Sample-Based Talking-Head Synthesis

Eric Cosatto

PhD. Thesis
October 2002
Swiss Federal Institute of Technology
Signal Processing Laboratory
Prof. Murat Kunt

Chapter 1: Introduction

1.1 Version Abrégée 2
1.2 Abstract 3
1.3 Overview 3
1.4 A natural way to interface with computers 4
1.5 An enabling technology for future access to information 5
    1.5.1 Talking-Heads for e-learning 5
    1.5.2 Talking-Heads for customer relations (e-care) 6
    1.5.3 Talking-Heads for e-commerce 6
    1.5.4 The need for video-realism 7
1.6 Previous Work 7
1.7 Our solution 9
1.8 References 10
1.9 Figures 11

Chapter 2: Creation of the Head Model (Analysis)

2.1 Introduction 16
2.2 Overview 17
2.3 Modeling 17
    2.3.1 A decomposition into facial parts 17
    2.3.2 Static 3D shape 18
    2.3.3 Soft Edges 18
    2.3.4 Dynamic Textures 19
2.4 Recording 19
    2.4.1 Overview 19
    2.4.2 Phonemes 19
    2.4.3 Coarticulation 20
    2.4.4 Visemes 20
    2.4.5 Defining a Corpus 21
    2.4.6 Audio-Video Recordings 21
    2.4.7 3D scanning 22
2.5 Phonetic Labeling of Audio Data 22
2.6 Image Analysis 23
    2.6.1 Introduction 23
    2.6.2 Locating Face and Measuring Facial Features 23
    2.6.3 Pose Estimation 27
    2.6.4 Sample Extraction and Normalization 30
    2.6.5 Sample Parameterization 30
2.7 References 32
2.8 Figures 34

Chapter 3: Animation of the Head Model (Synthesis)

3.1 Introduction 57
3.2 Architecture 58
3.3 Lip Synchronization 58
    3.3.1 Introduction 58
    3.3.2 Obtaining a target phoneme input 59
    3.3.3 Building a Search Graph 60
    3.3.4 Calculating the Target Cost 60
    3.3.5 Pruning the Graph 61
    3.3.6 Visual Features 62
    3.3.7 Calculating the Concatenation Cost 62
    3.3.8 Viterbi Search 63
    3.3.9 Transitions 63
    3.3.10 Objective Evaluation 64
    3.3.11 Subjective Evaluation 65
3.4 Visual Prosody 65
    3.4.1 Prosody of Head Movements 66
    3.4.2 Analysis of Head Movements during Speech 66
    3.4.3 Visual Prosody using a library of standard normalized motions 67
    3.4.4 Visual Prosody from Recorded Video Segments 67
    3.4.5 ‘Mechanical’ Facial Motions 67
    3.4.6 Inserting Expressions 68
3.5 Implementation 68
    3.5.1 Development Environment 68
    3.5.2 Real-Time Synthesis 69
3.6 Applications 70
    3.6.1 Automated Newscaster 70
    3.6.2 Help-Desk 71
3.7 References 72
3.8 Figures 74

Chapter 4: Conclusions

4.1 General Conclusions 95
    4.1.1 Model Creation 95
    4.1.2 Synthesis 96
4.2 Future Work 97
4.3 Technical Contributions 97
4.4 Personal Contributions 98
4.5 Acknowledgements 98
4.6 Curriculum Vitae 99
    4.6.1 Professional experience 99
    4.6.2 Education 99
    4.6.3 Other 99
    4.6.4 Publications 100
    4.6.5 Patents 101

CHAPTER 1

INTRODUCTION

Table of Contents

1.1 Version Abrégée 2
1.2 Abstract 3
1.3 Overview 3
1.4 A natural way to interface with computers 4
1.5 An enabling technology for future access to information 5
    1.5.1 Talking-Heads for e-learning 5
    1.5.2 Talking-Heads for customer relations (e-care) 6
    1.5.3 Talking-Heads for e-commerce 6
    1.5.4 The need for video-realism 7
1.6 Previous Work 7
1.7 Our solution 9
1.8 References 10
1.9 Figures 11

1.1 Version Abrégée

Dans ce travail, nous présentons un système capable de générer par ordinateur des animations vidéo très réalistes d'une personne qui parle à partir de données textuelles. Les approches infographiques traditionnelles n'ayant pour l'instant pas réussi à produire des agents virtuels ayant une apparence réaliste et qui, lorsqu'ils parlent, peuvent être pris pour une vraie vidéo, nous avons décidé d'attaquer le problème d'un angle différent. L'approche traditionnelle en infographie consiste à modéliser la forme de la tête a partir de polygones en trois dimensions (3D). Une couche intelligente est ensuite rajoutée pour permettre un contrôle de haut niveau comme par exemple des actions musculaires. Pour terminer, une simple image (ou texture) de la personne est appliquée afin de donner une apparence naturelle au modèle. Cependant, l'apparence extrêmement complexe du visage humain (la plasticité des lèvres et de la langue, les innombrables rides, plis et crevasses de la peau ainsi que le système pileux) de même que la dynamique subtile de l'articulation (mécanique de la rotation de la mâchoire, du mouvement de la langue et des lèvres, effets de co-articulation venant de l'adaptation du cerveau aux contraintes mécaniques de l'appareil vocal) rendent presque impossible la modélisation analytique complète d'une tête parlante. De plus, les nombreuses approximations et simplifications nécessaires à ce type d'implémentation 'analytique' résultent en général en des animations à l'apparence synthétique.

Notre approche est d'éviter complètement cette modélisation analytique et plutôt d'utiliser des échantillons vidéo enregistrés d'une personne qui parle. L'apparence naturelle est ainsi inhérente aux images enregistrées et la dynamique de l'articulation peut être évaluée par une analyse audio-visuelle des séquences vidéo. L'inconvénient de ce genre d'approches dites 'par échantillonnage' est un manque de flexibilité aussi bien pour la synthèse de nouvelles séquences parlées (qui n'ont pas été enregistrées) que pour le rendu des échantillons d'image sous une nouvelle vue (différente de celle sous laquelle ils ont été enregistrés). L'élément clef de cet ouvrage est de définir une modélisation de la tête qui permette un rendu flexible sous une gamme limitée mais utile de vues, ainsi que de développer les algorithmes qui permettent la synthèse d'animation parlées quelquonques a partir d'un nombre limité d'échantillons enregistrés.

Notre modélisation de la tête s'appuie sur une décomposition hiérarchique en parties contenant l'information sur la forme (en 3D) ainsi que l'information de texture (sous forme d'échantillons-image). L'information 3D de chaque partie du visage est codée de façon à ce que la forme 3D puisse accommoder toutes les textures. Elle est utilisée aussi bien pour extraire les échantillons-image lors de la phase d'analyse (création du modèle) que pour reproduire chaque partie du visage sous des vues différentes lors de la synthèse afin de permettre un rendu des mouvements de la tête.

Dans l'optique de générer des animations parlée de nature générale (et non pas limitées à un domaine particulier) à partir d'une quantité limitée d'échantillons-image, notre système utilise un ensemble de caractéristiques extraites des échantillons-image, tels que le contexte phonétique, les dimensions et positions des lèvres ainsi que la position de la tête en 3D, couplés à une approche en programmation dynamique destinés à découvrir la séquence optimale d'échantillons-image produisant une animation régulière et sans à-coups du texte cible. Notre système s'appuie en entrée sur une annotation phonétique du texte cible, ce qui permet d'utiliser aussi bien une séquence audio enregistrée et annotée qu'une séquence audio de synthèse.

Nous avons développé plusieurs applications dont l'interface utilisateur s'appuie sur nos têtes parlantes. La première est un lecteur automatique des nouvelles du jour. Ce système récupère sur certains sites Web le contenu en nouvelles journalières et crée dynamiquement un nouveau site où notre tête parlante, en récitant pour chaque titre un petit résumé, offre à l'utilisateur une expérience nouvelle, de style télévisuel. Une seconde application est d'offrir sur le Web un système de service clientèle qui recrée l'apparence visuelle et restitue l'aspect humain d'une interaction entre un client et le représentant du produit. Pour cela nous avons connecté notre tête parlante à un système existant de service clientèle basé sur une interaction vocale par téléphone.

1.2 Abstract

In this thesis we present a system that produces photo-realistic computer animations of a person talking from general text. Recognizing that traditional computer graphics has so-far failed to create photo-realistic computer agents that, when speaking, can be mistaken from video of real humans, we set out to attack the problem from a different angle. In traditional computer graphics, a 3D wireframe model is created that represents the shape of the head. Then an "intelligent layer" is added to provide high-level control such as muscle-actions. Finally a single texture map is used to provide natural appearance of the skin. But the complexities of the face's appearance (highly deformable lips and tongue, countless creases and wrinkles of the skin) and the complex dynamics of speech (mechanics of the jaw, tongue and lips, coarticulation effects stemming from the brains adaptation to the vocal apparatus) make it almost impossible to model a talking-head in such an 'analytical' way. The numerous approximation and shortcuts used by such systems result in unnatural animations.

Our approach is to bypass the complexities linked to analytical modeling and instead use recorded data samples of the subject speaking. The natural appearance is then inherent to the recorded images and the dynamics of speech are captured by audio-visual analysis of the recorded video sequences. The typical drawback of such sample-based approaches is a lack of flexibility, both in producing new speech animation from the recorded samples, and in showing samples under new views (different from the view under which they were recorded). The key elements of this work are to define a model that allow for flexibility in showing the head under a limited (but useful) range of views and develop algorithms that enable the production of general speech from a limited amount of recorded data.

The head model is a hierarchical set of facial parts containing 3D shape information as well as a set of image samples. The 3D shapes models the "overall" shape of the facial parts and does not attempt to model the details which are instead captured and rendered via the sample images. These 3D shapes are used to render these facial parts under a range of views by texture-mapping the image samples, thus allowing head movements.

To produce general speech from a limited set of recorded samples, we use high-level features such as phonetic context, lip geometry and head pose with a dynamic programming approach to find the optimal sequence of samples that produces the smoothest animation of the target speech. Our system typically uses a TTS (text-to-speech synthesizer) to synthesize the audio track, but we can also work from a phonetically labeled recorded audio track.

We have developed several applications with the talking-head agent. One is a fully automatic news reader that grabs textual newsfeed from the Internet and dynamically creates a web site with video of our talking-head speaking the news. In another application we have hooked our talking-head agent to a TTS-based customer service system to provide a visual, human presence for users interacting with the system.

1.3 Overview

This work is articulated around two main chapters: analysis and synthesis plus an introduction and a conclusion. The introduction motivates the use of talking-heads in computer user interfaces using several areas of application to illustrate the importance and the potential benefits of this technology. The introduction also covers previous work on this subject and introduces our approach, comparing it with other systems.

The analysis chapter discusses the creation process of a sample-based talking-head model that can generate realistic speech animations. First it describes in section 2.3 the underlying hierarchical head model with its three-dimensional facial parts and dynamic textures. The recording phase is then illustrated in section 2.4, covering audio, visual and 3D data capture. Then the analysis of the raw data is described for each channel. In particular, for the visual channel (section 2.6), the process goes from low-level feature extraction to precise location of facial elements and to head pose estimation. Finally the construction of a database of facial parts is described that uses results of the analysis phase.

The synthesis chapter discusses the generation of speech animations from the talking-head model. After a description of the overall architecture of the system (section 3.2), the synthesis of the mouth as it articulates speech is discussed in section 3.3 followed by the synthesis of head movements and facial expressions in section 3.4. Details of the implementation of the system are then discussed in section 3.5 and two applications are presented in section 3.6.

Chapter 4 concludes this work and discusses several issues related to the system's performance and results. Ideas for future work are also developed.

1.4 A natural way to interface with computers

The interface between computers and users has been constantly evolving. At the dawn of computing, punch-cards were the only way to program and interact with computers. The creation of a program was a far cry from today's highly dynamic and interactive debugging paradigm. The stuff had better be right the first time or one had to go through a lengthy cycle (days) again. In essence, the interface was so inefficient that it prevented all but a few people to use the system. Following technological breakthroughs, the computer quickly evolved. Processing power, memory and data storage followed the exponential path that we all know. At the interface level, keyboards provided a more flexible medium. But monochrome alphanumeric displays, still the norm until the mid eighties, left software interfaces at their simplest level. Users were forced to learn obscure key combinations to perform actions and to decode cryptic text messages to interpret the machine's output. This clumsy interface left the door open to manipulation errors and erroneous reading of outputs, resulting in a loss of productivity. As the tasks performed by computers evolved from simple to extremely complex, their user's interface essentially remained the same, creating a virtual bottleneck.

The advent of the mouse together with graphical displays provided a new medium for user interface design, which is still being explored by researchers today. Computer aided design (CAD) was among the first type of applications benefiting from this new interface technology. Software such as word processor and spreadsheets instantly became popular. As the interface evolved, the interaction with the machine became more efficient, less frustrating, making it possible for less trained people to harness the power of computing.

While the mouse and the graphical screen provided an easier way to select and present data, it essentially remains a textual interface where the users inputs data by typing and receives textual answers to be read from the screen.

More recently Automatic Speech Recognition (ASR) and Text-To-Speech synthesizers (TTS) have added a new mode of interaction: the voice. Of particular interest for telephone-based interfaces, this new technology has also started to penetrate the realm of computer interfaces. While very useful in some particular applications such as dictation or for visually impaired people, it hasn't really had a wide success.

People usually don't expect to hear from or speak to a computer. It simply doesn't embody a real human being and hence we do not feel like addressing it as such. Talking to a screen feels somewhat strange and hearing a computer speak to you even more so. We argue that adding a talking-face to the voice will alleviate this issue of unnaturalness of the voice only interface thus enabling a wider range of applications to be voice-enabled. Early studies (with less than perfect talking-heads) [19] have shown that 65% of users prefer a talking-head interface over voice-only or text-only interfaces in the context of an e-commerce setup.

In this work, we are not addressing the generation of the voice, but rather concentrate of synthesizing the visual speech channel in the form of video-realistic talking-heads. In fact in many applications, the voice of a speaking person can be more easily recorded than the video (no need for makeup, dressing, lighting, etc.). In such cases, we can use the recorded voice and add the synthesized talking-head on the visual channel. When the content needs to be synthesized dynamically on the other hand, we use off-the-shelve text-to-speech (TTS) engines.

1.5 An enabling technology for future access to information

With the advent of the Internet, people are starting to use the computer in a very different way. Instead of programming it to perform tasks, many use it as a medium for delivering information and entertainment. With the forthcoming 3G mobile phones as well as with the wider acceptance of 805.11b and Bluetooth wireless connectivity, small appliance-like computers are emerging that are allowing instant access to information at any time and in any place. Such instant access to information will become part of our everyday life; we will become accustomed to it and will expect it to be available in any situation, the way we expect our cell phones to work in any place.

In this context, intuitive and efficient interfaces will be a must. One cannot expect people to interrupt their tasks and spend time to navigate endless menus and cluttered screens using clumsy little buttons to access needed information. Instead the ultimate user interface will be nothing less than a perfectly natural talking-head agent who can be addressed via natural language. A trusted travel agent or a familiar company spokesperson will address you in a natural fashion in your language of choice, understand your queries and be able to make airline reservations or access your account quickly and efficiently with minimal distraction and effort from you. An added benefit of talking-heads is that they help understand voice-enabled interfaces in noisy environments (studies have shown that understanding is improved by 16% [19]). The positive effect of the visual channel on the understandability is due to people’s ability to lip-read.

1.5.1 Talking-Heads for e-learning

One of the main challenges in e-learning is to grab and direct the attention of the learner. In “real” situations, this task is performed by the teacher or the presenter. People generally need help to focus on particular aspects of the material and understand the main issues. This is best achieved by a live presentation. People are good at communicating extra-verbal information and a presenter knows exactly what he or she is talking about and will indicate to the audience what the essential aspects of the contents are and what can be safely ignored. When translated into text this type of information is generally lost. Furthermore, people when reading text, often skip paragraphs or read diagonally in order to accelerate the process. This often leads to misunderstandings.

E-learning not only encompasses remote teaching of school classes (see Figure 1.1 for a mock-up view of an e-learning session in a school setting) and business training but also includes giving instruction on how to use complex systems such as computers, software, home theaters, etc., or even more mundane appliances such as bread-makers and exercise machines. For example, complex software packages such as Turbotax (a software that helps you file your tax return) or Adobe Premiere (a software to create digital movies) have been including video clips of experts teaching how to use the software. These clips are very useful at focusing people’s attention to the main issues, directing them to particular features and making them aware of specific difficulties. They are instrumental in avoiding errors and costly calls to customer service and thus have been integral part of the products since many years, despite the high cost of producing and delivering these videos via CD-ROM’s or videocassettes. While in many cases the content of these e-courses is known in advance and does not evolve significantly, there is often still a need to upgrade and evolve it to reflect new features in a product or changes in the structure of a course. Producing new sets of videos is again costly and, worse yet, the original actors might not be on hand for the shoot, resulting in a loss of continuity or the need to re-shoot the entire set of courses. And in certain case, such as a well-known company spokesperson, it might actually be a big deal to replace the talent.

Talking-heads solve these problems because the model creation process is performed only once and the model can then be reused ad-infinitum to create new content. While the initial cost involved in creating a high-quality talking-head model might be superior to a shooting session, in the long run, the re-usability of the model makes it economically appealing. A noteworthy aspect of e-learning is that the talking-head models need to be of sufficient quality that they can be mistaken for real video. If visual or auditory artifacts are present, it might detract the user from concentrating on the presented material, or worse, in the case of the company spokesperson, it might end-up ridiculing the agent and hence be detrimental to the image of the company.

A similar area of application is expert systems. For example, a system providing automated medical advice to patients based on the input of symptoms is a prime candidate for a talking-head interface. In the medical context, the “humanness” of the automated system is key for its wide acceptance. The user must be able to trust the advice given by the system and he or she must be able to communicate effectively the symptoms. An accompanying talking-head, with the appropriate medical doctor attire and the right reassuring expressiveness, is a key enabling factor for the whole concept. See Figure 1.2 for a mock-up view of a health-care kiosk.

Another area related to e-learning is the automated delivery of news via a newscaster talking-head. The content of news is by essence dynamic and changes very often. The shooting of newscasts is an industry in itself and well-known anchor persons are very expensive hires. Talking-head can be a very effective and economical solution to such news broadcasts. Once the model is created, only the text feed is needed to continuously and automatically generate newscasts. A well known such virtual newscaster is Ananova [1]. While Ananova is clearly (and intentionally) non-realistic, it illustrates the concept of a more TV-like experience on the web.

1.5.2 Talking-Heads for customer relations (e-care)

Customer relation management (CRM) is a growing industry, which worldwide market size is forecasted to reach upward of 30 billion dollars in 2008 (source: Frost&Sullivan). Many large companies have sizeable customer relation management centers where hundreds of operators answer to customer requests. To reduce the number of human operators, companies have turned to systems that automatically answer, route and serve these calls. Typically the systems can only serve (successful completion) about half of the requests and route the difficult cases to human operators. Examples include call centers for government agencies, telephone companies, banks, etc. as well as automated check-in kiosks for airlines, hotels and car rental companies, automated cash-registers in supermarkets, etc.

While automating customer relations is a pressing goal for many companies, the danger exists that these systems, by their complexities and inefficiencies will confuse and anger customers and ultimately result in loss of business. In the case of call centers, the voice-only telephone interface is not always adequate to handle complex transactions and often involves the lengthy navigation of several sub-menus making it annoying and inefficient for the user. Recently, the introduction of new technologies such as automatic speech recognition, natural language understanding and dialog management, has improved the user interface of these automated telephone answering systems by letting the user simply state his or her request in a natural manner. While user feedback has been positive for these new systems, the voice-only interface still restricts the amount of data that can be exchanged between the user and the system. CRM service on screen-enabled terminals such as PDAs (see Figure 1.3 for a mock-up view of an e-care session on a PDA), notebook or desktop computers and kiosks are generally easier to navigate, yet they lack the human presence. By adding a talking-head to these screen-based CRM systems, one can combine the accessibility of screen displays and the human touch given by the agent. The virtual agent in this context of customer relation and assistance can help in several ways. First it welcomes the customer in a much more engaging and convincing way than simple text and logo. It then briefly describes the main options making it immediately obvious to the user what can be done. At this point “power users” might directly “click” their ways through the system to quickly access data without further interaction with the agent. For users new to the system, the agent can actually guide them through the system and react intelligently to their input and requests. The agent can help “fill-in” waiting time due to database access or slow network connection by appropriate behavior, expressions and the judicious use of standard prompts. Finally the agent might suggest other options in case the user isn’t successful in using the system, or might congratulate the user for a successful transaction and extending to him or her the company’s thanks and greetings.

1.5.3 Talking-Heads for e-commerce

While the success of e-commerce is undeniable, it hasn’t reached the widespread acceptance one might have expected. Many people use it to buy small items such as books and CD’s. It is also very successful at selling technology-related items such as personal computers and their accessories to computer-educated people. However, for more pricey items such as furniture, cars, appliances, real-estate and art, most people are still holding-off buying online. Much of this reluctance essentially stems from the fact that most people are not giving much trust to an automated system. People need to see a human face to get the impression they are making a good deal. They need to be reassured that their choice is the right one; they need confirmation that a transaction is completed, and want to be congratulated for their purchases. This type of human interaction cannot be achieved online by standard text and graphics-based user interfaces. However, as shown by subjective tests we performed [19], talking-heads can approach this level of "humanness" and, to some extent, restore the broken trust link between the user and the computer (an increase in trust of 30% has been shown when a talking-head is used versus text-only and voice-only interfaces). Another advantage in introducing this human factor in the transaction is the "pushiness" of human agents. While disliked in extreme cases, pushy salespersons are known to strike more deals. For many people their always is a "little hill" to climb when buying expensive items and they often need a little push to go over it. This psychological obstacle can only be lifted by human interaction: convincing, reassurance, flattery, etc. We have observed that talking-heads can fulfill this missing key element in e-sales transactions. Several companies such as Motorola and the real-estate company RE/MAX are experimenting with talking-heads to interact with their customers on their e-commerce web-sites.

1.5.4 The need for video-realism

In most of these applications, the degree of realism needed to achieve the desired effect is high. In fact, except for entertainment purposes, people don't expect to interact with non-realistic faces. In our experience talking-head exhibiting artifacts are disliked. In most of today's available talking-head products, artifacts in the articulation of the mouth are clearly visible. First, the way the lips move is wrong and second the mouth cavity is only superficially rendered with overly white blobs for the teeth and pure black for the rest of the cavity. Furthermore, the head movements are typically very mechanical, as are the eye blinks and eyebrows movements. The presence of these artifacts makes it obvious it is a synthetic talking-head and often makes the head look ridicule and end up distracting the user. This is clearly not acceptable for serious application such as e-commerce, e-learning and e-care. Furthermore, to improve understanding in noisy conditions via lip-reading of the talking-head, realistic lip movements are necessary.

However, there is a real danger in moving towards realistic talking-heads. When people see cartoons or synthetic head models, they do not necessarily expect a realistic behavior. Behavior being both the low-level behavior of facial elements when they are animated to produce speech or emotions and the high-level behavior of the agent in particular situations (speech related expressions, gestures, conversational cues, listening postures, turn taking, etc.). Conversely, a realistic looking talking-head will substantially raise people's expectations of its behavior. And by raising people's expectations we consequently run the risk of turning them off if these expectations are not met. On the other hand, only more realistic talking head can have a sufficient effect on users that they contribute to enhance the interface. This effect is illustrated in Figure 1.4 .

1.6 Previous Work

Several different techniques exist for modeling the human head, achieving various degrees of realism and flexibility. Most approaches use 3D meshes to model in fine detail the shape of the head [11][18][20]. These models are usually created using advanced 3D scanning techniques such as a CyberWare® range scanner. Some of them include information on how to move vertices according to physical properties of the skin and the underlying muscles [18]. Another approach is to fit a generic 3D model to image data by solving a dynamic system incorporating either optical flow and edge constraints [10] or stereo disparity maps [14]. A technique for matching a 3D model to a single photograph is proposed in [6] where the 3D model is deformed in the directions of several principal components to fit the given 2D image. To obtain a natural appearance, these 3D models typically use a 2D image of the subject that is texture-mapped onto the 3D model. As long as these 3D models do not incur any plastic deformations but only rigid movements of the entire head, images of high quality can be produced (providing the 3D shape has been captured with high precision). However, when plastic deformations occur, the shape of some polygons of the 3D model will change while the pixels from the 2D image that have been allocated to these polygons will remain the same, resulting in "stretch" and "squeeze" artifacts. While this is tolerable for small deformations of surfaces with little texture, the animation of a human mouth articulating speech, because of the hundreds of little wrinkles on the lips and the texture of the surrounding skin, results in unnatural appearances and a "synthetic look".

An alternative approach is based on morphing between 2D images. These techniques can produce photo-realistic images of new shapes by interpolating between two existing shapes. Morphing of a face requires precise specifications of the displacements of many points in order to guarantee that the results look like real faces. Most techniques therefore rely on a manual specification of the morph parameters [23]. Beymer et al. [3] and Bichsel [4] have proposed image analysis methods where the morph parameters are determined automatically, based on optical flow. While this approach gives an elegant solution to generating new views from a set of reference images, one still has to find the proper reference images. Moreover, since the techniques are based on 2D images the range of expressions and movements they can produce is rather limited. Ezzat et al. [12][13] have demonstrated a sample-based talking head system that uses morphing to generate intermediate appearances of mouth shapes from a very small set of selected mouth samples. While morphing generates smooth transitions between mouth samples, this system does not model the whole head and does not synthesize head movements and facial expressions. Another class of systems are based on machine learning techniques. For example Brandt et al. [5] demonstrated a system that directly learns a mapping between audio and lip position. While the idea of bypassing the phonetic representation of speech is attractive, the system is only able to drive a deformable mesh and hence is not capable of producing realistic lip animation.

Recently there has been a surge of interest in sample-based techniques (also referred to as data-driven) for synthesizing photo-realistic scenes. These techniques generally start by observing and collecting samples that are representative of the signal to be modeled. The samples are then parameterized so that they can be recalled at synthesis time. Typically samples are processed as little as possible to avoid distortions. A talking-head synthesis technique based on recorded samples that are selected automatically has been proposed in [7]. This system can produce videos of real people uttering text they never actually said. It uses video snippets of tri-phones (3 subsequent phonemes) as samples. Since these video snippets are parameterized with phonetic information, the resulting database is very large. Moreover, this parameterization can only be applied to the mouth area, precluding the use of other facial parts such as eyes and eyebrows that are carrying important conversational cues.

Other researchers explored ways of sampling both texture and 3D geometry of faces [16][21], producing realistic animations of facial expressions. These systems use multiple cameras or facial markers to capture the 3D geometry and texture of the face in each frame of video recordings. Deriving the exact geometry in the mouth and mouth cavity areas as they undergo plastic deformations remains, however, difficult. Extensive manual measuring of the images in [21] and a complex data capture setup involving six cameras and hundreds of facial markers in [16], are required resulting in a labor intensive capture process, especially when tens of thousand of video frames have to be analyzed, making them unsuitable for synthesis purposes.

Other systems worth mentioning are the ones that made their ways into products. Pulse3D [22] is releasing a talking-head player called Veepers. Veepers features a very small footprint player as well as very small file size models, making it well suited for low bandwidth situations. Also the model can be created from a single photograph, simplifying the task of model creation. While the animated talking-heads in Veepers are attractive in their simplicity and quickness to download, they lack natural lip articulation, expressions and head movements. Other commercially available systems exist from LIPSinc [17] and Anthropics [2]. Their talking heads have a more synthetic look but a more extensive animation of head movements, but they share with Veepers the poor lip synchronization. Lots of development work has gone into these systems to make them easy and quick to download, install and integrate into such programs as email clients. Indeed much of the hurdles in having talking-heads technology widely accepted eventually reside in solving these type of issues.

1.7 Our solution

Our aim is to approach video realism and this work is about building a system that is capable of producing talking-head animations that approach the quality of a real video taken from a person speaking. Contrary to most commercial implementations, which prioritize speed over quality, we built our system from the ground-up for uncompromised quality.

The basic reason for that choice stems from the original goal of the project, which was to build an ultra low-bandwidth video-telephony system. The model of the speaking person and the parameters of the speech animation were to be extracted at the sender terminal via image analysis and transmitted to the receiver where they would be used to re-synthesize a video animation of the speaker. Of course in this context, the appearance of the users must not be changed so they can be recognized and not ridiculed. But because video-telephony didn't have the widespread success that would have warranted further investments in research, the project was instead redirected towards improving computer user interfaces.

Our original choice to aim for video-realism led us to use a sample-based (or data-driven) approach. In this approach, models of objects are created not based on their physical properties but rather by capturing their appearance and behavior into samples that can be later reassembled to produce new appearances and behaviors. In our system we build the talking-head models by capturing images from a person via a video camera. The compelling advantage of sample-based methods is that they completely bypasses physical modeling of objects. While, for simple rigid objects, physical modeling can produce rendered scenes with striking realism, the highly complex plasticity and dynamics of the talking face would require such a large amount of modeling that it is still by today's standards infeasible (if not for technical reasons, economical reasons certainly prohibit this approach). Several attempts have been made in that direction and while they provided interesting insights into how to model skin tissues and muscles [18], they did not result in compelling animations (in particular speech animations). The sample-based approach is the only way (at least at current technology level) to build a talking-head model of a person that, when speech-animated, is of sufficient quality that it can be mistaken from a video recording of that person.

An essential limitation of the sample-based approach in modeling talking-heads (and objects in general) is that the view under which the object can be seen is limited by parallax effects. The images as they are captured by a camera are formed by a projection of the 3D world onto the camera's CCD. Hence, in principle, only the exact same view can be reproduced. In practice, fortunately, the important parts of the face, such as the mouth, eyes and forehead are fairly planar and speech production is generally performed in a frontal fashion, the speaker facing the person he or she addresses. This makes it possible to reuse image samples of these facial parts in a different view as the one under which they were captured. In our experience a range of ±15 degrees from the original view results in almost no visible artifacts. This makes it possible to implement head movements, a necessary feature for realistic animations, while using a sample-based approach. An illustration of the relationship between flexibility in animation and realism is shown on Figure 1.5

Comparing with existing systems, our approach relates most closely to the one of Bregler et al. in their implementation of Video-Rewrite [7]. What differentiate ourselves the most from Video-Rewrite is the higher degree of 3D modeling as well as our finer-grained audio-visual unit selection module. This provides our system with better head movements and livelier lip animations with a smaller database. A subjective comparison table of several speech-enabled talking-head systems that are either available as products or have been presented in the literature is shown on Figure 1.6 .

1.8 References

[1] Ananova. http://www.ananova.com/video/

[2] Anthropics. http://www.anthropics.com/

[3] Beymer, D., Poggio, T., “**Image Representation for Visual Learning”**, Science, vol. 272, pp.1905-1909, 28 June 1996.

[4] Bichsel, M., “**Automatic Interpolation and Recognition of Faces by Morphing”**, Proc. 2nd Int. Conf. on Automatic Face and Gesture Recognition, pp.128-135, IEEE CS Press, 1996.

[5] Brand, M., “**Voice Puppetry”,** ACM SIGGRAPH, 1999.

[6] Blanz, V., Vetter, T., “**A Morphable Model for the Synthesis of 3D Faces”**, Proc. SIGGRAPH'99, pp.353-360, ACM SIGGRAPH, July 1999.

[7] Bregler, C., Covell, M., Slaney, M., “**Video Rewrite: Driving Visual Speech with Audio”**, ACM SIGGRAPH, 1997.

[8] Cohen, M.M., Massaro, D.W., “**Modeling Coarticulation in Synthetic Visual Speech”**, Models and Techniques in Computer Animation, Springer Verlag, 1993.

[9] Cosatto, E., Graf, H.P., “**Sample-Based Synthesis of Photo-Realistic Talking-Heads”**, IEEE Computer Animation, 1998.

[10] DeCarlos D. and Metaxas D., “**Optical flow constraints on deformable models with applications to face tracking”**, International Journal of Computer Vision, Vol. 38, No. 2, pp99-127, 2000.

[11] Escher, M., Magnenat-Thalmann, N., “**Automatic 3D Cloning and Real-Time Animation of a Human Face”**, Proc. of Computer Animation, pp.58 - 66, IEEE Computer Society 1997.

[12] Ezzat, T., Poggio, T., “**MikeTalk: A Talking Facial Display Based On Morphing Visemes”**, IEEE Computer Animation, 1998.

[13] Ezzat, T., Geiger, G., Poggio, T., “**Trainable Videorealistic Speech Animation”**, acm Transactions on Graphics, proceeding of SIGGRAPH 2002, vol. 21, n. 3, pp. 388-397, July 2002.

[14] Fua, P., “**Reconstructing complex surfaces from multiple stereo views”**. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1078--1085. IEEE Computer Society Press, June 1995.

[15] Graf, H.P., Cosatto, E., Potamianos, G., “**Robust Recognition of Faces and Facial Features with a Multi-Modal System”**, IEEE Systems, Man and Cybernetics, pp. 2034-2039, 1997.

[16] Guenter, B., Grimm, C., Wood, D., Malvar, H., Pighin, F., “**Making Faces”,** ACM SIGGRAPH, pp. 55-66, 1998.

[17] Lipsinc. http://www.lipsinc.com/

[18] Lee, Y., Terzopoulos, D., Waters, K., “**Realistic Modeling for Facial Animations”**, Computer Graphics (SIGGRAPH '95 Conf. Proc.), pages 55--62, Aug. 1995.

[19] Pandzic, I., Ostermann, J., Millen, D., “**User evaluation: Synthetic talking faces for interactive services**”, The Visual Computer, Volume 15, Issue 7/8, pp 330-340, Nov 04, 1999.

[20] Parke, F. I., Waters, K**., “Computer facial animation**”, Wellesley MA: A K Peters, 1996.

[21] Pighin, F., Hecker, J., Lichinski, D., Szeliski, R., Salesin, D.H., “**Synthesizing Realistic Facial Expressions from Photographs”**, Proc. of SIGGRAPH'98, pp.75-84, ACM SIGGRAPH, July 1998.

[22] Pulse3D. http://www.pulse3d.com

[23] Seitz, S.M., Dyer, C.R., “**View Morphing”**, Proc. SIGGRAPH'96, pp.21-30, ACM SIGGRAPH, July 1996.

1.9 Figures

Figure 1.1 Mock-up of an e-learning session in a classroom.
Figure 1.2 Mock-up of a health care kiosk.
Figure 1.3 Mock-up of an e-care session on a hand-held device.
Figure 1.4 Realism raises expectations.
Figure 1.5 Realism versus flexibility in animation.
Figure 1.6 Subjective comparisons of talking-head systems.

Figure 1.1. Remote teaching, e-learning mock-up. A teacher’s face and demeanor captivates and engages the student to a higher degree than traditional interfaces, resulting in more efficient and rewarding e-learning sessions.

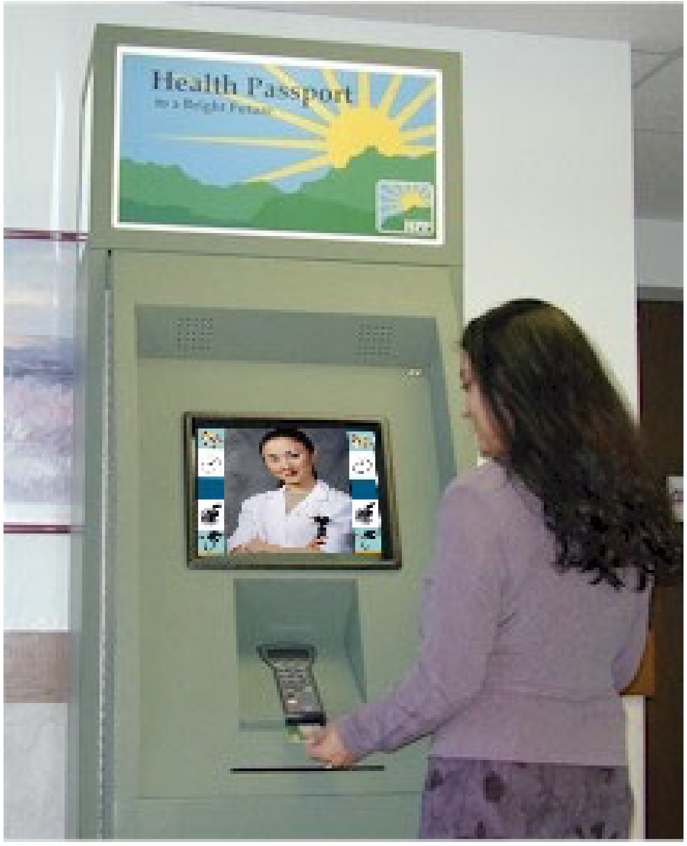


Figure 1.3. Mock-up of an automated healthcare kiosk. The user inserts her health card which contains her medical history and the talking head interacts with her to get the symptoms and issue the diagnostic. Trust in the system is much improved by the presence of the virtual doctor.

