

Figure 10. Compare the estimated MUPs with the original MUPs. The content of the corresponding speech track is "A bird flew on lighthearted wing."



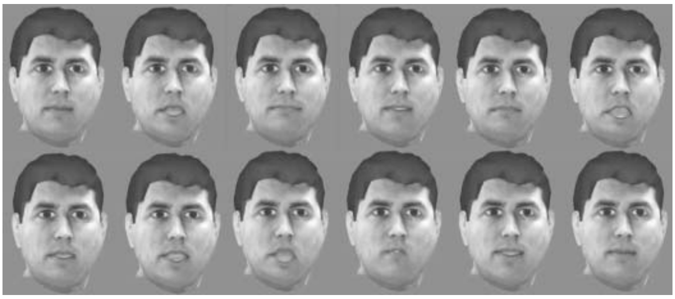
magnitude of the MUPs. The solid red trajectory is the original MUPs, and the dashed blue trajectory is the estimation results.

We reconstruct the facial deformation using the estimated MUPs. For both the ground truth and the estimated results, we divide the deformation of each marker by its maximum absolute displacement in the ground truth data. To evaluate the performance, we calculate the Pearson product-moment correlation coefficients (R) and the mean square error (MSE) using the normalized deformations. The Pearson product-moment correlation ($0.0 \leq R \leq 1.0$) measures how good the global match is between the shapes of two signal sequences. A large coefficient means a good match. The Pearson product-moment correlation coefficient R between the ground truth $\{\vec{d}_n\}$ and the estimated data $\{\vec{d}_n'\}$ is calculated by

$$R = \frac{tr(E[(\vec{d}_n - \vec{\mu}_1)(\vec{d}_n' - \vec{\mu}_2)^T])}{\sqrt{tr(E[(\vec{d}_n - \vec{\mu}_1)(\vec{d}_n - \vec{\mu}_1)^T])tr(E[(\vec{d}_n' - \vec{\mu}_2)(\vec{d}_n' - \vec{\mu}_2)^T])}} \qquad (4)$$

where $\vec{\mu}_1 = E[\vec{d}_n]$ and $\vec{\mu}_2 = E[\vec{d}_n']$. In our experiment, R = 0.952 and MSE = 0.0069 for training data and R = 0.946 and MSE = 0.0075 for testing data.

Figure 11. Typical animation frames. Temporal order: from left to right; from top to bottom.



The whole animation procedure contains three steps. First, we extract audio features from the input speech. Then, we use the trained neural networks to map the audio features to the visual features (i.e., MUPs). Finally, we use the estimated MUPs to animate a personalized 3D face model in iFACE. Figure 11 shows a typical animation sequence for the sentence in Figure 10.

Our real-time speech-driven animation can be used in real-time two-way communication scenarios such as video-phone and virtual environments (Leung et al., 2000). On the other hand, existing off-line speech-driven animation, e.g., Brand (1999), can be used in one-way communication scenarios, such as broadcasting and advertising. Our approach deals with the mapping of both vowels and consonants, thus it is more accurate than real-time approaches with only vowel-mapping (Morishima & Harashima, 1991; Goto, Kshirsagar & Thalmann, 2001). Compared to real-time approaches using only one neural network for all audio features (Massaro et al., 1999; Lavagetto, 1995), our local ANN mapping (i.e., one neural network for each audio feature cluster) is more efficient because each ANN is much simpler. Therefore, it can be trained with much less effort for a certain set of training data. More generally, speech-driven animation can be used in speech and language education (Cole et al., 1999), as a speech understanding aid for noisy environments and hard-of-hearing people and as a rehabilitation tool for facial motion disorders.

Human Emotion Perception Study

The synthetic talking face can be evaluated by human perception study. Here, we describe our experiments which compare the influence of the synthetic talking face on human emotion perception with that of the real face. We did similar experiments for 2D MU-based speech-driven animation (Hong, Wen &

Huang, 2002). We videotape a subject who is asked to calmly read three sentences with three different facial expressions: (1) neutral, (2) smile, and (3) sad, respectively. The 3 sentences are: (1) "It is normal," (2) "It is good," and (3) "It is bad." The associated information is: (1) neutral, (2) positive, and (3) negative. The audio tracks are used to generate three sets of face animation sequences. All three audio tracks are used in each set of animation sequences. The three sets are generated with a neutral expression, smiling, and sad, respectively. The facial deformation due to speech and expression is linearly combined in our experiments. Sixteen untrained human subjects, who never used our system before, participate in the experiments.

The first experiment investigates human emotion perception based on either the visual-only or audio-only stimuli. The subjects are first asked to infer their emotional states based on the animation sequences without audio. The emotion inference results in terms of the number of the subjects are shown in Table 1. As shown, the effectiveness of the synthetic talking face is comparable with that of the real face. The subjects are then asked to listen to the audio and decide the emotional state of the speaker. Note that the audio tracks are produced without emotions. The results in terms of the number of the subjects are shown in Table 1.

Table 1. Emotion inference based on visual only or audio only stimuli. "S" column: Synthetic face; "R" column: Real face.

		Facial Expression						Audio		
		Neutral		Smile		Sad		1	2	3
		S	R	S	R	S	R			
Emotion	Neutral	16	16	4	3	2	0	16	6	7
	Happy	0	0	12	13	0	0	0	10	0
	Sad	0	0	0	0	14	16	0	0	9

Table 2. Emotion inference results agreed with facial expressions. The inference is based on both audio and visual stimuli. "S" column: Synthetic face; "R" column: Real face.

		Facial Expression			
		Smile		Sad	
		S	R	S	R
Audio-visual relation	Same	15	16	16	16
	Opposite	2	3	10	12

The second and third experiments are designed to compare the influence of a synthetic face on bimodal human emotion perception and that of the real face. In the second experiment, the subjects are asked to infer the emotional state while observing the synthetic talking face and listening to the audio tracks. The third experiment is the same as the second one except that the subjects observe the real face, instead. In each of the experiments, the audio-visual stimuli are presented in two groups. In the first group, audio content and visual information represent the same kind of information (e.g., positive text with smiling expression). In the second group, the relationship is the opposite. The results are combined in Table 2.

We can see the face movements and the content of the audio tracks jointly influence the decisions of the subjects. If the audio content and the facial expressions represent the same kind of information, the human perception of the information is enhanced. For example, when the associated facial expression of the positive-text-content audio track is smiling, nearly all subjects say that the emotional state is happy (see Table 2). The numbers of the subjects who perceive a happy emotional state are higher than those using only one stimulus alone (see Table 1). However, it confuses human subjects if the facial expressions and the audio tracks represent opposite information. An example is shown in the fifth and sixth columns of Table 2. The audio content conveys positive information, while the facial expression is sad. Ten subjects report sad emotion if the synthetic talking face with a sad expression is shown. The number increases to 12 if the real face is used. This difference shows that the subjects tend to trust the real face more than the synthetic face when the visual information conflicts with the audio information. Overall, the experiments show that our real-time, speech-driven synthetic talking face successfully affects human emotion perception. The effectiveness of the synthetic face is comparable with that of the real face, even though it is slightly weaker.

Conclusions

This chapter presents a unified framework for learning compact facial deformation models from data and applying the models to facial motion analysis and synthesis. This framework uses a 3D facial motion capture database to learn *compact* holistic and parts-based facial deformation models called MUs. The MUs are used to approximate arbitrary facial deformation. The learned models are used in robust 3D facial motion analysis and real-time, speech-driven face animation. The experiments demonstrate that robust non-rigid face tracking and flexible, natural face animation can be achieved based on the learned models. In the future, we plan to investigate systematic ways of adapting learned models for new people, capturing appearance variations along with geometric deformation in motion capture data for subtle, yet perceptually important, facial deformation.

Acknowledgment

This work was supported in part by National Science Foundation Grant CDA 96-24386, and IIS-00-85980. We would thank Dr. Brian Guenter, Heung-Yeung Shum and Yong Rui of Microsoft Research for the face motion data.

References

Aizawa, K. & Huang, T. S. (1995). Model-based image coding, *Proceeding of IEEE*, 83, 259-271.

Basu, S., Oliver, N. & Pentland, A. (1999). 3D modeling and tracking of human lip motions. *Proceeding of International Conference on Computer Vision*, 337-343.

Brand, M. (1999). Voice puppetry. *Proceeding of SIGGRAPH 1999*, 21-28.

Cohen, I. et al. (2002). Facial expression recognition from video sequences. *Proceeding of IEEE International Conference on Multimedia and Expo*, 121-124.

Cole, R. et al. (1999). New tools for interactive speech and language training: Using animated conversational agents in the classrooms of profoundly deaf children. *Proceedings of ESCA/SOCRATES Workshop on Method and Tool Innovations for Speech Science Education.*

DeCarlo, D. (1998). Generation, Estimation and Tracking of Faces. Ph.D. thesis, University of Pennsylvania.

Eisert, P., Wiegand, T. & Girod, B. (2000). Model-Aided Coding: A New Approach to Incorporate Facial Animation into Motion-Compensated Video Coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 10(3), 344-358.

Ekman, P. & Friesen, W. V. (1977). Facial Action Coding System. Consulting Psychologists Press.

Essa, I. & Pentland, A. (1997). Coding Analysis, Interpretation, and Recognition of Facial Expressions. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 757-763.

Facial muscle image. (2002). Retrieved from the World Wide Web: http://sfghed.ucsf.edu/ClinicImages/anatomy.htm.

Goto, T., Kshirsagar, S. & Thalmann, N. M. (2001). Automatic Face Cloning and Animation. *IEEE Signal Processing Magazine*, 18(3), 17-25.

Guenter. B. et al. (1998). Making Faces. *Proceeding of SIGGRAPH 1998*, 55-66.

Hong, P., Wen, Z. & Huang, T. S. (2001). iFACE: A 3D synthetic talking face. *International Journal of Image and Graphics*, 1(1), 19-26.

Hong, P., Wen, Z. & Huang, T. S. (2002). Real-time speech driven expressive synthetic talking faces using neural networks. *IEEE Transactions on Neural Networks*, 13(4), 916-927

Jolliffe, I. T. (1986). *Principal Component Analysis*. Springer-Verlag.

Kshirsagar, S., Molet, T. & Thalmann, N. M. (2001). Principal Components of Expressive Speech Animation. *Proceeding of Computer Graphics International,* 38-44.

Lavagetto, F. (1995). Converting speech into lip movements: A multimedia telephone for hard of hearing people. *IEEE Transactions on Rehabilitation Engineering,* 90-102.

Lee, D. D. & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401, 788-791.

Leung, W. H. et al. (2000). Networked Intelligent Collaborative Environment (NetICE). *Proceeding of IEEE International Conference on Multimedia and Expo*, 55-62.

Marschner, S. R., Guenter, B. & Raghupathy, S. (2000). Modeling and Rendering for Realistic Facial Animation. *Proceedings of Workshop on Rendering*, 231-242.

Massaro, D. W. (1998). *Perceiving talking faces: From speech perception to a behavioral principle*. Cambridge, MA: MIT Press.

Massaro, D. W. et al. (1999). Picture my voice: audio to visual speech synthesis using artificial neural networks. *Proceeding of Audio-Visual Speech Processing*, 133-138.

Morishima, S. & Harashima, H. (1991). A media conversion from speech to facial image for intelligent man-machine interface. *IEEE Journal on Selected Areas in Communications,* 4,594-4,599.

Morishima, S., Ishikawa, T. & Terzopoulos, D. (1998). Facial muscle parameter decision from 2D frontal image. *Proceedings of the International Conference on Pattern Recognition,* 160-162.

Noh, J. & Neumann, U. (2001). Expression Cloning. *Proceeding of SIGGRAPH 2001*, 277-288.

Pandzic, I., Ostermann, J. & Millen, D. (1999). User evaluation: Synthetic talking faces for interactive services. *The Visual Computer*, 15, 330-340.

Parke, F. I. (1974). A Parametric Model of human Faces. Ph.D. thesis, University of Utah.

Parke, F. I. & Waters, K. (1996). *Computer Facial Animation*. A. K. Peters.

Pelachaud. C., Badler, N. I. & Steedman, M. (1991). Linguistic issues in facial animation. *Proceeding of Computer Animation*, 15-30.

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceeding of IEEE*, 77(2), 257-286.

Rabiner, L. & Juang, B. H. (1993). *Fundamentals of Speech Recognition*. Prentice-Hall.

Rao, R. & Chen, T. (1996). Exploiting audio-visual correlation in coding of talking head sequences. *Proceeding of Picture Coding Symposium*.

Reveret, L. & Essa, I. (2001). Visual Coding and Tracking of Speech Related Facial Motion. *Proceeding of Workshop on Cues in Communication*.

Stork, D. G. & Hennecke, M. E. (Eds.). (1996). *Speechreading by Humans and Machines: Models, Systems and Applications*. Springer.

Tao, H. (1998). Non-rigid motion modeling and analysis in video sequence for realistic facial animation. Ph.D. thesis. University of Illinois at Urbana-Champaign.

Tu, J. et al. (2003). Coding face at very low bit rate via visual face tracking. *Proceeding of Picture Coding Symposium*, 301-304.

Wachtman, G. S. et al. (2001). Automated tracking of facial features in facial neuromotor disorders. *Plastic and Reconstructive Surgery*, 107, 1124-1133.

Waters, K. (1987). A muscle model for animating three-dimensional facial expression. *Computer Graphics*, 22(4), 17-24.

Waters, K. & Levergood, T. M. (1993). DECface, an automatic lip-synchronization algorithm for synthetic faces. Cambridge Research Lab Technical Report, CRL 93-4.

Chapter XI

Synthesis and Analysis Techniques for the Human Body:

R&D Projects

Nikos Karatzoulis
Systema Technologies SA, Greece

Costas T. Davarakis
Systema Technologies SA, Greece

Dimitrios Tzovaras
Informatics and Telematics Institute, Greece

Abstract

This chapter presents a number of promising applications and provides an overview of recent developments and techniques in the area of analysis and synthesis techniques for the human body. The ability to model and to recognize humans and their activities by vision is key for a machine to interact intelligently and effortlessly with a human inhabited environment. The chapter analyzes the current techniques and technologies available for hand and body modeling and animation and presents recent results of *synthesis and analysis techniques for the human body reported by R&D projects worldwide. Technical details are provided for each R&D project and the results are discussed and evaluated.*

Introduction

Humans are the most commonly seen moving objects in one's daily life. The ability to model and to recognize humans and their activities by vision is key for a machine to interact intelligently and effortlessly with a human inhabited environment. Because of many potentially important applications, examining human body behavior is currently one of the most active application domains in computer vision. This survey identifies a number of promising applications and provides an overview of recent developments in this domain (Hillis, 2002).

Hand and body modeling and animation is still an open issue in the computer vision area. Various approaches to estimate hand gestures and body posture or motion from video images have been previously proposed (Rehg & Kanade, 1994; Lien & Huang, 1998; Zaharia, Preda & Preteux, 1999). Most of these techniques rely on 2-D or 3-D models (Saito, Watanabe & Ozawa, 1999; Tian, Kanade & Cohn, 2000; Gavrila & Davies, 1996; Wren, Azarbayejani, Darell & Pentland, 1997) to compactly describe the degrees of freedom of hand and body motion that has to be estimated. Most techniques use as input an intensity/color image provided by a camera and rely on the detection of skin color to detect useful features and to identify each body part in the image (Wren, Azarbayejani, Darell & Pentland, 1997). In addition, the issue of hand and body modeling and animation has been addressed by the Synthetic/Natural Hybrid Coding (SNHC) subgroup of the MPEG-4 standardization group to be described in more detail in the following.

In Sullivan & Carlsson (2002), view-based activity recognition serves as an input to a human body location tracker with the ultimate goal of 3D reanimation. The authors demonstrate that specific human actions can be detected from single frame postures in a video sequence. By recognizing the image of a person's posture as corresponding to a particular key frame from a set of stored key frames, it is possible to map body locations from the key frames to actual frames using a shape-matching algorithm. The algorithm is based on qualitative similarity that computes point-to-point correspondence between shapes, together with information about appearance.

In Sidenbladh, Black & Sigal (2002), a probabilistic approach is proposed to address the problem of 3D human motion modeling for synthesis and tracking. High dimensionality and non-linearity of human body movement modeling is avoided by representing the posterior distribution non-parametrically. Authors Bobick & Davis (2001) introduced a real-time human activity recognition method, which was tested on aerobic exercises. This method is based on a two-component image representation of motion, the Motion Energy Image, MEI, a binary image, which displays where motion has occurred during the movement of the person, and the Motion History Image, MHI, a scalar image, which indicates the temporal history of motion. MEI and MHI temporal templates are then matched to store instances of views of known actions.

Recently, in Sminchisescu & Triggs (2001), 3D human motion tracking from monocular image sequences is achieved by fitting a 3D human body model, consisting of tampered superellipsoids, on image features (edges and motion attributes) by means of an iterative cost function optimization scheme. Also, Plänkers & Fua (2001) present a framework that retains an articulated structure represented by sticks, but replace the simple geometric primitives by soft objects. This results in a realistic model where body parts such as the chest, abdomen or biceps muscles are well modeled.

The main objective of the chapter is to present and analyze the results of synthesis and analysis techniques for the human body reported by R&D projects worldwide. Human body synthesis and analysis is a very important research area with a large number of industrial applications. The examined technological area has produced impressive research results, which, in many cases, have emerged as successful consumer applications, especially in the media and film-making markets. The annual SIGGRAPH Conference is an excellent focal point to monitor scientific results and their use in several pilot applications. The elimination of hidden lines in wire-frame renderings, texture mapping, ray-traced images, animation and expression methodologies are only a few milestones during recent years in the quest to capture reality. The chapter aims to demonstrate the incorporation of recent and innovative techniques in human body modeling, animation and transmission in specific applications developed in R&D projects worldwide.

The chapter is organized as follows: the next section provides an overview of the fundamental standards, either established or emerging, which enable the design and development of interoperable, expandable, reusable and cost-effective modeling and animation applications. In the section following, a brief presentation of on-going and state-of-the-art R&D projects in the area of human (or human parts) analysis and synthesis is presented. This section focuses on developed or “under development” projects, mainly dealing with transferring research results to real life applications. The fourth section deals with a detailed presentation of four recently started European R&D projects that constitute major applications of the analysis/synthesis technology, developed with the author’s contribution. These sets of applications utilize technologies for 3D reconstruction of the human body and animation using image sequence processing and graphical modeling. In most cases, reported reconstruction accuracy is being pursued to map the analysis of the real person with the virtual human (humanoid) in terms of anthropometrical characteristics. The latter applications include:

a) The reconstruction of the music teacher training when he/she demonstrates typical playing methodologies for various musical instruments.

b) The use of human body motion estimation and tracking techniques in the post-production industry incorporating immersive techniques.

c) The use of human body augmentation for the development of virtual mirrors for novel e-commerce applications.

d) The use of 3D humanoids in training ergonomics.

Finally, conclusions are drawn in the final section.

Standards

The main tool introduced for the description of 3D "worlds" is the Virtual Reality Modeling Language (VRML). Technically speaking, VRML is neither virtual reality, nor a modeling language. Virtual reality typically implies an immersive 3D experience (such as the one provided by a head-mounted display) and various 3D input devices (such as digital gloves). VRML neither requires, nor preludes immersion. Furthermore, a true modeling language would contain much richer geometric modeling primitives and mechanisms. VRML provides a bare minimum of geometric modeling features and contains numerous features far beyond the scope of a modeling language (Carrey & Bell, 1997). VRML was designed to create a more "friendly" environment for the World Wide Web. It provides the technology that integrates three dimensions, two dimensions, text and multimedia into a coherent model. When these media types are combined with scripting languages and Internet capabilities, an entirely new genre of interactive applications becomes possible (Carrey & Bell, 1997).

X3D (X3D Task Group) is the next-generation open standard for 3D on the web. It is the result of several years of development by the Web 3D Consortium's X3D Task Group and the recently-formed Browser Working group. The needs that the standard meets are:

- Compatibility with existing VRML content, browsers, and tools.
- Extension mechanism to permit the introduction of new features, a quick review of advancements, and the formal adoption of these extensions into the specification.
- Small, simple "core" profile for widest-possible adoption of X3D support, both importing and exporting.
- Larger, full-VRML profile to support existing rich content.
- Support for different encoding procedures, including XML for tight integration with Web technologies and tools.
- Architecture and process to advance the specifications and technology rapidly.

X3D addresses the limitations of VRML. It is fully specified, so content will be fully compatible. It is extensible, which means X3D can be used to make a small, efficient 3D animation player, or can be used to support the latest streaming or rendering extensions. It supports multiple encoding and various APIs, so it can easily be integrated with Web browsers through XML, or with other applications. In addition to close ties with XML, X3D is the technology behind MPEG-4's 3D support.

The H-Anim (Humanoid Animation Working Group) model is the most commonly used approach to represent human beings. H-Anim is a set of specifications for description of human animation, based on body segments and connections. According to the H-Anim standard, the human body consists of a number of segments (such as the forearm, hand and foot) which are connected to each other by joints (such as the elbow, wrist and ankle). An H-Anim file contains a set of Joint nodes that are arranged to form a hierarchy. Each Joint node can contain other joint nodes, and may also contain a segment node, which describes the geometry of the body part associated with that joint. Each segment is a normal VRML transform node describing the 3D shape of the body part (the set of points that constitutes a 3D surface). The Segments can also have a number of site nodes, which define locations relative to the segment. The Sites can be used for attaching clothing and jewelry, and work as end-effectors for inverse kinematics applications. They can also be used to define eye points and viewpoint locations. A Segment node may contain a number of displacer nodes that specify which vertices within the segment correspond to a particular feature or configuration of vertices. H-Anim will be used to describe the gestures.

MPEG-4 (MPEG-4 standard) is a toolkit for developing networked multimedia applications based on any combination of audio, video, 2D, and 3D content. Specifically, the MPEG-4 standard was designed for delivering both static and interactive multimedia content to any platform over any network. Based on the Virtual Reality Modeling Language (VRML) standard developed by the Web3D Consortium, MPEG-4 has been under development since 1993 and today is ready for use. The first generation of MPEG-4 content servers and authoring tools are now available. Advances in the MPEG-4 standard still continue, particularly in the area of 3D data processing, offering a unique opportunity to generate new revenue streams by way of MPEG-4 and related MPEG activities standards. The Animation Framework eXtension (AFX) (MPEG-4 AFX), for example, is a joint Web3D-MPEG effort that will define new 3D capabilities for the next version of the MPEG-4 standard. Similarly, the MPEG group has recently initiated an effort to develop standards for Multi-User capabilities in MPEG-4 (MPEG-4, requirements for Multi-user worlds).

The issue of hand and body modeling and animation has been addressed by the Synthetic/Natural Hybrid Coding (SNHC) subgroup of the MPEG-4 standardization group. More specifically, 296 Body Animation Parameters (BAPs) are defined by MPEG-4 SNHC to describe almost any possible body posture, 28 of which describe movements of the arm and hand. Most BAPs represent angles of rotation around body joints. Due to the fact that the number of parameters is very large, accurate estimation of these parameters for luminance or color images is a very difficult task. However, if depth images from a calibrated camera system are available, this problem is significantly simplified.

MPEG-4 originally focused on video and FBA (Face and Body Animation) coding. The MPEG-4 FBA framework is limited to human-like virtual character animation. Recently, the FBA specifications have been extended to the so-called Bone-Based Animation (BBA) (Sévenier, 2002) specifications in order to animate any articulated virtual character (Preda & Preteux, 2002).

Relative R&D Projects Worldwide

Computer vision-based techniques are not mature enough to be used for industrial applications and, thus, semi-automatic systems are usually adopted. Traditional systems have evolved from the 90s on the analysis of applied human motion (analysis of sports performance, choreographic movements, medicine and health, robotics, etc.) towards current semi-automatic systems based on the implantation of optical markers, which are afterwards digitalized by stereoscopic images.

Nowadays, the technology of human motion capture and human body synthesis has achieved a reasonable degree of maturity. This can be deduced from the fact that, currently, there exist several commercial systems that permit the capture/

Table 1. Comparison of current motion capture systems (IST HUMODAN, 2001).

Systems	Advantages	Disadvantages
Optical Systems	▪ Optical data are extremely accurate in most cases. ▪ A larger number of markers can be used. ▪ It is easy to change marker configurations. ▪ It is possible to obtain approximations to internal skeletons by using groups of markers. ▪ Performers are not constrained by cables. ▪ Optical systems allow for a larger performance area than most other systems. ▪ Optical systems have a higher frequency of capture, resulting in more samples per second.	▪ Optical data requires extensive post-processing. ▪ The hardware is expensive, costing between $100,000 and $250,000. ▪ Optical systems cannot capture motions when markers are occluded for a long period of time. ▪ Capture must be carried out in a controlled environment, away from yellow light and reflective noise.
Magnetic Trackers	▪ Real-time data output can provide immediate feedback. ▪ Position and orientation data are available without post-processing. ▪ Less expensive than optical systems, costing between $5,000 and $150,000. ▪ The sensors are never occluded. ▪ It is possible to capture multiple performers interacting simultaneously with multiple set-ups.	▪ The tracker's sensitivity to metal can result in irregular output. ▪ Performers are constrained by cables in most cases. ▪ Magnetic trackers have a lower sampling rate than some optical systems. ▪ The capture area is smaller than is possible with optical systems. ▪ It is difficult to change marker configurations.
Electro-mechanical Body Suits	▪ The range of capture can be very large. ▪ Electromechanical suits are less expensive than optical and magnetic systems. ▪ The suit is portable. ▪ Real-time data collection is possible. ▪ Data is inexpensive to capture. ▪ The sensors are never occluded. ▪ It is possible to capture multiple performers simultaneously.	▪ Low sampling rate. ▪ They are obtrusive due to the amount of hardware. ▪ The systems apply constraints on human joints. ▪ Configuration of sensors is fixed. ▪ Most systems do not calculate global translations without a magnetic sensor.

recognition of human motion and its reconstruction and analysis for different applications. Nevertheless, the current limitations of these tools limit the range of application of this technology to off-line analysis.

The principal technologies used today are optical, electromagnetic, and electro-mechanical human tracking systems. Table 1 summarizes the advantages and disadvantages of the different human tracking systems.

Therefore, there is a lot of future in human motion tracking and recognition systems that are based on optical devices and do not include any wearable markers. By this method, the person can move freely around an indoor environment. Of course, this kind of system has some minimal requirements and the algorithms must be very robust to recover the complete motion parameters accounting for illumination changes and occlusions.

Moreover, there is also a lot of work that remains to be seen in the area of subtly introducing 3D technologies in areas of real scientific needs, as the ones previously mentioned. In the following, we briefly present recent or on-going projects focusing on these issues.

In the following, an overview of R&D projects dealing with human body modeling and animation is presented. The R&D projects presented in this chapter were selected to be representative and also to include successfully completed projects, as well as on-going projects focusing on innovative ideas. All projects deal with human body analysis/synthesis techniques. However, their main focus varies depending on the application. For this reason, the projects are categorized according to their three main areas of focus:

- Model/motion acquisition and applications
- Human body animation and applications
- Human body analysis/synthesis techniques optimized for transmission

These three fields, i.e., modeling, animation and transmission, represent the main research areas in human body analysis/synthesis and are also the main focus of the majority of R&D projects which are, of course, more application-oriented. The images used in the following section in order to illustrate the concept and the results of the presented projects are copyrighted by the corresponding projects.