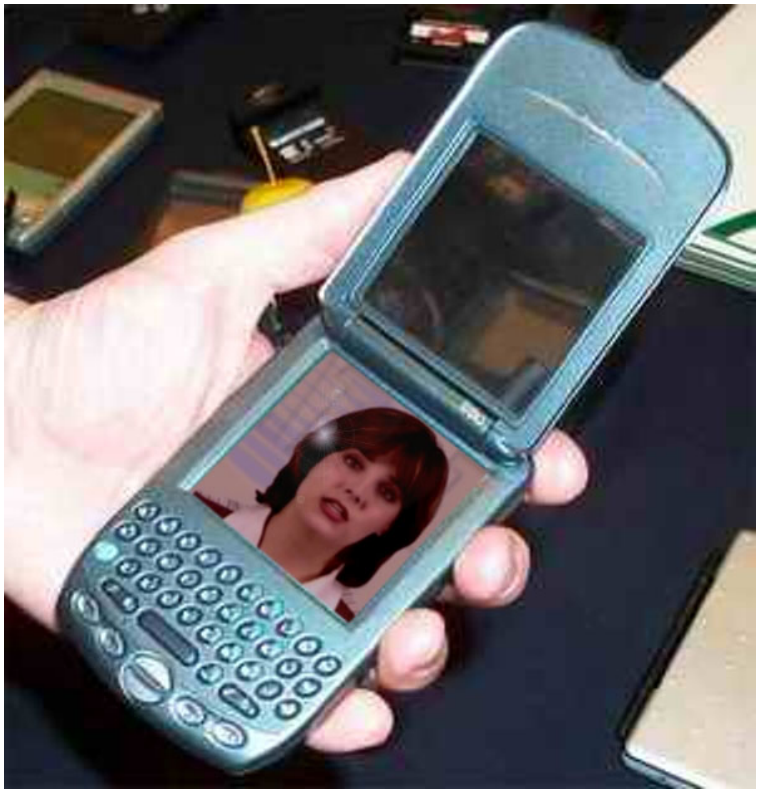


Figure 1.4. e-care session mock-up. A customer accesses his account through a virtual agent who assists him. The agent also provides brand awareness by emulating a well-known spokesperson. The visual presence of the agent makes the interaction more human and also provides for superior understanding in noisy conditions (lip-reading).

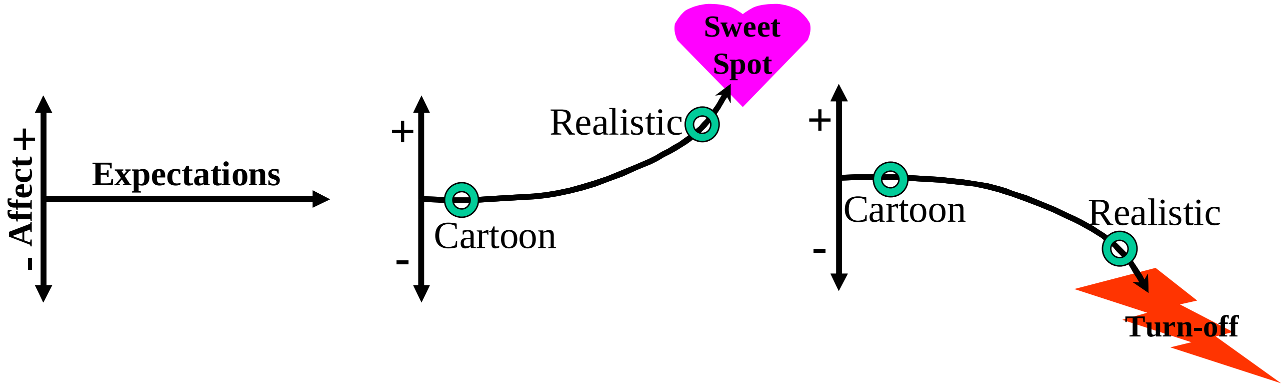


Figure 1.5. Talking head’s affect versus expectations. While cartoonish or synthetic-looking talking-head do not elicit large expectations in terms of behavior, more realistic looking faces produce more pronounced reactions in people. A positive affect is created when both appearance and behavior are realistic. On the other hand a realistic appearance (that raises expectations) combined with a synthetic behavior will likely turn-off users.

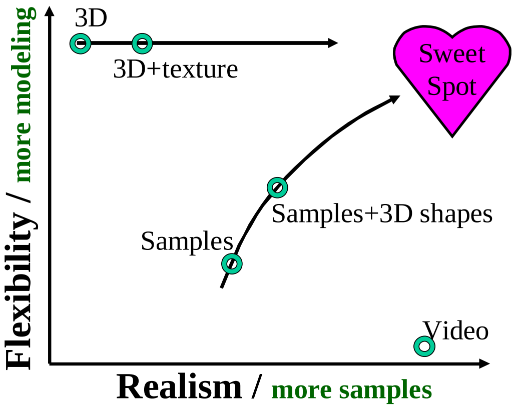


Figure 1.6. Flexibility versus Realism for different approaches. While recorded video offers the most realism, it completely lacks flexibility: what has been recorded is what you can show. On the other hand, a fully model-based 3D talking-head is completely flexible (can be shown in any view, under any lighting conditions) but, because of simplifications (due to the enormous complexities in modeling the human face) lacks realism. The sample-based approach allows combining the realism of video with the flexibility of the 3D modeling. As more and more 3D information is put into the sample-based model, it reaches the sweet spot of perfect realism and full flexibility. The 3D approach also reaches the sweet spot as more and more texture information is added to the model.

	Anthropics [1]	Veepers [2]	M.I.T. [3]	VideoRewrite [4]	AT&T VTTS (this work)
Realism	3	1	9	7	10
lip synch	1	1	8	7	10
Expressions	5	1	1	1	8
synthesis speed	10	10	1	6	8
model creation (easy=10, hard=1)	5	8	3	1	1
Installation, use (easy=10, hard=1)	10	10	1	1	1
bandwidth	10	10	1	1	1

Figure 1.7. A subjective comparison of speech-enabled talking-heads. The scale goes from 1 (worse) to 10 (best). While productized talking-heads excel in easy installation, model creation and usually provide streaming capabilities of animation parameters making the delivery very low bandwidth, they typically lack realism, both in terms of lip-synch and expressions. Conversely, realistic talking-heads typically require larger model and longer model creation time and thus have not been prime candidate for productization yet.

[1] Anthropics. http://www.anthropics.com/

[2] Pulse3D. http://www.pulse3d.com

[3] Ezzat, T., Geiger, G., Poggio, T., “**Trainable Videorealistic Speech Animation**”, acm Transactions on Graphics, proceeding of SIGGRAPH 2002, vol. 21, n. 3, pp. 388-397, July 2002.

[4] Bregler, C., Covell, M., Slaney, M., **Video Rewrite: Driving Visual Speech with Audio**, ACM SIGGRAPH, 1997.

CHAPTER 2

CREATION OF THE HEAD MODEL (ANALYSIS)

Table of Content

2.1 Introduction 16
2.2 Overview 17
2.3 Modeling 17
  2.3.1 A decomposition into facial parts 17
  2.3.2 Static 3D shape 18
  2.3.3 Soft Edges 18
  2.3.4 Dynamic Textures 19
2.4 Recording 19
  2.4.1 Overview 19
  2.4.2 Phonemes 19
  2.4.3 Coarticulation 20
  2.4.4 Visemes 20
  2.4.5 Defining a Corpus 21
  2.4.6 Audio-Video Recordings 21
  2.4.7 3D scanning 22
2.5 Phonetic Labeling of Audio Data 22
2.6 Image Analysis 23
  2.6.1 Introduction 23
  2.6.2 Locating Face and Measuring Facial Features 23
  2.6.3 Pose Estimation 27
  2.6.4 Sample Extraction and Normalization 30
  2.6.5 Sample Parameterization 30
2.7 References 32
2.8 Figures 34

2.1 Introduction

The creation of human face models that are both photo-realistic and can be animated is a challenging problem. The human face is highly complex: the convoluted geometric shape of the skull, the many forms the lips can exhibit during speech, the subtle variation in the rise of the eyebrows, the precise movement of the eyes, the rich appearance of the skin with its countless number of creases, grooves and wrinkles, as well as subtle variations in texture and color, the fuzziness of hair. All contribute to making it extremely difficult for a computer to synthesize an absolutely natural-looking face. To make matters worse, we are all highly trained from birth to read the subtlest expressions and interpret even the tiniest facial movements. Hence the slightest mistake made in reproducing a human face will be immediately noticed. In fact, to date and to the best of our knowledge, a *Turing Test* has not yet been passed by a facial animation program.

Ever since Ekman and Friesen in 1976 developed an objective technique for measuring facial movements [11], several researchers have strived to develop an analytical model of the face that can produce photo-realistic animations. These models typically start with a set of three-dimensional polygons that define the shape of the head and its various parts. An 'intelligent' layer is then defined to emulate facial actions. For example, Terzopulos et al. [28] have presented a head model in which the skin is modeled with several layers of spring lattices (to account for the different elasticity of skin and fatty tissues) and muscle actions are applied that simulate wrinkling. Kalra et al. [25] perform facial animations using free form deformation to deform volumetric objects by manipulating control points arranged in three dimensional cubic lattices. For comprehensive surveys of these techniques, see [34] and [36]. However, in a departure from the pure analytical model, many of these systems use a captured image of the face that is texture-mapped onto the polygon mesh. This shortcut avoids the complication of modeling the surface of the skin (pores, facial hair, dimples, oiliness, etc.). A trade-off of this shortcut, however, is that this single texture image is constantly stretched to accommodate skin movements and deformations such as when the jaw rotates as the mouth opens. This image stretching substantially alters the visual quality of the skin texture and results in an unnatural appearance.

The other difficulty in the analytical approach is to model the complex behavior of the mouth as it articulates speech. Modeling the complex dynamics of speech articulation not only requires a complete understanding of the mechanical aspects of the vocal apparatus but also a deep understanding of how the brain has adapted to it to produce speech in an optimal way. Most systems that use this approach employ simplified coarticulation models [1] that generally produce very contrived-looking lip animations. Hence, despite extensive efforts and fine-grained modeling attempts, the ultimate goal of a perfect facial animation that can be mistaken from a real video remains elusive. Such animations always have an eerie look and one always can tell it is computer generated.

A way to bypass the complexities of such analytical modeling and avoid its shortcomings is to learn a model from recorded data. In our approach, we record video and audio of a person articulating speech and producing facial movements. This way, we capture both the rich appearance of the mouth as it articulates and its complex behavior in time. We then use image and audio analysis techniques to create a database of labeled image samples that is later used to create new animations.

The number of samples needed to synthesize animations depends on the type of target animations. Unconstrained speech will require all possible appearances of the mouth articulating words. Ezzat et al. [13] have proposed a technique that produces speech animations from a small set of mouth shapes that corresponds to salient stationary appearances taken by the mouth during speech. These salient mouth shapes are often called visemes. Morphing between two visemes then produces transitory shapes. However this is only an approximation of real transitory movements and in the case of speech production, it typically results in loss of lip synchronization with the auditory part. Hence the need to record not only the stationary phase of phonemes but also all transitions in-between. This typically results in a large number of samples.

Because in a sample-based approach, all information available to the system to synthesize animations resides in the collected samples, great care has to be taken when recording these samples. For example, in a talking-head application, where speech production is the goal, the mouth has to be sampled in sufficient number to cover all possible speech postures. Furthermore, it is very important that the samples are captured under similar conditions than the desired target animations, that is, the subject should be speaking in a natural, unconstrained way during the recording process, so that the samples are characteristic of such a condition. If the subject is forced to over-articulate a set of phonemes, or if the subject is mechanically constrained as to avoid head movements (which facilitates the extraction of normalized mouth image samples), the captured samples won't look natural and will not be usable to recreate realistic speech animations. Allowing the recorded subject to produce speech in a natural fashion results in an unpredictable placement of the facial parts on the recorded image. To handle this large number of unconstrained samples, manual labor is not feasible and automated image analysis techniques are necessary.

2.2 Overview

This chapter is organized as follows. In section 2.3 we introduce a modelization of the head that supports independent sample-based animations of several facial parts in 3D. The key aspect of this model is that it allows sample-based rendering of facial parts in 3D by allowing multiple texture samples to be used on the same wireframe and by providing a way for facial parts to be merged seamlessly into a whole head.

In section 2.4, we describe the process of recording image samples of the head in unconstrained conditions during speech production, as well as generating emotions. In particular, we discuss the creation of a viseme-balanced corpus of speech. Finally the recording of the 3D shape of the head is covered.

In section 2.5, we briefly cover the audio analysis of the recordings. This analysis provides the basis for labeling video samples with a phonetic context which can be used to select proper samples in the synthesis phase (see Chapter 3).

In section 2.6, we describe the process of analyzing the video in order to extract normalized image samples of facial parts. This process starts by locating facial parts on the image samples and estimating the 3D pose (angles and position) of the head. Then, using the estimated pose of the head together with the 3D model of facial parts such as mouth and eyes, image samples are extracted and normalized. Each sample is finally labeled and stored in a database.

2.3 Modeling

2.3.1 A decomposition into facial parts

When using image samples to create an animatable model of the face, it is important to identify separate facial parts that can be sampled and animated independently of other parts. If the entire head were to be modeled as one object, one would have to sample face images with all appearances of mouth *combined* with all appearances of eyes and eyebrows. This would result in an unnecessary high number of samples that would be extremely difficult to capture in their entirety. To avoid this "curse of dimensionality", we divide the face into a set of facial parts that are modeled independently. Each facial part is made of a 3D wireframe model that describes its 3D shape and a set of texture maps that describes its possible appearances.

A first step in modeling the face is to note that facial movements of the upper part of the face are generated independently from the movements of the lower part. Rising of the eyebrows, for example, is not correlated to specific movements of the lower part of the mouth and hardly affects the cheeks, lips and chin. Conversely, movements of the lips do not affect the eyebrows. Movements of the eye globes are also independent from those of the surrounding skin and eyebrows. Following research by Munhall and Vatikiotis-Bateson [30] and after studying, how facial expressions are generated by humans [11], and how they are depicted by artists and cartoonists, we divide the head into two main overlapping parts. The upper part comprises the eyebrows, part of the forehead, the eyes and the upper part of the cheeks. The eye globes are modeled separately to allow flexible animation of eye movements. The lower part of the head comprises the cheeks, mouth and chin. It is important that the mouth part encompasses not only the lips but also the cheeks and chin. Movements of the chin and cheeks are highly correlated to those of the lips and they need to be modeled as a single part. The remaining parts of the head, including nose, ear, hair and surrounding skin patches are grouped into a single part called "base-face". Figure 2.1 illustrates this decomposition into facial parts.

2.3.2 Static 3D shape

To capture facial parts samples and to be able to show them under different viewpoints, one needs to capture the 3D shape *together* with the texture. But, for several reasons, ranging from the difficulty of capturing 3D shape in real-time to memory constraints and synthesis complexity, we do not capture the shape information for each and every image sample of a facial part. Guenter et al. [18] and Kalberer et al. [24] are exploring this very promising approach but many hurdles remain such as database size and real-time rendering. Instead, we define one single "average" shape for each facial part. Figure 2.2 illustrates this concept.

During the modeling process, these 3D shapes are used to extract (un-project) pixels from recorded images and re-project them into normalized samples. Such normalized samples are shown in Figure 2.2 (a and f). We use a frontal pose (all angles set to zero) for the normalization re-projection. More details on this can be found in section 2.6.4.

The inverse action is applied during synthesis and is often called texture-mapping. The normalized samples are re-projected from the normalized pose into a new pose. However, because the "average" shape does not exactly match the actual 3D shape of a given sample of that facial part, the re-projection will induce distortions. The key is to limit the range of viewing conditions, both when capturing the samples, and when using them for animation. This way, the distortions induced by re-projecting the samples are kept to a minimum. In the realm of talking-heads, the limitations on the viewpoints are not terribly important. As a talking-head speaks it rarely turns away more than 10 degrees from the camera and thus exhibits a frontal view most of the time. Under these conditions, our approach provides superior realism over traditional single-texture approaches. In Figure 2.3 , a simplified "average" 3D shape of the mouth comprising only 62 polygons is shown to produce acceptable results under different viewpoints for both an open mouth and pursed lips. The viewpoints used in the figure are the maximum allowable rotation around the x and y axis. Beyond these values, distortions become noticeable.

2.3.3 Soft Edges

All normalized samples of a given facial part are aligned and of the same size. Hence, we define a *single* alpha-mask that is used to seamlessly combine any sample of the facial part with the base face (Figure 2.2 b and g). An alpha-mask is a bitmap of the same size as the normalized samples with each pixel having a value between 0 and 1. On Figure 2.4 , black areas of the alpha-mask represent a zero value, while white areas represent a one value. The texture value $S_\mathbf{p}$ of a sample at pixel **p**(x,y) is combined with the texture $D_\mathbf{q}$ of a base-face' pixel **q**(x1,y1) using the alpha value $\alpha_\mathbf{p}$ of the alpha-mask at pixel **p**(x,y) to produce a new pixel $D'_\mathbf{q}$:

$$D'_\mathbf{q} = S_\mathbf{p} \cdot \alpha_\mathbf{p} + D_\mathbf{q} \cdot (1 - \alpha_\mathbf{p})$$

Equation 2-1

This operation is commonly known as alpha-blending and is supported on hardware by most video cards. Hence, no penalty is associated with this operation in terms of compute cycles. While the (x,y) coordinates of pixel **p** are integers, the destination pixel **q** 's coordinates are generally real values and bi-linear interpolation is used to compute the final values sent to the frame-buffer pixels.

We create these facial parts' alpha-masks by defining two contours: An inner contour *IC* is drawn around the area of the facial part that will be visible. An outer contour *OC* is then drawn around *IC* defining the area of the facial part that will not be visible. Outwards of *OC*, the mask's pixels are set to zero (fully transparent) and inwards of the inner contour *IC*, the mask's pixels are set to one (fully opaque). Between the contours, the mask is partially transparent to the sample:

$$\alpha_{\mathbf{p}} = \begin{cases} if \quad is\_inside(IC,\mathbf{p}) \rightarrow 1 \\ elseif \quad is\_inside(OC,\mathbf{p}) \rightarrow \dfrac{dist(OC,\mathbf{p})}{dist(OC,\mathbf{p}) + dist(IC,\mathbf{p})} \\ else \rightarrow 0 \end{cases}$$

Equation 2-2

Where *is_inside*(*C*,**p**) is true when the point **p** is enclosed by the contour *C*. And where *dist*(*C*,**p**) is the shortest distance from the point **p** to the contour *C*.

2.3.4 Dynamic Textures

Instead of modeling a facial part with a deformable 3D wire-frame and a *single* texture, our approach is to model it by a *fixed* 3D wire-frame and multiple textures. While the former has the advantage of arbitrary viewpoints and the drawback of poor photo-realism, the latter has the advantage of photo-realism and the drawback of limited viewpoints.

To create animations, samples are loaded in sequence as facial parts textures. Each facial part has a database of all needed samples to synthesize relevant animations. For example, the eye part not only includes a sample for an open and a closed eye, but also all the transitory positions in-between. Hence an animation of an eye-blink is created by sequencing samples from an open to a closed eye. While the eye requires in the order of tens of samples, animating the mouth requires two orders of magnitude more samples. To recreate speech, all possible appearances of the mouth need to be stored as samples, not only the so-called visemes (key shapes of the mouth) but also all transitions in-between. Recording, storing and using these samples in an efficient way to produce photo-realistic animations is a complex task. The following section details the recording process.

2.4 Recording

2.4.1 Overview

In this section, we describe the process of recording audio-video footage that will be later processed to collect facial part samples. This process starts by defining a script that will be used to direct the talent. Most of the difficulty is in collecting speech samples and this section will focus on that task. We first introduce the concepts of phonemes and visemes that are the driving forces in the creation of the script. Then we will describe the recording session and finally the recording of the 3D shape of the talent's head.

2.4.2 Phonemes

A phoneme is an atomic unit of speech sound. And each sound is produced by a distinctive articulatory behavior. The following refers to the production of present-day English but most of the principles described can be applied to other languages. Phonemes can be divided into consonant and vowels.

In producing vowels, the flow of air from the lungs through the vocal tract is not impeded (other than by the vibrating glottal chords) and the various sounds are produced by changes in the vocal chords and the shape of the mouth cavity. In modern day English, the length of a vowel is not a distinction factor. Vowels can be classified by the jaw position, the point of articulation, the shape of the lips, etc.

To produce consonants, the flow of air from the lungs is stopped or impeded. Consonants can be voiced or voiceless, indicating whether the vocal chords are vibrating or not during their production. Consonants can be further classified by the manner of articulation. The following articulation can be found in present-day English: stops (flow of air stopped: p,b,t [unvoiced], and d,k,g [voiced]); affricates (flow of air stopped, then slowly released to produce a friction sound: c,j); fricatives (flow of air slowed to produce a friction sound, f,v,th,s,z,sh,h); nasals (flow of air forced through the nose: m,n,ng); resonant (flow of air modulated or impeded by the tongue or the lips: l,r,w,y). Figure 2.5 shows the various parts of the human vocal tract involved in speech production.

There are several lists of phonetic units that have been widely used in speech recognition research, e.g. the DARPABET, TIMIT, CMU, etc. We use the DARPAbet because the labeler and text-to-speech synthesizer we use support this phonetic alphabet. The full list of this phonetic alphabet is shown in Figure 2.6 . Some phonemes represent an aggregate of two sounds and are called diphthongs. For example, the word ‘boy’ translates into ‘b’-‘Y’ with the phoneme ‘Y’ represents the ‘oy’ sound. Because diphthongs often involve two distinct postures of the vocal apparatus, we approximate them by two basic phonetic units. In our example, ‘boy’ thus translates into ‘b’-‘ao’-‘y’. The total number of phonemes we use after diphthong breakup is 36.

2.4.3 Coarticulation

As shown on Figure 2.5 , the articulation of phonemes involves visible parts of the face, namely the lips, tongue and the oral cavity (which can be expanded by dropping the jaw). Hence a mapping can be derived from the shape of the mouth to the phoneme being articulated. When we speak, however, we always co-articulate phonemes, because the speed with which our articulatory apparatus can assume the position and shape needed to articulate a given phoneme is not infinite. Hence, as we quickly articulate sequences of phonemes to form speech, the vocal apparatus acts as a low-pass filter. But the brain has been able to adapt to these mechanical limitations. Some phonemes can be ‘under-articulated’ and still produce an acceptable sound. So, for example, as we pronounce the word ‘stew’, the lips start pursing for the pronunciation of the ‘ew’ sound during the pronunciation of the ‘st’ sound which is hardly affected by it. Figure 2.7 illustrates the effect of coarticulation.

Coarticulation allows us to speak much faster than if we would simply produce each phoneme separately. On the other hand, coarticulation makes it more difficult to lip-read. Only few people are able to infer words from only looking at lips movements, without hearing any sound. And they often resort to contextual cues to disambiguate. Similarly, it would be naïve to label mouth samples with only the current phoneme being pronounced. We take coarticulation effects into account by always considering a phoneme in a phonetic context and not alone. Coarticulation typically extends to 300ms of the current phoneme.

But again, instead of trying to model coarticulation analytically, we make use of the recorded mouth samples. By labeling each mouth sample with its phonetic context, we are able to find optimal arrangements of mouth sequences that account for coarticulation. Chapter 3 explains this approach. Additionally, our approach captures the specifics of the person being recorded and the particular way he or she speaks.

2.4.4 Visemes

While most phonemes produce different appearances of the mouth, some classes of phonemes can be identified that produce similar appearances of the mouth. These classes are called visemes. For example, whether a consonant is voiced or voiceless only affects the vocal tract and doesn’t change the visual appearance of the mouth. Voiced and voiceless stops ‘b’ and ‘p’ produce very similar appearances, so do the voiced and voiceless fricatives ‘v’ and ‘f’. Clustering can be more or less aggressive and Figure 2.8 shows how we define five different hierarchical level of clustering. From 36 phonemes, we first cluster into 25 visemes; the next clusters contain, respectively, 20, 18, 10 and 7 visemes. We arrived at this clustering mostly by studying how the vocal apparatus articulates phonemes. We tried to run clustering algorithms over recorded normalized samples of the mouth using, as metric features, the measured mouth width and height as well as principal components of the mouth samples. However this approach proved unfruitful because we couldn't effectively un-coarticulate the image samples to isolate phoneme groups.

2.4.5 Defining a Corpus

When recording the talent, we want to make sure we capture most of the possible shapes the mouth can assume during speech. For this purpose we define a corpus of text that will 'exercise' the talent's mouth to assume most of these possible shapes. To account for coarticulation we consider three consecutive phonemes (tri-phones) and their viseme equivalent (tri-visemes). Figure 2.9 shows how various corpuses of text cover tri-viseme sequences. We consider 6 different corpuses consisting of single words dictionaries of different sizes and collections of sentences. The largest dictionary (200'000 words) is assumed to contain most existing tri-visemes. Figure 2.10 relates the size of the corpuses to how much of existing tri-visemes they contain. The selection of a corpus is a trade-off between size, complexity and coverage. We have to keep in mind that the selected corpus needs to be spoken by a talent. Hence it is best to avoid complex or little used words that might be mispronounced by the talent. We also try as much as possible to minimize the size of the corpus so as to make the recording session shorter. It wouldn't be practical to ask the talent to pronounce 200'000 words! Hence we settle for a corpus of 300 short sentences. These sentences were constructed by A. Syrdal [43] to cover most English diphones (a transition from one phoneme to the next). Hence they offer a good compromise between size and viseme coverage as shown on Figure 2.10 . This corpus results in about 80K frames when the video is sampled at 60 frames per second.

2.4.6 Audio-Video Recordings

Because we strive for photo-realism, we try to record video images in good conditions with high quality equipment. Also, to enable accurate and automated phonetic annotation of the audio track, we record in a silent room with a high-quality directional microphone and with the recording equipment in a separate room. Figure 2.14 shows several pictures taken during the recording session. To ensure a frontal position of the face and to facilitate the job of the talent, we use a teleprompter.

Once the corpus has been recorded on digital videotapes (D1, DV), it needs to be digitized, organized and the audio needs to be phonetically labeled. To support a high compression ratio while keeping the quality of the images, we opted for the MPEG2 standard for video compression [19][10]. MPEG2 encoder/decoder source code is readily available, making the integration of the software much easier and more portable. Also MPEG2 is now widely used in the DVD industry and hence a lot of affordable solutions allow real-time encoding of video sequences. Talking-head sequences typically contain a high degree of temporal redundancy that can be detected and compressed via MPEG2 codecs, achieving higher ratio of compression.

Compression schemes that make use of temporal redundancy typically do not support random access to individual video frames. In particular, MPEG2 is defined as a bitstream and does not provide a direct way to access a given frame within a sequence. We therefore developed an original solution to allow random access. The first time an MPEG2 file is opened, our software parses the entire file and creates an index giving for each frame within the sequence, the list of frames (and their position in the bitstream) that need to be decoded in order to decode that frame. An example of such an index is given in Figure 2.11 .

Using a tape-deck, we proceed to manually locate recorded sentences of our corpus on the video tape and encode them into separate computer files. For each sentence three files a created:

- one MPEG2-encoded video file
- one uncompressed PCM audio file.
- one ASCII text file containing the exact textual transcript of what is said in the audio file.

The text transcript files cannot be directly created from the recording script. Pronunciation and general reading errors that were not caught during the recording process (for which the talent has repeated the sentence) need to be removed at this stage. Either the sentence is discarded from the corpus or in certain cases the mispronounced words can be changed in the text file and, if necessary, its phonetic transcript added to the pronunciation dictionary (used for automatic phonetic labeling of the audio corpus; see section 2.5).

2.4.7 3D scanning

Part of the recording process is the capture of the 3D shape of the head and of the various facial parts. Traditionally, facial meshes have been constructed manually from two or more views of the face, or by manually digitizing plaster casts. This was a laborious and error prone process. Fortunately, capturing the three-dimensional shape of faces has recently become easier thanks to many affordable systems available on the market. 3D Laser scanners, such as the one manufactured by Cyberware, Inc. [6] circle around a person's head to acquire high resolution, regularly sampled range and reflectance information. High accuracy (0.5 mm) can be achieved with these types of systems and they have been used extensively for capturing faces [27].

Other methods reconstruct the 3D shape using images of a subject captured simultaneously from different viewpoints [16]. Companies such as InSpeck Inc. [22] and Eyetronics Inc. [12] offer 3D scanning products based on calibrated apparatus that project structured light on the subjects face. These systems are typically less accurate than laser-based scanner with accuracies ranging from 1 to 2mm, but are more portable and less expensive.

We've had the opportunity to use a Cyberware Head Laser Scanner to capture the 3D shape of our head models. The scanner is made of a scan-head that rotates around the subject and captures range information as well as color information (details in Figure 2.12). The system is connected to a workstation via a SCSI interface. The result of a scan is a cylindrical range and color map. Several examples of such cylindrical range and color maps are given on Figure 2.13 . Provided software tools convert this representation into a standard 3D polygonal model with an associated color texture map.

2.5 Phonetic Labeling of Audio Data

In our process, we now wish to label each recorded video frame with its phonetic context. For this purpose we use the audio files in conjunction with the text transcript files to align the phonetic interpretation of the text transcript to the recorded audio data. Tools are readily available to perform this task; they are commonly referred to as aligners. Aligners use a dictionary of word pronunciations. For each word that appears in the corpus of recorded text, there must be an entry in the dictionary with a list of possible pronunciations using a phonetic alphabet. Aligners can be trained for specific voices in order to increase the accuracy of the alignment. For this purpose we have recorded an extra set of sentences (audio and text data only, no video). Once the aligner has been trained with that data and once the audio and text files of the corpus have been corrected for mispronunciations (see section 2.4.6), the aligner automatically produces a phoneme file for each sentence. An example of such a phoneme file is shown in Figure 2.15

At this point the database is composed, for each sentence of the corpus, of three files: a video file in MPEG2 format, an audio file in PCM format, a phoneme file in ASCII format. To insure proper alignment of the audio with the video as well as verify the correctness of the aligner, we have developed a graphical interactive tool that allows to simultaneously visualize a video frame, its audio context and its phonetic context. A screenshot of this tool is shown in Figure 2.16 . By looking at certain phonemes and their corresponding audio signatures, one can assess the alignment of the audio and the video streams of a sequence. In particular, the voiced plosive b as well as the voiceless plosive p are formed by completely stopping the flow of air somewhere in the vocal apparatus, and then releasing the air. By opening the lips, the sudden release of the pent-up air creates a small explosive sound. By aligning the first video frame showing the parting lips with the beginning of this plosive sound, the streams can be aligned up to the sampling rate of the video stream (which is typically lower than that of the audio stream).

2.6 Image Analysis

2.6.1 Introduction

Sample-based synthesis methods use a database of recorded samples to synthesize a new signal. A crucial step in the production of this database is the analysis of the signal to locate the position of samples, followed by their normalization. The latter step is necessary to remove unwanted “dimensions” of the signal. In the case of an audio signal for a text-to-speech synthesis system, for example, the amplitude of the signal is normalized. In our case we need to ‘remove’ the rotation and translation of the head, so that all the extracted samples of facial parts appear to be viewed from exactly the same position and angle.

We first need to locate within a frame the position of the head, then of the mouth and eyes areas. While we need to capture and analyze a large number of video frames (typically over 80K), we have full control of the background, lighting and recording equipment. Hence we are not dealing with difficult backgrounds or wildly varying scales and poses of the head. Instead, we focus on finding, with sub-pixel accuracy, the position of specific facial features. This process is described in paragraph 2.6.2. Next, a normalization step will aim at removing variations due to changes in the pose of the head during the course of the recordings. First the head pose has to be estimated (paragraph 2.6.3), then normalized samples are obtained by factoring the pose out of the image (paragraph 2.6.4). The normalized samples are then annotated with a set of features (paragraph 2.6.5) to make their retrieval and comparison efficient during synthesis.

2.6.2 Locating Face and Measuring Facial Features

Face location and recognition is a very active subject in the world of computer vision and pattern recognition research. Many approaches have been described to track, locate and recognize faces or facial parts. In fact, there are probably as many different approaches as there are real life problems to solve. The “holy grail” of computer vision, which is to find a general technique that can solve all computer-vision problems, has not been discovered (and, most likely, never will). Hence, the most successful techniques so far have been the ones which were specifically designed to solve a given class of problems. For example, eigenfaces [31] have been proven successful at locating human faces in very cluttered images, however, the precision with which this technique can locate faces is limited and the computational cost is generally high. Color segmentation [14][21][26] is ‘CPU-friendly’ but works well only for situations where the lighting is controlled, while color histograms provide a more robust detection in less constrained situations [2]. Shape analysis and local detectors have been used by Jacquin et al. [23] to locate and track the outline of the head, and by Burl et al. [3], coupled with statistical models, to find areas of facial features. Motion information is often combined with color to locate and track faces [32]. Edge and texture information have been used in [5] and anthropomorphic rules such as the inherent symmetry of the face are exploited by Reisfeld et al. [41] and by Shdaifat et al. [42].

But the key to a robust and precise head and facial parts location system is combining these different approaches. Based on our previously published work [17], we propose a common representation of facial elements that allows combining efficiently the output of shape, color and motion analysis. We then describe how facial elements are combined to form whole faces using hierarchical shape filters and n-gram searching. Using the results of the previous steps, we now extract tight windows around facial features and perform an other iteration of color analysis to measure the precise shape of these features. Finally, to provide the precision needed for accurate pose estimation we need to find a set of four or more facial features with sub-pixel accuracy.

Common Representation of Candidate Facial Parts

To combine the output of very different type of analyses such as color, shape and motion, we need a common intermediate representation of candidate facial parts. We first define a head model that contains all facial parts with their sizes and relative positions. This model has been created by measuring the size, shape, orientation and relative position of facial features on frontal pictures of 35 people. In this application, the model is tailored to the particular person of whom we are analyzing the video. The output of each analysis module then provides a list of candidate facial elements with a label and an associated area marked by a 'blob' of connected pixels. These blobs of pixels can be easily measured and compared with the given face model. The blobs are obtained by performing a connected component analysis on a binary image [20]. This representation is suited for all three modes of analysis as shown in the following sections.

Shape/Texture analysis

In this first level of analysis we look for the position of the mouth, eyes and eyebrows. These facial elements exhibit a similar horizontally elongated rectangular shape. Hence we use a set of image processing filters designed to enhance areas where such shapes are present. First, the image is filtered with a high-pass filter, removing the lowest spatial frequencies. We are not interested in slow-changing colors across the face but rather in abrupt changes marking the presence of facial elements such as lips, eyes or the boundary between hair and skin. This step is implemented by subtracting a 'running' average of a rectangular window across the image. The second step convolves a structuring kernel with the image to enhance areas of a certain size and shape. Again, we are looking for the mouth, the eyes and the brows in a frontal image of the face. Hence we adjust the size of the structuring kernel to that of these facial elements. Since we know the image resolution and the overall size of the head, we roughly estimate the optimal size of the averaging window (first step) and of the structuring kernels (second step). To compensate for varying illumination in time, we use an adaptive thresholding technique. It is implemented by computing the numbers of runs (run-length encoding) as a function of the threshold and by taking the threshold that produces the maximum number of runs:

$$thresh = MAX_t\left(nb\_runs(RLE(BIN(t, im)))\right)$$

Equation 2-3

where *BIN(t,im)* is the thresholding operator on image *im* with threshold *t*, *RLE(im)* is a run-length encoding operator on image *im* and *nb_runs(im)* counts the number of runs in a run-length encoded image. The process of shape analysis is illustrated in Figure 2.17

To fine-tune the five free parameters (size of averaging window (*awx*,*awy*), size of structuring kernel (*kx*,*ky*) and offset of adaptive threshold (*off*)) we use the following technique. We first mark the position and size of the mouth, nostrils and eyes on a training set of images. Then we define a quality measure of the resulting thresholded image by verifying that blobs of connected pixels of the right size are located at the marked position of the facial elements. If for all facial elements there exists a blob of the correct size at the correct position, a score of 100 is given. We then scan the five dimensional space of parameters to find the combination of parameters that provides the best quality measure. Figure 2.18 shows the scanning of parameter *kx* and Figure 2.19 shows the scanning of parameter *off* while all other parameters remain constant.

Color Analysis

We use color analysis to find candidate patches of consistent color. In an initialization step we sample color pixels of such patches from several training images, thus defining a list of bounding boxes in the Hue-Saturation-Luminance (HSL) color space. This color space provides better separation of facial elements using either the saturation (lip versus non-lip color) or the hue (skin versus non-skin color). The HSL space is defined as follows (*r*, *g* and *b* respectively the red, green and blue components):

$$hue = \begin{cases} r = \max(r,g,b) \text{ AND } g = \min(r,g,b) \rightarrow 300^{\circ} + \dfrac{\max(r,g,b) - b}{\max(r,g,b) - \min(r,g,b)} \\ \cdots \\ b = \max(r,g,b) \; AND \; r = \min(r,g,b) \rightarrow 180^{\circ} - \dfrac{\max(r,g,b) - g}{\max(r,g,b) - \min(r,g,b)} \end{cases}$$

$$sat = \frac{\max(r,g,b) - \min(r,g,b)}{\max(r,g,b) + \min(r,g,b)} \qquad lum = \frac{\max(r,g,b) + \min(r,g,b)}{2}$$

Equation 2-4

A binary image is then created by assigning ‘black’ to pixels belonging to a bounding-box from the list of defined boxes and ‘white’ to pixels falling outside. The image is then smoothed and connected components are extracted. We apply this process to find the contour of the face, hair and whole head using sampled color pixels from the skin and hair. The color analysis process is illustrated in detail in Figure 2.20 .

This approach is very simple, yet quite robust because the hue of the skin does not vary a lot even when the illumination changes (except when the camera’s CCD goes into saturation). Furthermore, our videos to be analyzed are taken under controlled conditions (lighting, background), thus making this approach suitable.

Motion Analysis

The motion within the picture is estimated by frame differencing. By thresholding the result of the difference between two consecutive frames, we identify areas where large displacements occur. Connected components of black pixels mark these areas.

Shape Filtering and N-gram Analysis

The color analysis as well as the texture and motion analysis produces connected components marking areas where a facial feature may be present. Now, combinations of these features that could represent a face have to be found. First, the shape of each individual connected component is analyzed and those that can definitely not represent a facial feature are discarded. This is done by comparing the size, shape (ratio of height and width) and position of the candidate facial feature with that of the head model. Then combinations of two connected components are tested whether they could represent a combination of two facial features, for example an eye pair, a brow pair or a nostril pair. This time, a pair of candidate facial features is compared with the model using relative position (distance between the two elements), alignment (whether the shape of the two components are matching) and orientation (the angle formed by the line connecting the pair of elements). A score is calculated that reflects how well the pair matches the head model. For example, a pair of facial parts *p1*and *p2* are scored based on their relative distance using the following equation:

$$score_{p_1}(p_2) = score_{p_1} + \left(1 - \frac{\left| dist(p_1, p_2) - distm(type(p_1), type(p_2)) \right|}{distm(type(p_1), type(p_2))} \right)$$

$$dist(p_1, p_2) = \text{distance between center of mass of two facial parts}$$

$$distm(part\_type_1, part\_type_2) = \text{distance in head model}$$

Equation 2-5

In the next step further combinations of facial features are evaluated: eye-pair and brow-pair; eye-pair and mouth; eye-pair, brow-pair and mouth; eye-pair, nostrils, mouth and head contour (from the color analysis); etc. Each time the combination is scored using the head model. Even though the number of combinations to evaluate grows exponentially with the number of facial features, since the number of candidates is kept low at each step by filtering, the overall complexity of the search is well within the range of real-time processing. The process of shape filtering and N-gram search is illustrated in Figure 2.21

This analysis is robust because it doesn't rely on any single facial feature to be present, instead the best scoring combination of features is taken. On a set of 35 different people speaking short sentences totaling over 30K frames, the algorithm found the position of enough facial features to correctly infer the position of the head in over 90% of the cases. When the videos of only one person are analyzed and the head model is trained on that single subject, the accuracy jumps to 98% over a database of over 200K images taken of the subject speaking naturally in a frontal pose.

Measuring facial features

Finding the exact dimensions of the facial features is more challenging because the person is moving the head during the recordings and is changing facial expressions while speaking. This leads to variations in the appearance of a facial feature and affects locally the lighting conditions. For example, during a nod a shadow may fall over the eyes. The shape analysis produces an approximate measure of the positions of facial features but not an accurate description of their shapes. We therefore need to analyze further the areas around eyes, mouth, and the lower end of the nose using color segmentation. In the training phase, areas of the lips and skin are sampled. From the results of the shape analysis, tight windows around the mouth, eyes and nostrils were isolated, allowing a cleaner sampling of the color pixels and, in turn, providing a cleaner separation between areas of skin, non-skin, lips, non-lips. The contours around the eyes are then extracted using an inverted image of the non-skin area of the eyes. The outer lip contour is extracted similarly from the inverted image of the non-skin areas of the mouth. The inner contour of the lips is extracted from the non-lips areas within the outer lips contour (inside of the mouth). This process is illustrated in Figure 2.22

The accuracy achieved for the dimensions of the mouth are typically ±2 pixels (standard deviation), where the width of the mouth is 60 to 100 pixels. However, sometimes, for example when the speaker smiles, strong shadows cover the mouth corners, leading to larger errors in the measurements of the mouth width. Moreover, color segmentation may not work well if the lip color is not sufficiently different from that of the surrounding skin. This is sometimes the case for elderly people or male speakers with narrow lips or with facial hair. We tested the accuracy of lip measurements in a large experiment for automatic lip reading [38][39], where 50 different people pronouncing over 5,000 utterances were recorded under varying lighting conditions. While color segmentation very often provides the lip outlines with remarkable accuracy, its robustness is not always sufficient. Recalibrating the color thresholds dynamically during the analysis by re-training the color segmenter every 20 to 50 frames can track changes in lighting conditions and improves the results.

High accuracy feature points

Correlation analysis is well suited for determining the location of feature points (such as the eye corners or the nostrils) in the face with the precision needed for measuring the head pose. In a fully automatic training phase, representative examples of feature points such as eye-corners, nostrils and mouth-corners are selected from the recorded video, using the results of the previous analysis steps (Figure 2.23 (a)). Then, the area around these points is cut out, generating, for each feature point, a set of kernels. For example, for the right mouth corner, nine examples are selected (Figure 2.23 (c)), based on the width and height of the mouth (3 different width times three different heights). During the analysis phase, for an image and a given feature point, a kernel is chosen with dimensions that are most similar to the ones on the image, and this kernel is scanned over the area around the feature point.

In order to verify that such an approach works reliably we studied the shapes of the correlation functions around the feature points. A correlation analysis or matched filter approach works well if the kernel resembles closely the image to be located. Feature points, such as mouth corners, change significantly in appearance depending on the head orientation, the lighting conditions and whether the mouth is open or closed. Therefore, a single kernel will not be sufficient for an accurate determination of the position. Tests with different mouth images indicated that nine kernels cover the range of appearances encountered here (Figure 2.23 (c)). The correlation function shows a prominent global minimum at the location of the mouth corner as is shown in Figure 2.23 (d). Moreover, the correlation function is monotonically decreasing towards the minimum. Qualitatively the same behavior is observed for all the feature points we investigated, which makes the correlation analysis suitable for a gradient descent approach, saving considerable computation time. Instead of a full correlation analysis where the kernel is scanned point-by-point over the whole image, we use a conjugate gradient technique [37][40], which finds the minimum typically in about ten steps. It requires evaluating numerically the gradient of the correlation function at each step. Yet, even with this additional computation, the conjugate gradient technique is one to two orders of magnitude faster than the full correlation. A typical size of the kernels is 20x20 pixels searching an area of 100x100 pixels, resulting, for 10 iterations, in a computation time of less than 20 ms on a 300MHz PC.

Only the luminance or the hue is used for this analysis and the kernels as well as the images are filtered with a high-pass filter before the correlation (Figure 2.23 (b)). For each mouth corner we use nine different kernels, for each eye corner three, and for each corner of the eyebrows one. The standard deviation in the measured location is typically about one pixel for the eye corners and filtering over time reduces the error to less than 0.5 pixels. The mouth corners are measured with a similar precision except when the mouth is very wide open so that the corner is essentially a straight vertical line. Then the vertical position may be less accurate, however, the horizontal position is still accurate. Feature points we measure in this way are mouth, nostrils, eyes and eyebrows. Knowing precisely the positions of the eye corners and the nostrils allows an accurate determination of the head pose. The locations of the eyebrows help identifying emotions while precise locations of the mouth corners are needed for selecting the mouth shapes for articulating speech. Sometimes the interior of the mouth is also analyzed in this way as well as the outer edges of the lips in the center of the mouth in order to get a better measure of lip protrusion and for estimating the mechanical stress put on the lips.

2.6.3 Pose Estimation

Knowing the values of both the object points $\mathbf{M_i}(x,y,z)$ and their corresponding projected points in the image plane $\mathbf{p_i}(x,y)$ and using the equations of projection, one can infer the pose of the object. This is a general problem known as *Perspective-n-Point* problem (or PnP problem). Several methods have been reported that solve it using the Newton-Raphson method [44][33] which is computationally expensive, may be slow to converge and requires an initial guess. We use instead a pose estimation technique reported by Dementhon et al. in [9] and [35]. In this method the pose is first estimated using a scaled orthographic projection and then it is iteratively refined to match a perspective projection.

Sophisticated techniques that use optical flow constraints coupled with generic 3D head models have been reported (e.g. [8][29]) that also solve the pose estimation problem. However, in our case, since we know precisely the geometry of the subject's face, we can afford this simpler, faster approach.

The equation of the scaled orthographic projection is the following:

$$\mathbf{M} \bullet \mathbf{B} \cdot s = \mathbf{p}$$

$$\begin{pmatrix} \mathbf{M_1} - \mathbf{M_0} \\ \cdots \\ \mathbf{M_n} - \mathbf{M_0} \end{pmatrix} \cdot \begin{pmatrix} i_x & j_x \\ i_y & j_y \\ i_z & j_z \end{pmatrix} \cdot s = \begin{pmatrix} \mathbf{p_1} - \mathbf{p_0} \\ \cdots \\ \mathbf{p_n} - \mathbf{p_0} \end{pmatrix}$$

Equation 2-6

where **M** is the *n* x 3 object matrix; **B** is the 3 x 2 basis matrix made of two column vectors **i**(x,y,z) and **j**(x,y,z) that represent the orthonormal basis of the image plane in which the object is projected; *s* is the scaling factor; **p** is the *n* x 2 image matrix of projected object coordinates. The scaled orthographic projection approximates perspective projection by assuming a common depth of all object points. One can also write the equations for ***perspective*** projection in the following way:

$$\begin{pmatrix} \mathbf{M_1} - \mathbf{M_0} \\ \cdots \\ \mathbf{M_n} - \mathbf{M_0} \end{pmatrix} \cdot \begin{pmatrix} i_x & j_x \\ i_y & j_y \\ i_z & j_z \end{pmatrix} \cdot \frac{f}{Z_0} = \begin{pmatrix} \mathbf{p_1} \cdot (1 + \varepsilon 1) - \mathbf{p_0} \\ \cdots \\ \mathbf{p_n} \cdot (1 + \varepsilon 1) - \mathbf{p_0} \end{pmatrix}$$

Equation 2-7

where f is the focal length (distance from focal point to image plane), the unknown Z_0 is the depth in world coordinates of the reference point of the object and

$$\varepsilon_i = \frac{\mathbf{M_0 M_1} \bullet \mathbf{k}}{Z_0} \quad \text{with } \mathbf{k} = \mathbf{i} \text{ x } \mathbf{j}.$$

Equation 2-8

By setting ε to zero, equation 2-7 is equivalent to equation 2-6. The key idea is then to iterate by first solving for **i, j,** and Z_0 and then compute the new ε using equation 2-8. Finally, using the new ε in equation 2-7 provides a better estimation of the pose (closer to the perspective projection). This algorithm is numerically very stable, even with measurement errors, and it converges in a few iterations.

We use six feature points of the face with relative positions that are reasonably stable during speech, namely the four eye corners and the two nostrils. The object matrix **M** is then a 5x3 matrix and we solve equation 2-7 by performing a singular value decomposition (SVD) of **M**. When the points are coplanar, as is the case for the four eye corners and two nostrils, solving equation 2-7 for ***i*** and ***j*** actually returns an ***i*** and ***j*** that lie in the plane of points. Let's call them $\boldsymbol{i_0}$ and $\boldsymbol{j_0}$. Then ***i*** and ***j*** can be computed by:

$$\begin{cases} \mathbf{i} = \mathbf{i_0} + \lambda \cdot \mathbf{u} \\ \mathbf{j} = \mathbf{j_0} + \mu \cdot \mathbf{u} \end{cases}$$

Equation 2-9

where **u** is the basis unit vector of the nullspace of **M** (the unit vector perpendicular to the plane of points; The vector **u** can be obtained from the last column of the second orthonormal matrix of the SVD of the object matrix). Since **i** and **j** must be perpendicular and of the same length (orthonormal basis), we obtain the following system:

$$\begin{cases} \lambda \cdot \mu = -\mathbf{i_0} \bullet \mathbf{j_0} \\ \lambda^2 - \mu^2 = \mathbf{j_0^2} - \mathbf{i_0^2} \end{cases}$$

Equation 2-10

This system yields two symmetrical solutions for ***i*** and ***j***:

$$\begin{cases} \mathbf{i} = \mathbf{i_0} + \lambda_0 \cdot \mathbf{u} & \mathbf{j} = \mathbf{j_0} + \mu_0 \cdot \mathbf{u} \\ \mathbf{i} = \mathbf{i_0} - \lambda_0 \cdot \mathbf{u} & \mathbf{j} = \mathbf{j_0} - \mu_0 \cdot \mathbf{u} \end{cases}$$

Equation 2-11

where λ_0 and μ_0 are the roots of equation 2-10. Since we have two solutions at each iteration, we would need to explore a tree of solutions to find the best one. In practice, however, branches with impossible solutions or lower quality solutions can be discarded at each step (except at the first iteration where we keep two solutions). The quality measure is simply a cumulative error between the calculated image points (projected model points using the calculated pose) and the actual image points. In our experiment with the face model, we typically see a convergence in less than five iterations.

Measurement errors on the positions of the feature points are typically less than one pixel. A study of the errors in the pose resulting from errors of the recognition is shown in Figure 2.24 . All possible combinations of recognition errors are calculated for a given perturbation (with 6 points and 9 possible errors, all $9^6 = 531441$ poses have been computed).

The final pose matrix is the product of three homogeneous matrices:

$$\mathbf{T} = \mathbf{Trans} \cdot \mathbf{Rot} \cdot \mathbf{Rotcen}$$

Equation 2-12

The first translation matrix **Rotcen** re-centers the object to the center of rotation used in the pose estimation (the point $\mathbf{M_0}$). The second is the rotation matrix obtained from the pose estimation algorithm and the third matrix **Trans** centers the object so that the perspective projection of the reference object point $\mathbf{M_0}$ falls on its corresponding image point $\mathbf{p_0}$:

$$\mathbf{T} = \begin{pmatrix} 0 & 0 & 0 & \frac{p0_x}{f} \\ 0 & 0 & 0 & \frac{p0_y}{f} \\ 0 & 0 & 0 & Z0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \mathbf{i_x} & \mathbf{i_y} & \mathbf{i_z} & 0 \\ \mathbf{j_x} & \mathbf{j_y} & \mathbf{j_z} & 0 \\ \mathbf{k_x} & \mathbf{k_y} & \mathbf{k_z} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 & 0 & -M0_x \\ 0 & 0 & 0 & -M0_y \\ 0 & 0 & 0 & -M0_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Equation 2-13

A 3D object point **O** is then transformed into a 2D image point **i** in the following way:

$$\mathbf{O'} = \mathbf{T} \cdot \mathbf{O} \qquad \mathbf{i} = \begin{pmatrix} \frac{f \cdot \mathbf{O'_x}}{\mathbf{O'_z}} \\ \frac{f \cdot \mathbf{O'_y}}{\mathbf{O'_z}} \end{pmatrix}$$

Equation 2-14

An example of a result of the pose extraction is shown in Figure 2.25 . The rotation angles are extracted from the rotation matrix and displayed along with corresponding images of the head.

When images are captured by cameras, the scene's colors are captured by the CCD elements into pixels. On the other hand, 3D object points are represented in metric units. In equations 2-6 and 2-7, projected object points are related to image points. This assumes that image points have been converted from their raw pixel position into metric units. To perform such conversion, one needs to know the intrinsic parameters of the camera that captured the images.

There are several techniques to calibrate automatically cameras (see [44] and, more recently, [46]). These techniques are most useful when low-end computer cameras are used, whose optics induces a fair amount of deformation. In our case, we use high-end camera optics and we only need to extract two intrinsic parameters, namely the number of CCD capturing elements per metric unit in both directions. Using a pinhole camera model and knowing the position of the focal point relative to the imaging plane (focal length), we capture the image of rulers placed in both horizontal and vertical directions at a known distance from the CCD of the camera (see Figure 2.26). We can thus easily infer the width and height of the CCD itself.

$$CCD\_width = \frac{focal\_length \cdot ruler\_width}{dist\_CCD\_to\_ruler - focal\_length}$$

Equation 2-15

$$pixels\_per\_mm\_x = \frac{CCD\_width\_in\_pixels}{CCD\_width}$$

Equation 2-16

$$p\_mm_x = \frac{p\_pix_x}{pix\_per\_mm\_x} \qquad p\_mm_y = \frac{p\_pix_y}{pix\_per\_mm\_y}$$

Equation 2-17

On our Sony DXC950 camera, we extract the following values: *pix_per_mm_x*=139.265 and *pix_per_mm_y*=126.234 and verify it for 5 different focal lengths. We use these calibration values to convert the positions in pixels on the images of measured feature points into metric units.

2.6.4 Sample Extraction and Normalization

Once the head pose on an image is known, the set of polygons (quadrilaterals) marking the 3D shape of a facial part is projected onto the image plane, resulting in a set of 2D quads (figure 5c). The same facial part's polygons are also projected using a frontal pose (all angles set to zero), resulting in a second set of 2D quads. Each quad of the first set is then mapped into its corresponding quad of the second (normalized) set. The pixels within the quads are mapped using bilinear interpolation [45]. The resulting image is a frontal, normalized view of the facial part. This operation is also called re-projection and is illustrated in Figure 2.27

The resolution of the normalized views should be adapted to the target output, so that no detail is lost and no unnecessary information is carried along. In order to preserve a maximum of details, we keep the number of pixels in the sample bitmaps roughly similar to that of the corresponding area in the original image (for example: 208x168 pixels for the mouth area). These bitmaps can be compressed using standard compression techniques such as JPEG and MPEG2 before being saved into a database. In this way, one minute of speech for the "mouth" facial part results in an MPEG2 file of size 12MB (208x168 pix at 60 frames/sec) with very little loss in quality.

2.6.5 Sample Parameterization

Once all samples of a face part are extracted and stored as bitmaps, several features need to be computed and attached to the samples to enable synthesis (see chapter 3). We use the following features:

- **Geometric features**

These are features derived from the type, relative position and size of facial parts present in the sample image. They are obtained from the facial analysis module and are re-projected to correspond to a normalized, frontal view of the head. For example, we describe the mouth with three parameters: The width (the distance between the two corner points), the y-position of the upper lip (the y-maximum of the outer lip contour) and the y-position of the lower lip (the y-minimum of the outer lip contour). Samples of other facial parts are parameterized in a similar way. Geometric features are used in the synthesis process to define a visual distance between samples.

- **PCA features**

While geometric features described above are useful to discriminate among a large number of samples of a given facial part, they cannot capture fine details of the appearance. Principal component analysis (PCA) captures the main directions of variance within the considered dataset and hence is well suited to compress the images of the database into small feature vectors. To compute PCA, the luminance values of the normalized bitmaps are first sub-sampled in both directions into M vectors of N pixels and stacked into a data matrix. The NxN covariance matrix of the NxM data matrix is first calculated. Then, from the singular value decomposition of the covariance matrix, the eigenvectors with the highest eigenvalues are retained.

Finally, each sample is projected onto each of the principal eigenvectors to obtain a feature vector. From the original normalized samples measuring 208x168 RGB pixels, retaining k PCA coefficients achieves a factor of 26208/k in compression. For example, 15 PCA coefficients account for over 95% of the variance in the input images, which is typically sufficient for the purpose of defining a distance between samples. Figure 2.28 shows the set of 12 ‘eigen-mouths’ extracted from the database of 200K mouth images.

Misalignment of samples (offsets, rotations, etc.) would prevent PCA to account for most of the variance with a small number of principal components, and hence would not be useful here. However, since the samples have been normalized (thus factoring out the head pose), PCA components are very good features for discriminating fine details in the shapes of facial parts.

Wavelet representation, Principal Component Analysis, and Linear Discriminant Analysis have been used successfully for lip-reading [38] and have been demonstrated to capture reliably the appearance of mouth and lip images. Here we choose Principal component analysis (PCA) since it provides a compact representation and captures the appearance of the mouth with just a few parameters. PCA is generally expensive to compute, especially for images, because of the large matrices that have to be handled at a complexity of order $O(n^3)$. Sub-sampling the images, and performing the analysis on the luminance channel only brings the size of the covariance matrix where it can be efficiently reduced by SVD. In our experiments we sub-sampled our images down to 30 by 18 pixels, resulting in vectors of size 540 and a covariance matrix only slightly larger than 1MB.

- **The head pose:**

These are the angles and the position of the head in the given image. This information is saved for each image that is used as base face. It is used at synthesis time to project facial parts samples onto the base face with the correct orientation.

- **Original sequence and frame number:**

These are the frame number and sequence number in the original recorded video data. This information is used when selecting samples for an animation to preserve whenever possible the inherent smoothness of real, recorded facial movements.

- **Phonetic information**

Each mouth sample is assigned a phoneme label which corresponds to the phoneme being uttered at the time the image was captured. This phonetic labeling process is described in section 2.5.

2.7 References

[1] Baird, H., **Anatomy of a versatile page reader**. Proceedings of the IEEE, 80(7):1059--1065, 1992.

[2] Birchfield, S., **An elliptical head tracker**; Proc. of 31st Asilomar Conf. on Signals, Systems and Computers, November 1997.

[3] Burl, M. C., Leung, T. K., Perona, P. **Face localization via shape statistics**; in M. Bichsel, ed., `Proc. Intl. Workshop on Automatic Face and Gesture Recognition', pp. 154-159, 1995.

[4] Cohen, M.M., Massaro, D.W., **Modeling Coarticulation in Synthetic Visual Speech**, Models and Techniques in Computer Animation, Springer Verlag, 1993.

[5] Cootes, T. F., Taylor, C. J., **Locating faces using statistical feature detectors**. In Proceedings, Second International Conference on Automatic Face and Gesture Recognition, pages 204--209. IEEE Computer Society Press, October 1996.

[6] Cosatto, E., Graf, H.P., "**Sample-Based Synthesis of Photo-Realistic Talking-Heads"**, IEEE Computer Animation, 1998.

[7] Cyberware Laboratory, Inc. "**4020/RGB 3D Scanner with Color Digitizer"**. Monterey, CA, 1990.

[8] DeCarlos D. and Metaxas D., "**Optical flow constraints on deformable models with applications to face tracking"**, International Journal of Computer Vision, Vol. 38, No. 2, pp99-127, 2000.

[9] Dementhon D.F., Davis L. S., "**Model-based object pose in 25 lines of code"**, Intl. Journal of Comput. Vision, 15:123--141, 1995.

[10] Ebrahimi, T., Kunt, M., "**Coding of moving pictures and associated audio for digital storage media up to about 1.5"**, ISO/IEC JTC1 CD 11172, MPEG, 1991.

[11] Ekman, P., Friesen, W., "**Manual for Facial Action Coding System**", Consulting Psychologists Press Inc., Palo Alto, CA, 1978.

[12] Eyetronics 3D Scanning solutions, www.eyetronics.com

[13] Ezzat, T., Poggio, T., "**MikeTalk: A Talking Facial Display Based On Morphing Visemes"**, IEEE Computer Animation, 1998.

[14] Faigin, G., "**The Artist's Complete Guide to Facial Expression**", Watson-Guptill, New-York, 1990.

[15] Fieguth, P., Terzopoulos, D., "**Color-based tracking of heads and other mobile objects at video frame rates**", Proc. of the IEEE CVPR, pages 21–27, 1997.

[16] Fua, P., "**Reconstructing complex surfaces from multiple stereo views**", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1078--1085. IEEE Computer Society Press, June 1995.

[17] Graf, H.P., Cosatto, E., Potamianos, G., "**Robust Recognition of Faces and Facial Features with a Multi-Modal System"**, IEEE Systems, Man and Cybernetics, pp. 2034-2039, 1997.

[18] Guenter, B., Grimm, C., Wood, D., Malvar, H., Pighin, F., "**Making Faces",** ACM SIGGRAPH, pp. 55-66, 1998.

[19] Haskell, B.G, Puri, A., Netravali, A.N., "**Digital Video: An Introduction to MPEG-2"**, Chapman & Hall, New York, 1997.

[20] Horn, B.K.P., "**Robot Vision**", MIT Press, McGraw Hill, 1986.

[21] Hunke, M., Waibel, A., **Face locating and tracking for human-computer interaction**; Proc. of the 28th Asilomar Conf. on Signals, Systems and Computers, pages 1277–1281, 1994.

[22] Inspeck Inc., www.inspeck.com

[23] Jacquin, A., Eleftheriadis, A., **Automatic location tracking of faces and facial features in video sequences**; In: Bichsel, M. Z. Ed. , Proc. 1st Internat. Workshop on Automatic Face- and Gesture-Recognition, Zurich, Switzerland, pp. 142--147.

[24] Kalberer, G., Van Gool, L., **Lip animation based on observed 3D speech dynamics**, Proceedings of SPIE, January 2001.

[25] Kalra P., Mangili A., Magnenat-Thalmann N., Thalmann D., **Simulation of Facial Muscle Actions based on Rational Free Form Deformation**; Proceedings Eurographics 92, pp. 65-69

[26] Kocheisen, M., **Head Detection on Low-Resolution Color Photos**, Proc. Workshop Machines that Learn, Snowbird, UT, 1996.

[27] Levoy, M. et al., The digital michelangelo project: 3d scanning of large statues. SIGGRAPH, 2000.

[28] Lee, Y., Terzopoulos, D., Waters, K., **Realistic Modeling for Facial Animations**. In Computer Graphics (SIGGRAPH '95 Conf. Proc.), pages 55--62, Aug. 1995.

[29] Li, H., Roivainen, P., Forcheimer, R., **3-D motion estimation in model-based facial image coding**; IEEE transactions on Pattern Analysis and Machine Intelligence 15(6), 545-55, 1993.

[30] Munhall, K. G. & Vatikiotis-Bateson, E. (1998) **The moving face during speech communication**. In R. Cambell, B. .Dodd, & D. Burnham (Eds.), Hearing by eye, Part 2: The psychology of speechreading and audiovisual speech. London: Taylor & Francis, Psychology Press.

[31] Turk, M. A., Pentland, A. P., **Face Recognition Using Eigenfaces** Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, pp. 586-591, June 1991.

[32] Lee, C. H., Kim, J. S., Park, K. H., **Automatic human face location in a complex background using motion and color information** Patt. Rec. 29(11), pp. 1877--1889, 1996.

[33] Lowe, D.G., **Fitting Parameterized Three Dimensional Models to Images**; IEEE Trans. On Pattern Analysis and Machine Intelligence, Vol. 13, pp.441-450, 1991.

[34] Noh, J.Y., Neumann, U., **A Survey of Facial Modeling and Animation Techniques**; 1998, USC Technical Report 99-705

[35] Oberkampf, D., Dementhon, D., Davis, L., **Iterative Pose Estimation Using Coplanar Feature Points**; Internal Report, CVL, CAR-TR-677, University of Maryland, 1993.

[36] Parke, F. I., Waters, K**., Computer facial animation**; Wellesley MA: A K Peters, 1996.

[37] Polak, E., **Computational Methods in Optimization**; Academic Press, New York, 1971.

[38] Potamianos, G., Graf, H.P., **Linear Discrimant Analysis for Speechreading**, IEEE Workshop on Multimedia Signal Processing, pp. 221-226, 1998.

[39] Potamianos, G., Graf, H.P., **Discriminative training of HMM stream exponents for AV speech recognition**; Proc. Int. Conf. Acoustic, Speech and Signal Processing (ICASSP-98), vol. 6, Seattle, WA, May 12-15, 1998, pp. 3733-3766.

[40] Press, W. H. Teukolsky, S. A. Vetterling, W. T. and Flannery. B. P. **Numerical Recipes in C**; Cambridge University Press, second edition, 1992.

[41] Reisfeld, D., Yeshurun, Y., **Robust detection of facial features by generalized symmetry** in Proceedings of the 11th International Conference on Pattern Recognition, The Hague, Netherlands, September 1992.

[42] Shdaifat, I., Grigat, R-R., Lutgert, S., **Viseme recognition using multiple feature matching;** EU-ROSPEECH,Aalborg, Denmark, September 20001.

[43] Syrdal, A. K., **Development of a female voice for a concatenative text-to-speech synthesis system**; Current Topics in Acoustic Research, vol 1, 1994, 169-181.

[44] Tsai R.Y., **A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses**, IEEE Journal of Robotics and Automation, Vol. RA-3, No. 4, pp. 323-344, August 1987.

[45] Wolberg G, **Digital Image Warping**; IEEE Computer Society Press, New York, 1990.

[46] Zhang, Z., **A Flexible New Technique for Camera Calibration**; Microsoft Research Technical Report MSR-TR-98-71, 1998.

2.8 Figures

Figure 2.1	Hierchical face model
Figure 2.2	3D shape of facial parts
Figure 2.3	Distortions of facial parts
Figure 2.4	Alpha-mask for facial parts
Figure 2.5	Human vocal tract
Figure 2.6	English phonemes
Figure 2.7	Coarticulation illustration
Figure 2.8	Hierarchical viseme clustering
Figure 2.9	Tri-viseme coverage
Figure 2.10	Corpus size versus tri-viseme coverage
Figure 2.11	MPEG2 index file
Figure 2.12	Cyberware Scanner
Figure 2.13	Examples of Cyberware scans
Figure 2.14	Recording Session
Figure 2.15	Example of phoneme file
Figure 2.16	Examiner tool
Figure 2.17	Steps of Shape Analysis
Figure 2.18	Shape Analysis (fine tuning of stucturing kernel)
Figure 2.19	Shape Analysis (fine tuning of thresholding)
Figure 2.20	Color Analysis (finding the whole head)
Figure 2.21	Shape Filtering and N-gram Analysis
Figure 2.22	Color Analysis (Measuring facial parts)
Figure 2.23	High precision feature points
Figure 2.24	Pose estimation precision
Figure 2.25	Pose estimation example
Figure 2.26	Camera model and calibration.
Figure 2.27	Sample Normalization
Figure 2.28	PCA

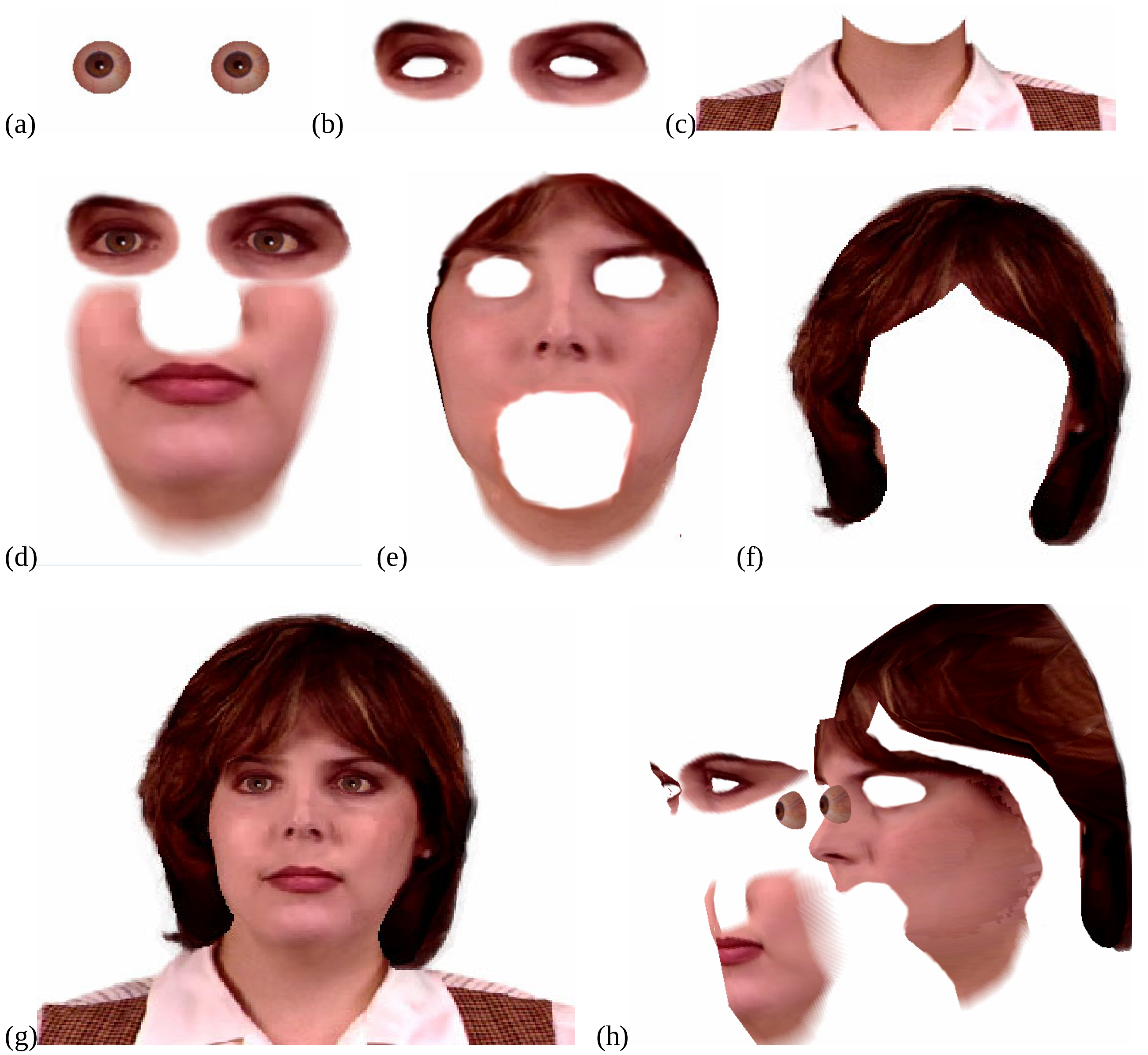


Figure 2.1. The hierarchical head model. To allow different areas of the head to be animated independently, the head is divided into facial parts. The eye-globes (a) are modeled as spheres to allow eye movements. The eye borders and eyebrows (b) allow independent animation of the upper part of the head such as eyebrow rising and eye squinting. The shoulder and neck area (c) is also modeled separately to allow head articulation. The mouth area (d) includes cheeks and chin. The “base face” (e) includes the nose and surrounding skin, as well as the hair (f). In (g) all parts are assembled (note: this is an actual rendering on a frontal pose, not a photograph!). Feathering (progressive alpha-blending) is used to prevent hard boundaries to appear at the junction of facial parts. In (h), a side view hints at the assembly process.

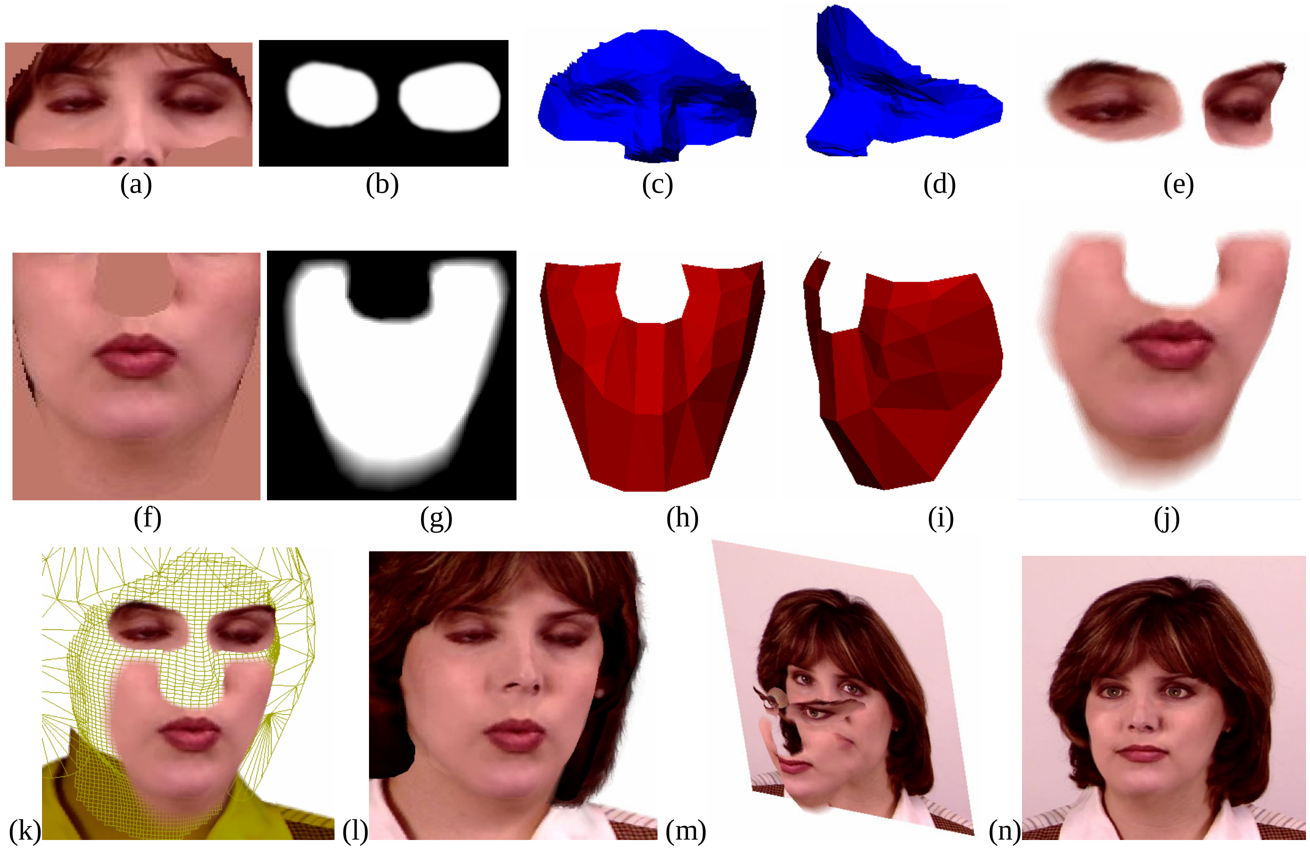


Figure 2.2. The facial parts of the head model. The first row (a-e) shows the eye part and the second row (f-j) shows the mouth part. The first column (a,f) shows the normalized texture, the second column (b,g) shows the texture mask, the third column (c,h) and the fourth column (d,I) show two different views of the 3D model of the parts and the fifth column (e,j) shows a view of the texture mapped, masked 3D shape of the parts. In (k) the underlying 3D "base face" is shown as a wire-frame and (l) shows the final assembly with the 3D "base face" being texture-mapped. Another modeling approach is shown in (m,n): the "base face" is modeled by a single plane, while the mouth and eye parts are modeled in 3D. Picture (n) is obtained by setting the pose of the facial parts to match the pose exhibited by the "base face" image.