Model/Motion Acquisition and Applications

VAKHUM

VAKHUM: Virtual Animation of the Kinematics of the HUMan for Industrial, Educational and Research Purposes (IST VAKHUM) aims to develop special services to improve the working environment of various fields that use computer models of human joints. These fields need high-quality data to perform their tasks correctly. These tasks are, for example: modeling of human joints, prosthesis design, car-crash simulation, medical education, and biomedical research. All of these tasks use anatomical and kinematics data, but they all encounter the same problem: no data reflecting the high percentage of morphological variations in the human species are easily available. Frequently, only normalized models are produced. Hence, the real relationships between the morphology and the kinematics of a specific subject cannot be foreseen with high accuracy. VAKHUM's main goal is to develop a database to allow interactive access of a broad range of data of a type not currently available, and to use this to create tutorials on functional anatomy. The data will be made available to industry, education and research. A source of high-quality data of both morphological and kinematics models of human joints will be created. The applied techniques will allow data to be obtained that is of potential interest in related fields across industry, medical education and research.

VAKHUM Technical Approach

Morphological data of human bones are initially collected from medical imaging procedures, mainly by computerised tomodensitometry (CT-Scan). The latter allows the construction of very accurate 3D bone models (Figure 1).

Several kinds of data will be available from the VAKHUM database. Not only raw data, but also surface and finite-element models are included. Surface models are useful for3D animation and/or education, while finite elements meshes are used to simulate the deformation and the mechanical stresses induced within living tissues by different motor tasks. They are essential in research, but also in clinical applications, such as the evaluation of the risk of bone fracture, or the planning of complex musculo-skeletal surgery. Finite elements simulations are also useful to teach musculo-skeletal biomechanics.

Figure 1. 3D bone models of the iliac bone. Left: Surface models using tiling techniques; Right: Finite elements model.

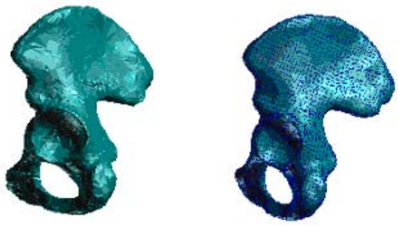
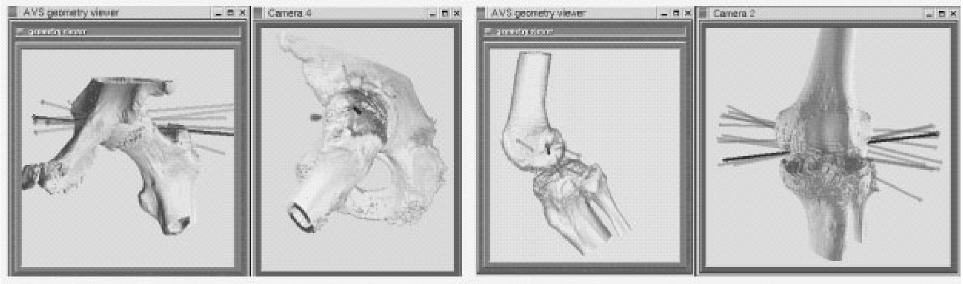


Figure 2. Joint kinematics. Left: Hip joint during a motion of flexion; Right: Knee joint flexion. Helical axes of motion are displayed, as well.

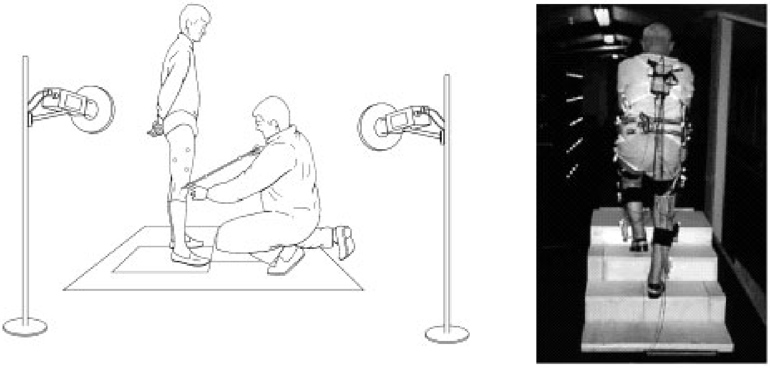


Kinematics is the study of motion. As part of the VAKHUM project, the motion of the human lower limb will be studied during several normal activities (walking, running, stair climbing). Several techniques can be used to study a motion, each of them having its own advantages and disadvantages. This data, associated with medical imaging, can bring forth new information on human kinematics (Figure 2). Unfortunately, electrogoniometry is difficult to use to study full-limb motion.

Other systems like motion-capture-devices using stereophotogrammetry (e.g., video cameras) allow us to study the relative angular displacement of the joints of a particular limb by tracking skin markers attached to a volunteer or patient during some activities (Figure 3).

VAKHUM deals with combining electrogoniometry and stereophotogrammetry to animate 3D models collected from medical imaging. This technique allows not only a combination of different data sources, but also a comparison of results obtained from different protocols, which currently poses an accuracy problem in

Figure 3. Left: Anatomical calibration of skin markers for gait analysis; Right: Kinematics analysis of stair climbing.



biomechanics due to a lack in standardisation. Furthermore, the 3D models produced by VAKHUM will be fully documented and established according the available guidelines from the International Society of Biomechanics (ISB). This should be a guarantee that the VAKHUM data will be widely accepted.

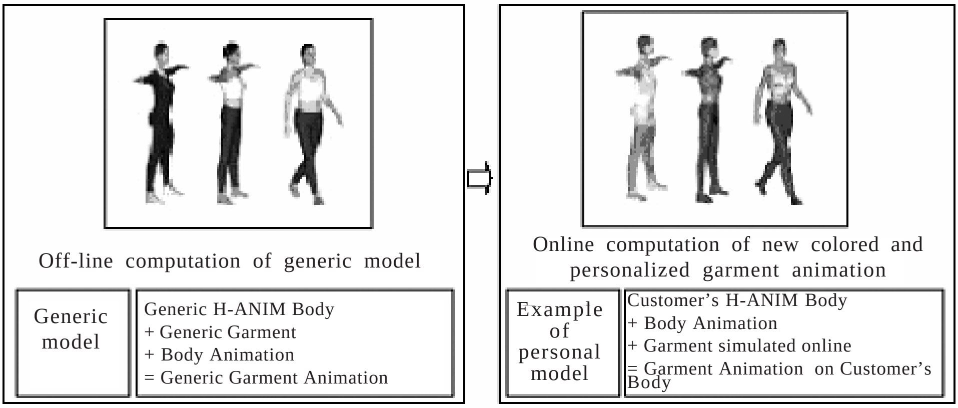
e-Tailor

e-Tailor: Integration of 3D Body Measurement, Advanced CAD, and E-Commerce Technologies in the European Fashion Industry (Virtual Retailing of Made-to-Measure Garments, European Sizing Information Infrastructure) project (IST e-Tailor) aims to establish an innovative paradigm for virtual retailing services of customized clothing by tackling related problems of different sizing systems, fitting problems, high cost, data privacy and lack of interfaces. The main goal of e-Tailor is to develop an advanced customized clothing infrastructure, enabling the customers to visualize themselves wearing clothes on offer at e-kiosks and Internet shops.

e-Tailor Technical Approach

The method adopted for both body presentation and animation for online web applications is to use generic models. The main advantage of using generic models is that one can give necessary information in a pre-processing stage and then online calculations can be performed quickly. This also provides animation information automatically. The basic idea is to consider an avatar or virtual model

Figure 4. From generic to personalized body and garment simulation and animation.


Off-line computation of generic model
Generic model
Generic H-ANIM Body
+ Generic Garment
+ Body Animation
= Generic Garment Animation
Online computation of new colored and personalized garment animation
Example of personal model
Customer's H-ANIM Body
+ Body Animation
+ Garment simulated online
= Garment Animation on Customer's Body

as a database of combined set of data, including 3D shape and the structure on how to animate it.

The human modeling approach starts from default virtual human generic models, including shape and animation structures, and the generic shape is transformed into a new virtual model, according to the scanned body information (Figure 4) (Katsounis, Thalman & Rodrian, 2002). For the generic body animation, a walking motion obtained from a VICON TM motion captured system was applied (Vicon, 2001). Six cameras with a simple rate of 120Hz are used along 25 mm markers.

Human Body Animation and Applications

Intelligent conversational avatar

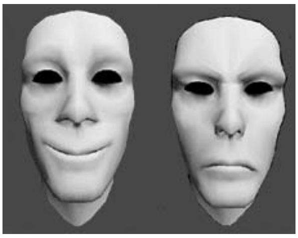
This project is being developed by the **HITLab** at MIT. The purpose of this project is to develop an Expert System and Natural Language Parsing module to parse emotive expressions from textual input. The information will be then used to set the graphical appearance of avatars in order to reduce the need to switch between messages.

Technical Approach

The Expert System parses the text to get the emotions the user desires to portray, taking into account cues present in text, such as: types of words used, contextual information, length of phrases typed, use of emotions, etc.

An agent entity is causing the display of the emotions of the person who is typing his text input. The agent is also propagating these emotions to a specific recipient. The Expert System is used to characterize the agent and his/her emotional states. Facts about emotional effects must be specified in terms of simple data structures or unambiguous natural language (NL) statements (e.g., if one is irritated, then further irritation can make the person angry) (Figure 5).

Figure 5. Emotional states (happy-unhappy).



To implement the emotional model, an unambiguous set of emotional categories with a certain degree associated has been used, and a transition function for perturbing the emotional states. For this function, the behavior of an exponential model has been used (e^{-j}). To obtain continuous changes, Fuzzy Logic was applied. The Expert System, itself, is implemented in CLIPS, a language developed by NASA. The 3D face model used was composed of polygons, which can be rendered with a skin-like surface material. A more detailed presentation of the project is provided in Intelligent Conversational Avatar.

FashionMe

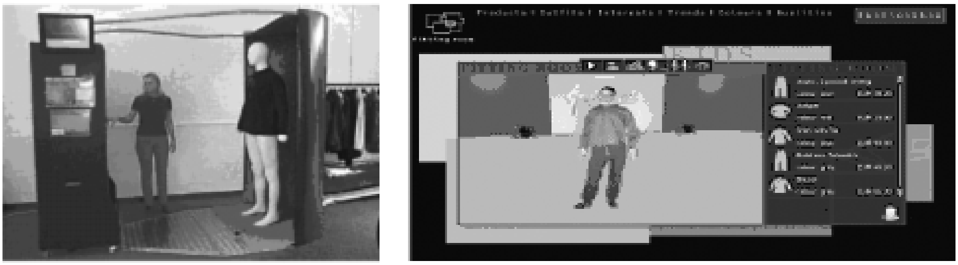
FashionMe: Fashion Shopping with Individualized Avatars project (IST FASHIONME) offers a breakthrough for the European fashion industry. Customers will be able to try on custom-tailored clothing "virtually" using avatars over the Internet (Figure 6). The FashionMe platform aims to provide the European fashion industry with an Internet infrastructure for avatars, including:

- Photorealistic avatars in fitting rooms.
- A website for avatars.
- A service for personalized avatar clothing.
- Selling avatar related products, such as clothing.
- An e-Commerce platform for purchasing clothes through the Internet.

FashionMe Technical Approach

In order to realize 3D garment visualization on an individualized avatar or to use a virtual catwalk displayed with regular web technologies, several components

Figure 6. The scanning and clothing adaptation systems used in FashionMe.



are needed. For example, an avatar must have certain prerequisites in order to meet these requirements. The three essential components are:

1. A 3D mesh representing the shape of the body,
2. Texture mapping on this mesh to provide the avatar with a realistic appearance,
3. An H-Anim model taken as a basis for the definition of motions. This also comprises the definition of joints with certain degrees of freedom, as well as adjustable length relations.

The term eGarments denotes digital, 3D models of real pieces of clothing. Most online product catalogs only consist of 2D pictures. The easiest way to produce eGarments is to generate data from a CAD program, which is used for the design and the cutting construction of the clothing. State-of-the-art of cutting construction is, however, in most cases only two-dimensional. In order to generate a 3D volume model of the garment, these faces must be sewed together virtually and then transferred into a 3D grid model. eGarments are produced in FashionMe in a multistage process. A real dummy is equipped with the specific garment and is then scanned in 3D. The basis for this method is that the naked dummy was scanned in the first instance so that the system knows the gauges of the dummy. In a second step, the dummy is scanned wearing the garment. By subtracting the known rough model from the dressed model, the necessary geometric data is computed. The eGarment consists of the offset of the dummy, the garment's surface, and the graphical information, which maps the surface and is used as a texture.

The focus of the scanning technology used in FashionMe (provided by the AvatarMe company) basically lies on the simple and fast generation of Internet-enabled, personalized avatars that have a realistic appearance, and not so much on an exact rendering of the actual gauges of a person. The scanner is accommodated in a booth that can also be set up in public places (Figure 6). The scanning process comprises digitally photographing the model from four different perspectives. Based on these digital views, the avatar is computed by means of existing rough models. In order to be able to move the model realistically, the avatar is assigned a skeleton. The assignment of all necessary points in the avatar for the individual joints is realized using predefined avatars. After size, weight, age and sex of the scanned person have been recorded, the most appropriate avatar is automatically selected from about 60 different pre-defined body models. Then, the avatar is personalized using texture mapping. This procedure provides a first skeleton, which can be further refined manually, in order to equip fingers with knuckles, for example.

The avatar produced using this procedure does not really comply with the exact body measures of the respective person. The quality, however, is sufficient to walk on the virtual catwalk and get a realistic impression of how the garment would actually look on one's body. The mesh of such an avatar consists of about 3.800 nodes. This figure can be considered a compromise between the desired accuracy and the minimal amount of data. The procedure is optimized for real-time web animation.

Moving the virtual skeleton of the user is referred in FashionMe as "the virtual catwalk." Virtual catwalk animates the avatar according to defined, sex-specific motion sequences. The motion sequences contain detailed, time-dependent descriptions of motions for every joint. These motion sequences are either generated artificially by means of a 3D editor, or digitized by motion capturing real movement of the users. Motion capturing (e.g., the technology of Vicon) (Vicon, 2001) records and digitizes even complex movements of a human model with all their irregularities in order to provide the highest possible realistic impression.

INTERFACE

The objective of the *INTERFACE: MultimodAl analysis/synthesis systEm for human INTERaction to virtual and augmented environments* project (IST INTERFACE) is to define new models and implement advanced tools for audio-video analysis, synthesis and representation, in order to provide essential technologies for the implementation of large-scale virtual and augmented environments.

The work is oriented to make man-machine interaction as natural as possible, based on everyday human communication by speech, facial expressions, and body gestures. Man-to-machine action is based on a coherent analysis of audio-video channels to perform either low-level tasks, or high-level interpretation and data fusion, speech emotion understanding or facial expression classification. Machine-to-man action, on the other hand, will be based on human-like audio-video feedback simulating a "person in the machine." A common software platform will be developed by the project for the creation of Internet-based applications. A case study application of a virtual salesman will be developed, demonstrated, and evaluated.

INTERFACE Technical Approach

Research within INTERFACE focused on the design and development of the following components: a) emotional speech analysis, b) emotional video analysis,

c) emotional speech synthesis, d) emotional facial animation and e) emotional body animation.

Emotional speech analysis: The aim of this task is to define a limited set of parameters that would characterize the broad emotion classes. To achieve that, a number of tools for low and high-level feature extraction and analysis were developed. Calculation of low-level features included phoneme segmentation, pitch extraction, energy measure and calculation of pitch and energy derivatives. The high-level features were grouped into five groups: The first group consists of high-level features that are extracted from pitch. The second group consists of features that are extracted from energy. The third group consists of features that are calculated from phoneme segmentation. The fourth group consists of features that are calculated from features calculated from pitch derivative. The fifth group consists of high-level features that are calculated from energy derivatives. A set of 26 different high-level features were analyzed, in order to define a set of high-level features that differentiated various emotions the most. All low and high-level features that did not show any capabilities to differentiate the emotions were excluded from the set.

Emotional video analysis: The research that is carried out within the project is focused on expression recognition techniques that are consistent with the MPEG-4 standardized parameters for facial definition and animation, FDP and FAP. To complement the expression recognition based on low-level parameters, additional techniques that extract the expression from the video sequence have also been developed. These techniques also need the location of the face and a rough estimation of the main feature points of the face. For this aim, there is an activity oriented to Low Level Facial Parameter extraction and another activity aimed at High Level Facial Parameter extraction or expression recognition. Another group of tools has been developed for the extraction of cues important for dialogue systems, such as gaze, specific movements like nods and shakes for approval or disapproval, attention or lack of attention.

Emotional speech synthesis: A proprietary tool for Text-To-Speech synthesis has been made compliant to MPEG-4 for its synchronization with animated faces, and addition of a new functionality needed for emotional synthesis: energy modifications. In particular, the modification was oriented to improve the phoneme&synchronization information that is necessary to interface the Facial Animation module.

Emotional facial animation: The first activity carried out was related to the enhancement of the facial expressions synthesis. With the help of an artist, a database of high-level expressions has been developed by mixing low-level expressions. A second activity was related to the development of a facial animation engine based on *Facial Animation Tables* (FAT), in order to accurately reproduce facial animations. For each *Facial Animation Parameter* (FAP), we need the FAT to control the displacements of the vertices corresponding to the *Facial Description Parameter* (FDP). The FAT can be defined with an animation engine or by a designer. The advantages of the FAT-based animation are that the expected deformations due to the animation are exactly mastered, the FAT can be defined by a designer and, thus, the animation can be balanced between realistic and cartoon-like behavior. The drawback is that each FAT is unique to a given face and it is required to build one FAT for each face (even if the topology of the mesh is the same), whereas an MPEG-4 animation engine can work with any face mesh and FDP data.

Emotional body animation: The server calculates the emotion from all man-to-machine action (text, speech and facial expression) and associates to this emotion a gesture that is translated into the corresponding Body Animation Parameters (BAP) file that is finally sent through Internet.

STAR

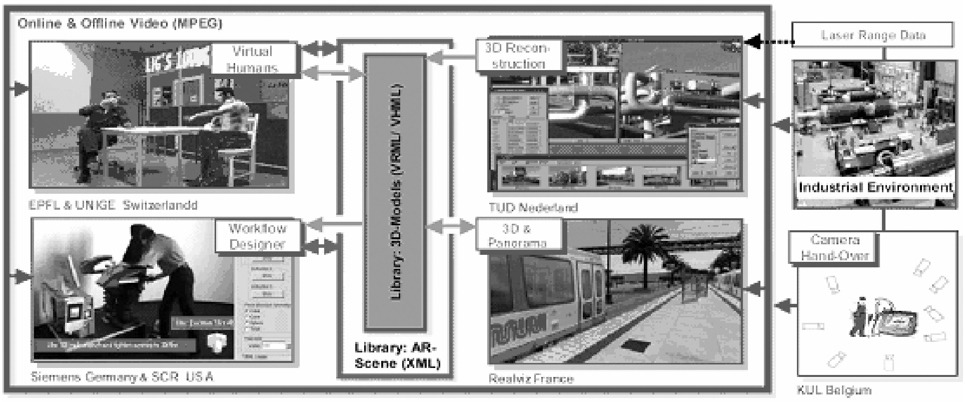
The (IST STAR) project *Service and Training through Augmented Reality (STAR)* is a collaboration between research institutes and companies from Europe and the USA. STAR's focus is the development of Mixed Reality techniques for training, documentation, and planning purposes.

To achieve these, the following components were developed (Figure 7):

- Automated 3D reconstruction of industrial installations,
- Automated view selection and camera handover,
- Manipulation of mixed objects by virtual humans,
- Approaches with different tracking and recognition of objects from video,
- User interface to create own Augmented Reality applications.

Using these components, it is possible to realize the following scenario: A worker is on-site preparing to perform a maintenance task. He is equipped with a laptop

Figure 7. STAR project components.



with a camera and a wireless connection to the local computer network. The camera captures the workspace in which the worker is operating and the video from the camera is transmitted over the network to an expert. The 3D position of the camera with respect to the scene is tracked automatically by the system, using feature-matching techniques. The expert can augment the video with a variety of relevant information: text, 3D, etc., and send the augmented view back to the worker over the network. He will then use this information to decide what steps to perform next.

STAR Technical Approach

The main component of STAR is the "virtual human," a set of connected software components that allows manipulation of mixed objects by virtual humans. The operator is able to add autonomous virtual humans into augmented environments. Both the user and autonomous virtual humans are able to move, animate, and deform objects, whether real or virtual. This involves replacing real objects with virtual ones when they are being manipulated.

The STAR project uses and investigates different edge-based and feature-based tracking algorithms in different components. Tracking information is then used to control the virtual human and manipulate the virtual objects.

BLUE-C Project

The *blue-c* (BLUE-C Project) is a joint research project between several institutes at ETH Zurich. The goal is to build a collaborative, immersive virtual environment, which will eventually integrate real humans captured by a set of video cameras. The project started in April 2000 and its first phase is expected to be completed by Spring 2003.

Mastering rapidly changing computing and communication resources is an essential key to personal and professional success in a global information society. The main challenge consists not only in accessing data, but rather in extracting relevant information and combining it into new structures. The efficient and collaborative deployment of applications becomes increasingly important as we find more complex and interactive tools at our disposal. Today's technology enables information exchange and simple communication. However, it often fails in the promising field of computer enhanced collaboration in virtual reality environments. Some improvements were made by coming-of-age virtual reality systems that offer a variety of instrumental tools for stand-alone visual analysis. Nevertheless, the crucial interaction between humans and virtual objects is mostly neglected. Therefore, successful models of truly computer supported collaborative work are still rare.

The blue-c project aims at investigating a new generation of virtual design, modeling, and collaboration environments. 3D human representations are integrated in real-time into networked virtual environments. The use of large screens and cutting-edge projection technology creates the impression of total immersion. Thus, unprecedented interaction and collaboration techniques among humans and virtual models will become feasible.

Blue-c Technical Approach

The blue-c features are:

- Full immersion of the participants in a virtual world,
- 3D rendered human inlays, supporting motion and speech in real-time,
- New interaction metaphors between humans and simulated artifacts of functional and/or behavioral nature.

The blue-c system foresees the simultaneous acquisition of live video streams and the projection of virtual reality scenes. Color representations with depth information of the users are generated using real-time image analysis. The computer-generated graphics will be projected onto wall-sized screens surrounding the user, allowing him to completely enter the virtual world. Multiple blue-c portals, connected by high-speed networks, will allow remotely located users to meet, communicate, and collaborate in the same virtual space. The blue-c system includes:

- A fully immersive 3D stereo projection theatre,
- Real-time acquisition of multiple video streams,
- 3D human inlays reconstructed from video images,
- Voice and spatial sound rendering,
- Distributed computing architectures for real-time image processing and rendering,
- A flexible communication layer adapting to network performance,
- Scalable hardware and software architecture for both fixed and mobile installations,
- An advanced application programming interface.

Applications of blue-c include architectural design, next-generation product development, and computer-aided medicine.

Human Body Analysis/Synthesis Techniques Optimized for Transmission

ATTEST

The *ATTEST: Multimodal Analysis/Synthesis System for Human Interaction to Virtual and Augmented Environments* (IST ATTEST) project will design an open, flexible and modular 3D-TV system, which can be used in a broadcast environment. It will be based on the concept of 2D video and synchronized depth information, assuring full compatibility with digital 2D-TV available today. As early as 1920, TV pioneers dreamed of developing high-definition 3D color TV, as only such would provide the most natural viewing experience. Today, the hurdle of 3D-TV still remains to be taken.

Essential requirements are the backwards compatibility with existing 2D broadcast and flexibility to support a wide range of different 2D and 3D displays. This can be achieved by providing depth as an enhancement layer on top of a regular 2D transmission. ATTEST will address the complete 3D-TV broadcast chain: 3D content generation (novel 3D camera and 2D-to-3D conversion of existing content), 3D video coding (complying with 2D digital broadcast and streaming internet standards), and visualization on novel single and multi-user 3D displays. The concept to provide 3D as synchronized 2D and depth information finally provides the flexibility to allow local customization of the depth experience.

ATTEST Technical Approach

In ATTEST the need for the 3D video content will be satisfied in two different ways:

- A range camera will be converted into a broadcast 3D camera, which requires a redesign of the camera optics and electronics to deliver a full resolution 3D camera, higher depth and pixel resolution;
- As the need for 3D content can only partially be satisfied by newly recorded material, ATTEST will also develop algorithms to convert existing 2D video material into 3D. Both offline (content provider) and online (set-top-box) conversion tools will be provided.

In the introduction period, 2D and 3D-TV sets will co-exist. ATTEST will, therefore, develop coding schemes within the current MPEG-2 broadcast standards that allow transmission of depth information in an enhancement layer, while providing full compatibility with existing 2D decoders. First, perceptual quality will be assessed through a software prototype, later a hardware real-time decoder prototype will be developed.

At present, a suitable glasses-free 3D-TV display that enables free positioning of the viewer is not available. Also, there is no suitable display for single users (3D-TV on PC), or for use in a typical living room environment. ATTEST will develop two 3D displays (single and multiple user) that allow free positioning within an opening angle of 60 degrees. Both are based on head tracking and project the appropriate views into the viewer's eyes.

ATTEST will deliver a 3D-TV application running on a demonstrator platform, with an end-to-end DVB delivery system. The 3D content will either be recorded with the ATTEST 3D camera, or will be converted from 2D video footage using the ATTEST 2D-to-3D conversion tools. ATTEST will build a real-time MPEG-2 base and 3D enhancement layer decoder and demonstrate optimized 3D video rendering on the ATTEST single and multi-user 3D displays.

VISICAST

The goal of *ViSiCAST: Virtual Signing: Capture, Animation, Storage & Transmission* project (IST VISICAST) is to improve the quality of life of Europe's deaf citizens by widening their access to services and facilities enjoyed by the community at large. The project identifies a number of aspects of life where the integration of deaf individuals in society would be improved if sign language communication was available: access to public services, commercial

Figure 8. TESSA, the TExt and Sign Support Assistant.

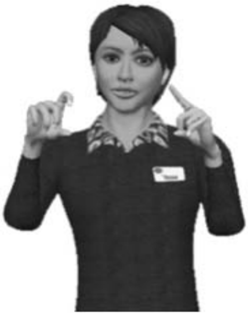
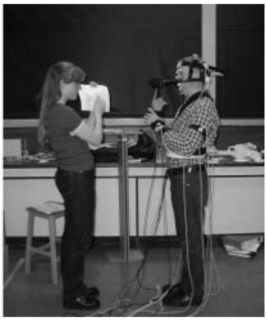


Figure 9. Signs being motion captured.



transactions and entertainment, learning and leisure opportunities, including broadcast and interactive television, e-commerce and the World Wide Web.

The movements of the virtual human are "copies" of those of a native sign language user. Software specially developed for the project captures the signer's hand, mouth and body movements using a variety of electronic sensors (Figures 8 and 9). These movements are then stored and used to animate the avatar when required.