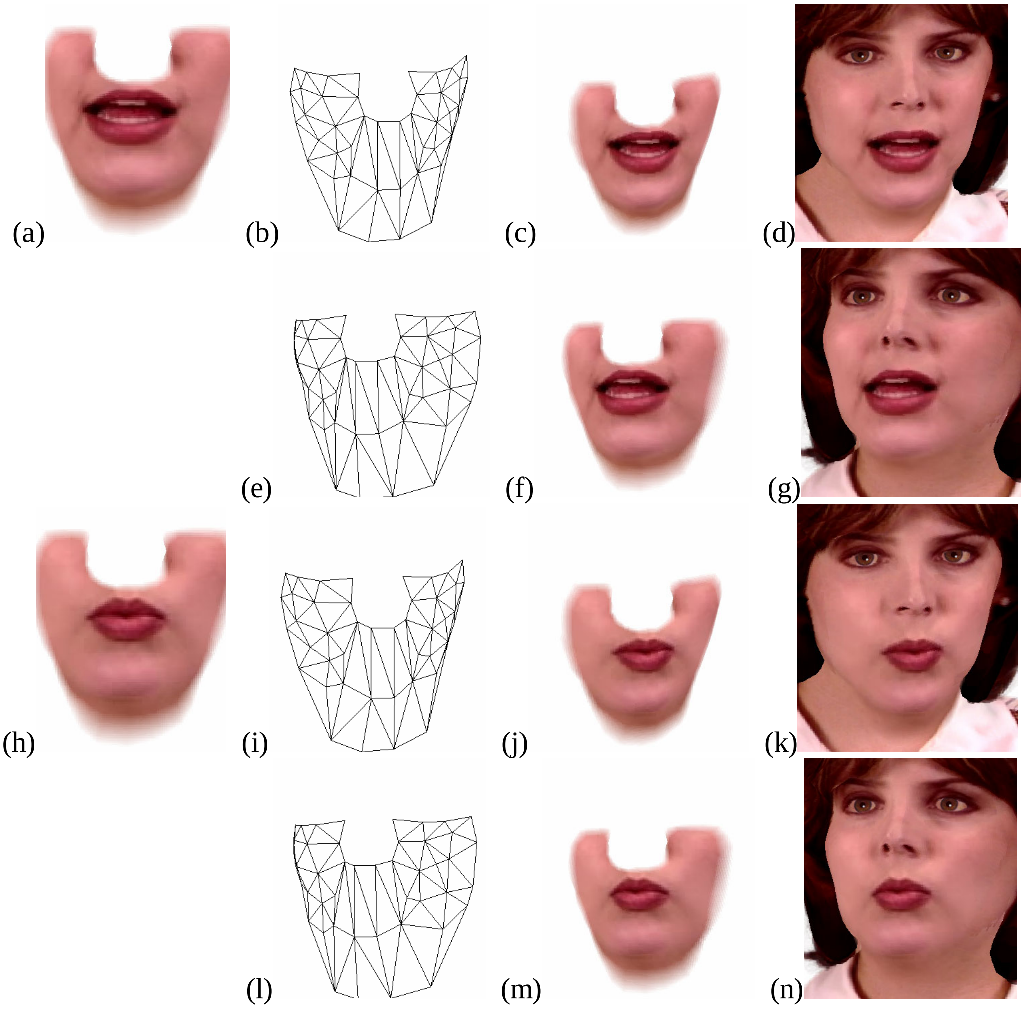


Figure 2.3. Distortions incurred by changing the viewpoint when using a single simplified 3D wire-frame. Bitmaps (a,h) show the original frontal view of an open mouth sample and a protruded mouth sample respectively. Bitmaps (b,c,d) and (I,j,k) show the samples with a rotation of 12deg. around the x-axis and 10deg. Around the y-axis. Bitmaps (e,f,g) and (l,m,n) show the samples with a rotation of –3deg. around the x-axis and –10deg. around the y-axis. The second column shows the wire-frame of the mouth under the given rotations. The third column shown the sample texture applied to the wire-frame and the fourth column shows the mouth integrated into the whole head. Both the protruded and open mouth samples look acceptable under these viewpoints.

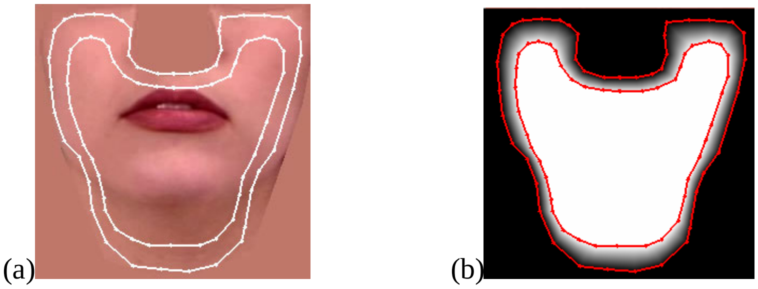

(a)
(b)

Figure 2.4. The creation of the alpha-mask for the mouth part. A normalized sample (a) is used to define two contours. Outside the outer contour the facial part's samples will be completely transparent, while inside the inner contour, they will be completely opaque. In between the mask gradually becomes transparent to enable a seamless blending of the facial part onto the base-face. Since all samples are normalized, only a single alpha-mask is needed. Notice that the mask is extended in the chin area to accommodate open mouths. The blending is happening in the neck area and the cheeks which are smooth, thus making the operation almost undetectable.

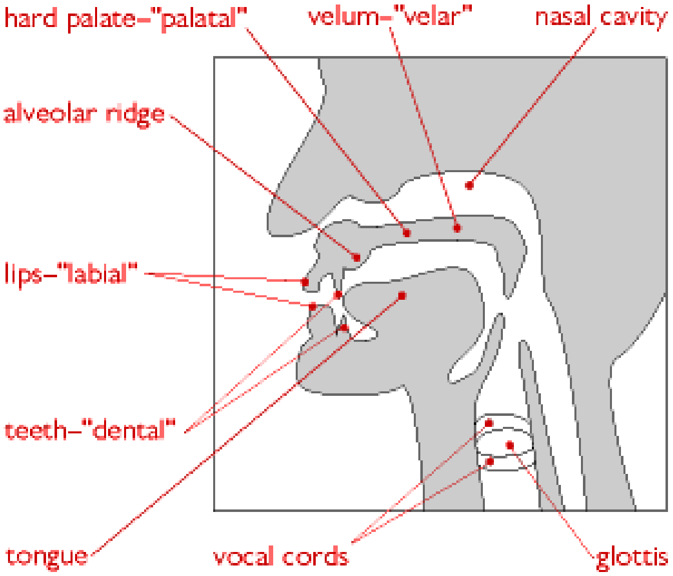


Figure 2.5. Representation of the various parts of the human vocal tract involved in the articulation of speech.

Vowels:

ae: bad
ah: bud
ax: about
aa: pot
ow: oat
ao: bought
eh: bed
iy: eat
ih: hit
uw: boot
uh: put

Diphthongs:

aw: how
ay: bite
ey: bait
oy: boy
el: bottle
en: button
em: Adam
er: bird

Consonants:

m : me (nasal-bilabial)
n : net (nasal-alveolar)
ng: sing (nasal-velar)
l : lot (resonant-alveolar-lateral)
r : rat (resonant-alveolar-retroflex)
w : we (resonant-bilabial)
y : yes (resonant-alveopalatal)
p : pot (stop-bilabial-voiceless)
b : bat (stop-bilabial-voiced)
t : tip (stop-alveolar-voiceless)
d : dip (stop-alveolar-voice
k : cot (stop-velar-voiceless)
g : gag (stop-velar-voiced)
q : bat (stop-glottal)
ch: cheap (affricate-voiceless)
jh: jot (affricate-voiced)
f : fin (fricative-labiodental-voiceless)
v : vat (fricative-labiodental-voiced)
th:thorn (fricative-interdental-voiceless)
dh: that (fricative-interdental-voiced)
s : son (fricative-alveolar-voiceless)
z : zoo (fricative-alveolar-voiced)
sh:shop (fricative-alveopalatal-voiceless)
zh:measure (fricative-alveopalatal-voiced)
hh: hoot (fricative-velar-voiceless)

Figure 2.6. A table of present-day English phonemes and their notation in the DarpaBet phonetic alphabet.

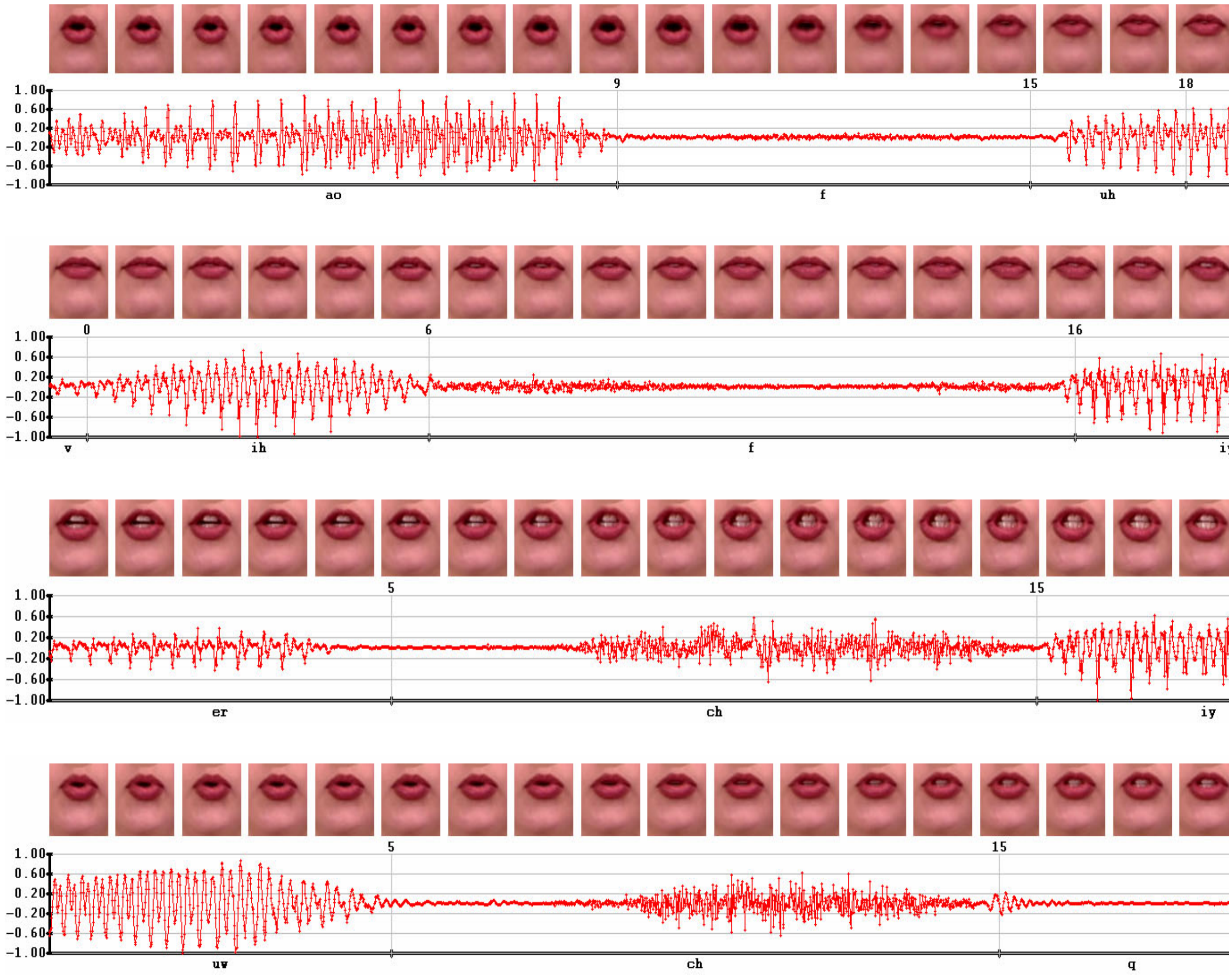


Figure 2.7. Examples of coarticulation. The 1st and 2nd graphs show how the lip shape for the phoneme ‘f’ varies depending whether the word ‘awful’ (1st) or ‘feel’ (2nd) is being pronounced. The 3rd and 4th graphs show how the lip shape for the phoneme ‘ch’ varies depending whether the word ‘cheek’ (3rd) or ‘pooch’ (4th) is being pronounced. The phoneme labels are shown on the bottom of the graphs, while the video frame numbers are shown below the mouth images. The sampling of the mouth is 60 frames per second. The span of the graph is 350 milliseconds or 21 video frames. The red signal is the speech signal, originally sampled at 16kHz.

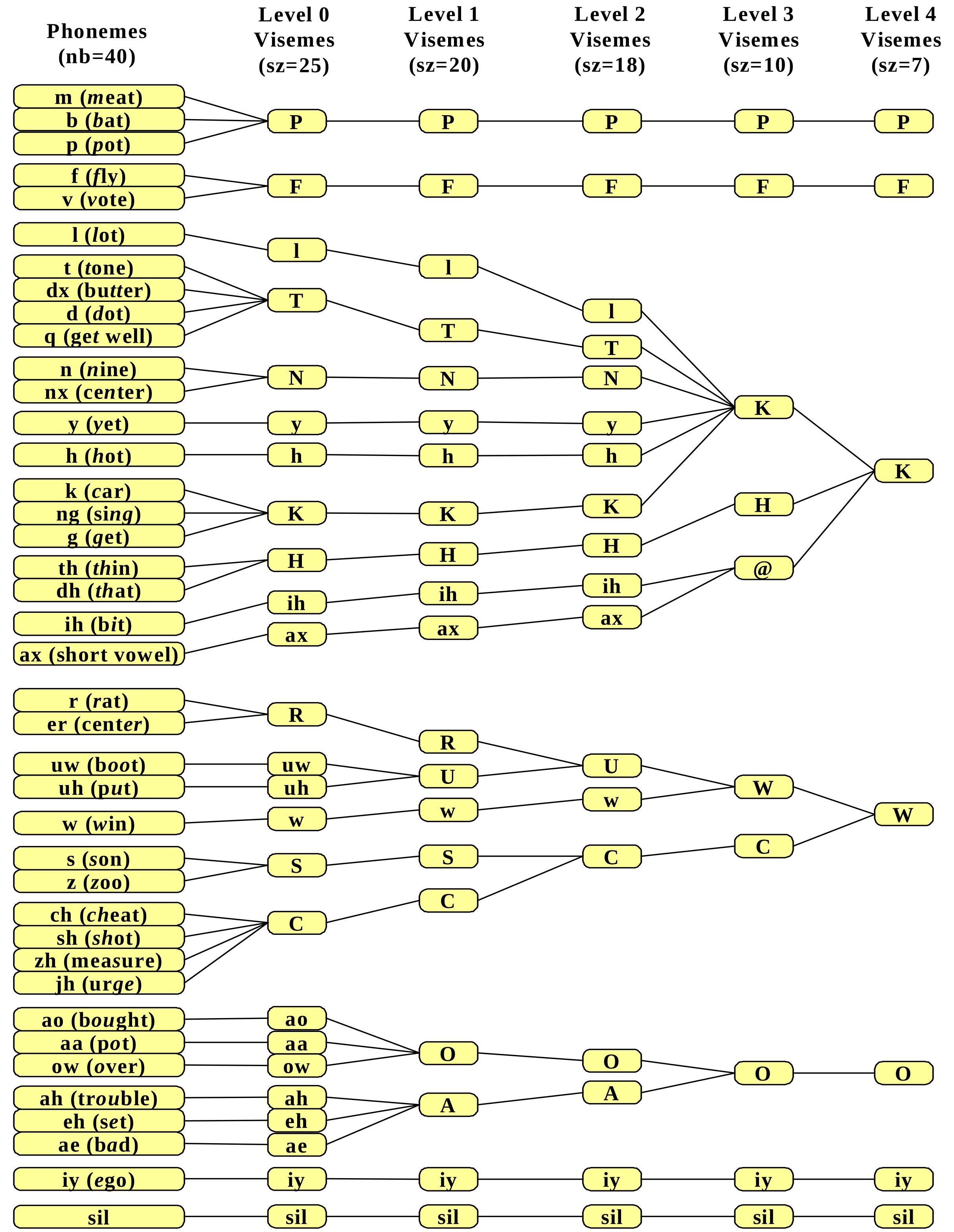


Figure 2.8. Phonemes are clustered hierarchically into groups based on the similarity of the appearance of the mouth. These clusters are called visemes. We define 5 levels of clustering, from less aggressive (level 0) to most aggressive (level 4). Viseme clusters containing more than one phoneme are labeled with a capital letter.

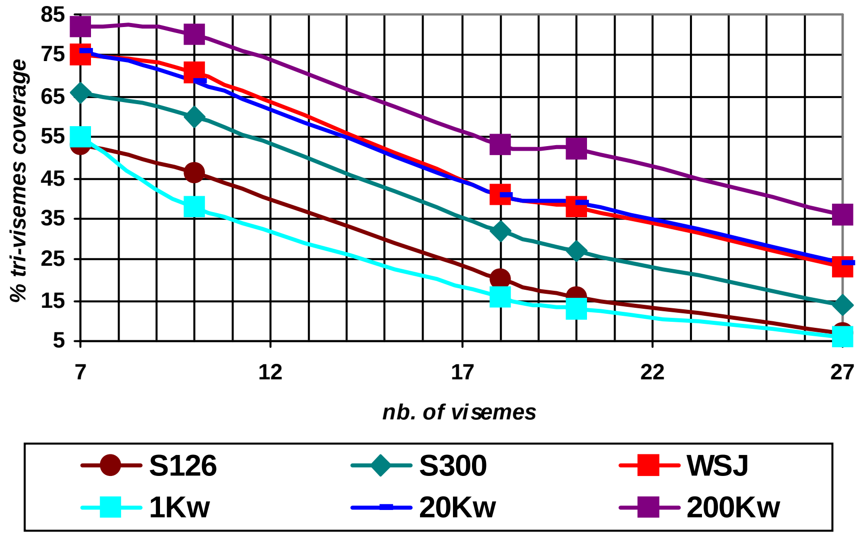


Figure 2.9. We have calculated the fraction of all possible tri-visemes that appear in 6 different text corpuses for five different viseme grouping schemes. These groupings are: group 0 (sz=7), group 1 (sz=10), group 2 (sz=18), group 3 (sz=20), group 4 (sz=27). The text corpuses are: S126 (a set of 126 sentences using common English words), S300 (a set of 300 short sentences covering most English diphones, totaling 2885 words), WSJ (a set of sentences from the Wall Street Journal totaling 15K words), 1Kw (a set of the 1000 most common English words), 20Kw (a dictionary containing 20K words), 200Kw (a dictionary containing 200K words).

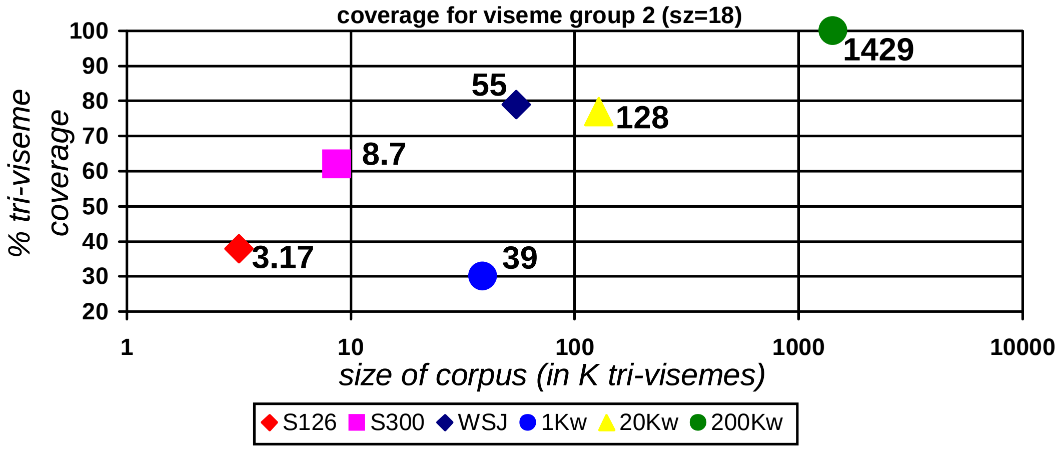


Figure 2.10. This is a cross-section of the above graph for number of visemes = 18. The vertical scale is normalized by the largest coverage value (of corpus 200Kw). On the horizontal axis, the size of each corpus is given by the number of tri-visemes they contain. The 'sweet spot' is the upper-left corner. The corpus we are using (S300) provides a reasonable alternative between complexity and coverage.

```
Fr= 0 Type= I D= 0  Bk= -1 Fw= -1 f= 0x0000000000000021
Fr= 1 Type= P D= 3  Bk=  0 Fw= -1 f= 0x000000000000fd64
Fr= 2 Type= B D= 1  Bk=  0 Fw=  1 f= 0x000000000001d33f
Fr= 3 Type= B D= 2  Bk=  0 Fw=  1 f= 0x0000000000024356
Fr= 4 Type= P D= 6  Bk=  1 Fw= -1 f= 0x0000000000033498
Fr= 5 Type= B D= 4  Bk=  1 Fw=  4 f= 0x0000000000047477
Fr= 6 Type= B D= 5  Bk=  1 Fw=  4 f= 0x0000000000057ab6
Fr= 7 Type= P D= 9  Bk=  4 Fw= -1 f= 0x00000000000681e4
Fr= 8 Type= B D= 7  Bk=  4 Fw=  7 f= 0x0000000000078820
Fr= 9 Type= B D= 8  Bk=  4 Fw=  7 f= 0x00000000000876f9
Fr=10 Type= P D= 12 Bk=  7 Fw= -1 f= 0x00000000000964ae
Fr=11 Type= B D= 10 Bk=  7 Fw= 10 f= 0x00000000000a804b
Fr=12 Type= B D= 11 Bk=  7 Fw= 10 f= 0x00000000000b731e
Fr=13 Type= I D= 15 Bk= -1 Fw= -1 f= 0x00000000000c667e
Fr=14 Type= B D= 13 Bk= 10 Fw= 13 f= 0x00000000000d88bd
Fr=15 Type= B D= 14 Bk= 10 Fw= 13 f= 0x00000000000e6db7
Fr=16 Type= P D= 18 Bk= 13 Fw= -1 f= 0x00000000000f53f7
Fr=17 Type= B D= 16 Bk= 13 Fw= 16 f= 0x0000000000106139
Fr=18 Type= B D= 17 Bk= 13 Fw= 16 f= 0x00000000001149ee
Fr=19 Type= P D= 21 Bk= 16 Fw= -1 f= 0x000000000012318b
Fr=20 Type= B D= 19 Bk= 16 Fw= 19 f= 0x0000000000133a81
Fr=21 Type= B D= 20 Bk= 16 Fw= 19 f= 0x0000000000141e91
Fr=22 Type= P D= 24 Bk= 19 Fw= -1 f= 0x00000000001501f2
Fr=23 Type= B D= 22 Bk= 19 Fw= 22 f= 0x0000000000161151
Fr=24 Type= B D= 23 Bk= 19 Fw= 22 f= 0x000000000016f826
Fr=25 Type= P D= 27 Bk= 22 Fw= -1 f= 0x000000000017ddd7
Fr=26 Type= B D= 25 Bk= 22 Fw= 25 f= 0x000000000018e958
Fr=27 Type= B D= 26 Bk= 22 Fw= 25 f= 0x000000000019ceb7
Fr=28 Type= I D= 30 Bk= -1 Fw= -1 f= 0x00000000001ab456
```

Figure 2.11. An MPEG2 .map file showing the indices, type and dependencies of each frame within the stream. Column 1 indicates the frame number in file order; column 2 indicates the type of MPEG2 frames (I=integral frame, P=predictive frame, B=bi-directional frame); column 3 indicates the frame number in display order; column 4 indicates, for B and P-frames which previously decoded frame (file order) is needed to decode that frame; column 5 indicates, for B-frames, which other previously decoded frame(file order) is needed to decode that frame (note that the display order of frames referenced in that column is always larger of that of the frame being decoded); column 6 indicates the byte position within the file where the frame start (this is a 64-bit value to allow for files larger then 2 Gigabytes).

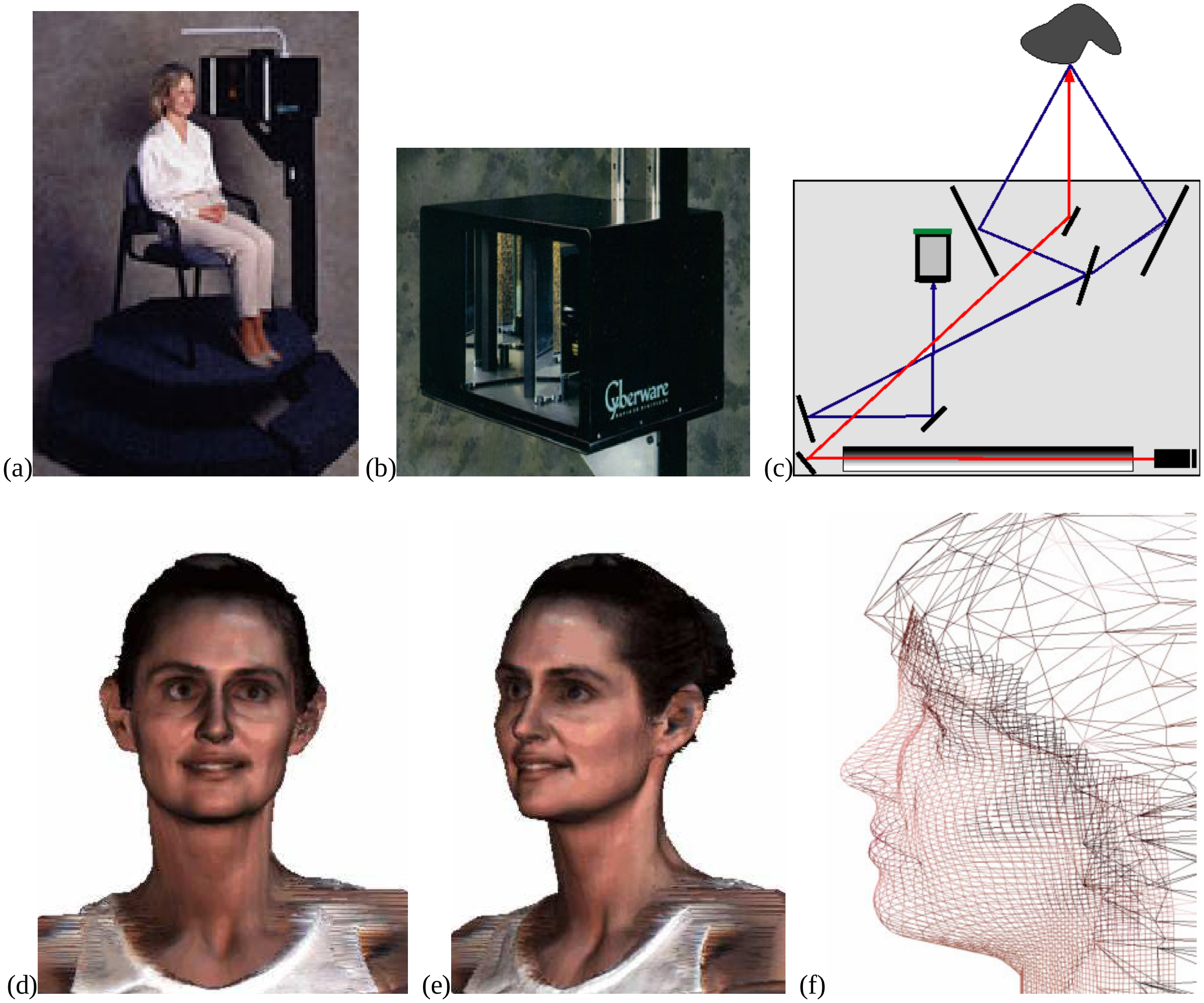


Figure 2.12. The Cyberware “Head & Face Color 3D Scanner” (a). A detail of the scan-head (b) shows surface mirrors. The schematic in (c) reveals how the system operates. A laser beam (red line) is shone on the subject and is reflected back (blue lines). Looking at the scanned object at an angle reveals the depth profile of the surface. The use of two mirrors reduces the risk of the surface not reflecting the beam (when the surface normal forms a too large an angle with the laser beam). The images from the two mirrors are combined using a beam splitter and are captured by a CCD camera (green). A servo-motor rotates the scan-head around the subject and captures range and color information (via a second CCD camera) at a pitch of 1/10th of a degree. The output is a “cloud” of 3D points each with its color value (R,G,B) (d,e). The resolution on the surface of the subject’s head is about 1/4th of a millimeter in both directions. The precision of the depth at each sample is about 0.3mm. Using software tools these point clouds are transformed into a standard 3D wire-frame model (f) with separate texture bitmaps. Images (a) to (e), courtesy of Cyberware Inc.

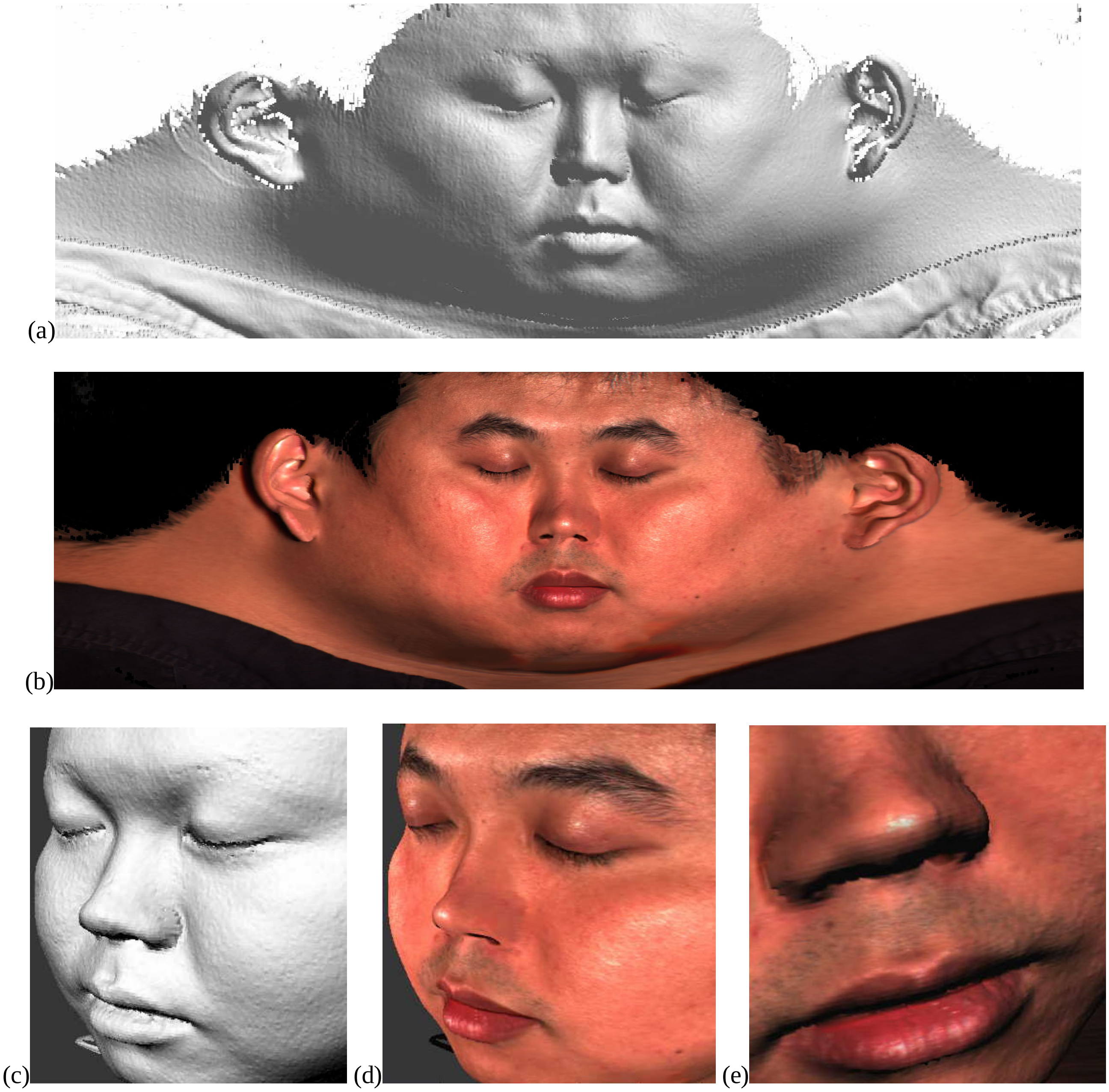


Figure 2.13. Examples of CyberWare scans at different stages. The range data (a), the texture (b) flattened out using cylindrical coordinates. Bitmap (c) shows a 3D view of the scan, (d) shows the same view with texture applied. In (e) a detail is shown with shading.

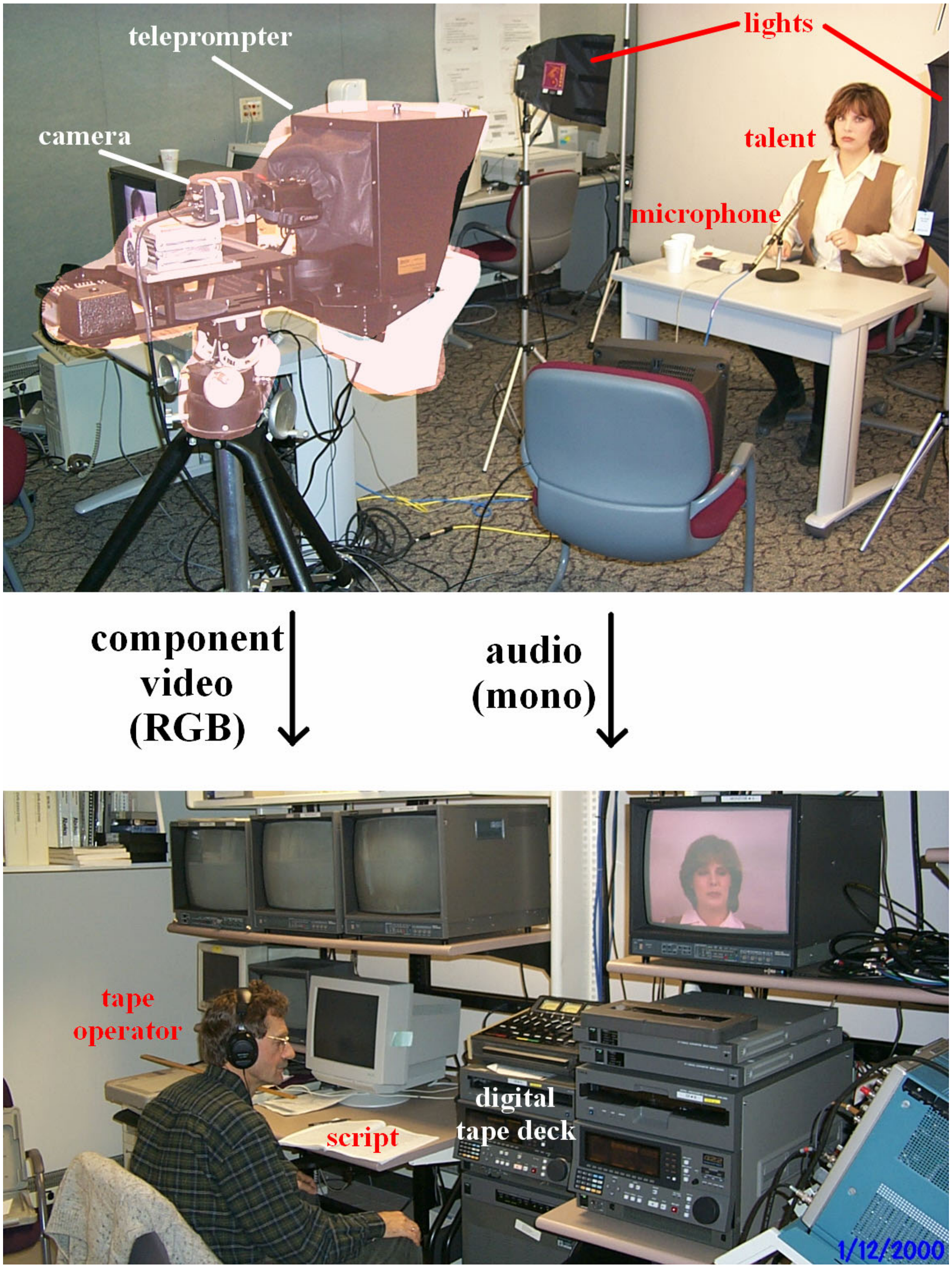


Figure 2.14. Snapshots of the recording session. The talent sits in a separate ‘silent’ room and reads from a teleprompter. A 3CCD camera transmits a RGB component signal to the recording room, which is recorded uncompressed digitally onto D1 tapes. The audio is also recorded digitally from a directional microphone onto the same tape.

```
separator : file s101_279.ar
   1.090000  :     SIL :     sil :    <SIL>
   1.160000  :      dh :        :
   1.220000  :      ih :       1 :     the
          -  :         :       0 :
   1.310000  :     SIL :     sil :    <SIL>
   1.380000  :       d :        :
   1.570000  :      uw :        :
   1.680000  :       k :       1 :    duke
          -  :         :       1 :
   1.830000  :     SIL :     sil :    <SIL>
   1.860000  :       d :        :
   1.940000  :      ih :        :
   2.000000  :       d :       1 :     did
          -  :         :       2 :
   2.040000  :      ih :       1 :      a
          -  :         :       3 :
   2.100000  :       g :        :
   2.210000  :      uh :        :
   2.260000  :       d :       1 :    good
          -  :         :       4 :
   2.340000  :      jh :        :
   2.580000  :      aa :        :
   2.620000  :       b :       1 :     job
          -  :         :       5 :
   2.900000  :     SIL :     sil :    <SIL>
   3.350000  :     SIL :     sil :    <SIL>
```

Figure 2.15. A phoneme file for the sentence: “The Duke did a good job.” Column 1 shows the time in seconds of the end of the phoneme; column 2 shows the phoneme name (in DarpaBET); column 3 shows the syllable number within a word and for entries with an “-“ in the first column, the syllable number within the whole sentence; column 4 shows the words.