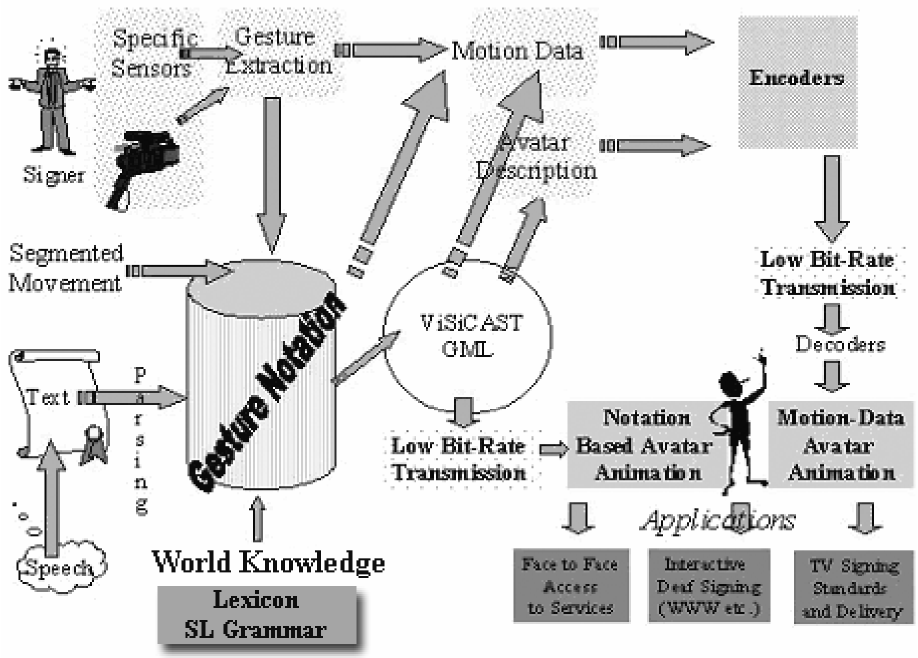
VISICAST Technical Approach

Individual components of shape and movement may be acquired live (motion capture of human signers) or constructed in 3D graphic space using physical modeling tools (Figure 10). Tools based on each system will be developed in parallel, evaluated, and deployed as appropriate. For use with the direct recording of signed sequences, the project will develop a refined suite of advanced Motion Capture Tools, forming a single, coherent, capture, recording and replay system, using robust techniques and equipment capable for use in TV studios and other industrial, non-laboratory settings.

An Internet browser plug-in has been developed, which allows viewing of text as signs. A version is provided free of charge to deaf users. Authoring tools have also developed to allow deaf people to construct their own deaf-signed web sites.

The applications of the virtual human signing system are going to be used in face-to-face transactions, such as post offices, health centers and hospitals, advice services, and shops. Currently, the scenario for these transactions in VISICAST is a post-office. The system allows the counter clerk serving the deaf customer to speak into a microphone and have his or her speech translated into on-screen virtual human signing. To improve the efficiency of the transactional system, it

Figure 10. VISICAST project concept.


Specific Sensors
Gesture Extraction
Motion Data
Encoders
Signer
Avatar Description
Segmented Movement
Low Bit-Rate Transmission
Decoders
ViSiCAST GML
Gesture Notation
Text
Parsing
Low Bit-Rate Transmission
Notation Based Avatar Animation
Motion-Data Avatar Animation
Applications
Speech
World Knowledge
Lexicon
SL Grammar
Face to Face Access to Services
Interactive Deaf Signing (WWW etc.)
TV Signing Standards and Delivery

incorporates available technologies to "read" limited signs from the deaf customer and translate these into text (or speech) that the counter clerk can understand.

Application Oriented Projects – Case Studies

Based on the results of the analysis of human (or human part) movement, a number of 3D synthesis applications have been developed and demonstrated in many research projects. In the following, we will present four such research projects selected in terms of utilizing technology research in cost-effective information society applications:

- Project HUMODAN: *An automatic human model animation environment for augmented reality interaction* is facing the issue of extracting human body motion for the post-production industry and media at large.

- Project IMUTUS: *Interactive Music Tuition System* is using 3-D technologies to reconstruct humanoid music teachers and demonstrate typical methodologies for performing various musical instruments.
- Project SHOPLAB: *The ShopLab-Network for Test and Design of Hybrid Shop Environments based on multi-modal Interface Technology* is introducing the aspect of a virtual mirror augmenting the human body for use in a commercial environment promoting shopping procedures.
- Project MIRTH: *Musculo-skeletal Injury Reduction Tools for Health and safety*, where 3-D technology is being utilized for determining ergonomics in the automotive industry design and training.

HUMODAN: An Automatic Human Model Animation Environment for Augmented Reality Interaction

The objective of the HUMODAN project (IST HUMODAN, 2001) is to design, develop and set up an innovative system for automatic recognition and animation of human motion in controlled environments. The most relevant and distinctive feature of this system with respect to existing technologies is that the individual being recorded will not wear any type of marker or special suit or other types of sensors. This system is expected to be highly useful in a wide range of technological areas, such as TV production, tele-presence, immersive and collaborative interactivity storytelling, medical diagnostic support, tele-operation, education and training.

In the field of multi-part systems, not much research work has been carried out so far in human motion reconstruction. Two noticeable exceptions are the simulation of human body motion in sports activities and the medical diagnosis of damages in the locomotor system, for example, gait analysis. In this case, the real motion of the sportsman has been recorded with video cameras. After adequate processing, the trajectories of several points are applied as inputs to a mechanical model of the human being. An Inverse Dynamics algorithm allows one to calculate the overall motion of the person, as well as the internal forces that are required to produce the motion, energy consumption, ground reactions, etc. Further post-processing of these outputs will estimate specific indices, for example, the performance of the athlete or the level of gait damage of a patient. Investigations on motion patterns are still in course and there are not yet automatic methods implemented.

Real-time dynamic simulation of mechanisms has been a very active research field recently, with applications mainly in robotics and vehicle simulation, considering different degrees of accuracy in the model. In this basis, HUMODAN is developing an innovative system for recognition of human motion based on image processing analysis/synthesis techniques. The system will be enhanced to recognize and analyze other biped and no-biped beings, like for example pet animals, robots, etc. In addition, it will be able to focus only in a part of the body but with high detail, like for example the hands or the face. The fundamental aim is to obtain a 3D model of the person or persons by means of a sequence of images taken from different viewpoints. With this information the aim is to carry out different tasks such as: realistic animation of a person, biomechanical study of sports or dance movements, recognition of a person (face and movements), integration of a virtual humanoid with real characters, interaction in a person and humanoid immersed environment, robot tracking of a person, etc.

To ensure the widest range of applications, the individual recorded will not wear any type of marker or special suit. To this end, biomechanical models will be constructed using a hierarchical and articulated structure in order to establish a correlation between each structural element of the biomechanical model with the analytical characteristics of the images obtained using different views. Innovative shape or part recognition techniques will be applied. The biomechanical model will include a knowledge database to retain high-level information of the motions.

To make the system usable it will also be necessary to develop specific applications and plug-ins to integrate the animation into end users tools such as digital TV production software, animation software and virtual environments like a CAVE. The HUMODAN system is expected to allow the development of new applications including: a) medical applications such as non-invasive diagnose of neurological damages affecting the locomotor system, b) surgery training, anatomy, traumatology, etc., training the use of tools or processes in dangerous environments, virtual laboratories, sports, etc.

HUMODAN technical approach

HUMODAN is an on-going project being mainly developed based on the Avango framework (Tramberend, 2000). Avango is a programming framework for building distributed, interactive VE applications. It uses the C++ programming language to define two categories of object classes. Nodes provide an object-oriented scene-graph API, which allows the representation and rendering of complex geometry. Sensors provide Avango with its interface to the real world and they are used to import external device data into an application.

Avango’s distribution paradigm is based on an object-based representation that matches well with scenegraph-based virtual environment development. All objects in Avango are field-containers. These field-containers represent the object's state information as a collection of fields. This method is similar to the method used in the Inventor ToolkitTM (Strauss, 1993) and in VRML. These field-containers support a generic streaming interface. This allows the field-container and its fields (i.e., the object and its state information) to be written to a stream and then reconstructed from a stream. This forms the basis for object distribution in Avango.

ShopLab: Network for the Test and Design of Hybrid Shop Environments

The objective of the ShopLab project (IST ShopLab) is the development of hybrid shop environments based on multimodal interfaces (and interspaces) which combine the virtues of real world shops with the additional value of digital technologies and services. The alienation of many people to new technologies will be reduced by integrating their needs and experiences in the design process. Therefore, test beds that are accessible to the public will be installed in a model shop and in real shops so that an active participation of users will be guaranteed. Both social and practical user acceptance can be tested in realistic environments. The enormous cultural value of traditional retail shops in different European regions, as well as the different European mentalities, will be regarded in the participatory design of the hybrid shop modules.

The ShopLab project consists of the following elements: hardware interface design, software interface design, shop construction and design, multimedia application design, usability and user acceptance test design and inter-cultural communication design for the development of hybrid shop environments. These environments consist of ShopLab modules such as Interactive Window, Interactive Shelf, Interactive Mirror and Interactive (Entrance) Space. These modules consist of H/W and S/W interface components that make use of multiple interaction modalities and include shop-specific multimedia applications and services.

ShopLab will merge real objects from shop display rooms with virtual elements and vice versa. The resulting hybrid shop environments will support multi-modal interaction and will appeal to multiple human senses (tactile, acoustic, visual, gesture/movement). The whole shop, including products and the shop interior, will be transformed into a "shop-terminal," providing access to additional information and services. Following the tangram metaphor, flexibility and adaptability will be key requirements for the ShopLab systems.

SHOPLAB technical approach

In order to clearly indicate the potential of the project, we will present in detail the functionality of the Virtual Mirror scenario, which is part of the ShopLab toolbox. In the case of small clothing shops, lack of space is often a factor that detracts from the shoppers' enjoyment. When trying different outfits on in a small cubicle, one cannot see how the clothes will look in a natural environment. Certain views of the clothes are difficult to obtain, for example, one cannot easily see how the trousers fit in the back. Moreover, some types of clothes, for example, a snowboarding garment, are particularly difficult to evaluate in the shop environment. The following ShopLab installation overcomes these problems. On one wall of a changing room imagine an ordinary mirror for traditional changing room evaluation, while on an adjacent wall a flat screen panel displays the customer in their new clothes in an appropriate environment (Figure 11). They will be able to see themselves easily from behind or sideways. The displayed artificial environment and lighting conditions make it easier to evaluate how the clothes would appear in a "real world" situation (Figure 12).

The ShopLab platform is mainly being developed using the ARTookit libraries (ARToolkit Library). ARToolkit is an open-source vision-tracking library that enables the easy development of wide range of Augmented Reality applications. The library has been inspired and implemented by Professor Hirokata Kato and Dr. Mark Billinghurst. The 3D human models will be developed based on the H-Anim standard. A portable 3D body scanner will be used for the modeling.

Figure 11. The concept of the Virtual Mirror.

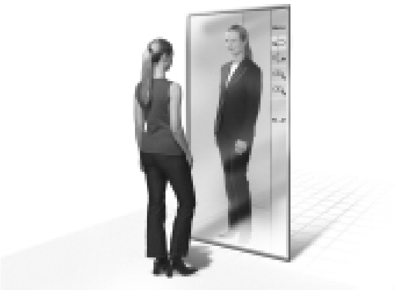
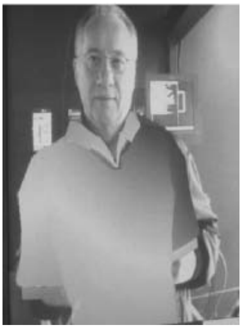


Figure 12. The Virtual Mirror in action.

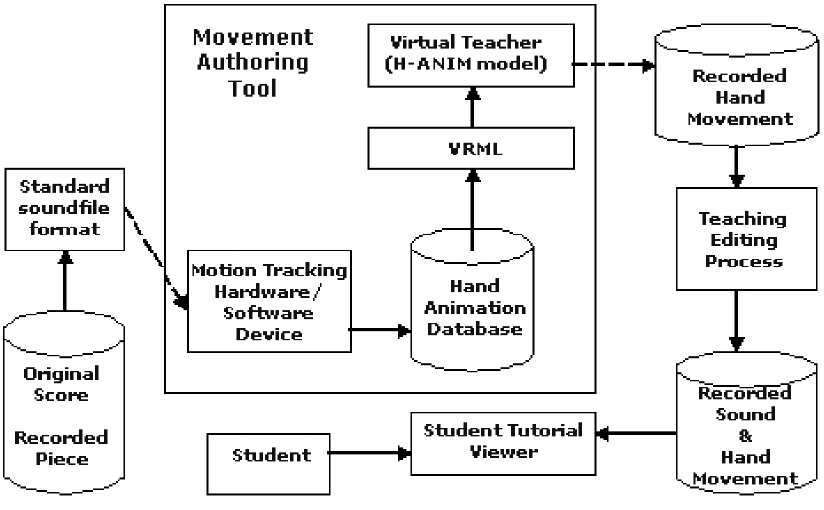


IMUTUS: Interactive Music Tutorial System

The main goal of the IMUTUS project (IST IMUTUS) is to provide an interactive music tuition multimedia system for training users on traditional instruments with no MIDI (Musical Instrument Digital Interface) output. The selected instrument is the recorder. The system will be based on audio/optical recognition, multimedia, virtual reality and audio-to-MIDI transformation technologies.

The most important aspect of IMUTUS in terms of human body analysis and synthesis is the Movement Authoring tool (Figure 13), which is part of the IMUTUS virtual reality module. A very important aspect in learning and practicing with a musical instrument is the fingering technique used. The movement of the hands and the positioning of the instrument between the lipsare only two examples of the kinematics involved in learning to play music. In order to support this learning aspect, the system will provide visual representations of the movement of the fingers of a virtual teacher synchronized with the music played. The teachers will have the opportunity to use a dedicated authoring tool in order to visually record their movements and convert them to 3D representations. The reason for not using simple video is that with 3D representations the learning user will have the possibility to freely navigate within the scenery and, thus, observe the movement of the teacher from different viewpoints, zooming in and out, and emphasizing aspects that he/she considers especially difficult (i.e., the position of two fingers next to each other).

Figure 13. The modules of the author movement tool.



The introduction of virtual reality will allow experimenting with new pedagogical mechanisms for teaching music and, more specifically, for correcting hand positioning. Entirely different from the adoption of simple videotapes, by using the VR it will be possible for the pupil to re-execute specific passages that are not planned in the video, but can be directly produced on the basis of the music score or MIDI. This is a high-level coding for the finger movements. The high-level coding allows the updating of the pupil database via Internet. It would be impossible to send MPEG videos to show the same gestures.

IMUTUS technical approach

Many researchers have worked on 3D finger representation, but their concentration was mainly on gesture recognition aspects, providing no timing or synchronization facilities for the movement of the fingers. The major problem in 3D finger representation and playback synchronization is the correct timing of the virtual fingers.

Concerning the gesture description, using separate controls for each geometry can be useful, but this increases the complexity of the system. The controls can be reduced using inverse kinematics methods by implementing only one control for each finger. For the gesture description subcomponent, a hybrid system with inverse kinematics and H-Anim models is adopted. The 3D finger representation module supports the following functions:

- Read initial position values (read position values for fingers).
- Read final position values (read position values for fingers).
- Get morphing total time (read time for morphing from initial to final state).
- Calculate intermediate position values of the morphing sequence (real-time finger position calculation and interaction with the 3D rendering engine).

Correct representation and modeling of the virtual hands can also be performed using virtual reality gloves (e.g., the CyberGlove of Immersion Technologies) (Immersion Corp., 2002). CyberGlove is a low-profile, lightweight glove with flexible sensors which can measure the position and movement of the fingers and wrist. It is based on a resistive, bend-sensing technology and its sensors are thin and flexible enough to produce almost undetectable resistance to bending. The device can provide accurate measurements for a wide range of hand sizes.

MIRTH: Musculo-skeletical Injury Reduction tools for Health and Safety

In this set of applications, we introduce yet another concept recently introduced in the Automotive Industry, by Daimler-Chrysler in Germany. It is the use of 3D human motion for teaching and benchmarking ergonomics in the workplace. A project running under the European Research and Development Programme "Growth," is called MIRTH and aims at producing a set of tools that perform ergonomic assessments of workplaces. The particular areas of application are computer workstations and office ergonomics, electronic assembly work, and car manufacturing production lines. The benefits of the project will include reducing the present high costs of injuries incurred as a result of work-related musculo-skeletical disorders.

The project will also develop an ergonomics-training tool in two versions, one for expert users (ergonomers, workplace designers) and one for non-expert users (office workers). The tool will be based on the concept of illustrating ergonomics

Figure 14. Different Modes of Illustration in MIRTH.

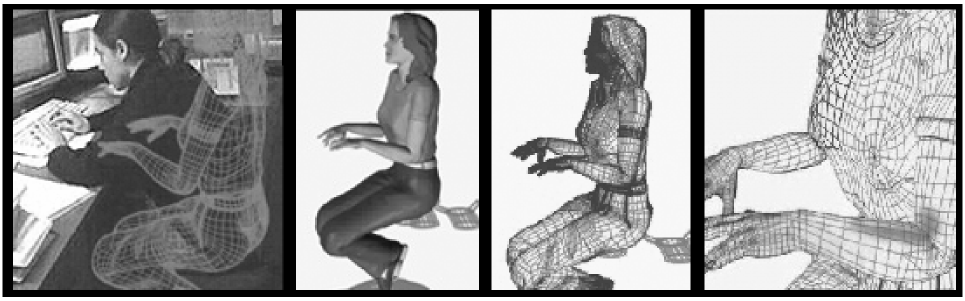
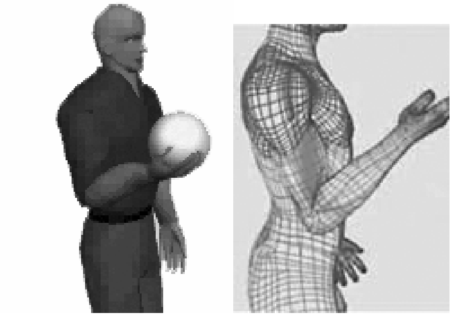


Figure 15. Illustrating the concepts of force and repetition.



concepts (Figure 14) such as posture, force and repetition by 3D animation and video effects (Figure 15).

Conclusions

This chapter presents recent results of synthesis and analysis techniques for the human body reported by R&D projects worldwide. The examined technological area has produced impressive research results, which have emerged as successful consumer applications, especially in the media, industrial and educational markets.

During the last decade, a very large number of articles have been published in the research area of human body modeling. However, computer vision-based human body analysis/synthesis techniques are not mature enough to be used for industrial applications and, thus, semi-automatic systems are usually adopted. Many issues are still open, including unconstrained image segmentation, real-time motion tracking, personalization of human models, modeling of multiple person environments and computationally efficient real-time applications. Each one of these topics represents a stand-alone problem and their solutions are of interest not only to human body modeling research, but also to other research fields.

Reduction of the processing time in order to support real-time applications is one of the major issues in human body modeling and animation. Reduction depends on the performance of the current computer systems (CPU and memory capabilities), which are constantly improving, and the computational complexity of the technical approaches used (matching/tracking algorithms, etc.), which, however, are not expected to significantly lower processing time. Thus, it is expected that in the near future algorithms, that nowadays are computationally prohibitive, will be feasible, giving rise to new human modeling and animation applications.

Realistic human body modeling and animation is considered essential in virtual reality applications, as well as in remote collaboration in virtual environments. Most R&D projects reviewed in this chapter are moving in this direction and their results are considered to be very interesting and promising. Virtual Reality applications today lack real-time human participation. Most applications have been shown as walk-thru types for virtual exploration of spatial data (virtual prototypes, architecture, cultural heritage, etc.) or as user interactive direct manipulation of data (training systems, education systems, etc.). Especially in digital storytelling, applications today are limited to predefined human participation, such as animated 3D cartoons and/or integration of pre-recorded video textures for the representation of humans. A life-like virtual character reconstructed and controlled by the behavior of a human in real-time would be desirable for many types of applications in digital storytelling, especially in the field of supporting performances in theatres with virtual background for human and synthetic actors. In the field of collaborative virtual environments, most VR systems do not support network distributable applications and, therefore, lack the means to represent a remote human team member in a collaborative application scenario. Examples using real-time video textures for remote participants have been demonstrated so far and have shown major problems in collaboration, as the perspective projection for the 3D virtual world and the remote participant (represented by real-time textures) were different.

Many R&D projects are still focusing their research on human body modeling and animation. It is expected that most of the current research problems reported will be solved within the next years and more real-time applications, especially in the multimodal interaction area, will be feasible. However, due to the nature of the human body modeling problem, completely general solutions (working under all circumstances and for all users) are not expected in the very near future.

References

ARToolKit Library. Retrieved from the World Wide Web: http://mtd.fh-hagenberg.at/depot/graphics/artoolkit/.

Blue-C Project. Retrieved from the World Wide Web: http://blue-c.ethz.ch/.

Bobick, A. F. & Davis, J. W. (2001). The recognition of human movement using temporal templates. *IEEE Trans. on PAMI*, 23(3), 257-267.

Carrey, R. & Bell, G. (1997). *The Annotated Vrml 2.0 Reference Manual. 1st edition*. Addison-Wesley.

Gavrila, D. M. & Davis, S. L. (1996). 3-D Model Based Tracking of Humans in Actions: a Multi-view approach. *Proceedings IEEE Conference. on Computer Vision and Pattern Recognition*, San Francisco, CA.

Hillis, D. (2002, July). *The Power to Shape the World*. ACM.

Humanoid Animation WorkingGroup. Retrieved from the World Wide Web: http://h-anim.org/.

Immersion Corp. (2002). Retrieved from the World Wide Web: http://www.immersion.com/, Immersion Corp.

Intelligent Conversational Avatar Project. Retrieved from the World Wide Web: http://www.hitl.washington.edu/research/multimodal/avatar.html.

IST. ATTEST. Multimodal Analysis/Synthesis System for Human Interaction to Virtual and Augmented Environments. Retrieved from the World Wide Web: http://www.extra.research.philips.com/euprojects/attest/.

IST. e-Tailor. Integration of 3D Body Measurement, Advanced CAD, and E-Commerce Technologies in the European Fashion Industry. Retrieved from the World Wide Web: http://www.atc.gr/e-tailor.

IST. FASHIONME. Fashion Shopping with Individualized Avatars. Retrieved from the World Wide Web: http://www.fashion-me.com/.

IST. HUMODAN. (2001). Technical Annex, 10-11. Retrieved from the World Wide Web: http://dmi.uib.es/research/GV/HUMODAN/.

IST. IMUTUS. Interactive Music Tuition System. Retrieved from the World Wide Web: http://www.ilsp.gr/imutus_eng.html.

IST. INTERFACE. MultimodAl Analysis/Synthesis SystEm for Human INTERaction to Virtual and Augmented Environments. Retrieved from the World Wide Web: http://www.ist-interface.org.

IST. ShopLab. Network for the Test and Design of Hybrid Shop Environments. Retrieved from the World Wide Web: http://www.shoplab.info.

IST. STAR. Service and Training through Augmented Reality (STAR). Retrieved from the World Wide Web: http://www.realviz.com/STAR/.

IST. VAKHUM. Virtual Animation of the Kinematics of the HUMan for Industrial, Educational and Research Purposes. Retrieved from the World Wide Web: http://www.ulb.ac.be/project/vakhum/.

IST. VISICAST. Virtual Signing: Capture, Animation, Storage & Transmission. Retrieved from the World Wide Web: http://www.visicast.co.uk/.

Katsounis, G. A., Thalmann, M. & Rodrian, H. C. (2002). E-TAILOR: Integration of 3D Scanners, CAD and Virtual-Try-on Technologies for Online Retailing of Made-to-Measure Garments. E-TAILOR IST project.

Lien, C. C. & Huang, C. L. (1998). Model-Based Articulated Hand Motion Tracking For Gesture Recognition. *IVC*, 16(2), 121-134.

MPEG-4 AFX. Retrieved from the World Wide Web: http://mpeg.telecomitalialab.com/working_documents/mpeg-04/snhc/afx_vm.zip.

MPEG-4 Requirements for Multi-user Worlds. Retrieved from the World Wide Web: http://mpeg.telecomitalialab.com/ working_documents/mpeg4/requirements/multiuser_worlds_requirements.zip,Multiuser Worlds and AFX CD, http://mpeg.telecomitalialab.com/working_documents/mpeg-04/systems/amd4.zip.

MPEG-4 Standard, Coding of Moving Pictures and Audio. Retrieved from the World Wide Web: http://mpeg.telecomitalialab.com/standards/mpeg-4/mpeg-4.htm.

Perales, F. J. & Torres, J. (1994). A system for human motion matching between synthetic and real images based on a biomechanical graphical model. IEEE Computer Society. Workshop on Motion of Non-Rigid and Articulated Objects, Austin, TX.

Plänkers, R. & Fua, P. (2001). Articulated soft objects for video-based body modelling. *IEEE International Conference on Computer Vision*. Vancouver, Canada.

Preda, M. & Preteux, F. (2002). Advanced animation framework for virtual character within the MPEG-4 standard. *Proc. IEEE International Conference on Image Processing (ICIP'2002), Rochester, NY.*

Preda, M. & Preteux, F. (2002). Critic review on MPEG-4 Face and Body Animation. *Proc. IEEE International Conference on Image Processing (ICIP'2002), Rochester, NY.*

Rehg, J. M. & Kanade, T. (1994). Visual Tracking of High {DOF} Articulated Structures: An Application to human hand tracking. *Proceedings of Third European Conference on Computer Vision. Austin*, TX, 2, 37-46.

Saito, H., Watanabe, A. & Ozawa, S. (1999). Face Pose Estimating System based on Eigen Space Analysis. ICIP99, Kobe, Japan.

Sévenier, M. B. (2002). FPDAM of ISO/IEC 14496-1/AMD4, ISO/IEC JTC1/SC29/WG11, N5471, Awaiji.

Sidenbladh, H., Black, M. J. & Sigal, L. (2002). Implicit probabilistic models of human motion for synthesis and tracking. *Proc. European Conf. on Computer Vision*. Copenhagen, Denmark.

Sminchisescu, C. & Triggs, B. (2001). Covariance scaled sampling for monocular 3D body tracking. *IEEE Int. Conf. on Computer Vision and Pattern Recognition*. Kauai Marriott, Hawaii.

Strauss, P. S. (1993). IRIS Inventor, A 3D Graphics Toolkit. *Proceedings of the 8th Annual Conference on Object-Oriented Programming Systems, Languages, and Applications*. Edited by A. Paepcke. ACM Press, 192-200.

Sullivan, J. & Carlsson, S. (2002). Recognizing and Tracking Human Action. *Proc. European Conf. on Computer Vision (ECCV)*, Copenhagen, Denmark.

Tian, Y., Kanade, T. & Cohn, J. F. (2000). Recognizing Upper Face Action Units for Facial Expression Analysis. *Computer Vision and Pattern Recognition (CVPR'00)*- Hilton Head, SC, 1,1294-1298.

Tramberend, H. (2000). Avango: A Distributed Virtual Reality Framework. GMD – German National Research Center for Information Technology.

Vicon Website. (2001). Retrieved from the World Wide Web: http://www.metrics.co.uk/animation/.

Wren, C. R., Azarbayejani, A., Darrell, T. & Pentland, A. (1997, July). Pfinder: Real-Time Tracking of the Human Body. *IEEE Trans. PAMI.*, 19(7), 780-785.

X3D Task Group. Retrieved from the World Wide Web: http://www.web3d.org/fs_x3d.htm.

Zaharia, T., Preda, M. & Preteux, F. (1999). 3D Body Animation and Coding Within an MPEG-4 Compliant Framework. Proceedings International Workshop on Synthetic-Natural Hubrid Coding and 3D Imaging (IWSNHC3DI99). Santorini, Greece, 74-78.

About the Authors

Nikos Sarris received his Ph.D. from the Aristotle University of Thessaloniki in "3D Modelling Techniques for the Human Face" and his Master of Engineering (M.Eng.) degree in Computer Systems Engineering from the University of Manchester Institute of Science and Technology (UMIST). He has worked as a Research Assistant in the Information Processing Laboratory of the Aristotle University of Thessaloniki for four years, where he participated in several national and international projects and coordinated a national research project funded by the Greek General Secretariat of Research and Technology. He has worked as a Research Fellow for the Informatics & Telematics Institute for three years, where he participated in several national and international projects and coordinated a Thematic Network of Excellence within the European Commission Information Society Technologies 5th Framework Programme. Dr. Sarris has fulfilled his military service in the Research & Informatics Corps of the Greek Army and has been a member of the Greek Technical Chamber as a Computer Systems and Informatics Engineer since 1996. His research interests include 3D model-based image and video processing, image coding, image analysis and sign language synthesis and recognition.