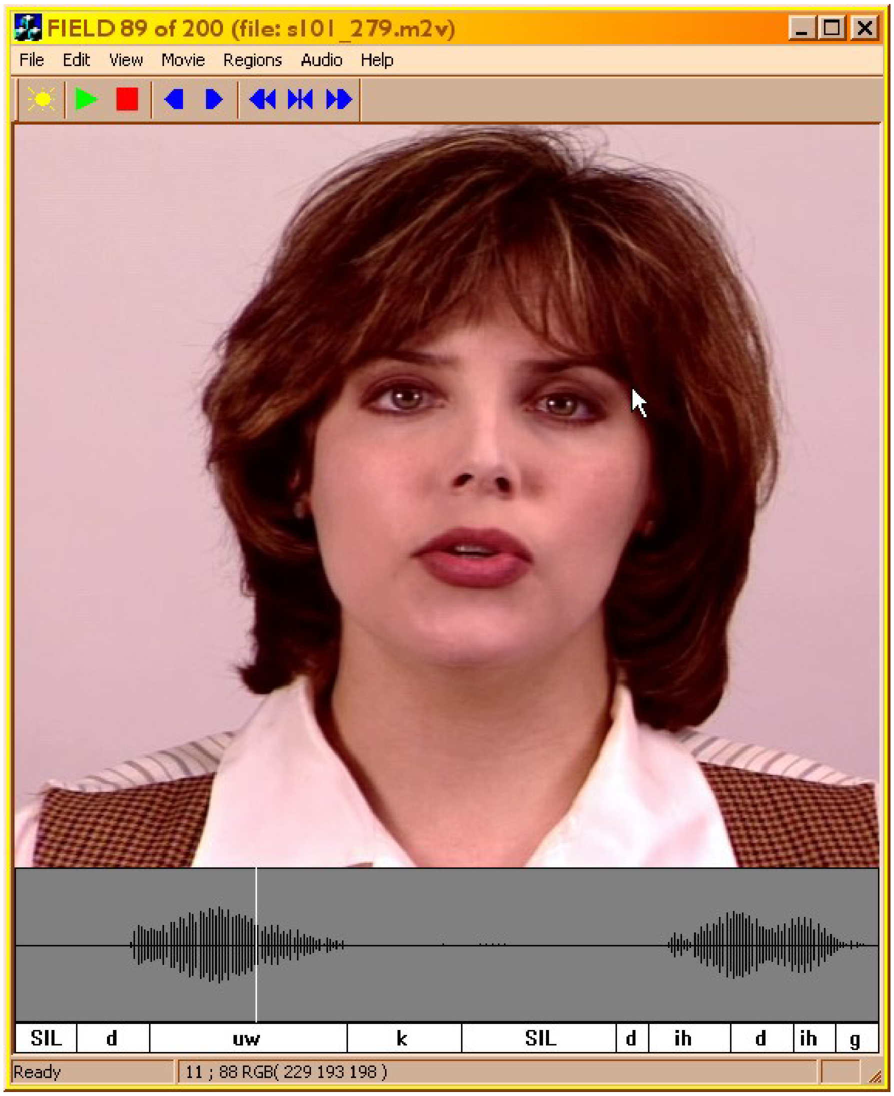

FIELD 89 of 200 (file: s101_279.m2v)
File
Edit
View
Movie
Regions
Audio
Help
SIL
d
uw
k
SIL
d
ih
d
ih
g
Ready
11 ; 88 RGB(229 193 198)

Figure 2.16. A screenshot of the examiner tool. Once a sequence (MPEG2 video file, PCM audio file and ASCII phoneme file) is loaded, simple keyboard or mouse commands allow navigation within a sequence. Below the video frame is the audio context showing the sound level at each sample. The extent of the context is variable and can be adjusted interactively. Below the audio context is the corresponding phonetic context showing the position and extent of each phoneme.

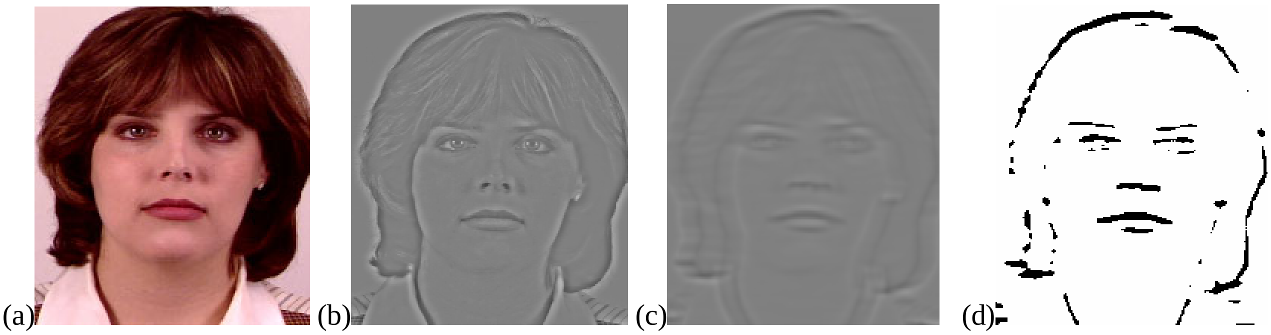


Figure 2.17. Steps of Shape Analysis. In (b) the luminance channel of the original image has been filtered to remove low frequencies. This is implemented by subtracting a running averaging window. The range of frequencies is controlled by the size of the averaging window. Here we used a 11x11 window. In (c) the high-pass filtered image has been convolved with a structuring kernel (a 17x3 rectangle) to enhance facial elements such as mouth eyes and eyebrows that exhibit such a rectangular shape. Finally in (d) the image has been thresholded adaptively.

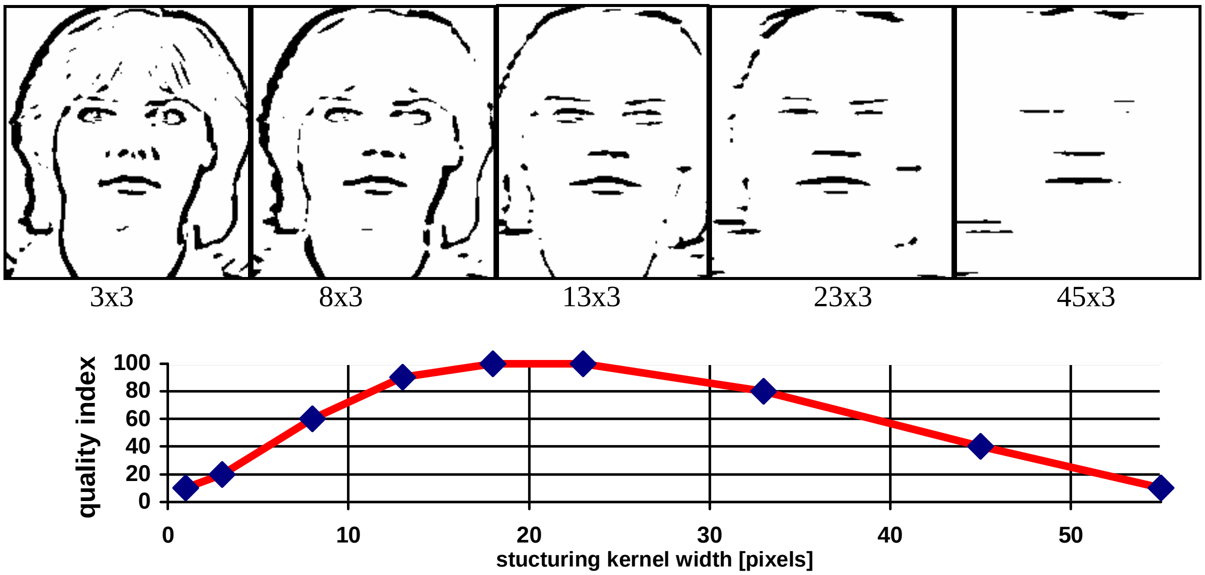


Figure 2.18. After applying a high-pass filter, a structuring kernel is used to enhance areas of a given size. Recognizing that areas of interest in the face are typically elongated horizontally, we choose a rectangular structuring kernel. The graph shows, for kernels of different sizes, the proportion of facial elements that are represented on the thresholded images by a single connected component of a correct size. The optimal size, as shown by the graph, is in accordance with the intuitive ‘guess’ of a kernel size matching the average size of facial parts of interest.

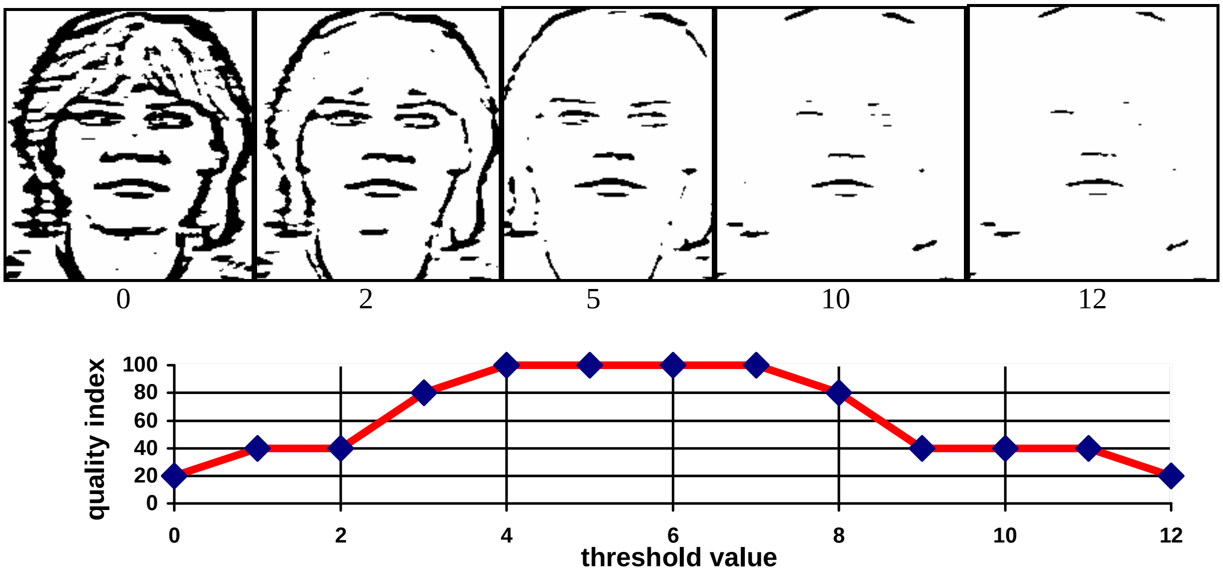


Figure 2.19. After applying a high-pass filter and a structuring kernel, adaptive thresholding is applied. The optimal value for the threshold is obtained as follows. First, the position and size of some facial elements (eyes, nostrils mouth) are marked, then the processing is applied and connected components are extracted. The quality index on the graph reflects the proportion of facial elements found (right position and size).

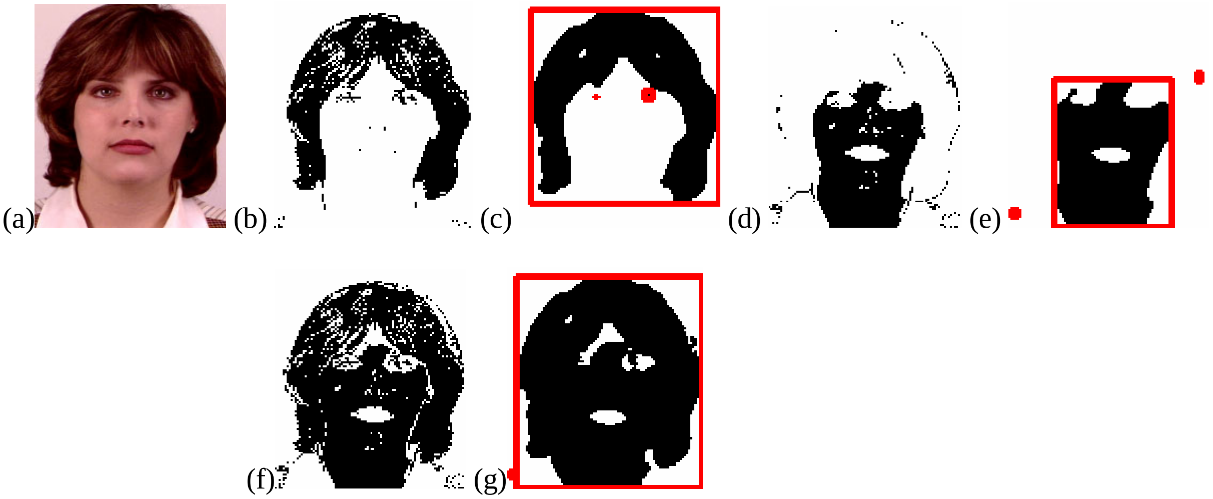


Figure 2.20. Steps of the color analysis. Bounding boxes are first defined in the Hue-Saturation-Luminance (HSL) space that enclose color samples from corresponding areas of the image. These bounding boxes are created interactively in an initialization phase. In (b), (d) and (f), the original image (a) is thresholded with black pixels being within, respectively, the hair bbox, the skin bbox, the union of the hair and skin bboxes. In (c), (e) and (g), the thresholded images are smoothed and connected components analysis is performed to extract the position of, respectively, the hair, the skin and the whole head.

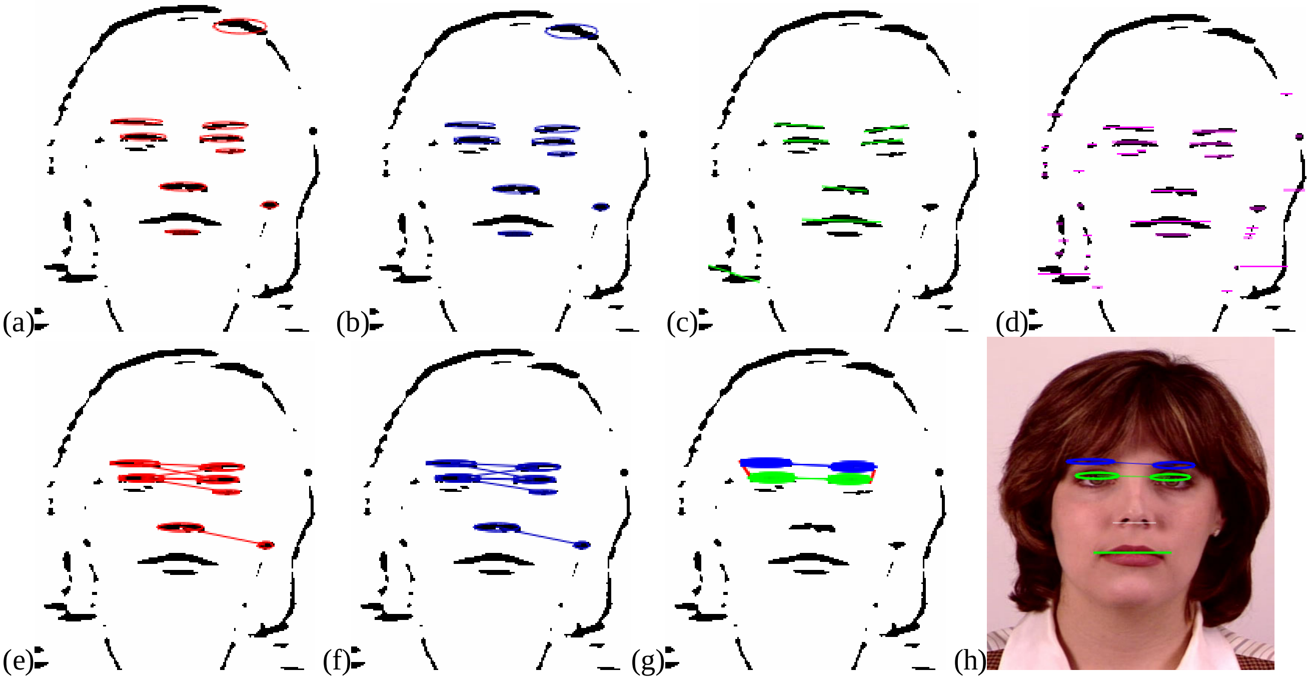


Figure 2.21. Steps of filtering and n-gram analysis. In (a) candidates blobs for an eye are marked by a red ellipse, in (b) candidate brows by a blue ellipse, in (c) candidate mouths by a green line, in (d) candidate nostril-pair by a magenta line. At this stage, blobs have been filtered for size, ratio and position using the head model. In the next step, eye-pairs (e) and brow-pairs (f) are evaluated and filtered for relative position and alignment (angle) using the head model. In (g) the combination of eye-pair and brow-pair are evaluated and once again filtered for relative position using the head model. Finally in (h) the combination of eye-pair, brow-pair, mouth and nostril-pair is evaluated and filtered using the head model.

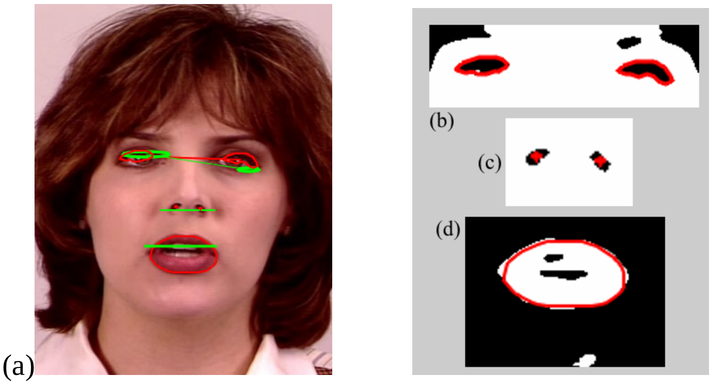


Figure 2.22. Color analysis of the facial parts. In a previous step, the rough position of the eyes, nostrils and mouth have been determined (green overlays in (a)). Using this information, windows are cut out of the image and color analysis is performed. In (b) the non-skin colors are shown in black and their contour is extracted (red). In (c) nostril colors are shown in black and their center of mass is shown in red. In (d) the lip color is shown in white and its contour extracted in red. Color analysis is performed by first defining bounding boxes of characteristic colors in the Hue-Saturation-Luminance (HSL) color space. This is typically done interactively on several representative frames. Then patches of consistent colors can be segmented out based on whether they fall within the bounding boxes or outside.

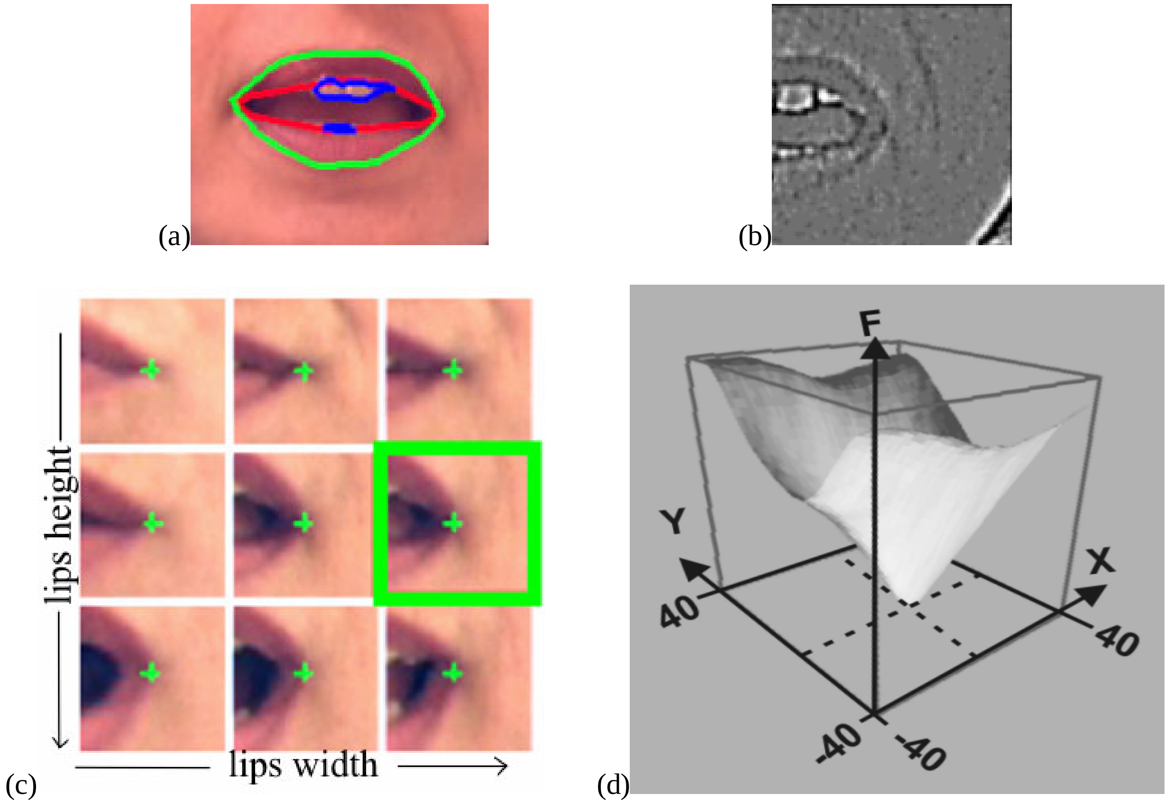


Figure 2.23. Precision analysis of feature points. In previous steps of the analysis process, facial parts have been measured; in this example, the outline of the lips has been measured (a). This analysis starts by defining an n-dimensional grid of samples, each axis of the grid being a measured feature of the facial part; in this example, a two-dimensional feature grid is defined, with the x-axis being the width of the lips and the y-axis, the height of the lips. This grid is first populated automatically with several candidates in each node, then the best suited candidate can be chosen interactively using a graphical interface. Using this grid, images are then analyzed by convolving the selected kernel from the grid (green square in (c) based on the measurements of the facial part) over the facial part area. To improve results, we use a high-pass filtered image (b) for the convolution operation. The result of the convolution exhibits a strong peak at the matching point (d) with monotonically decreasing slopes, making the use of a conjugate gradient descent approach feasible.

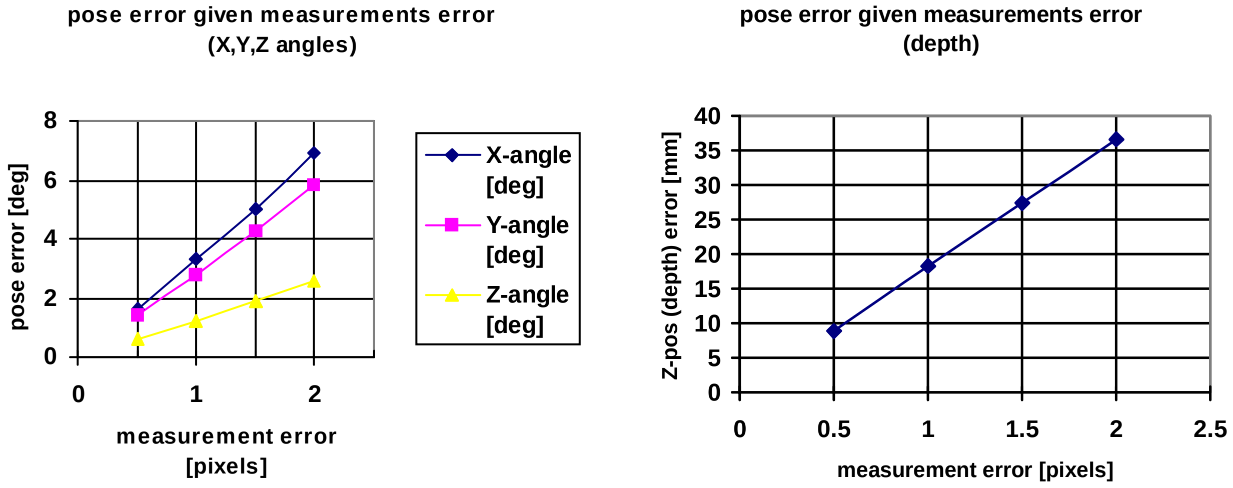


Figure 2.24. Pose estimation accuracy. These charts show the errors in angle (left) and depth (right) estimation, given the error in the measured feature points. These are averaged values over all possible combinations of measurement errors involving a total of 0.5, 1, 1.5 and 2 pixels respectively. Six feature points are used: four eye corners and 2 nostrils.

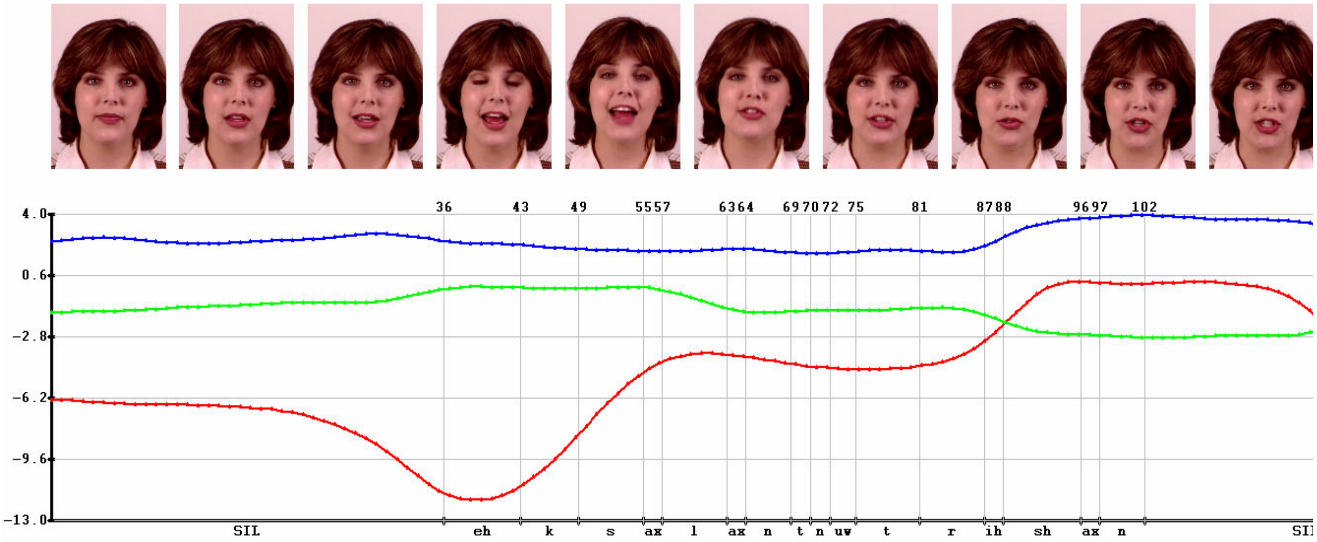


Figure 2.25. Pose estimation example. This graph shows the output of the pose estimation process. In red is the rotation angle around the x-axis (yaw), in green, around the y-axis (pitch) and in blue, around the z-axis (roll). A picture of the recorded head is shown every 183 milliseconds. The vertical axis of the graph shows the angles in degrees, while the horizontal axis shows time (bottom are phoneme boundaries, top are frame number (at 60 frames per second)). The clip's duration is 2.2 seconds.

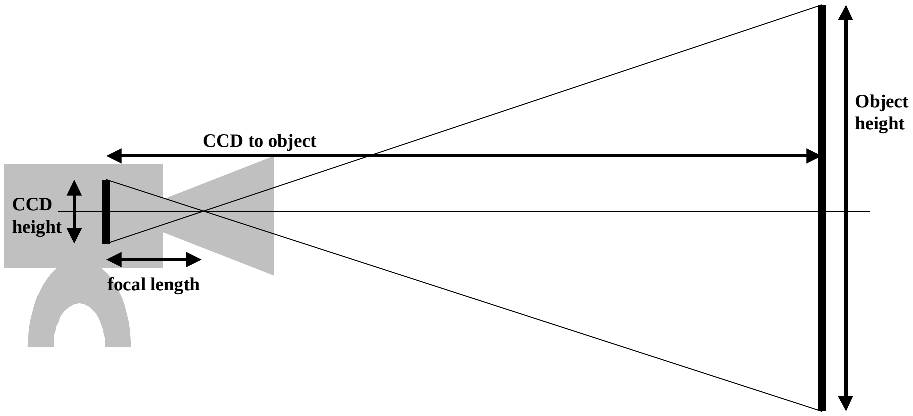


Figure 2.26. Cameral model and calibration. Knowing the distance from the CCD to the object, the size of the object and the focal length, one can easily infer the size of the CCD. Then, from the resolution of the CCD in pixels, we derive two camera calibration values: *pix_per_mm_x* and *pix_per_mm_y*. These are used to convert pixel positions into metric units.

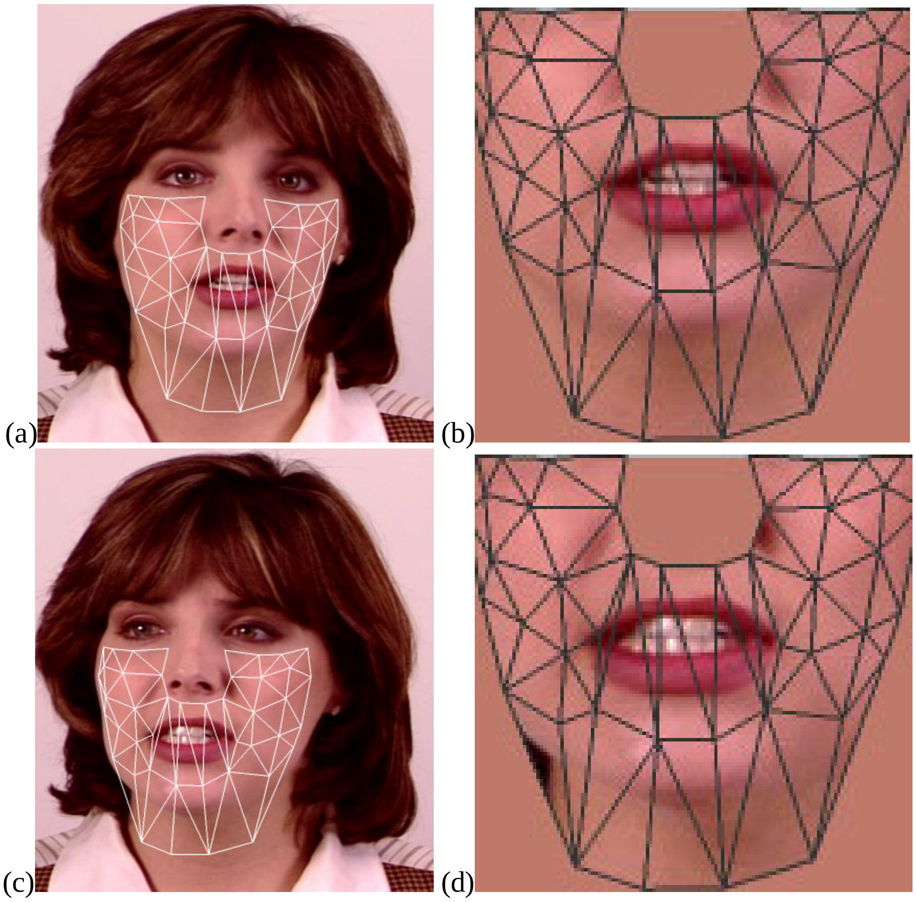


Figure 2.27. Sample extraction and normalization. In (a) and (c), the 3D model of the mouth is projected onto an image using the head pose information. In (b) and (d) the same 3D model is projected using a normalized pose (here, all angles at zero). Pixels are then extracted from the image into the normalized sample using traditional texture mapping techniques.

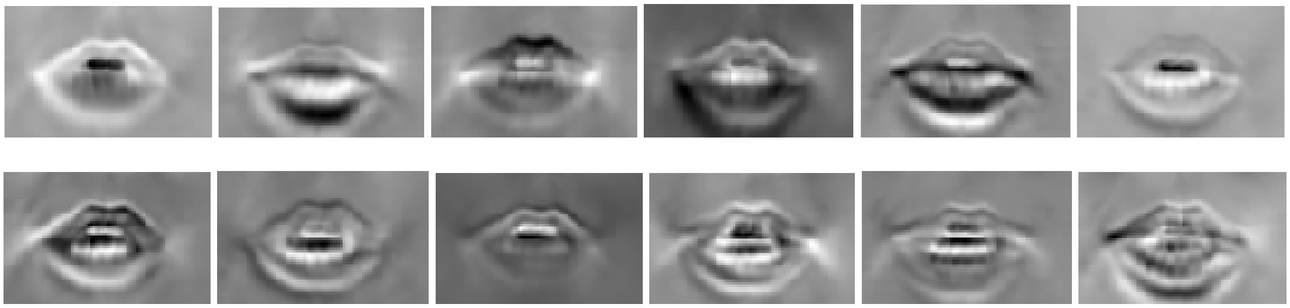


Figure 2.28. Principal Component Analysis. These are the first twelve principal components of the database of 200K normalized mouths. Only the luminance channel is used for the analysis. The resolution of the normalized mouths is 86x54 pixels.

CHAPTER 3

ANIMATION OF THE HEAD MODEL (SYNTHESIS)

Table of Content

3.1 Introduction....57
3.2 Architecture....58
3.3 Lip Synchronization....58
    3.3.1 Introduction....58
    3.3.2 Obtaining a target phoneme input....59
    3.3.3 Building a Search Graph....60
    3.3.4 Calculating the Target Cost....60
    3.3.5 Pruning the Graph....61
    3.3.6 Visual Features....62
    3.3.7 Calculating the Concatenation Cost....62
    3.3.8 Viterbi Search....63
    3.3.9 Transitions....63
    3.3.10 Objective Evaluation....64
    3.3.11 Subjective Evaluation....65
3.4 Visual Prosody....65
    3.4.1 Prosody of Head Movements....66
    3.4.2 Analysis of Head Movements during Speech....66
    3.4.3 Visual Prosody using a library of standard normalized motions....67
    3.4.4 Visual Prosody from Recorded Video Segments....67
    3.4.5 ‘Mechanical’ Facial Motions....67
    3.4.6 Inserting Expressions....68
3.5 Implementation....68
    3.5.1 Development Environment....68
    3.5.2 Real-Time Synthesis....69
3.6 Applications....70
    3.6.1 Automated Newscaster....70
    3.6.2 Help-Desk....71
3.7 References....72
3.8 Figures....74

3.1 Introduction

In Chapter 2, we have described a model and its associated creation process that allows synthesis of photo-realistic talking-head animations. In this chapter we will discuss how talking-head animations (and in particular speech animation) are created using the model. In section 3.2 , we briefly describe the architecture of the overall synthesis module, essentially illustrating the flow of operations and data between the model creation (off-line part; see Chapter 2) and the synthesis (on-line part; see Chapter 3).

As previously discussed, we are using a sampled-based or data-driven approach to talking head animation. Inherent to this approach is the photo-realistic appearance of the head model. Because of the realistic appearance, expectations for its animation are raised considerably compared to synthetic-looking or cartoon-like models. Because in our system the talking head looks like a real person, we expect its animation to be equally realistic. In fact, it has been shown that 'schizophrenic' animations, where one aspect of the model is realistic while another appears synthetic or cartoonish, are disliked. For example, wireframe head models are often texture-mapped with real photographs. When these models are speech animated, suddenly they look ridiculous, because the mouth animation is not as realistic as their appearance. In fact, this has been exploited for entertainment purposes in our PlayMail [30] system where people can create their own models with a single image and a few mouse clicks and get them speech animated and sent via email. Even though the animations are far from realistic, this service has been quite successful and people use it for its entertainment value, essentially making fun of the models being animated. However in more serious applications such as customer relation management (CRM), the focus is not entertainment but rather trust building, brand-awareness and guidance, all of which call for a realistic virtual agent. There are several requirements that need to be met for a speech animation to be labeled video-realistic. First and foremost is the realism of the lip movements and how well they are synchronized with the audio track. A realistic mouth animation involves the entire vocal apparatus: lips, jaw, teeth and tongue need equal attention and need to be precisely modeled and animated. The high plasticity of the mouth and the influence of coarticulation make it all but impossible for a fully model-based system to produce realistic animations. Again, the advantage of the image-driven approach is to bypass modeling by capturing the entire mouth area into image samples. The difficulty then resides in concatenating small units of recorded video speech into a new animation that is both synchronized with the audio and smooth in appearance. Essentially, our approach is to build an animation graph in which elemental units of video speech are linked into a multitude of possible animation paths. The optimal path is then calculated based on costs assigned to each node and arc of the graph. The synthesis of lip movements is described in section 3.3

Realistic and synchronized lip animation is a necessary condition for a realistic animation, but it is by no means sufficient. To produce believable animations, the motion of the entire head, as well as of the eyes and eyebrows has to be synchronized with the speech in a natural and meaningful way. Analog to the prosody in speech, we address visual prosody in section 3.4 . While lip movements are directly related to the production of sounds, the movements of the head, eyes and eyebrows are related to the meaning or the context of speech and ultimately linked to mood and personality. This high-level aspect of the behavior of virtual agents is not addressed in this work. For very interesting forays into that universe, the reader is referred to work by MIT AI lab [36] and MIRALab [19]. We instead concentrate our efforts on speech-related facial expressions (section 3.4.6).

In section 3.5 we discuss the implementation of the entire talking-head system. The overall size of the code base exceeds 100K lines of C++ code. This is a rather large system and hence has been implemented in a modular way. A MATLAB-style development environment has been developed to allow flexible access to the algorithms via a scripting language. Integral part of this development environment is also a powerful GUI (Graphical User Interface) that allows interactive display and manipulation of graphical objects of both 2D and 3D nature. This development environment has proved crucial in successfully developing and combining a wide range of algorithms (image recognition as well as image synthesis) into dedicated real-time applications.

In section 3.6 we describe two different applications that we have implemented to demonstrate the talking-head technology. These applications are of an exploratory nature. Common use of talking heads, and particularly video-realistic talking heads, is only beginning to emerge and it is still unclear in what areas and for what purpose they will provide superior user interfaces. Hence the need to implement such demo applications and collect users' feedback.

3.2 Architecture

The architecture of the synthesis module is illustrated on Figure 3.1 . The diagram shows two main elements. The off-line part describes the flow of operations involved in creating the talking-head model from video recordings. The on-line part describes the flow of operations that starts from text or from a labeled audio request, and results in a speech animation.

3.3 Lip Synchronization

3.3.1 Introduction

One of the most important but also most elusive aspects of speech animation is the synchronization of lip movements on the visual channel with the acoustic channel, often referred-to as lip-synch. Even though we do not actively track the movements of the lips as we listen to speech, we often subconsciously use the lip movements to disambiguate acoustic information. People with hearing impairments use the visual lip information more actively and, using contextual clues, can often lip-read entire conversations. We are all highly trained in associating lip movements with acoustic speech and hence are very quick at noticing any discrepancy. Poorly synchronized or dubbed movies, for example, are generally disliked, in particular in America where people are not used to them (in Europe, where dubbed movies are more frequent, people are seemingly more tolerant to loss of lip synchronization). A speech animation is perceived either to have lip-synch or not to have lip-synch. There is unfortunately no graceful degradation of perceived quality. Viewers typically can tell something is wrong with the animation but are generally unable to pinpoint the exact moments when there is a mismatch between audio and visual stimuli. When lip-synch is lost, a viewer will ignore the lip movements and instead focus on the audio or the body language. A typical example is video-conferencing over ISDN connections where the reduced frame-rate results in loss of lip-synch. Users often resort to wide-angle framing, thus avoiding close-ups where the loss of lip-synch would be more noticeable.

While a real person produces speech by positioning the vocal apparatus to produce particular sounds, a synthetic character is typically animated using information from the sound channel (either recorded audio or synthesized text-to-speech (TTS) audio). A system that synthesizes such speech animations has to learn a mapping from the audio channel to the visual channel. Researchers have devised different solutions to this mapping problem. Early work by Parke [29] uses a simple mapping from phonemes to morph targets and linear interpolation to animate a simple wire-frame mouth model. This approach is still widely used in cartoon animation [22]. Cohen and Massaro [5] refined that model by incorporating coarticulation. In their model, morph targets are modeled by a set of action parameters (lip pull, tongue forward, jaw drop, etc.), then each phoneme is given coarticulation strength, extent and weight for each action parameter. Coarticulation is modeled by summing the contribution of neighboring phonemes weighted by an exponential decay. The weights for the action parameters were coded manually by observing people speaking.

Modeling the complex dynamics of speech articulation, however, not only requires a complete understanding of the mechanical aspects of the vocal apparatus but also a deep understanding of how the brain has adapted to it to produce speech in an optimal way. Hence, despite a high degree of sophistication, these coarticulation models often produce very contrived-looking (under-articulated) and poorly synchronized lip animations that appear far from video-realistic.

More recently, researchers have presented systems that attempt to learn a coarticulation model from recorded data. Matt Brand has demonstrated a system called "Voice Puppets" [3], in which a direct mapping from the audio track to action parameters is learned. This system is useful because it completely bypasses the phonetic level, mapping directly from the waveform to facial actions. Unfortunately, it is only capable of learning a limited number of facial action parameters, making it unsuitable for realistic lip animation. In fact, no trainable coarticulation model has been shown that is able to learn a sufficient number of parameters from data to create realistic lip animations.

A noteworthy attempt at photo-realistic talking-head animation by Ezzat et al. [8] uses morphing of key-frames to produce transitions between a low number of recorded visemes. While producing smooth-looking animations, this approach results in poor lip-synch because the dynamics of coarticulation aren't captured by the essentially linear transition scheme. In a more recent work [9], the authors model coarticulation by synthesizing trajectories in a 46-dimensional space of morph-targets. Phonemes are represented in this space by multidimensional Gaussians, the magnitudes and variances of which are learned from recorded data. The variance of a Gaussian implicitly models coarticulation effects, albeit in a rather under-articulated way.

Instead of Ezzat's approach, we favor a method borrowed from the area of text-to-speech (TTS) synthesis called unit selection [18]. In this method, acoustic units of speech are recorded into a database and later reassembled to form new speech. Target and transition costs are defined and the optimal concatenation (as defined by the cost function) is obtained using a Viterbi search [37]. This approach has recently produced TTS systems with very natural sounding voices [2]. In Video Rewrite [4], a system that allows visual dubbing, Bregler et al. use this approach to produce realistic animations. Their solution uses video units spanning three phonemes (triphones). Such long units result in a large database (there are about 40 phonemes in the English language, theoretically resulting in 40^3) and a limited number of available candidate per given triphone, in turn, making it necessary to use extensive smoothing for the transitions between concatenated units. This typically reduces the quality of the lip-synch.

Our approach also relies on recorded units of video. See chapter 2 for details on recording, extracting and normalizing video samples of the mouth and other facial parts. Because the video units of speech are recorded from a real person speaking, they are inherently realistic. Furthermore they capture the particulars of the person being recorded, each individual having its own way to articulate speech. The difficulty then resides in selecting the optimal sequence of units from the database and to concatenate the selected units in a way that the final animation appears smooth, free of jerks or harsh transition from one unit to the next, all the while being in synchrony with the audio track. Instead of using fixed units spanning diphones or triphones (as in Video Rewrite), we use variable length units and let the system decide on the optimal concatenation. We use a combination of three costs to achieve both lip-synch and smoothness with minimal need for blending between recorded video segments. The resulting animations appear both realistic and lip-synchronized.

While ultimately, speech animations have to be evaluated by subjective testing, it would be too costly to tune every parameter of the unit selection process that way. In particular, when computation complexity has to be reduced for real-time applications, a trade-off has to be made between quality and speed. Such trade-off is accomplished by varying parameters of the unit-selection module. There are several such parameters, making it impractical to tune them manually via subjective testing. Instead we define an objective quality measure that can be computed automatically based on smoothness and synchronicity. Using this measure, we have been able to find threshold values for several parameters and quality/speed ratios.

3.3.2 Obtaining a target phoneme input

A speech animation starts with a target phoneme string. This phoneme input can be obtained either from the output of a Text-to-speech synthesizer (TTS) or from recorded and labeled speech. Most TTS systems use internal phonetic representations to produce audio speech, and hence this information is readily available. In Microsoft's speech interface (SAPI) [32], which has become a de-facto standard interface for TTS systems, phonemes can be requested from the TTS module. Phonetic information consists of set of phonemes and their duration. In this work, we use the DARPA phonetic alphabet (see section 2.4.2).

While TTS systems have made tremendous progress in naturalness, they are still a far cry from real recorded speech. In instances where the most natural animations are needed and where offline recording of audio speech is affordable (such as pre-recorded prompt of a company spokesperson), labeling recorded speech audio is desirable. In fact, a common remark among early users of our system was that they disliked the discrepancy between the very natural looking agent and a TTS voice that sounded somewhat synthetic. With recent advances in naturalness, the discrepancy has been reduced, but unfortunately hasn't completely disappeared.

To accommodate both TTS and recorded audio, our system uses phonetic labeling, providing a single, convenient interface to both inputs. Recorded audio has to be labeled using a phonetic labeler or aligner. These tools are readily available (for example the CMU SPHINX-II system [17] for speaker-independent labeling and aligning). Because we need to label a large corpus very precisely, we instead use a trainable aligner tool (an in-house version derived from Entropics' "Aligner" [39]). In our recordings, we had the talent speak for about two hours, providing enough speech data to train the aligner. Recorded speech has so far not been the focus of our research; hence we haven't invested resources into speech labeling. In fact, phonetic labelers are used for automatic speech recognition (ASR) and thus are designed to recognize the entire phoneme set. In our case, we are rather interested in the visual mapping of the audio channel and hence would train a labeler specifically to recognize visually different phonemes.

3.3.3 Building a Search Graph

At this point we have obtained a phonetic target defining the duration of each phoneme of the speech to be animated. From the total duration of the target speech, we derive the number of video frames in the final animation. Speech animations are generally played at 30 frames per second.

We now build an animation graph with n states corresponding to the n frames of the final animation. Each state of the final animation (one video frame) is populated with a list of candidate nodes (a recorded video sample from the database). Each state is fully connected to the next and concatenation costs are assigned for each arc, while target costs are assigned to each node. Figure 3.2 illustrates the structure of such a graph. A Viterbi search trough the graph finds the optimum path (the one that generates the lowest cost).

The task of the unit selection is to balance two competing goals. On the one hand we want to insure lip synchronization. Working towards this goal, the *target cost* uses phonetic and visemic context to select a list of candidates that most closely match the phonetic and visemic context of the target. The context spans several frames to ensure that coarticulation effects are taken into account.

On the other hand we want to ensure smoothness in the final animation. To achieve this goal, we want to be able to use the longest possible original segments from the database. The *concatenation cost* works towards that goal by penalizing segment transitions and insuring that when we need to transition to another segment, we choose a candidate that is visually close to its predecessor, thus generating the smoothest possible transition. This cost has two distinct components (the skip cost and the transition cost) because the visual distance between two frames cannot be perfectly characterized (otherwise, the transition cost alone would suffice). The feature vector that is used to calculate the visual distance between candidates provides only a limited, compressed view of the original, so that the distance measured between two candidates in the feature space cannot always be trusted to ensure perfect smoothness of the final animation. Also the visual distance does not account for the "direction" of change and hence might interpret two successively recorded images to have a large visual distance. The additional skip cost is a hint given to the system that consecutively recorded frames ***are*** smoothly transitioning.

3.3.4 Calculating the Target Cost

The target cost is a measure of how much distortion a given candidate's features have, compared to the target features. The target feature vector is obtained from the phonetic annotation of a given frame of the final animation. The target feature vector at frame t, $\mathbf{T(t)}=\{ph_{t-nl}, ph_{t-nl-1},\ldots,ph_{t-1}, ph_t, ph_{t+1},\ldots ph_{t+nr-1}, ph_{t+nr}\}$ is of size $nl+nr+1$, where nl and nr are, respectively, the extent in frames of the coarticulation left and right of the target ph_t (the phoneme being spoken at frame *t*). A weight vector of the same size is given by ***W(t)***= $\{w_{t-nl}, w_{t-nl-1},\ldots,w_{t-1}, w_t, w_{t+1},\ldots w_{t+nr-1}, w_{t+nr}\}$,

$$w_i = e^{-\alpha|t-i|}, i \in \left[t - nl; t + nr\right]$$

Equation 3-1

This weight vector simulates coarticulation by giving an exponentially decaying influence to phonemes, as they are further away from the target phoneme. Typical value for *nr*, *nl*, α are 10, 10, 0.3.

For a given target and weight vector, the whole database is now searched to find the best candidates. A candidate extracted from the database at frame *u* has a feature vector ***U(u)***=$\{ph_{u-nl}, ph_{u-nl-1},\ldots, ph_{u-1}, ph_u, ph_{u+1},\ldots ph_{u+nr-1}, ph_{u+nr}\}$. It is compared with the target feature vector. The target cost for frame *t* and candidate *u* is given by:

$$TC(t,u) = \frac{1}{\sum_{i=-nl}^{nr} w_{t+i}} \sum_{i=-nl}^{nr} w_{t+i} \cdot M(T_{t+i}, U_{u+i})$$

Equation 3-2

where $M(ph_i\,, ph_j)$ is a *p*x*p* "viseme distance matrix" where *p* is the number of phonemes in the alphabet. This matrix denotes visual similarities between phonemes. For example, the phonemes {m,b,p}, while different in the acoustic domain, have a very similar appearance in the visual domain and their "viseme distance" will be small. This matrix is populated fully automatically from the recorded database. For every phoneme of the recorded dataset, a visual feature vector is extracted from the image sample corresponding to the center point of the phoneme segment. Then, a statistical analysis is performed and a median visual feature vector is obtained for each phoneme of the alphabet. The matrix is then filled using the Euclidian distance in visual feature space. This way, the system is also able to adapt to each individual's particular way of articulating. For some people, the visual appearance of the phoneme 'o' might be close to that of an 'a', while others might articulate the phoneme 'o' in a significantly more rounded fashion. This adaptability of the unit selection module is key to synthesizing smooth animations for a variety of speakers.

In summary, the target cost *TC* is a measure of similarity of the ***audio-visual coarticulation context*** between a database candidate and the target. See Figure 3.3 for an example of target cost calculation.

3.3.5 Pruning the Graph

The complexity of the Viterbi search goes in $O(n*c^2)$ where n is the number of states and c is the average number of nodes per state. This would make it impractical to populate each state with all the candidates from the database, which contains about 80 thousand recorded samples. Another consideration is the avoidance of the full traversal of the database for each state of the graph. To solve this complexity issue, we pre-compute a lookup table with entries corresponding to a chosen visemic classification. Each entry is populated with a list of database samples that match its viseme. For example, the closure viseme, corresponding to the phonemes 'm', 'b', 'p' is populated with the samples, which were labeled with those phonemes. Then, when populating a state of the graph, only the list matching the states' target viseme is searched for candidates. Using phonemes instead of visemes to index this table lookup would run the risk of missing valid candidates for a given target, given the relatively small size of the database.

In situations where processing time supersedes the precision of the result, it is desirable to further prune the graph before performing the search for the shortest path. Since the target cost is a measure of the acoustic match of the candidate, we use this value to rank candidates. As they are inserted into the graph, they are put into a sorted list. The list is then pruned to a length ("beam width") given by the following equation:

$$length = \begin{cases} minlen \quad \text{when} \quad head\_of\_list \le mincost \\ \dfrac{(maxlen - minlen)\cdot(head\_of\_list - mincost)}{(maxcost - mincost)} + minlen \quad \text{when} \quad head\_of\_list \le maxcost \\ max\_len \quad \text{otherwise} \end{cases}$$

Equation 3-3

where *head_of_list* is the cost at the head of the sorted list (highest cost), *maxlen* and *minlen* are, respectively, the maximum and minimum desired length of the list, *mincost* and *maxcost* control the shape of the pruning as illustrated in Figure 3.4 .

3.3.6 Visual Features

We define a distance between two candidates in the graph based on the visual dissimilarities between them. For this purpose, we use a set of features that capture the visual appearance of the lips, jaw and inner mouth. To capture the overall shape of the lips, we use their measured width and height (see section 2.6.2). To capture appearance of the inner mouth, we use PCA coefficients (see section 2.6.5). Each feature is given a weight proportional to its discrimination ability, which is calculated fully automatically using the recorded database. Using their phonetic annotation and a given visual feature, the overlap of phoneme classes is evaluated in the one-dimensional space of the selected feature. The discrimination factor of a visual feature is inversely proportional to the sum of the overlap of phoneme classes. Figure 3.5 shows the overlap of phoneme classes for one of the features. Figure 3.6 shows the distance of several candidates to a target using the Euclidian distance in the visual feature space.

3.3.7 Calculating the Concatenation Cost

A concatenation cost is now set for all arcs of the graph. The concatenation cost measures the visual distance between a candidate of a given state and a candidate of the previous state. The candidates, *u1* (from state *i*) and *u2* (from state *i-1*), have a feature vector *U1, U2*, respectively. The concatenation cost is then given by *CC(u1,u2)=f(U1,U2) + g(u1,u2)*, where:

$$f(U1,U2) = \frac{1}{\sqrt{k}}\sqrt{\sum_{i=1}^{k}(U1_i - U2_i)^2}$$

Equation 3-4

is the Euclidian distance in the feature space. This cost reflects the visual difference between two candidate images as captured by the chosen visual features. Alternatively, when speed is critical, we use the block distance:

$$f(U1,U2) = \frac{1}{k}\sum_{i=1}^{k}\left|U1_i - U2_i\right|$$

Equation 3-5

$$g(u1,u2) = \begin{cases} 0 \quad when\left|fr(u1) - fr(u2)\right| = 1 \quad \text{AND} \quad seq(u1) = seq(u2) \\ w_1 \quad when\left|fr(u1) - fr(u2)\right| = 0 \quad \text{AND} \quad seq(u1) = seq(u2) \\ w_2 \quad when\left|fr(u1) - fr(u2)\right| = 2 \quad \text{AND} \quad seq(u1) = seq(u2) \\ \ldots \\ w_{p-1} \quad when\left|fr(u1) - fr(u2)\right| = p-1 \quad \text{AND} \quad seq(u1) = seq(u2) \\ w_p \quad when\left|fr(u1) - fr(u2)\right| \ge p \quad \text{OR} \quad seq(u1) \ne seq(u2) \end{cases}$$

Equation 3-6

where $0<w_1<w_2<\ldots<w_p$, $seq(u)$ = *recorded_sequence_nb* and $fr(u)$ = *recorded_frame_nb,* is a cost for skipping consecutive frames of a sequence. This cost helps the system avoid switching too often between recorded segments, thus keeping as much as possible of the integrity of the original recordings. We set $p=5$ and let the w_i increase exponentially. In this way, the small cost of w_1 and w_2 allows for varying the length of a segment by occasionally skipping a frame or repeating a frame to better fit the target. The high cost of w_5, however, ensures that skipping more than 5 frames incurs a high cost, avoiding jerkiness in the resulting animation.

3.3.8 Viterbi Search

The graph G={$S_0, S_1,\ldots, S_n$} with states S_i and candidates $S_{i,j}$ has been constructed with a target cost *TC* for each candidate and concatenative cost *CC* for each arc going from state S_i to state S_{i+1}. A path {$p_0, p_1,\ldots, p_n$} through this graph generates the following accumulated cost:

$$c = WTC \cdot \sum_{t=0}^{n} TC(t, S_{t,p_t}) + WCC \cdot \sum_{t=1}^{n} CC(S_{t,p_t}, S_{t-1,p_{t-1}})$$

Equation 3-7

The best path through the graph is the path that produces the minimum cost. The weights *WTC* and *WCC* are used to fine-tune the emphasis given to concatenation cost versus target cost, or in other words to emphasize acoustic versus visual matching. A strong weight given to concatenation cost will generate very smooth animation but the synchronization to the speech might be lost. A strong weight given to target cost will generate an animation which is perfectly synchronized to the speech but that might appear visually choppy or jerky due to a high number of switches between database sequences. Determination of these coefficients is accomplished by subjective and objective evaluation of test sequences (see section 3.3.10 and 3.3.11).

There are c^n different paths through a graph with n states and c candidates per state. However, by evaluating for each node the lowest cost to the previous state, all subsequent evaluations of paths through that node from its previous state are reduced to that arc with the lowest cost. By (forward) iterating through the graph and keeping only the lowest cost arc from each node to its previous state, and then backtracking the optimal path, the entire evaluation of the graph is reduced to $n*c^2$ paths. Figure 3.7 shows the best path and the associated node and concatenation costs for the text ‘Upton’.

3.3.9 Transitions

Once the best path through the animation graph has been identified, an animation ‘script’ is obtained that designates for each frame of the animation, which particular image sample has to be retrieved from the database. Because of the effect of the skip and concatenation costs, the animation script is made of segments of consecutively recorded image samples. The Viterbi search has done its best to minimize visual differences at the junction of these segments; however, because the database has a limited size, many junctions remain for which a large visual difference exists. These abrupt transitions are generally noticeable on the final animation and greatly contribute to lower its overall quality. We use gradual blending of the entire image samples to smooth the animation over these junctions. In fact the animation ‘script’ contains two tracks: the main track and the blending track, as well as a blending factor. Figure 3.8 shows an example of such an animation script. The blending factor gradually blends a segment into its neighbor over a series of *k* frames. The length *k* of the transition is proportional to the visual distance at the junction of the segments but is bounded by *minlen* and *maxlen* parameters.

We have experimented with two types of blending. Morphing provides very smooth transitions but is expensive to compute and also tends to change the dynamics of the animation, sometimes resulting in poor lip-synch. Also, for large visual discontinuities, morphing might produce noticeable artifacts. Instead we prefer pixel lending, which is very cheap to compute, simulates motion blur and does not change the dynamics of the mouth articulation. Figure 3.9 illustrates qualitatively the difference between morphing and pixel blending.

3.3.10 Objective Evaluation

Inherent to the unit selection process is a kind of unpredictability. With a database of over 80 thousand samples, the animation graph can grow quite large and results in many possible paths. Even though there is only one ‘best’ path, there might be several, quite different paths through the graph that produce very similar scores. A slight change in one parameter of the search might produce a new animation that, while qualitatively similar, is in fact made of a majority of new image samples. Ultimately, the only way to evaluate the quality of a parameter set is by subjective evaluation. However, these tests are quite costly in time and manpower (plus, you quickly run out of subjects willing to participate in the evaluations). It is therefore necessary to have a set of objective evaluator functions that can assign scores and quickly diagnose defects in a synthesized animation.

First, let’s review the free parameters of the unit selection process. They can be categorized in two classes. The first class is designed to trade off lip-synchronization for smoothness and includes the following parameters: the skip cost weights (see section 3.3.7) and the weight given to the visual distance in feature space (we assume a target cost weight of 1). The second class of parameters trades off quality for speed and includes the size of the database and parameters controlling the pruning of the graph (see section 3.3.5). The two classes of parameters can be tuned independently.

The trade-off between smoothness and lip-synchronization is in fact very subjective and application dependent. To compute smoothness we use a combination of two evaluators: the average segment length and the average visual distance between segments. A segment is a chunk of consecutively recorded image samples. By assigning a large weight to the skip cost, the unit selection process is encouraged to choose longer segments from the database. Having longer segments brings the animation closer to recorded quality and minimizes the number of transitions where visual jerks might occur. However, putting too much emphasis on the skip cost will force the unit selection to pick longer and longer segments, disregarding the target cost (acoustic match) and resulting in a badly synchronized animation. Figure 3.10 shows the relationship between the average segment length and the average target cost for a set of 20 different animations.

The average visual distance between segments is an indication of the actual amount of visual jerkiness present between two consecutive segments. Assigning a large weight to the visual distance at segment’s boundaries will encourage the unit selection to choose segments which endpoints match visually, thus producing a smooth visual transition. However, too strong an emphasis on the visual distance weight will result in forcing the unit selection to pick visually matching segments and ignore their target cost, thus resulting again in a poorly synchronized animation. Figure 3.11 shows the relationship between average visual distance and average target cost for a set of 20 different animations.

The trade-off between quality and speed can be measured quite efficiently using similar evaluators. The average target cost will tell us if the animation is synchronized, the average segment length and average visual distance, if the animation is smooth. Figure 3.12 shows these two evaluators as a function of the database size.

Graph pruning is controlled by four parameters (see section 3.3.5). We define a sparse sampling of the parameter space and then compute four evaluators for each entry: two for smoothness, one for synchronicity and one for the actual size of the graph. By giving weights to these evaluators we compute a score for each set of parameters. Twenty test sequences, totaling 2823 phonetic segments are used for each evaluation. We then find the optimal set of parameters for a given set of weights that gives us a range for refining the procedure with finer sampling of the parameter space. Figure 3.13 and Figure 3.14 show cross-sections of the parameter space for two of the pruning parameters.

To further diagnose problems in the unit selection process, we using a graphical tool that allows us to compare the dynamics of speech between recorded and synthesized speech segments. We first isolate a sentence from the database and synthesize that very same sentence using the remainder of the database, excluding that sentence (“leave-one-out” approach). We then plot the mouth height for both the recorded and synthesized version of the selected sentence. Figure 3.15 shows several examples of such plots. These plots are useful for pinpointing problems such as missed mouth closures, loss of synch or odd movements. The ground truth plot serves as a reference and large discrepancies are indicative of animation problems.

3.3.11 Subjective Evaluation

In the work of Cohen and Massaro [5], talking head speech animations are evaluated on whether viewers can lipread monosyllabic English words as they are articulated by the synthetic talking head. While this study gives a useful insight into the visualization of individual phonemes, it only provides superficial modeling of the coarticulation effect. The evaluation of a talking head has to be made on much longer speech segments than monosyllabic words. We evaluate our system on a set of 15 sentences (averaging 4 seconds each). To evaluate unit selection of speech animations, we only synthesize the mouth area. Hence, we avoid artifacts related to the overall head modeling and animation and rather concentrate on the lip animation. We have conducted several small-scale informal tests involving 3-5 people that helped us focus on specific aspects of the synthesis that weren’t satisfactory. We have also conducted more formal subjective testing involving dozens of participants. These tests, however, were geared towards evaluating animations of the entire head, often within an application. For example, we have tested intelligibility of our sample-based talking-head animation compared with synthetic 3D models and audio-only presentations, and we found significant improvements with the talking heads with the sample-based model edging out the synthetic 3D model. Another test we performed relates to the trust a user gives to a computer system. We found that talking head augmented interfaces significantly increase trust. Complete details of these tests can be found in [28].

An interesting effect that we have observed many times when performing our evaluations is the so-called McGurk effect [23]. Our brain performs a merging of the visual and audio sensory inputs and when presented with discrepancy between both inputs, most of us will perceive a ‘fused’ input, which might correspond to neither the audio, nor the visual input. The classical example comes from McGurk’s experiment where viewers were presented with a visual stimulus of a head articulating ‘GA’ and an audio stimulus saying ‘BA’. A large proportion of the viewers (98%) report perceiving ‘DA’. In our experiments we also noted this effect when, as a result of a problem in the unit selection process, for example, the lip animation of the word ‘message’ missed its closure. All of the viewers could swear hearing ‘Nessage’ and yet, closing their eyes, they would all hear ‘message’. While seemingly anecdotal, the McGurk effect has strong implications for our system because it means that a problem in the unit selection might not only result in some random visual artifacts but might easily produce misunderstandings; a case where adding a visual channel does, in fact, reduce intelligibility. More often, however, does the talking head actually help comprehension of the audio channel.

Another issue worth mentioning is the amount of articulation going into the animation. Beskow et al. [1] have studied the effect of over-articulation and, while they found that over-articulation is preferred by hearing impaired people (a fact that we have confirmed by letting a hearing impaired colleague play with our system), it does not significantly help comprehension and is often disliked in normal situations. On the other hand under-articulation results in dull animations, but is sometimes preferred in order to mask artifacts due to poor modeling of the mouth. In our system, we are of course bound by the recorded samples and cannot completely control the final amount of articulation. However, having had naturalness as our primary goal, we have always asked our talents to speak in a natural way, avoiding under and over articulation.

3.4 Visual Prosody

Visual prosody is the analog of acoustic prosody in the visual domain. It refers to head and facial movements that accompany speech and are often meant to augment it. Head nods, eyebrow raising and frowning, eye rotation, eyelid position, head tilts, smiles, frowns, etc. are only a fraction of facial motions we use on a regular basis when we speak. While some of these facial movements are uncorrelated with the content of speech, others are synchronized and their timing is therefore important.

While acoustic speech prosody has been widely studied, visual prosody has only very recently started to receive attention from researchers. In [16], Granstrom et al. find that visual cues have significant influence on the perception of virtual agents. They rank smile, eyebrows and head movements as the most significant parameters and find a correlation between eyebrow raising and perceived prominence. Costa et al. [7] have demonstrated a system that learns a mapping between the acoustic signal and facial action parameters, in particular movements of the eyebrows.

We can distinguish three types of facial motions: mechanical, emotional and prosodic. Mechanical facial motions are typically serving a mechanical or biological purpose, and hence are essential for conveying a natural talking head. These motions include reactive head motions linked to the rotation of the jaw as well as head motions due to actions and relaxations of the neck muscles, eye-blinks necessary for eye wetting and saccadic eye motions typical of gazing. Because these features are generic to all human beings, we come to expect them from a natural looking talking head and their absence makes the animation look completely fake. Because these movements are happening regularly and are not synchronized with speech production, their timing is not important and they can be modeled using probabilistic state machines. Researchers have actually studied some of the motions and models have been created. For example Lee et al. [20] have studied in detail the saccadic motions of the eyes. Section 3.4.5 describes the implementation of these “mechanical” facial motions.

Emotional facial expressions are related to the mood or emotional state of the speaker, as well as to the high-level textual content of the speech. They vary greatly among individuals and are therefore hard to study systematically. Also, in the context of a virtual agent synthesis, they constitute a large hurdle because they require a high-level understanding of the text being spoken as well as a modeling of the mood and emotional state of the virtual agent. While these expressions are important, they are not essential to a believable speech animation. However, we cannot ignore the expressive power of smiles, frowns, eyebrow raise, eye widening, etc. In many applications it is desirable to enliven speech animation with such expressions. Because there is no system able to perform contextual text understanding, we limit ourselves to inserting manually such expressions in speech animations. Section 3.4.6 describes how expressions can be easily inserted into an animation through manual insertion of markers in the target text.

Prosodic facial movements are used to stress words or syllable, indicating their prominence. They include head motions, such as short nods and eyebrow movements. They happen in synchrony with the speech, making their timing particularly important. In the following paragraphs, we study the correlation between head movements and prosody. Eyebrows movements are somewhat more delicate to animate and often convey more information than a simple emphasis. We chose to animate eyebrows through the same manual marking as emotional facial movements.

3.4.1 Prosody of Head Movements

Understanding quantitatively the correlations between head movements and spoken text is important for synthesizing video-realistic talking heads. Talking heads appear much more engaging when they exhibit realistic motion patterns. For example, head nods often accompany stressed syllables and are sometimes used to emphasize a word.

In our experiments, we found that animations with head movements that are synchronized with the speech are rated higher than those with unsynchronized head movements (where the head movement is extracted from a recorded but unrelated sentence). Random movements (uncharacteristic of real movements) are rated lower and finally a total absence of motion is rated the lowest.

3.4.2 Analysis of Head Movements during Speech

We analyze quantitatively head and facial movements that accompany speech and investigate how they relate to the text’s prosodic structure. On our corpus of recorded data, we use the head pose vector (see section 2.6.3) of entire sentences. On these sentences, we also obtain prosodic annotation from a prosody analysis tool [35]. Using the ToBI set [33] of prosodic events (phrase boundaries and pitch accents, see Figure 3.16 for a list of these events) this tool is trained to predict such events from audio and textual input (and hence it can be used on input sources from both TTS and recorded audio). We then perform a correlation analysis between predicted prosodic events and actual head movements as captured by the pose estimation. Using our recorded corpus of sentences, we analyzed, for four different speakers, the correlation between head movements and prosodic events, and found that a small set of simple motion patterns are repeatedly applied in synchrony with the main prosodic events. Direction and strength of head movements vary widely from one speaker to another, yet their timing is typically well synchronized with the spoken text. Figure 3.17 illustrates the correlation between predicted prosodic events from the ToBI set and actual head movements. We derived the following probabilistic model for simple head movements synchronized with pitch accents '*':

P(^ | *) = 0.42	^ = simple nod
P(~ | *) = 0.18	~ = nod + overshoot
P(/ | *) = 0.2	/ = sudden sideways motion

More details on this study can be found in [14].

3.4.3 Visual Prosody using a library of standard normalized motions

From the analysis described in the previous paragraph, we extract head motions corresponding to the set of prosodic events and store them as vectors in a library. In order to build animations using these recorded vectors, they need to be normalized. In particular, we force the entry and exit points of the motion to correspond to the frontal pose (all angles at zero). Therefore, we find the points, left and right of the prosodic events on the extracted motions in the library, where the pose is closest to the frontal pose and choose these points for the start and end of the motion segments. If these endpoints are not exactly the frontal pose, the motion vector is extended and interpolated (3^{rd} degree polynomial interpolation).

To synthesize a head motion animation, the prosody predictor analyzes the target audio and text and, for each predicted prosodic event, a corresponding head motion is obtained from the prosody library (a particular motion is selected randomly from the list of available motions for that prosodic event). A randomly generated base motion is first created to simulate muscle actions and relaxations of the neck. We use sinusoids of random amplitude and time constant for each angle and translation. The normalized motion segments are then superimposed on the base motion at prosodic events. Figure 3.18 shows a synthesized head motion graph.

The 3D head model described in section 2.3 is used to create animations where the head movements are simulated by applying the prosodic motion. The center of rotation is first defined at the base of the neck. In reality there are several rotation points at each vertebrae articulation along the upper part of the spine. To simulate this high degree of articulation, we allow translations at the base of the neck (shoulders) and rotations at the base of the head. This simplified articulation model is illustrated in Figure 3.19 . The rotation angles and translation components from the synthesized prosodic motion are applied to the head while an additional translation component is applied to the shoulder (this is part of the randomly generated base motion).

3.4.4 Visual Prosody from Recorded Video Segments

When using the 2.5D head model (see Figure 2.2 (m,n)) we have to use a different approach for the synthesis of visual prosody. The 2.5D head model uses a recorded video sequence to render the 'base head'. Hence the library of head motion cannot be used directly to animate the head motion. Instead we use the predicted prosodic events to search the database of recorded sentences for segments where the prosodic content closely matches the target prosody. This simple approach works when target and library segments are of similar length. To create longer animations using this approach is difficult because the visual appearance of subsequent segments is not guaranteed to match, potentially resulting in large artifacts.

3.4.5 'Mechanical' Facial Motions

These motions, serving a mechanical or biological purpose include reactive head motions, eye-blinks and eye globe motions. The small reactive motion of the head results from a sudden opening of the jaw after a closure. Using the phonetic labeling of the target text, we identify such occurrences and trigger a small nod using the same approach as described in the previous paragraphs.

We implement eye-blinks and eye globe motions with probabilistic state machines. Figure 3.20 illustrates such a state machine. These state machines were designed from the observed behavior of these facial elements. To animate eye-blinks, we associate an image sample of the facial part to each state of the state machine and produce the animation by cycling though its states.

To animate the eye and emulate the saccadic eye motion typical of gazing, we model the eye globes separately using two half spheres. An imaginary gaze point is moving in space, characteristic of the direction to which a person looks while focusing alternatively between his or her interlocutor and an imaginary desk. The two eye globes are following the gaze point in a way illustrated in Figure 3.21 . The gaze, additionally to moving back and forth between the interlocutor and the desk, exhibits a random saccadic motion around its main direction. This saccadic movement is characteristic of how people look at an object, constantly shifting the focus point to cover the entire object.

3.4.6 Inserting Expressions

To avoid the complexity (and shortcomings) of high-level textual analysis, we only provide manual insertion of emotional expressions. Markers can be inserted between sentences in the target text. The facial parts, type of expression, length and intensity level are controlled by parameters of the marker. Such markers assume the following form:

"I'm very happy to be here. Expr(mouth, *smile_1*, 1300, 60) However, I'm sad because my sister was not able to join me. Expr(mouth, *frown_2*, 2400, 50) Expr(eyebrows, *frown_1*, 2400, 40)"

During the recording session, we have asked the talent to perform several emotional expressions, including various degrees of smile (from a smirk to a wide laugh), as well as frowns (from sadness to disgust). The talent was asked to always start from a neutral, relaxed state. We have recorded these performances for both the mouth and the eyes. In the given example, these expressions are labeled 'smile_1' and 'frown_2' for the mouth and 'frown_1" for the eyes. In the database, these expressions are stored as integral segments. During synthesis, these segments are inserted between speech animations of sentences. Depending on the desired length and intensity of the expression (for example 1300 milliseconds and an intensity of 60% for the first emotion above) the recorded segment is ramped up, maintained and ramped down. It is also important that the transition from the end of the speech to the beginning of the emotional expression happens seamlessly. The mouth should first transition from speech to neutral and then from neutral to emotion, thus avoiding large visual jerks. We have recorded emotion segments that always start from a neutral state, making it possible to seamlessly insert them into silences of speech animations. Figure 3.22 illustrates this concept.

3.5 Implementation

3.5.1 Development Environment

Over the course of this project, we've developed a large number of algorithms and the code base has grown steadily to over 200K lines of C++ code. Image recognition modules include: connected-components, color-clustering, motion analysis, convolution with multiple kernels, conjugate-gradient descent, n-gram search, shape filtering and clustering, model-based scoring, contour finding, pose-estimation, principal components analysis, and wavelet decomposition. Image synthesis modules include: Viterbi graph search, 3D object manipulation, texture mapping with alpha blending, pixel-flow, transitions morphing and blending. Other modules include support for file formats such as: bitmaps (TIFF, JPEG), audio (WAV), video (AVI, MPEG2), 3D objects (Open Inventor [38]), web pages (HTML), phoneme files.

In order to create, debug and access these components in a flexible way to quickly build new applications, we designed a development environment that integrates all the modules and provides a simple scripting language coupled with a powerful graphical user interface (GUI). The scripting language features loops, conditional branching, subroutines, arithmetic operations (integer, floats), vector operations and string operations. Figure 3.23

shows an example of such a script. The scripts are interpreted at execution time, making it possible to quickly experiment with different parameters and algorithms. The GUI offers object-based display of recognition results by allowing the overlay of shapes marking recognized features in an image. Because these shapes are stored and displayed independently of an image, they can be easily selected, manipulated and edited by simple mouse clicks. The GUI also supports the interactive display, manipulation and editing of 3D objects. It also allows the display of graphs, charts, scatter plots and supports the overlay of phonetic information. Figure 3.24 shows a screenshot of the development environment.

The GUI part of the development environment was implemented using the Microsoft foundation classes (MFC). Within MFC, the multiple document interface (MDI) provides support for display and event handling of multiple types of documents (in our case, text editing for scripts, image/objects display).

Together the scripting language and the GUI offer an environment similar to MATLAB. However, we decided early on to use the scripting language only as a way to combine algorithms and modules into applications while writing the core algorithms in C++, thus avoiding inefficiencies and enabling real-time performance.

3.5.2 Real-Time Synthesis

In many applications, the synthesis of the talking head has to happen in real-time. A typical example is a dialog situation where the user enters a query and the computer agent needs to respond immediately. In this scenario, two different constraints are imparted onto the talking-head synthesis module. Firstly, the head needs to start talking within a reasonable delay from the time of the request. Unless the agent explicitly indicates to the user that it needs time to find an answer, this latency should not exceed half a second. Longer delays make the agent look stupid. Secondly, the rate at which the head is animated during speech should be close to 30 frames per second. Lower rates result in loss of lip-synch.

The constraint on the latency turns out to be critical in our approach because substantial computation is required to find the optimal concatenation of image samples of a speech animation. This computation has to happen before the animation starts playing. Hence we need to optimize this part of the code. We first profiled the code and identified the most used code segments. One is a weighted dot product corresponding to the calculation of the target cost (see section 3.3.4) and the other one is the block distance used in the calculation of the concatenation cost (see section 3.3.7). In both cases we are able to use short integers to represent the value and thus we can take advantage of parallel computation offered by the MMX set of instruction on Intel platforms. These instructions allow the execution of 4 arithmetic operations on short integers in one clock cycle (addition, subtraction, multiply-add, etc.). As expected, we obtain approximately a factor of 4 speedup over the compiled C++ code. In the next step, we reduced the amount of computation by only evaluating subsets of the database when searching for candidates. This is implemented using table look-ups (see section 3.3.5 for more details). We also keep the size of the graph under control by pruning low-scoring candidates and allowing only a maximum number of candidates per state (see also section 3.3.5 for more details).