Michael G. Strintzis is the Director of Informatics & Telematics Institute (Centre of Research and Technology Hellas) and a Professor of Electrical and Computer Engineering at Aristotle University of Thessaloniki. He received the Diploma degree in electrical engineering from the National Technical University of Athens, Athens, Greece, in 1967, and his M.A. and Ph.D. degrees in electrical engineering from Princeton University, Princeton, NJ, in 1969 and 1970, respectively. He then joined the Electrical Engineering Department at the University of Pittsburgh, Pittsburgh, PA, where he served as Assistant Professor (1970–1976) and Associate Professor (1976–1980). Since 1980, he has been Professor of

Electrical and Computer Engineering at the University of Thessaloniki, Thessaloniki, Greece and, since 1999, Director of the Informatics and Telematics Research Institute, Thessaloniki. His current research interests include 2D and 3D image coding, image processing, biomedical signal and image processing, and DVD and Internet data authentication and copy protection. Dr. Strintzis is currently a member of the management committee of the European programme “Information Society Technologies” (IST) and of the Joint Communication Board of the European Space Agency.

* * * * *

Niki Aifanti is an Assistant Researcher in the Informatics & Telematics Institute, Greece. She received her B.S. degree in Electrical and Computer Engineering from the Aristotle University of Thessaloniki in 2000 and the M.Sc. in Multimedia Signal Processing and Communications from the University of Surrey, UK, in 2001. She is currently a Ph.D. candidate in Aristotle University of Thessaloniki. Her research interests include gesture recognition, human motion analysis, 3D human body pose reconstruction and 3D body modeling.

Ana C. Andrés del Valle received the Spanish State degree of Telecommunications Engineering from the ETSETB or Barcelona Technical School of Telecom at UPC, Barcelona, Spain. In September 2003, she will receive her Ph.D. degree from Télécom Paris after doing research at the Multimedia Communications Department of the Eurecom Institute, Sophia Antipolis, France. As a researcher, she has cooperated with several telecom companies: AT&T Labs – Research, New Jersey (1999) and France Telecom R&D – Rennes (during her Ph.D.). In academics, Ana C. Andrés has supervised student research, has written several publications and prepared specialized tutorials related to Human Body Motion Analysis and Synthesis. In 2002, she was a visiting professor at the Computer Science and Mathematics Department of the University of the Balearic Islands (Spain). Her research interests are image and video processing for multimedia applications, especially facial expression analysis, virtual reality, human computer interaction and computer graphics.

Themis Balomenos was born in Athens, Greece, in 1977. He received the Diploma in Electrical and Computer Engineering from the National Technical University of Athens (NTUA), Greece, in 2000. Since 2001, he has been pursuing his Ph.D. and working as a Researcher at the Image, Video, and Multimedia Systems Laboratory in NTUA, in the fields of Human Machine Interaction and Computer Vision. He is a member of the Technical Chamber of

Greece. His research interests include computer vision, human motion and gesture analysis, human machine interaction, and biomedical applications. He has published five papers in the above fields.

Costas T. Davarakis is the C.E.O. of Systema Technologies SA based in Athens, Greece. His research interests include interactive 3D modeling, virtual and augmented reality and technology enhanced learning. He received a Greek diploma in Electrical Engineering from the University of Patras in 1983, an M.Sc. in Digital Systems from Brunel University of West London in 1985 and a Ph.D. in Computer Engineering from the University of Patras in 1990. He has been a Post-Doctoral Researcher in the Greek Computer Technology Institute until 1993. In the past 10 years he founded, organized and led the R&D activities of the company.

Athanasios Drosopoulos was born in Lamia, Greece, in 1976. He received his degree in Computer Science from the University of Ioannina in 1998. Currently, he is a Ph.D. candidate in the Image, Video and Multimedia Systems Laboratory of the Electrical and Computer Engineering Department of the National Technical University of Athens. His main research interests include 3D motion and structure estimation, non-rigid motion estimation, facial feature segmentation and gesture analysis. He has published three journal and seven conference papers in these fields. He is also a member of the EUNITE network on intelligent technologies.

Jean-Luc Dugelay (Ph.D., 1992, IEEE M'94-SM'02) joined the Eurecom Institute, France (Sophia Antipolis) in 1992, where he is currently a Professor in charge of image and video research and teaching activities inside the Multimedia Communications Department. Previously, he was a Ph.D. candidate at the Department of Advanced Image Coding and Processing at France Telecom Research in Rennes, where he worked on stereoscopic television and 3D motion estimation. His research interests are in the area of multimedia signal processing and communications; including security imaging (i.e., watermarking and biometrics), image/video coding, facial image analysis, virtual imaging, face cloning and talking heads. His group is currently involved in several national and European projects related to image processing. Jean-Luc Dugelay is currently an Associate Editor for the *IEEE Transactions on Multimedia*. He is a member of the IEEE Signal Processing Society, Image and Multidimensional Signal Processing Technical Committee (IEEE IMDSP TC). Jean-Luc Dugelay serves as a Consultant for several major companies, in particular, France Telecom R&D.

Peter Eisert is the Head of the Computer Vision & Graphics Group of the Image Processing Department at the Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute. He received his diploma degree in Electrical Engineering from the University of Karlsruhe, Germany, in 1995. He then joined the Telecommunications Institute of the University of Erlangen, Germany, where he worked on facial animation, model-based video coding, 3D geometry reconstruction and light field coding. He was a member of the graduate research center "3D image analysis and synthesis" and involved in the organization of multiple national and international workshops. After receiving his Ph.D. in 2000, he joined the Information Systems Laboratory, Stanford University, as a postdoc. Since 2002, he has been with the Fraunhofer Institute for Telecommunications. His present research interests focus on 3D image and image sequence processing, as well as algorithms from computer vision and graphics. He is currently working on facial expression analysis and synthesis for low-bandwidth communication, 3D geometry reconstruction, and acquisition and streaming of 3D image-based scenes. He is a Lecturer at the Technical University of Berlin and the author of numerous technical papers published in international journals and conference proceedings.

Nikos Grammalidis is an Associate Researcher in the Informatics & Telematics Institute, Greece. He received his B.S. and Ph.D. degrees in Electrical and Computer Engineering from the Aristotle University of Thessaloniki, in 1992 and 2000, respectively. His Ph.D. dissertation was titled, "Analysis, Coding and Processing of Multi-view Image Sequences Using Object-Based Techniques." Prior to his current position, he was a researcher on 3D Imaging Laboratory at the Aristotle University of Thessaloniki. His main research interests include image compression, 3D data processing, multimedia image communication, 3D motion estimation, stereo and multiview image sequence coding. His involvement with those research areas has led to the co-authoring of more than 10 articles in refereed journals and more than 30 papers in international conferences. Since 1992, he has been involved in more than 10 projects, funded by the EC and the Greek Ministry of Research and Technology.

E. A. Hendriks received his M.Sc. and Ph.D. degrees from the University of Utrecht in 1983 and 1987, respectively, both in physics. In 1987 he joined the Electrical Engineering faculty of Delft University of Technology (The Netherlands) as an Assistant Professor. In 1994 he became a member of the Information and Communication Theory of this faculty and since 1997 he heads the computer vision section of this group as an Associate Professor. His interest is in computer vision, low-level image processing, image segmentation, stereoscopic and 3D imaging, motion and disparity estimation, structure from motion/ disparity/silhouette and real time algorithms for computer vision applications.

Pengyu Hong received his B.Eng. and M.Eng. degrees from Tsinghua University, Beijing, China, and his Ph.D. degree from University of Illinois at Urbana-Champaign, Urbana, all in Computer Science. Currently, he is a Postdoctoral Researcher at School of Public Health, Harvard University, USA. His research interests include human computer interaction, computer vision and pattern recognition, machine learning and multimedia database. In 2000, he received the Ray Ozzie Fellowship for his research work on facial motion modeling, analysis and synthesis.

Thomas S. Huang received his B.S. in Electrical Engineering from National Taiwan University, Taipei, Taiwan, China; and his M.S. and Sc.D. Degrees in Electrical Engineering from the Massachusetts Institute of Technology, Cambridge, Massachusetts. He was on the Faculty of the Department of Electrical Engineering at MIT from 1963 to 1973; and on the Faculty of the School of Electrical Engineering and Director of its Laboratory for Information and Signal Processing at Purdue University from 1973 to 1980. In 1980, he joined the University of Illinois at Urbana-Champaign (USA), where he is now a William L. Everitt Distinguished Professor of Electrical and Computer Engineering, and Research Professor at the Coordinated Science Laboratory, and Head of the Image Formation and Processing Group at the Beckman Institute for Advanced Science and Technology. Dr. Huang's professional interests lie in the broad area of information technology, especially the transmission and processing of multidimensional signals. He has published 12 books, and over 300 papers in network theory, digital filtering, image processing, and computer vision. He is a Fellow of the International Association of Pattern Recognition, IEEE, and the Optical Society of American. He has received a Guggenheim Fellowship, an A.V. Humboldt Foundation Senior U.S. Scientist Award, and a Fellowship from the Japan Association for the Promotion of Science. He received the IEEE Acoustics, Speech, and Signal Processing Society's Technical Achievement Award in 1987, and the Society Award in 1991. He is a Founding Editor of the *International Journal Computer Vision, Graphics, and Image Processing*; and Editor of the Springer Series in Information Sciences, published by Springer Verlag.

Spiros Ioannou was born in Athens, Greece, in 1975. He received the Diploma in Electrical and Computer Engineering from the National Technical University of Athens (NTUA), Greece, in 2000. Since 2000, he is pursuing his Ph.D. and working as a Researcher at the Image, Video, and Multimedia Systems Laboratory in NTUA, in the fields of Human Machine Interaction and Computer Vision. He is a member of the Technical Chamber of Greece. His research interests include image segmentation, computer vision and facial feature extraction.

Gregor A. Kalberer is a Ph.D. student at the Computer Vision Lab BIWI, D-ITET, ETH Zurich, Switzerland. He received his M.Sc. in Electrical Engineering from ETH Zurich in 1999. His research interests include computer vision, graphics, animation and virtual reality. He is a member of the IEEE.

Markus Kampmann was born in Essen, Germany, in 1968. He received his Diploma degree in Electrical Engineering from the University of Bochum, Germany, in 1993 and his Doctoral degree in Electrical Engineering from the University of Hannover, Germany, in 2002. From 1993 to 2001, he was working as a Research Assistant at the Institut für Theoretische Nachrichtentechnik und Informationsverarbeitung of the University of Hannover, Germay. His research interests were video coding, facial animation and image analysis. Since 2001, he has been working with Ericsson Eurolab in Herzogenrath, Germany. His work fields are multimedia streaming, facial animation and computer graphics.

Nikos Karatzoulis (M.Sc.), born in 1974, he has been studying at the University of Sunderland (B.A., Business Computing, 1994-1997) and at the University of Leeds (M.Sc., Distributed Multimedia Systems, 1997-1998). His main area of interest is Virtual Environments. Currently, he participates in several IST projects: IMUTUS, HUMODAN and SHOPLAB.

Kostas Karpouzis was born in Athens, Greece, in 1972. He graduated from the Department of Electrical and Computer Engineering, the National Technical University of Athens in 1998 and received his Ph.D. degree in 2001 from the same university. His current research interests lie in the areas of human computer interaction, image and video processing, 3D computer animation and virtual reality. He is a member of the Technical Chamber of Greece and a member of ACM SIGGRAPH and SIGCHI societies. Dr. Karpouzis has published seven papers in international journals and more than 25 in proceedings of international conferences. He is a member of the technical committee of the International Conference on Image Processing (ICIP) and Co-editor of the Greek Computer Society newsletter. Since 1995, he has participated in eight research projects at Greek and European levels.

Aggelos K. Katsaggelos received his Diploma degree in Electrical and Mechanical Engineering from the Aristotelian University of Thessaloniki, Thessaloniki, Greece, in 1979 and his M.S. and Ph.D. degrees, both in Electrical Engineering, from the Georgia Institute of Technology, Atlanta, in 1981 and 1985, respectively. In 1985, he joined the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, Illinois, USA, where he is currently a Professor, holding the Ameritech Chair of Information Technology. He is the past Editor-in-Chief of the *IEEE Signal Processing Magazine* (1997-2002), co-author of *Rate-Distortion Based Video Compression* (Kluwer 1997), editor of *Digital Image Restoration* (Springer-Verlag 1991), and co-editor of *Recovery Techniques for Image and Video Compression and Transmission,* (Kluwer 1998) and a number of conference proceedings. He is the co-inventor of eight international patents, a Fellow of the IEEE (1998), and the recipient of the IEEE Third Millennium Medal (2000), the IEEE Signal Processing Society Meritorious Service Award (2001), and an IEEE Signal Processing Society Best Paper Award (2001).

Stefanos Kollias was born in Athens in 1956. He obtained his Diploma from the National Technical University of Athens (NTUA) in 1979, his M.Sc. in Communication Engineering in 1980 from UMIST in England and his Ph.D. in Signal Processing from the Computer Science Division of NTUA. He has been with the Electrical Engineering Department of NTUA since 1986, where he serves now as a Professor. Since 1990, he has been the Director of the Image, Video and Multimedia Systems Laboratory of NTUA. He has published more than 120 papers in the above fields, 50 of which were published in international journals. He has been a member of the Technical or Advisory Committee or invited speaker in 40 International Conferences. He is a reviewer of 10 IEEE Transactions and of 10 other journals. Ten graduate students have completed their doctorate under his supervision, while another 10 are currently performing their Ph.D. thesis. He and his team have been participating in 38 European and National projects.

Gauthier Lafruit was a Research Scientist with the Belgian National Foundation for Scientific Research from 1989 to 1994, being mainly active in the area of wavelet image compression implementations. Subsequently, he was a Research Assistant with the VUB (Free University of Brussels, Belgium). In 1996, he became the recipient of the Scientific Barco award and joined IMEC (Interuniversity MicroElectronics Centre, Leuven, Belgium), where he was involved as Senior Scientist with the design of low-power VLSI for combined JPEG/wavelet compression engines. He is currently the Principal Scientist in the Multimedia Image Compression Systems Group with IMEC. His main interests include progressive transmission in still image, video and 3D object coding, as well as scalability and resource monitoring for advanced, scalable video and 3D coding applications. Gauthier Lafruit is the author/co-author of around 60 scientific publications, 50 MPEG standardization contributions, five patent(s) (applications) and has participated (and been appointed as evaluator) in several national and international project(s) (proposals).

B. J. Lei received his B.Sc. in Computer Software and M.Sc. in Parallel Network Computing from Xi'an Jiaotong University, China in 1995 and 1998, respectively. He is now a "post-doc" in the Information and Communication Theory Group at the Technical University of Delft, The Netherlands. His main research interests are in low-level image processing, 3-D imaging, computer vision, and multimedia applications.

Tiehan Lv is a graduate student and member of the Embedded Systems Group in the Department of Electrical Engineering at Princeton University, USA. He received an M.A.E. in Electrical Engineering from Peking University, China. He is a member of the ACM.

Sotiris Malassiotis was born in Thessaloniki, Greece, in 1971. He received his B.S. and Ph.D. degrees in Electrical Engineering from the Aristotle University of Thessaloniki, in 1993 and 1998, respectively. From 1994 to 1997, he was conducting research in the Information Processing Laboratory of Aristotle University of Thessaloniki. He is currently a senior researcher in the Informatics & Telematics Institute, Thessaloniki. He has participated in several European and National research projects. He is the author of more than ten articles in refereed journals and more than 20 papers in international conferences. His research interests include stereoscopic image analysis, range image analysis, pattern recognition, and computer graphics.

Pascal Müller is a Consultant to the Computer Vision Lab of the ETH Zurich (Switzerland) and also works as Technical Director for the production company Centralpictures. He received his Master's Degree in Computer Science from the ETH Zurich in 2001. His research areas are computer animation, procedural/ physical modeling and sound-sensitive graphics.

Burak Ozer received his B.S. degree in Electrical and Communications Engineering from Istanbul Technical University in 1993, his M.S. degree in Electrical Engineering from Bogazici University, Istanbul, Turkey, in 1995, and his Ph.D. degree in Electrical and Computer Engineering from New Jersey Institute of Technology in 2000. Currently, he is a Research Staff Member at the Department of Electrical Engineering, Princeton University, USA. He was a Post-Doctoral Researcher at the same department in 2001. His research interests include real-time systems, smart cameras, surveillance systems, digital image and video libraries, pattern recognition, video/image compression and 2D/3D object modeling. He is a member of the IEEE and member of the Embedded Systems Group in the Department of Electrical Engineering at Princeton University.

Marius Preda received an Engineer Degree in Electronics from the Polytechnical Institute of Bucarest, in 1998, and a Ph.D. Degree in Mathematics and Informatics from the University Paris V - René Descartes, Paris, in 2001. He started his career as a Production Engineer at "Electronica Aplicata" (Bucharest) and then as a Researcher at University "Electronica si Telecommunicatii" (Bucharest). During his Ph.D. studies, he was an R&D Engineer in the ARTEMIS Project Unit at INT (Evry, France), where he is currently R&D Project Manager. His main research interests include generic virtual character definition and animation, rendering, low bit-rate compression and transmission of animation, multimedia composition and multimedia standardization. He has been actively involved (around 50 contributions) in MPEG-4 since 1998, especially focusing on synthetic objects coding. He is the main contributor of the new animation tools dedicated to generic synthetic objects, promoted by MPEG-4 as part of the "Animation Framework eXtension" specifications.

Françoise Preteux graduated from the Ecole des Mines de Paris (EMP) and received her Doctorat d'Etat ès Sciences Mathématiques from the University of Paris VI, in 1982 and 1987, respectively. After working as a Research Engineer at the Center for Mathematical Morphology of EMP, she held a position as Professor at the Ecole Nationale Supérieure des Télécommunications de Paris (1989-1993). Since 1994, she has been a Professor at the Institut National des Télécommunications, being successively the Head of the Signal & Image processing Department (1994-1998) and of the ARTEMIS Project Unit (1999-present). She is the (co)-author of over 80 major scientific papers within the field of stochastic modeling and mathematical morphology, medical imaging segmentation, 3D modeling/reconstruction, indexing techniques and digital image coding. She is a regular reviewer for international journals and a member of international conference program committees. She actively contributes to the MPEG standardization process, being the Deputy Head of the French Delegation for MPEG-7, the official liaison between SC29-WG11 and CEN-ISSS and the France representative at the ISO SC 29.

Amaryllis Raouzaiou was born in Athens, Greece, in 1977. She graduated from the Department of Electrical and Computer Engineering, the National Technical University of Athens in 2000 and she is currently pursuing her Ph.D. degree at the Image, Video, and Multimedia Systems Laboratory at the same University. Her current research interests lie in the areas of synthetic-natural hybrid video coding, human-computer interaction and machine vision. She is a member of the Technical Chamber of Greece. She is with the team of IST project ERMIS (Emotionally Rich Man-Machine Interaction Systems). She has published three journal articles and eight conference papers in the above fields.

Ioan Alexandru Salomie received his M.Sc. degree in Mechanics and Machines Construction from the "Politechnica" University of Cluj-Napoca, Romania, in 1989 and his M.Sc. degree in Applied Computer Science from the Vrije Universiteit Brussel (VUB), Belgium, in 1994. Since October 1995, he has been a member of the Department of Electronics and Information Processing (ETRO) at VUB. He has a rich experience in software design and development of tools for image and data visualization, image analysis, and telemedicine. His research has evolved in the direction of surface extraction, coding, and animation of polygonal surface meshes, and he is currently finishing his Ph.D. on this topic. Since 2000 he has been actively involved in the SNHC group (Synthetic Natural Hybrid Coding) of MPEG-4, and is the main contributor to the MeshGrid surface representation in SNHC.

Angel Sappa received the Electro-Mechanical Engineering Degree in 1995 from the National University of La Pampa, La Pampa-Argentina, and the Ph.D. degree in Industrial Engineering in 1999 from the Polytechnic University of Catalonia Barcelona-Spain. From 1999 to 2002 he undertook post-doctorate research in the field of 3D modeling of rigid objects at the LAAS-CNRS, Toulouse-France and at Z+F UK Ltd. Manchester-UK. Since August 2002 to August 2003 he was with the Informatics and Telematics Institute, Thessaloniki-Greece, as a Marie Curie Research Fellow. Since September 2003 he has been with the Computer Vision Center, Barcelona, Spain. His research interests span a broad spectrum whiting the 2D and 3D image processing topics. His current research interests are focused on model-based segmentation, 3D modeling and 2D-to-3D conversion from image sequences. He is also interested in finding connections and overlapping areas between computer vision and computer graphics.

Nicolas Tsapatsoulis was born in Limassol, Cyprus, in 1969. He graduated from the Department of Electrical and Computer Engineering, the National

Technical University of Athens in 1994 and received his Ph.D. degree in 2000 from the same university. His current research interests lie in the areas of human computer interaction, machine vision, image and video processing, neural networks and biomedical engineering. He is a member of the Technical Chambers of Greece and Cyprus and a member of IEEE Signal Processing and Computer societies. Dr. Tsapatsoulis has published 10 papers in international journals and more than 30 in proceedings of international conferences. He served as Technical Program Co-Chair for the VLBV'01 workshop. He is a reviewer of the *IEEE Transactions on Neural Networks* and *IEEE Transactions on Circuits and Systems for Video Technology* journals.

Jilin Tu received his B.Eng. degree and M.Eng. Degree from Huazhong University of Science and Technology, Wuhan, China, and his M.S. degree from Colorado State University. Currently, he is pursuing his Ph.D. degree in the Department of Electrical Engineering at University of Illinois at Urbana-Champaign (USA). His research interests include facial motion modeling, analysis and synthesis; machine learning and computer vision.

Dimitrios Tzovaras is a Senior Researcher Grade C (Assistant Professor) at the Informatics & Telematics Institute, Greece. He received his Diploma in Electrical Engineering and his Ph.D. in 2D and 3D Image Compression from the Aristotle University of Thessaloniki, Greece, in 1992 and 1997, respectively. Prior to his current position, he was a leading researcher on 3D Imaging at the Aristotle University of Thessaloniki. His main research interests include human and body modeling and animation 3D data processing, multimedia image communication and virtual reality. His involvement with those research areas has led to the co-authoring of over 20 articles in refereed journals and more than 50 papers in international conferences. He has served as a regular reviewer for a number of international journals and conferences. Since 1992, Dr. Tzovaras has been involved in more than 20 projects, funded by the EC and the Greek Ministry of Research and Technology.

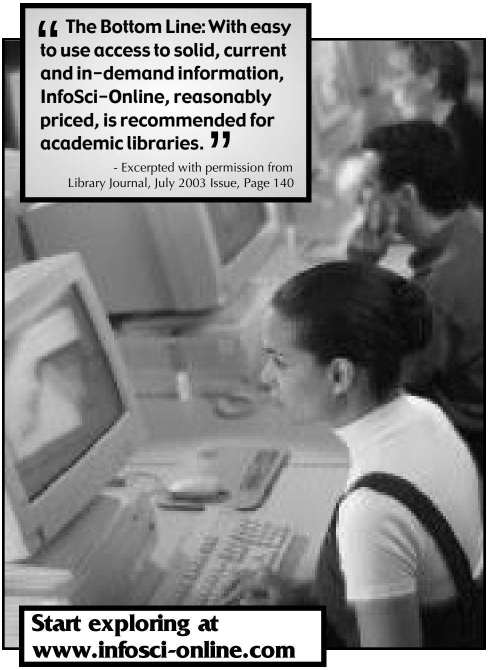
Luc Van Gool is Professor for Computer Vision at the University of Leuven in Belgium and at ETH Zurich in Switzerland. He is a member of the editorial board of several computer vision journals and of the programme committees of international conferences about the same subject. His research includes object recognition, tracking, texture, 3D reconstruction, and the confluence of vision and graphics. Vision and graphics for archaeology is among his favourite applications.

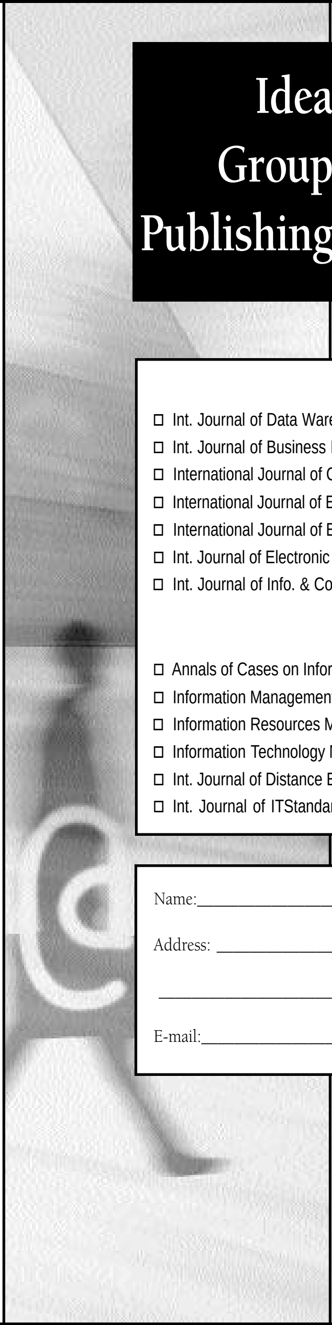
Zhen Wen received the B.Eng. degree from Tsinghua University, Beijing, China, and the M.S. degree from University of Illinois at Urbana-Champaign, Urbana, both in computer science. Currently, he is pursuing his Ph.D. degree in the Department of Computer Science at University of Illinois at Urbana-Champaign. His research interests include facial motion modeling, analysis and synthesis; image based modeling and rendering; machine learning and computer vision; multimedia systems and communication.

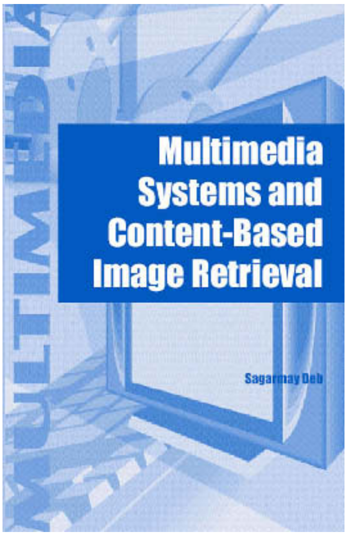
Wayne Wolf is professor of electrical engineering at Princeton University. Before joining Princeton, he was with AT&T Bell Laboratories, Murray Hill, New Jersey. He received his B.S., M.S., and Ph.D. degrees in electrical engineering from Stanford University in 1980, 1981, and 1984, respectively. His research interests include embedded computing, VLSI systems, and multimedia information systems. He is the author of Computers as Components and Modern VLSI Design. Dr. Wolf has been elected to Phi Beta Kappa and Tau Beta Pi. He is a Fellow of the IEEE and ACM and a member of the SPIE and ASEE.

Liang Zhang was born in Zhejiang, China in 1961. He received his B.Sc. degree from Chengdu Institute of Radio Engineering in 1982, his M.Sc. degree from Shanghai Jiaotong University in 1986 and his Doctoral degree in electrical engineering from the University of Hannover, Germany, in 2000. He was working as an assistant from 1987 to 1988 and as a lecturer from 1989 to 1992 in the Department of Electrical Engineering, Shanghai Jiaotong University. From 1992 to 2000, he was a research assistant at the Institut für Theoretische Nachrichtentechnik und Informationsverarbeitung, University of Hannover, Germany. Since 2000, he has been with Communications Research Centre Canada. His research interests are image analysis, computer vision, and video coding.

Index






MULTIMEDIA
Multimedia Systems and Content-Based Image Retrieval
Sagarmay Deb