The constraint on the frame rate poses an equally difficult problem. To synthesize one frame, our approach requires several bitmaps to be loaded and then texture-mapped using the 3D model of the talking-head. The time budget for synthesizing one frame is about 30 milliseconds making it necessary to optimize both the bitmap access from disk and the display. To provide fast access to the bitmaps on disk we archive the bitmaps into a single file and instruct the operating system to memory-map the file. This technique typically provides a speed-up of a factor of three over conventional file access (a mouth bitmap is loaded in about 10 milliseconds instead of 30 ms). To enable fast display of the talking head, we extensively use OpenGL [24]. OpenGL provides a complete set of primitives for the display of 3D graphics. OpenGL was designed specifically so that its primitives can be hardware accelerated on dedicated video cards. Thanks to the rapid rise in popularity of video games, video cards performance has increased faster than predicted by Moore’s law (double the performance every 18 months). Today, affordable video cards are capable of displaying over 5 million polygons per second. Our head model is typically displayed in 5 milliseconds, which includes projection, texture-mapping, alpha-blending and hidden face removal.

Using these optimizations, we implemented a player that is capable of producing real-time speech animations on conventional modern PC's (900Mhz Pentium IV + nVidia GeForce256 AGP video card). The latency on this system is about 1/20th of the animation duration.

3.6 Applications

We have envisioned talking head synthesis as a core technology that can be inserted into many applications in order to augment their user interface. As such we have *not* focused our resources in developing full-fledged applications around the talking head. Instead, we have mostly focused our efforts to perfect the quality and speed of the synthesizer. The talking-head technology is still in its infancy however, and there is a need to bootstrap its acceptance with compelling applications. We therefore have developed two demo applications destined to strike the imagination of viewers and trigger the integration of the talking head into real products.

3.6.1 Automated Newscaster

Our talking-head player is capable of real-time synthesis; however, it requires installation and occupies disk space and hence is not yet suitable for large distribution over the Internet. Because of bandwidth issues, most people are reluctant to install new software and spend time waiting for it to download. An alternate way to distribute talking-head animations over the Internet is to stream complete audio-video animations directly to client players such as Microsoft Windows Media Player, Apple Quicktime or RealNetworks Real Player. Most PCs are equipped with such players making it a preferred way to distribute talking-head content. At this point we haven't developed the capabilities to synthesize and stream talking-head animations in real time to these types of media players. Instead, we implemented the automated newscaster application which does not require real-time synthesis and hence allows us to produce off-line encoded audio-video files that are later streamed to client players via a media server (Real Server, Windows Media Server, QuickTime Server).

The automated newscaster application periodically checks the Internet for news updates. After downloading headlines and summaries for sections such as business, international, national, technology and sports, it dynamically builds a web site that provides the user with a multi-modal, interactive browsing of the latest news. A screenshot of the business news page from this site is shown in Figure 3.25 . On the left side of the screen our talking-head is speaking a summary of the headlines, while on the right side of the screen, the user can click on any headline and have the talking-head directly skip to it or alternatively access the full text for reading. The talking-head animation is generated entirely automatically from the textual content downloaded from the Internet. Figure 3.26 shows the flow of operations involved in synthesizing a news animation from the downloaded textual content. Html source is first obtained from the Internet and is then parsed to retrieve the relevant textual data. Many news web sites on the Internet, such as the New-York-Times [25] provide XML tags identifying logical parts within the HTML code. For example, headline summaries are identified by the <NYT_SUMMARY> tag. This makes the parsing robust against changes in the layout and style of the pages. The text needed to create the talking-head animation is prepared for TTS synthesis by adding pauses between headlines and at the beginning and end of the animation. For this purpose, a special marker is inserted in the text; for example: "\!si=300" will insert a pause of 300 milliseconds. Later, emotions, as well as music jingles will be inserted during these silences. The marked-up text is then sent to the TTS module and the resulting phoneme and audio blocks are used to calculate the visual animation. The animation is then rendered to memory and the image buffer is sent together with the audio to the video encoder, resulting in an encoded video file. The video file is placed on the media server and its reference is inserted in the produced web page. We currently use the RealNetworks Real Player and Server to encode and stream the talking-head animations.

To allow the user to interactively choose any headline, we produce as many separate video files as there are headlines. If the user doesn't select any headline in particular, the page plays all headlines. To allow for such flexibility, we use the SMIL language (Synchronized Multimedia Interface Language) [34] to play individual or chained-up video files. Figure 3.27 shows an example of a SMIL file that was automatically generated for this application.

3.6.2 Help-Desk

The Help-Desk application is a demonstration of our real-time player in a dialog situation between a user and a virtual customer service agent. It follows a client-server type of architecture and hence requires the installation of the talking-head player. On the other hand, it allows very low bitrate communication between the server and the player because only animation parameters and audio (when no TTS server is available at he client) need to be sent. The client player is responsible for playing speech animations of the virtual customer service agent sent by the server and for capturing the user's input and sending it to the server. The server receives the user's input, interprets it, generates a response, computes the associated speech animation and sends the animation's parameters and audio to the client player. Figure 3.28 shows the client-server architecture of the Help-Desk application. We use a natural language understanding module (NLU) to interpret the user's input [21], while the dialog itself is directed by a dialog management module [11]. These modules are part of a commercial system known as "How May I Help You?" [12], currently used by AT&T to run customer relation management systems (CRM) over phone lines.

To appear realistic in a dialog situation, the virtual agent needs to exhibit idle and listening behavior while not speaking. Idle behavior includes mechanical facial motions described in section 3.4.5 , while listening behavior refers to a more complex set of facial motions and expressions used by protagonists during dialogs to indicate understanding or disagreement, facilitate turn-taking and interruption, suggest reflection or expectation. In its current implementation however, the talking-head player only supports idle behavior in the form of mechanical facial motions such as eye-blinks, eye motions and small head movements.

3.7 References

[1] Beskow J, Granström B & Spens K-E., "**Articulation strength - Readability experiments with a synthetic talking face"**, Proc of Fonetik 2002, TMH-QPSR, 44: 97-100

[2] Beutnagel, M., Conkie, A., Schroeter, J., Stylianou, Y., Syrdal, A., "**The AT&T Next-Gen TTS System",** J. Acoust. Soc. Am., Vol. 105, Pt. 2, p. 1030, 1999.

[3] Brand, M., "**Voice Puppetry",** ACM SIGGRAPH, 1999.

[4] Bregler, C., Covell, M., Slaney, M., "**Video Rewrite: Driving Visual Speech with Audio"**, ACM SIGGRAPH, 1997.

[5] Cohen, M.M., Massaro, D.W., "**Modeling Coarticulation in Synthetic Visual Speech"**, Models and Techniques in Computer Animation, Springer Verlag, 1993.

[6] Cosatto, E., Graf, H.P., "**Sample-Based Synthesis of Photo-Realistic Talking-Heads"**, IEEE Computer Animation, 1998.

[7] Costa, M., Chen, T., Lavagetto, F., "**Visual Prosody Analysis for Realistic Motion Synthesis of 3D Head Models"**, Proc. of ICAV3D'01 - International Conference on Augmented, Virtual Environments and 3D Imaging, Mykonos, Greece, May 30-June 01, 2001, pp. 343-346.

[8] Ezzat, T., Poggio, T., "MikeTalk: A Talking Facial Display Based On Morphing Visemes", IEEE Computer Animation, 1998.

[9] Ezzat, T., Geiger, G., Poggio, T., "**Trainable Videorealistic Speech Animation"**, ACM Transactions on Graphics, Proc. of SIGGRAPH 2002, vol. 21, n. 3, pp. 388-397, July 2002.

[10] Faigin, G., "The Artist's Complete Guide to Facial Expression", Watson-Guptill, New-York, 1990.

[11] Gorin, A.L., Wright, J.H., Riccardi, G., Abella, A., and Alonso T., "**Semantic information processing of spoken language**", Proc. of ATR Workshop on MultiLingual Speech Communication, 2000.

[12] Gorin, A.L., Riccardi, G., and Wright, J.H., "**How may i help you?**" Speech Communication, 23:113--127, 1997.

[13] Graf, H.P., Cosatto, E., Potamianos, G., "**Robust Recognition of Faces and Facial Features with a Multi-Modal System"**, IEEE Systems, Man and Cybernetics, pp. 2034-2039, 1997.

[14] Graf, H.P., Cosatto, E., Strom, V., Huang, F.J., "**Visual Prosody: Facial Movements Accomanying Speech**", Proc. Int. Conf. on Automatic Face and GestureRecognition, 2001.

[15] Guenter, B., Grimm, C., Wood, D., Malvar, H., Pighin, F., "**Making Faces",** ACM SIGGRAPH, pp. 55-66, 1998.

[16] Granström, B.A., House, D., Swerts, M.G., "**Multimodal Feedback Cues in Human-Machine Interactions"**, Proceedings of Speech Prosody 2002, Aix-en-Provence, April 11-13, 2002.

[17] Huang, X., Alleva, F., Hon, H., Hwang, M., Lee, K., Rosenfeld, R., "**The SPHINX-II Speech Recognition System: An Overview**", Computer Speech and Language, v. 2, p. 137-148, 1993.

[18] Hunt, A., Black, A., "Unit selection in a concatenative speech synthesis system using a large speech database", ICASSP, vol. 1, pp. 373-376, 1996.

[19] Kshirsagar, S., Magnenat-Thalmann, N., "**A Multilayer Personality Model**", in 2nd International Symposium on Smart Graphics, June 2002.

[20] Lee, S.P., Badler, J., Badler, N., "**Eyes Alive**", acm Transactions on Graphics, proceeding of SIGGRAPH 2002, vol. 21, n. 3, pp. 637-644, July 2002.

[21] Levin, E., Pieraccini, R., Eckert, W., Fabbrizio, G. D., & Narayanan, S., "**Spoken language dialogue: From theory to practice**", Proc. IEEE Workshop on Automatic Speech Recognition and Understanding, ASRUU99, 1999.

[22] MAGPIE speech lip-synch animation software: http://www.thirdwishsoftware.com/magpiepro.html

[23] McGurk, H., MacDonald, J. "**Hearing lips and seeing voices"**, Nature, 264, 1976, pp 746-748.

[24] Neider, J., Davis, T., and Woo, M.. "**OpenGL programming guide**", Addison-Wesley, 1993.

[25] The New-York Times: http://www.nytimes.com

[26] Oberkampf, D., Dementhon, D., Davis, L., "**Iterative Pose Estimation Using Coplanar Feature Points**", Internal Report, CVL, CAR-TR-677, University of Maryland, 1993.

[27] Ostermann, J. "**Animation of synthetic faces in MPEG-4",** IEEE Computer Animation, pp. 49-55, 1998.

[28] Pandzic, I., Ostermann, J., Millen, D., "**User evaluation: Synthetic talking faces for interactive services**", The Visual Computer, Volume 15, Issue 7/8, pp 330-340, Nov 04, 1999.

[29] Parke, F.I., "**A parametric model for human faces**", PhD Dissertation, University of Utah, Department of Computer Sciences, 1974.

[30] PlayMail by AT&T Labs - Research: http://playmail.research.att.com

[31] Potamianos, G., Graf, H.P., "**Linear Discriminant Analysis for Speechreading"**, IEEE Workshop on Multimedia Signal Processing, pp. 221-226, 1998.

[32] SAPI. Microsoft Speech API: http://www.microsoft.com/IIT/OnlineDocs/intro2SAPI.html

[33] Silverman, K., Beckman, M., Pitrelli, J., Ostendorf, M., Wightman, C., Price, P., Pierrehumbert, J. and Hirschberg, J., "**TOBI: A standard for labeling English prosody"**, Proceedings ICSLP-92, Banff, Vol. 2: 981-984.

[34] SMIL. "**Synchronized Multimedia Integration Language (SMIL 2.0) Specification**", W3C Proposed Recommendation (05 June 2001), http://www.w3.org/TR/smil20/.

[35] Strom, V., Heine, H., “**Utilizing prosody for unconstrained morpheme recognition**”, Proc. European Conf. on Speech Communication and Technology, Budapest 1999.

[36] Velásquez, J., “**Modeling Emotions and Other Motivations in Synthetic Agents”**, In: Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97). Providence, RI: MIT/AAAI Press, 1997.

[37] Viterbi, A.J., “Error bounds for convolutional codes and an asymptotically optimal decoding algorithm”, IEEE trans. on Information Theory, 1967.

[38] Wernecke, J., “The Inventor Mentor: Programming Object-Oriented 3D Graphics with Open Inventor”, 2nd Ed., Reading, Massachusetts, Addison Wesley, 1994.

[39] Wightman, C., Talkin, D., "**The Aligner**", Entropic Research Laboratory 1994.

3.8 Figures

Figure 3.1 Architecture of the Synthesis module.
Figure 3.2 Synthesis Graph.
Figure 3.3 Target Cost Calculation.
Figure 3.4 Effect of Pruning.
Figure 3.5 Discrimination power of visual features.
Figure 3.6 Visual distance.
Figure 3.7 Best path through the synthesis graph.
Figure 3.8 Animation script.
Figure 3.9 Transitions: blending versus morphing.
Figure 3.10 Objective evaluation: average segment length versus average total target cost.
Figure 3.11 Objective evaluation: average total visual distance versus average total target cost.
Figure 3.12 Objective evaluation: average target cost, average segment length versus database size.
Figure 3.13 Graph pruning (parameter 1)
Figure 3.14 Graph pruning (parameter 2)
Figure 3.15 Objective evaluation: mouth-height plots.
Figure 3.16 Prosodic events (ToBI set).
Figure 3.17 Prosodic events (graph of x-angle head rot).
Figure 3.18 Synthesized synchronized prosody motion.
Figure 3.19 Simplified head articulation model.
Figure 3.20 Probabilistic state machine for random facial movements.
Figure 3.21 Eye globes and gaze.
Figure 3.22 Expression segments insertion.
Figure 3.23 Example of script
Figure 3.24 Screen-shot of the development environment
Figure 3.25 Screen-shot of the automatic newscaster application.
Figure 3.26 Architecture of the automatic newscaster application.
Figure 3.27 Example of a SMIL file.
Figure 3.28 Architecture of a help desk application.

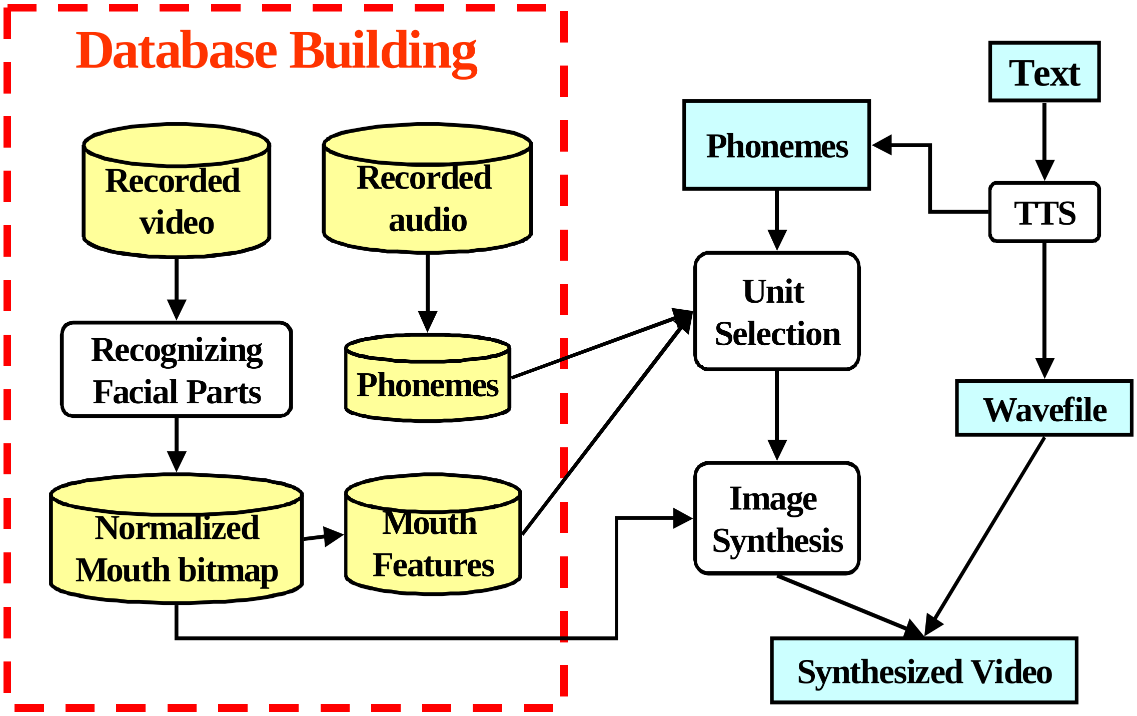


Figure 3.1. Architecture of the synthesis module. On the left side the off-line process of building the database of the head model from a/v recordings is outlined. From the video channel, the recognition extracts the facial parts and creates a database of normalized samples. Features are then extracted from the normalized samples. Using the audio channel, the data is phonetically labeled. This process is done only once, off-line. On the right side, the on-line synthesis process is outlined. Starting from text, the TTS module produces audio and phonemes (alternatively, we can also start from labeled recorded audio). The phoneme information is then used by the unit selection process to find the optimal concatenation of samples from the database. A talking head animation is then rendered using the calculated animations of the mouth as well as other facial parts. Finally, combining the audio with the synthesized talking-head animation produces a video.

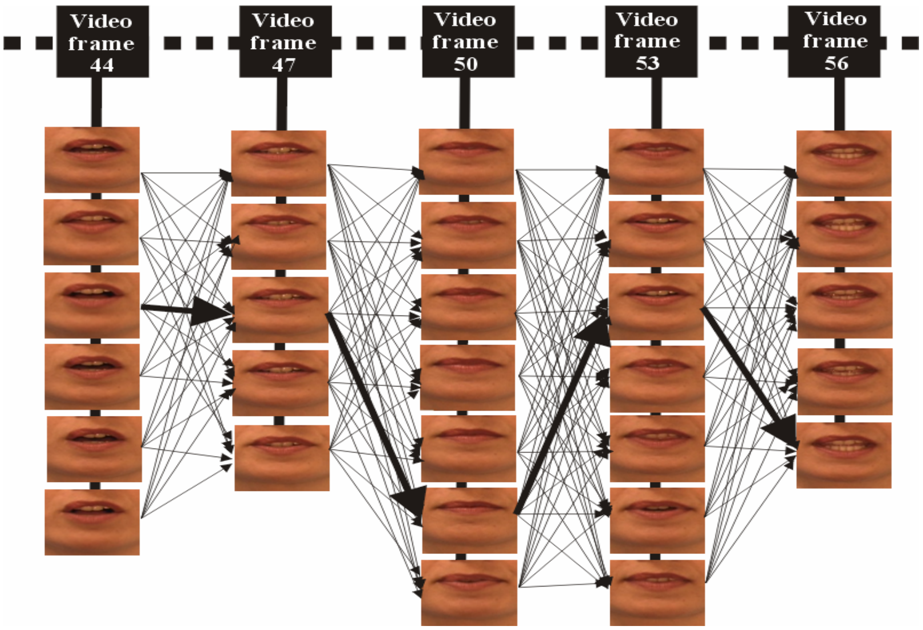


Figure 3.2. A slice of a synthesis graph. For each frame of the final animation, a node is created in the synthesis graph. Each node is populated with a variable number of candidates during the unit selection process. Candidates of a node are fully connected to the candidates of the following node. The graph represents all possible animations that can be created by concatenating one candidate at each node. By assigning costs to arcs and using a Viterbi search algorithm, an optimal path (as defined by the cost function(s)) can be determined to produce a desired animation.

```
target:     aaaaaaaaabbbb b boooooooooooo
candidate:  aaaaammmmmmmm m uuuuuuuuuuuuu
distance:   0000099990000 0 9555555555555
weight:     1111222334568 9 9865433222111
```

$$\text{node cost} = \frac{\text{match} \bullet \text{weight}}{|\text{weight}|} = 3.8$$

Figure 3.3. Example of target cost calculation. The target (green) is the phonetic annotation of the speech to be synthesized. Each column represents one frame of the animation. A phoneme has a particular duration and typically spans multiple frames. The center column is the current animation frame for which we are trying to find the best matching candidate. The candidate (red) represents the phonetic context of one recorded sample from the database. The distance (blue) measures the ***visual*** similarity between two phonemes (see section 2.4.4 for details on how this distance is obtained). Finally the weight vector (blue) is an exponential decay around the center frame. It simulates coarticulation by decreasing the influence as a function of the distance from the currently articulated phoneme. The target cost is then simply the weighted dot product of the target vector and the candidate vector.

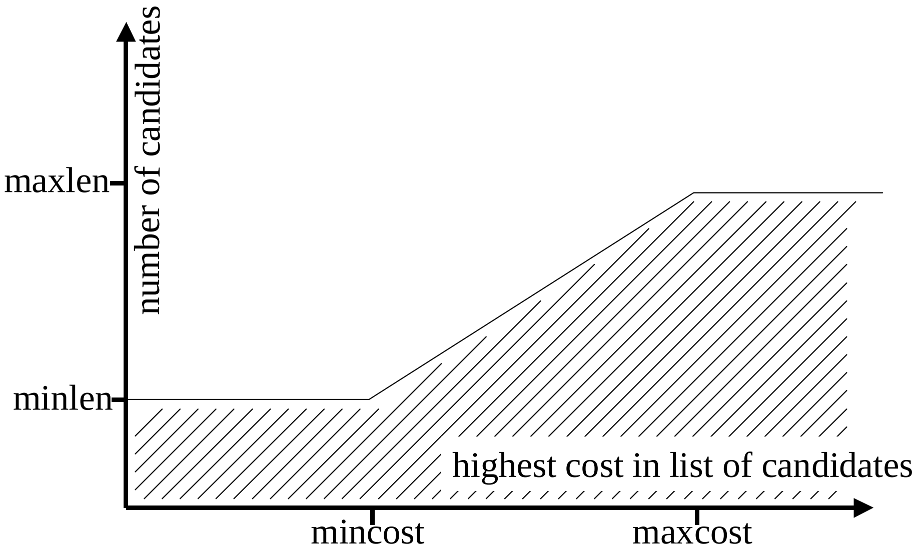
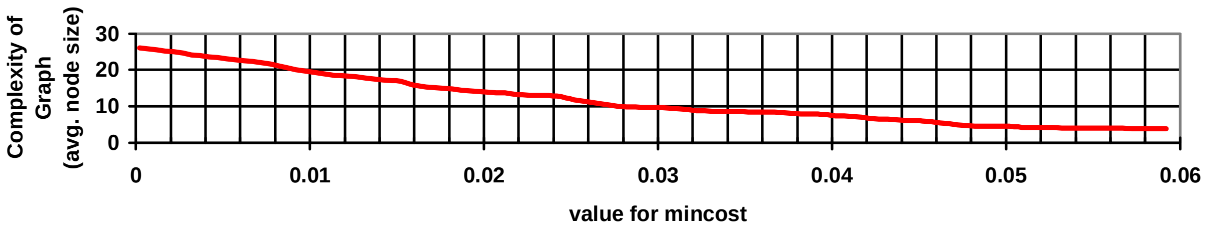
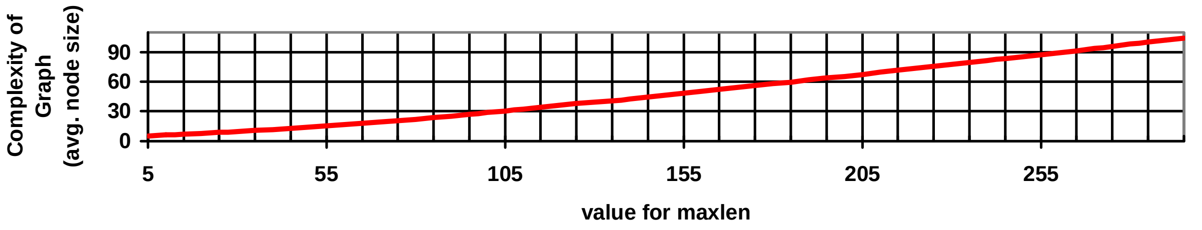


Figure 3.4. Illustration of graph pruning. The vertical axis of the graph shows the number of candidates at a node of the graph and the horizontal axis shows the highest cost in the sorted list of candidates at a node. The shaded area is the zone where all graph nodes must fall after the pruning process is completed. This zone is defined by four numbers: the minimum necessary entries in the sorted list, the maximum allowable entries in the sorted list and two cost values (mincost, maxcost) defining the slope of the pruning. If a given sorted list falls outside the shaded area, candidates are removed until it falls within. Qualitatively this process has the effect of allowing more candidates when their cost is higher, giving the algorithm more choices when the fit isn't good. For a given target, the complexity of the graph is shown below as a function of the mincost and maxlen values. Figure 3.13 and 3.14 shows the effect of pruning on the quality of synthesized animations.





```
*** Feature # 0
00 |
01 |
02 |
03 |
04 |                                    _  b          m  p
05 |                                    _  b          m  p
06 |                                    _  b          m  p
07 |                                    _  b   f      m  p     v
08 |                                    _  b   f      m  p     vw
09 |                          U         _      f               vw
10 |                          U         _      f           rstuvw  z
11 |         D                U         _    d             rstuvw  z
12 |         D          O    TU         _    d    i  l     rstu w yz
13 |         D    I     O    TU         _    d    i  l n   rstu w yz
14 |   >    CD    I     O    T     Z    _    d  g ijkl n   rstu   y
15 | < >    CD    I     O   ST     Z    _    de g ijkl n   r tu   y
16 | < >    CD    I     O   ST     Z         de ghijkl n   r      y
17 | < >    C     I     O   S      Z         de ghijkl n   r      y
18 | < >    C     I     O   S      Z         de ghijkl n          y
19 | < >                O   S      Z         de gh  k  no
20 | < >                O                 a   e gh  k   o
21 | <                  O                 a   e gh      o
22 | <                                    a   e  h      o
23 | <                                    a      h      o
24 | <                                    a             o
25 | <                                    a             o
26 |                                      a             o
27 |                                      a             o
28 |                                      a             o
29 |                                                    o
30 |
```

Figure 3.5. Discrimination of visual features. This graph shows the spread (30 bins) of all phonemes as a function of visual feature #0 (the measured height of the mouth). We can see that this feature clearly separates closures (‘m’, ‘b’, ‘p’, ‘f’, ‘v’) from openings (‘a’=’ae’, ‘o’=’ao’, ‘e’=’eh’, ‘I’=’iy’, ‘O’=’aa’, ‘<’= ‘ah’, ‘>’=’ow’). Other phonemes (those with mouth semi-open) are not separated. We used 8215 samples corresponding to the stationary point of every phoneme from the database of recordings.

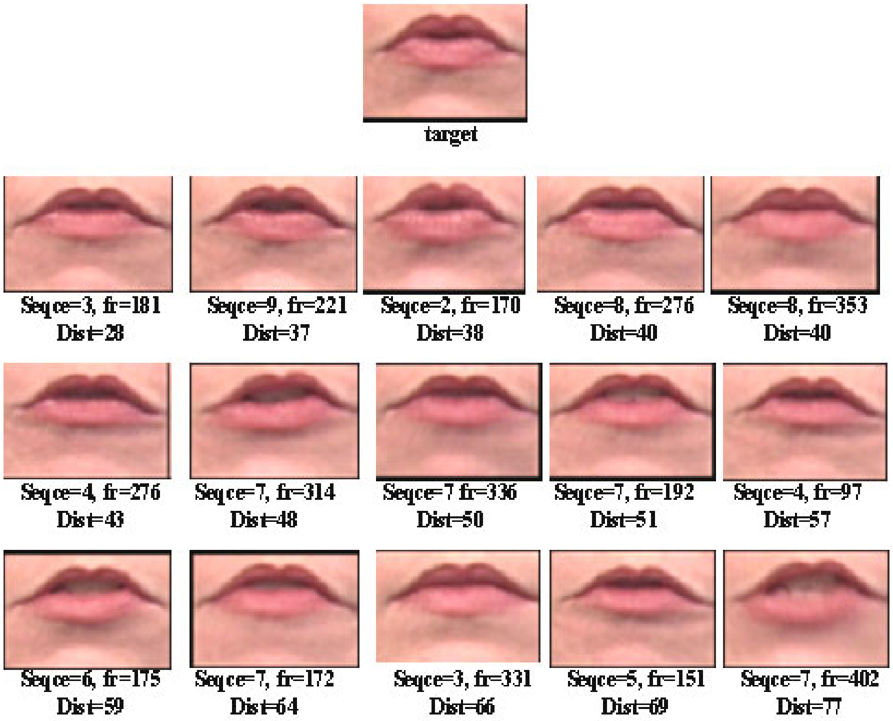


Figure 3.6. Visual distance in feature space. Fifteen different mouths are shown with their distance to the target mouth image. The distance is the weighted block distance in feature space. The features include measured width and height of the mouth and PCA coefficients.

```
frame   target       selected       acc. cost  tgt cost   conc. cost   selected sample
--------------------------------------------------------------------------------------
0043  SI ah p      b  ah m        0.9113 [0.0872; 0.0000] S=s101_214; 082
0044  SI ah p      b  ah m        0.9724 [0.0611; 0.0000] S=s101_214; 083
0045  SI ah p      b  ah m        1.0176 [0.0452; 0.0000] S=s101_214; 084
0046  SI ah p      b  ah m        1.0589 [0.0413; 0.0000] S=s101_214; 085
0047  ah p  t      ah m  ae       1.1044 [0.0455; 0.0000] S=s101_214; 086
0048  ah p  t      ah m  ae       1.1595 [0.0551; 0.0000] S=s101_214; 087
0049  ah p  t      ah m  ae       1.2257 [0.0662; 0.0000] S=s101_214; 088
0050  ah p  t      ah m  ae       1.2995 [0.0738; 0.0000] S=s101_214; 089
0051  ah p  t      ah m  ae       1.3822 [0.0826; 0.0000] S=s101_214; 090
0052  ah p  t      r  p  ih       4.0209 [0.0528; 2.5859] S=s101_214; 265
0053  ah p  t      r  p  ih       4.0733 [0.0523; 0.0000] S=s101_012; 266
0054  ah p  t      r  p  ih       4.1493 [0.0760; 0.0000] S=s101_012; 267
0055  p  t  n      p  ih n        4.2521 [0.1028; 0.0000] S=s101_012; 268
0056  p  t  n      p  ih n        4.3661 [0.1140; 0.0000] S=s101_012; 269
0057  p  t  n      p  ih n        4.4622 [0.0961; 0.0000] S=s101_012; 270
0058  t  n  s      ih n  y        4.5253 [0.0631; 0.0000] S=s101_012; 271
0059  t  n  s      ih n  y        4.5581 [0.0328; 0.0000] S=s101_012; 272
```

Figure 3.7. A slice of the best path after performing a Viterbi search for the text ‘**`Upton`**’. The first column is the frame number of the animation. The second column shows the phonetic context (3 phonemes: one before, one after) of the target text. The third column shows the phonetic context of the best selected candidate. The fourth column shows the accumulated target cost, the fifth column the target cost for the frame and the sixth column the concatenation cost for the frame. Note that the concatenation cost is zero when successive candidates form a recorded sequence (see last column with the recording sequence and frame number). In this example the word ‘**`Upton`**’ did not exist in the database. The unit selection has found two segments from the recorded sequences ‘`mister b`**`aum`**` asked`’ and ‘`your o`**`pin`**`ion`’, that when concatenated, approximate in the best possible way (based on what’s available from the recorded database) the target word ‘**`Upton`**’.

```
F= 0055 ( s101_127 ; 072 ; 1.00 )  ( s101_127 ; 072 ; 1.00 )
F= 0056 ( s101_127 ; 073 ; 1.00 )  ( s101_127 ; 073 ; 1.00 )
F= 0057 ( s101_127 ; 074 ; 1.00 )  ( s101_127 ; 074 ; 1.00 )
F= 0058 ( s101_127 ; 075 ; 0.67 )  ( s101_110 ; 242 ; 0.33 )
F= 0059 ( s101_110 ; 242 ; 0.67 )  ( s101_127 ; 075 ; 0.33 )
F= 0060 ( s101_110 ; 243 ; 1.00 )  ( s101_110 ; 243 ; 1.00 )
F= 0061 ( s101_110 ; 244 ; 1.00 )  ( s101_110 ; 244 ; 1.00 )
F= 0062 ( s101_110 ; 245 ; 1.00 )  ( s101_110 ; 245 ; 1.00 )
F= 0063 ( s101_110 ; 246 ; 1.00 )  ( s101_110 ; 246 ; 1.00 )
F= 0064 ( s101_110 ; 247 ; 1.00 )  ( s101_110 ; 247 ; 1.00 )
F= 0065 ( s101_110 ; 248 ; 0.86 )  ( s101_267 ; 253 ; 0.14 )
F= 0066 ( s101_110 ; 249 ; 0.71 )  ( s101_267 ; 253 ; 0.29 )
F= 0067 ( s101_110 ; 250 ; 0.57 )  ( s101_267 ; 253 ; 0.43 )
```

Figure 3.8. A slice of an animation script. To allow for transitions, two animation tracks are used. In each track, we find the recorded sequence number and frame number and the blending factor at the transition point. A final composite sample is synthesized by blending the samples referenced by each animation track using the given blending factor. The blending factor is ramped up and down according to the length of the segments involved.

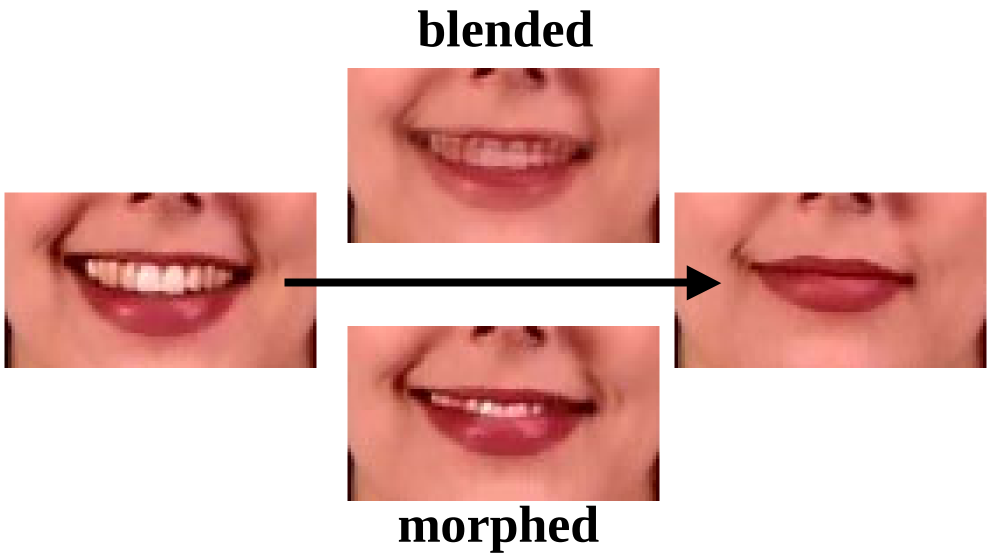


Figure 3.9. Difference between pixel morphing and blending. Here, the large visual difference between the two target images results in visible artifacts when simple pixel blending is used. A more complex and expensive algorithm uses dense-field pixel-flow to find forward and backward mappings from the two target images into the morphed image. Hole-filling methods are used for pixels that have no correspondences in neither target images.

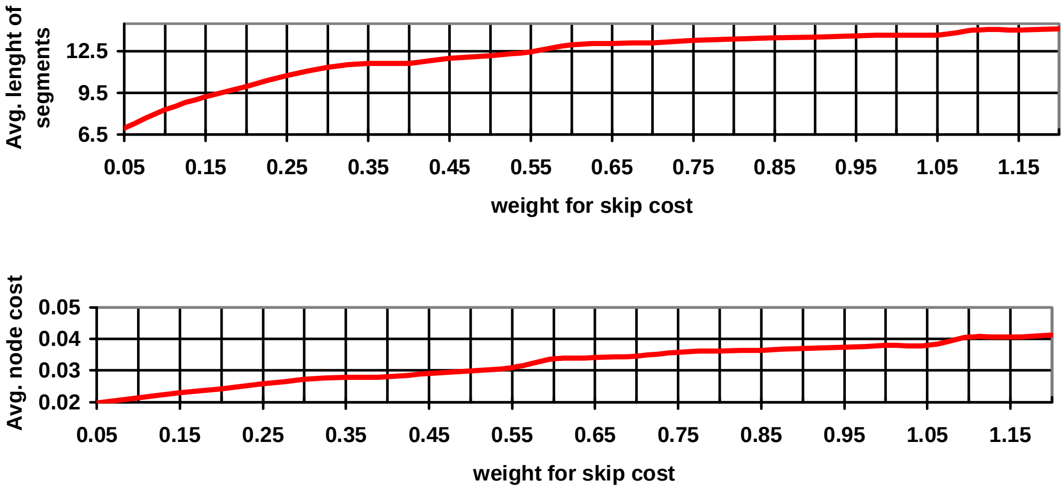


Figure 3.10. Influence of the skip cost. The weight given to the skip cost is shown on the horizontal axis of the two graphs. The top graph shows that the average length of the segments making up the animation is growing as the weight given to the skip cost increases. Penalizing skips results in longer segments. On the other hand the same increase results in higher node costs, meaning that the synchronization gets worse. The average segment length behaves asymptotically while the average node cost grows linearly as the weight given to the skip cost increases. This suggests an optimal value for the skip cost weight around 0.6. These values were obtained from the synthesis of 20 sentences (s030_030 to s030_049), totaling 2823 phonetic segments.

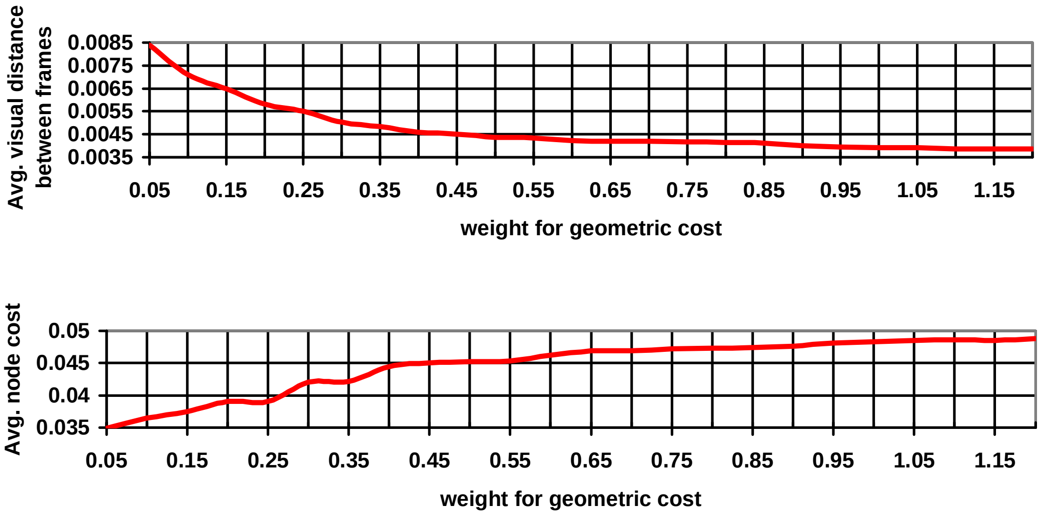


Figure 3.11. Influence of geometric cost. The weight given to the geometric cost is shown on the horizontal axis of the two graphs. The top graph shows that the average visual distance between frames making up the animation is decreasing as the weight given to the geometric cost increases. On the other hand the same increase results in higher node costs, meaning that the synchronization gets worse. These graphs suggest an optimal value for the geometric cost weight around 0.25. These values were obtained from the synthesis of 20 sentences, totaling 2823 phonetic segments.

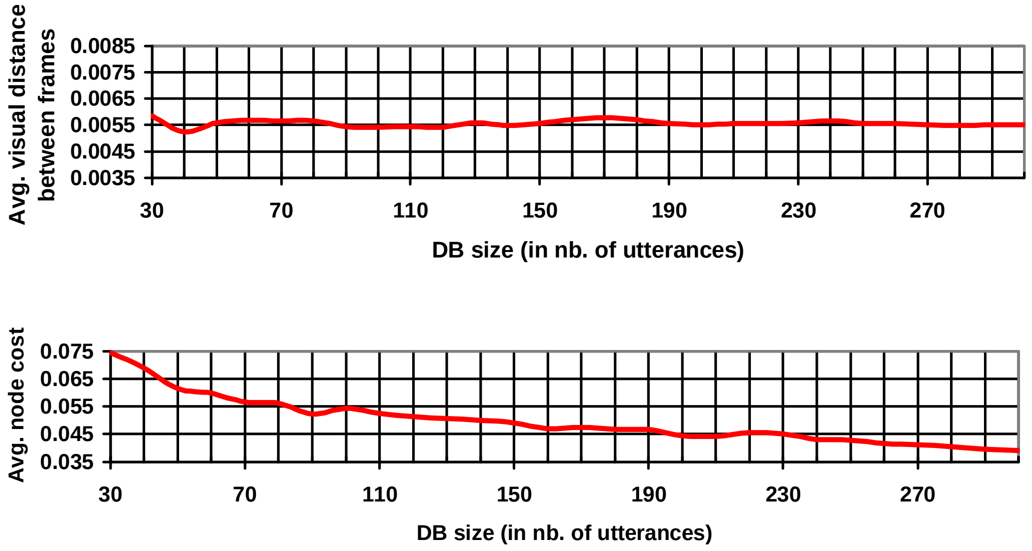


Figure 3.12. Influence of the database size. The size of the database is shown on the horizontal axis in number of utterances. Each utterance in the DB is on average 300 video frames. As expected, the size of the database has no direct influence on the visual smoothness of synthesized utterances (this is mostly influenced by the weight given to the skip and geometric costs). However, as the size of the database grows, the average node cost clearly decreases, indicating better lip synchronization.

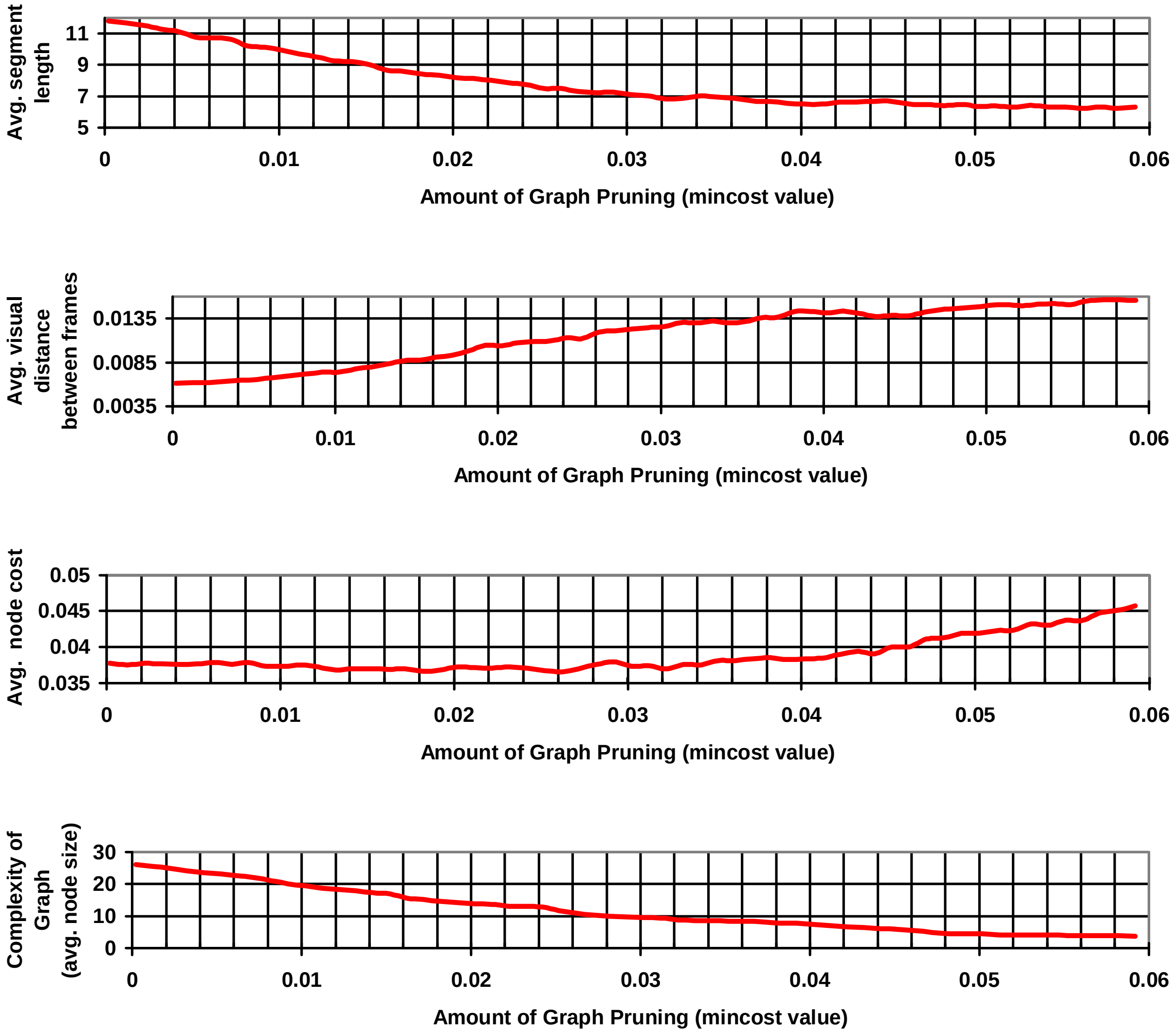


Figure 3.13. Graph pruning. The pruning of the graph is determined by four variables: ***mincost***, ***maxcost***, ***minlen***, ***maxlen*** (see Figure 3.3 for an explanation of these variables). The result of the pruning is evaluated by 4 values: the average segment length, the average visual distance, the average node cost and the graph size. The amount of graph pruning is shown on the horizontal axis by the ***mincost*** parameter. The other fixed parameters are: ***minlen***=5, ***maxlen***=100, ***maxcost***=0.2.

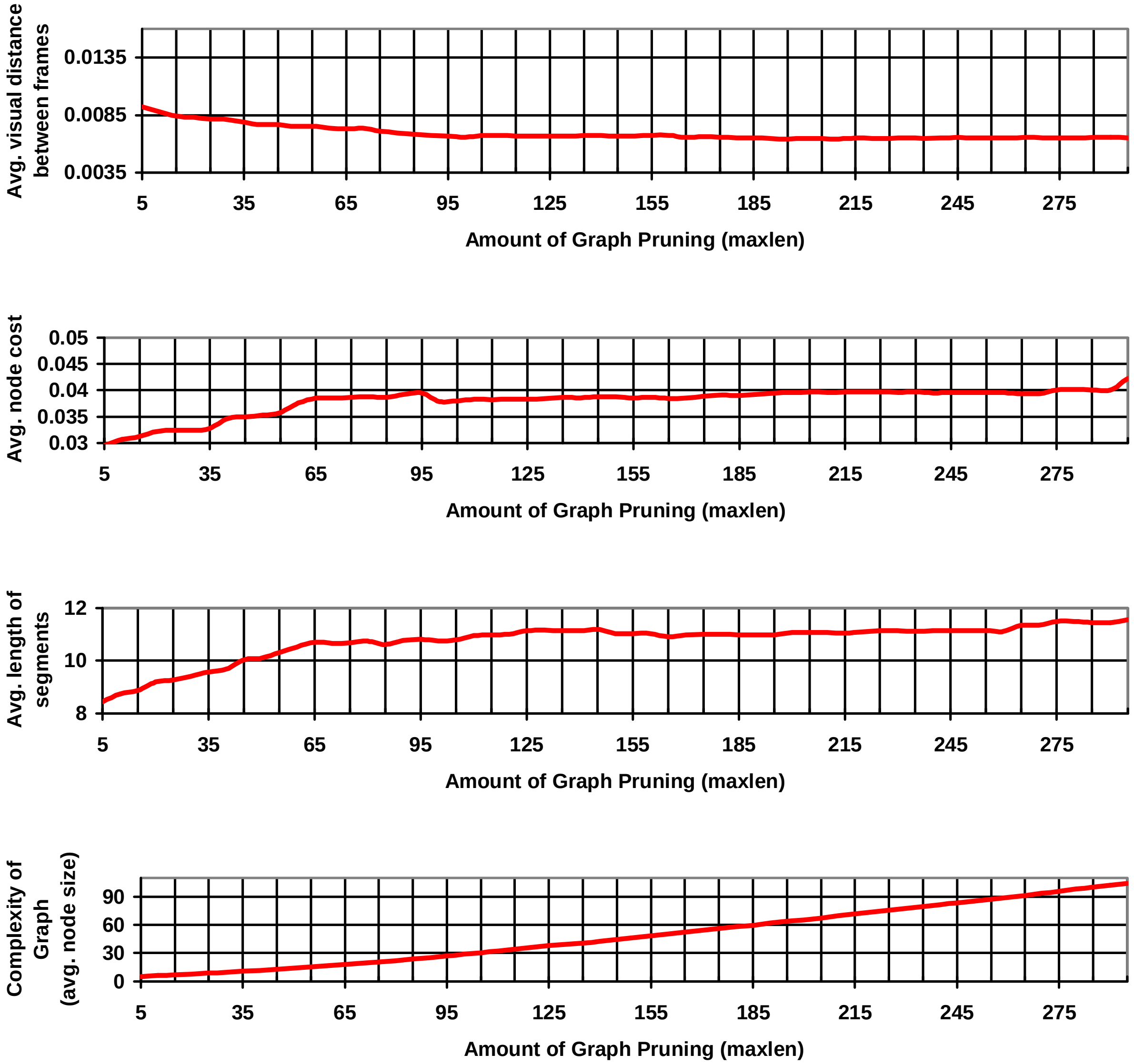


Figure 3.14. Graph pruning. The pruning of the graph is determined by four variables: ***mincost***, ***maxcost***, ***minlen***, ***maxlen*** (see Figure 3.3 for an explanation of these variables). The result of the pruning is evaluated by 4 values: the average segment length, the average visual distance, the average node cost and the graph size. The amount of graph pruning is shown on the horizontal axis by the ***maxlen*** parameter. The other fixed parameters are: ***minlen***=5, ***mincost***=0.006, ***maxcost***=0.2. The graphs suggest an optimal value for ***maxlen*** around 65.

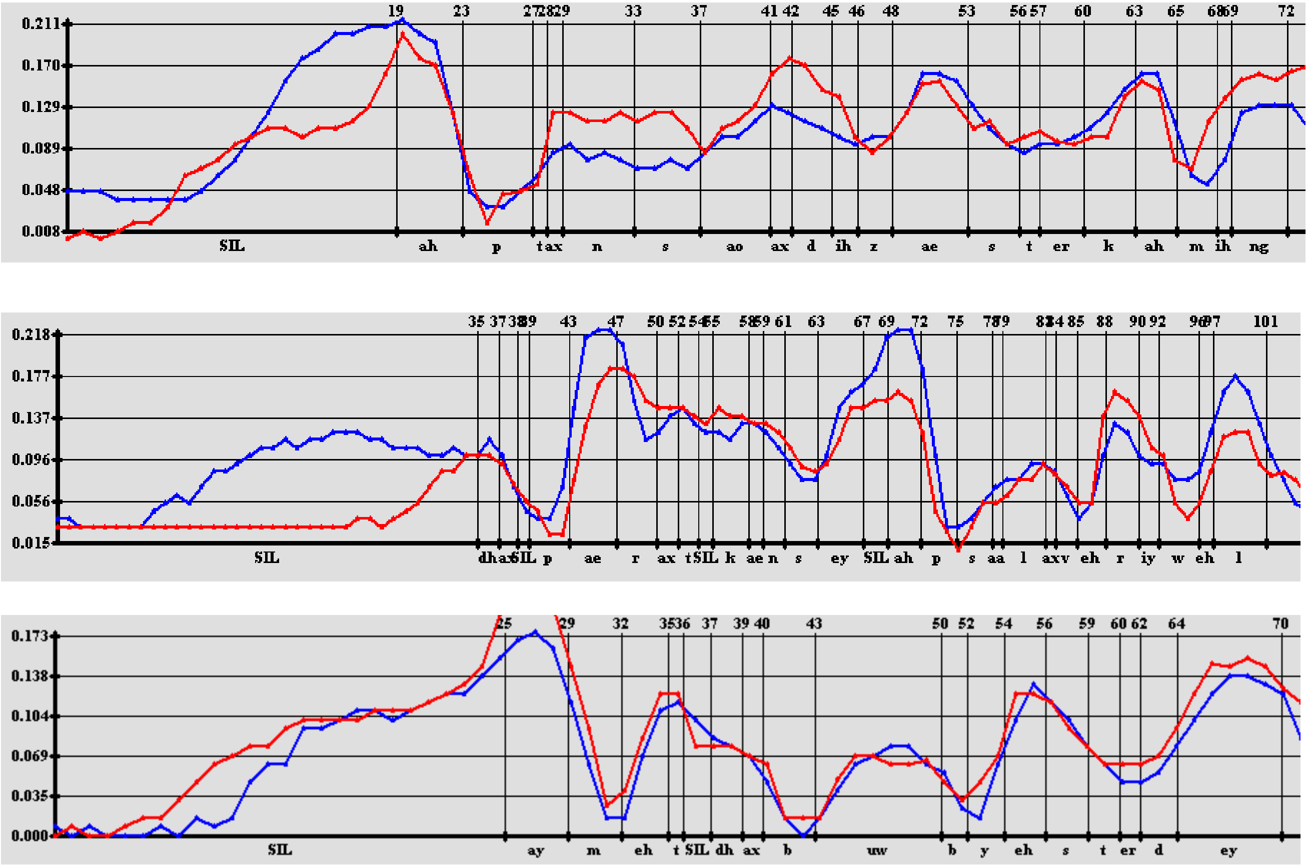


Figure 3.15. Mouth height plots. These plots show the measure of the mouth opening as a function of time for three sentences from our recorded corpus. The blue curves are measured on the recorded sequences while the red curves are measured from animations that were synthesized from the phonetic annotation of the recorded sequences. On the horizontal axis, the numbers on top indicate the video frame number, while the bottom part shows the labels of the phonetic segments. Even though the curves don’t match perfectly the resulting animations look fine. One of the most important things to look at on these curves is how well closures match both in terms of timing and amplitude. The three sentences are from top to bottom: “Upton saw a disaster coming”, “The parrot can say Uppsala very well” and “I met the boob yesterday”.

Symbol of pitch accent	Movement of the pitch of the fundamental frequency (F0)
H*	High - upper end of the pitch range.
!H*	Down-stepped high; somewhat lower than H.
L+H*	Low, moving high.
L*	Low - lower end of pitch range

Phrase boundary	Movements of F0
H-H%	Pitch high and rising higher towards end; typical for yes-no question.
L-H%	Pitch low and rising towards end; typical for comma.
L-L%	Pitch low, staying low; typical for end of a statement.

Figure 3.16. ToBI set of prosodic events. These tables show the main prosodic events of the English language annotated with the ToBI convention.

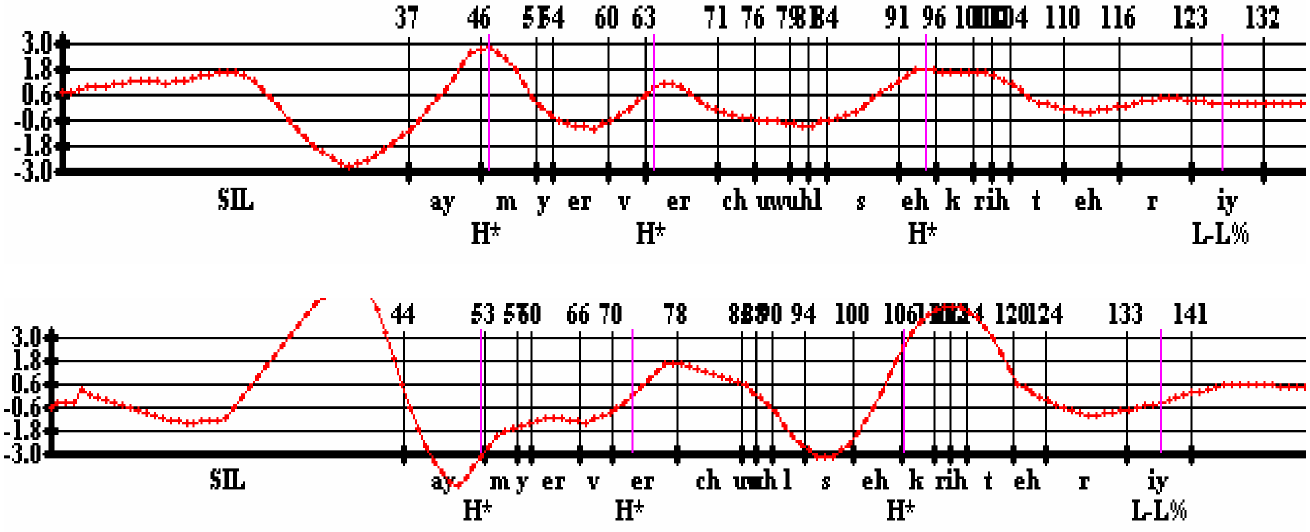


Figure 3.17. Prosodic events. These graphs show the head movements (rotation around x axis, or pitch angle) as a function of time. On the horizontal axis, the top numbers indicate the frame number and on the bottom, the first rows shows the phonetic segments and the second rows shows the prosodic events (ToBI set annotations) that were predicted from the text (also indicated by pink vertical lines). The two graphs show the sentence: "I am your virtual secretary", the top one uttered in a neutral manner and the bottom one in a happier, more expressive fashion. We have found correlations between predicted prosodic events and head movements.

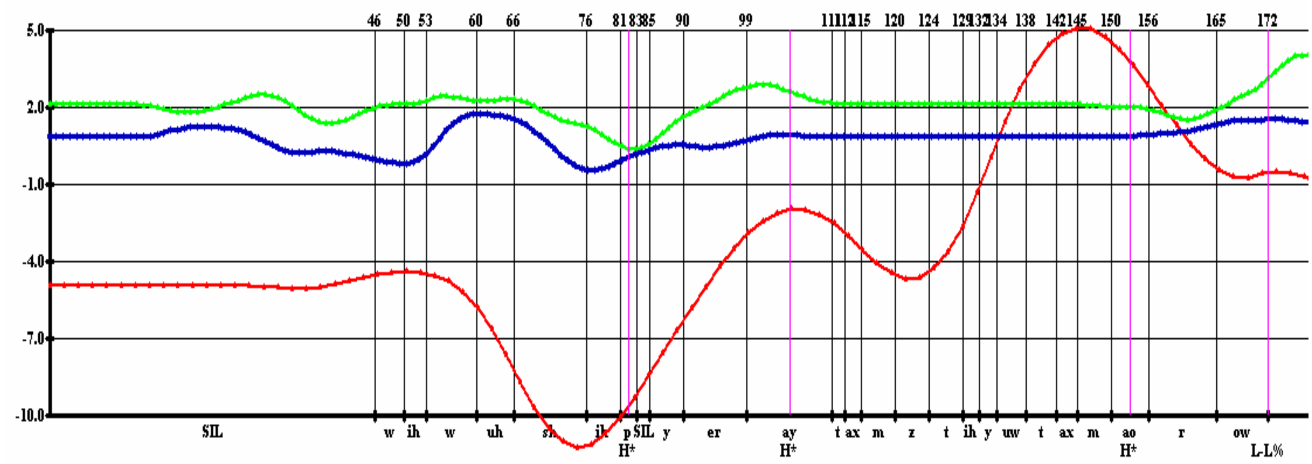


Figure 3.18. Synthesized head motion. The graph plots the head movements as a function of time. The red curve shows the pitch angle (rotation around the x-axis), the green curve the yaw angle (rotation around the y-axis) and the blue curve the roll angle (rotation around the z-axis). On the horizontal axis, the top numbers indicate the frame number and on the bottom, the first rows shows the phonetic segments and the second rows shows the prosodic events (ToBI set annotations) that were predicted from the text and used to drive the synthesis of the head motion. The trajectories are computed by adding normalized recorded segments of head motion around prosodic events.

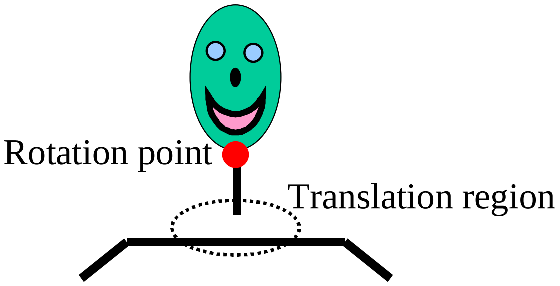


Figure 3.19. Simplified articulation model of the head. We approximate the complex mechanics of the spine in the region from the shoulders to where it attaches to the skull. For the rotation component of the skull, a single rotation point is defined at the base of the skull, and for the translation component, the head-neck compound is allowed to shift over the shoulders. A combined translation component of the head-neck-shoulder is superimposed. The disconnect of the head-neck relative to the shoulders is tolerable for small shifts because we use images that are feathered (edge-blended) together, masking potential artifacts.

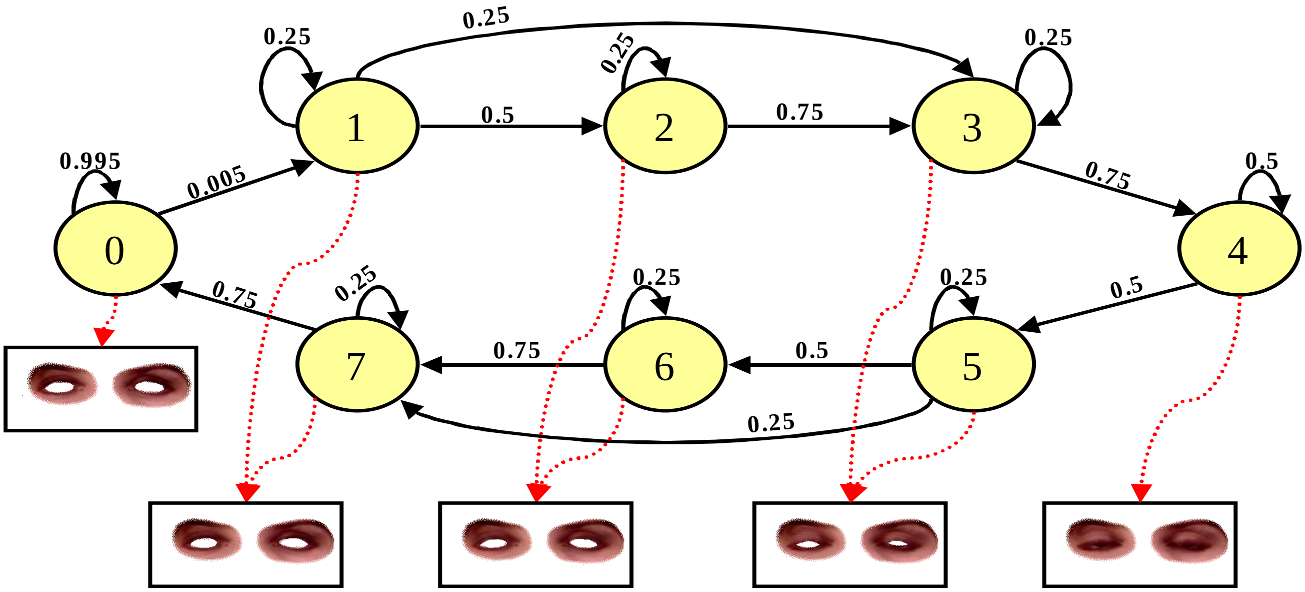


Figure 3.20. State machine for eye-blinks. By observing eye-blinks in our recordings, we obtained this probabilistic state machine. Transition probabilities are indicated along the arcs and each state corresponds to a bitmap of the eyes. For each frame of an animation, a probabilistic transition is performed using the computer pseudo-random number generator, resulting in an eye-blink of varying characteristics.

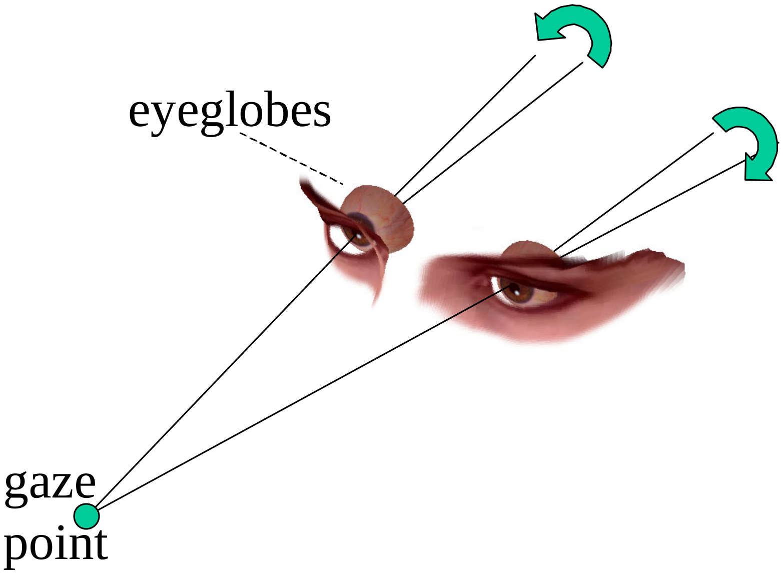


Figure 3.21. Eye-globes and gaze. The eye-globes are modeled with two half-spheres. Their rotations relative to the eye cavity is controlled by the gaze point. The ray going from the globe center through the pupil is constrained to reach the gaze point, allowing solving for the angles.

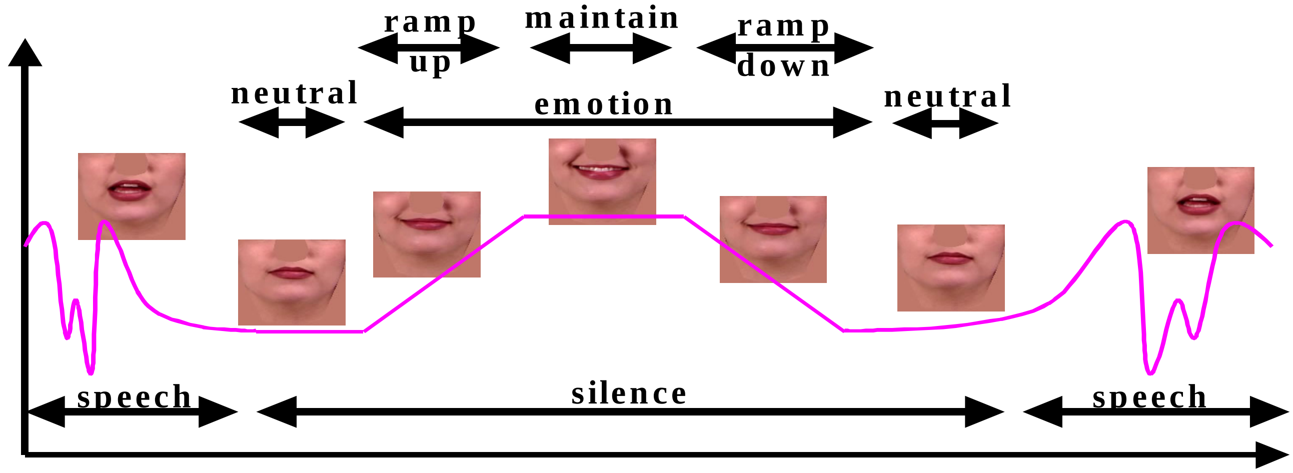
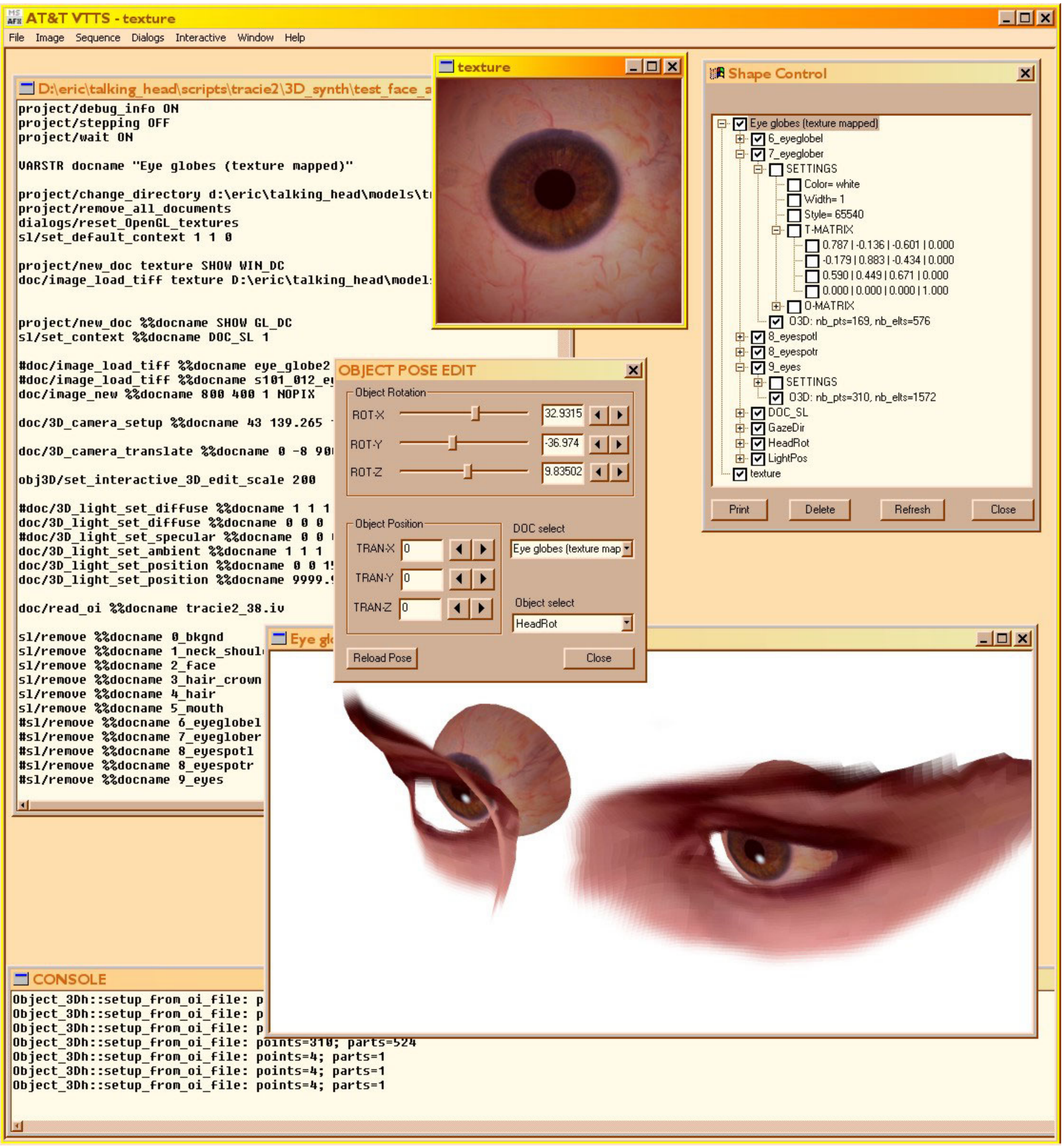


Figure 3.22. Emotion insertion. This graph symbolically shows how we insert emotions inside silence segments of an animation. The emotion segments were recorded from a neutral to an emotional appearance. Hence they can be easily blended inside a silence segment where the mouth is also in a neutral appearance. The intensity of the inserted emotion can be controlled by simply choosing how far into the database emotion segment it is ramped up.

```
VARBLOCK1 avg_seglen_v 0

# loop over min beam width (%L1)
LOOP %%minlen_start %%minlen_stop %%minlen_step
  # loop over max beam width (%L2)
  LOOP  %%maxlen_start %%maxlen_stop %%maxlen_step
    # loop over min cost (%L3)
    LOOP %%mincost_start %%mincost_stop %%mincost_step
      # loop over max cost (%L4)
      LOOP %%maxcost_start %%maxcost_stop %%maxcost_step

        VARINT min_len %L1
        VARINT max_beam_width %L2

        CALL unitsel_do_viterbi_analyze.seq

        unitsel/get_result_param avg_seglen p true
        block1/push_back %%avg_seglen_v %%p

      ENDLOOP
    ENDLOOP
  ENDLOOP
ENDLOOP
```

Figure 3.23. Example of script. The scripting language we developed is line-based and supports minimal loop and conditional function calls. In this example, four parameters are scanned for pruning the search graph and the results are saved in vectors.



D:\eric\talking_head\scripts\tracie2\3D_synth\test_face_a

```
project/debug_info ON
project/stepping OFF
project/wait ON

VARSTR docname "Eye globes (texture mapped)"

project/change_directory d:\eric\talking_head\models\t
project/remove_all_documents
dialogs/reset_OpenGL_textures
sl/set_default_context 1 1 0

project/new_doc texture SHOW WIN_DC
doc/image_load_tiff texture D:\eric\talking_head\model


project/new_doc %%docname SHOW GL_DC
sl/set_context %%docname DOC_SL 1

#doc/image_load_tiff %%docname eye_globe2
#doc/image_load_tiff %%docname s101_012_e
doc/image_new %%docname 800 400 1 NOPIX

doc/3D_camera_setup %%docname 43 139.265

doc/3D_camera_translate %%docname 0 -8 90

obj3D/set_interactive_3D_edit_scale 200

#doc/3D_light_set_diffuse %%docname 1 1 1
doc/3D_light_set_diffuse %%docname 0 0 0
#doc/3D_light_set_specular %%docname 0 0
doc/3D_light_set_ambient %%docname 1 1 1
doc/3D_light_set_position %%docname 0 0 1
doc/3D_light_set_position %%docname 9999.

doc/read_oi %%docname tracie2_38.iv

sl/remove %%docname 0_bkgnd
sl/remove %%docname 1_neck_shoul
sl/remove %%docname 2_face
sl/remove %%docname 3_hair_crown
sl/remove %%docname 4_hair
sl/remove %%docname 5_mouth
#sl/remove %%docname 6_eyeglobel
#sl/remove %%docname 7_eyeglober
#sl/remove %%docname 8_eyespotl
#sl/remove %%docname 8_eyespotr
#sl/remove %%docname 9_eyes
```

```
☑ Eye globes (texture mapped)
    ☑ 6_eyeglobel
    ☑ 7_eyeglober
        ☐ SETTINGS
            ☐ Color= white
            ☐ Width= 1
            ☐ Style= 65540
            ☐ T-MATRIX
                ☐ 0.787 | -0.136 | -0.601 | 0.000
                ☐ -0.179 | 0.883 | -0.434 | 0.000
                ☐ 0.590 | 0.449 | 0.671 | 0.000
                ☐ 0.000 | 0.000 | 0.000 | 1.000
            ☐ O-MATRIX
        ☑ O3D: nb_pts=169, nb_elts=576
    ☑ 8_eyespotl
    ☑ 8_eyespotr
    ☑ 9_eyes
        ☐ SETTINGS
        ☑ O3D: nb_pts=310, nb_elts=1572
    ☑ DOC_SL
    ☑ GazeDir
    ☑ HeadRot
    ☑ LightPos
☑ texture
```

Object Rotation

ROT-X 32.9315

ROT-Y -36.974

ROT-Z 9.83502

Object Position

TRAN-X 0

TRAN-Y 0

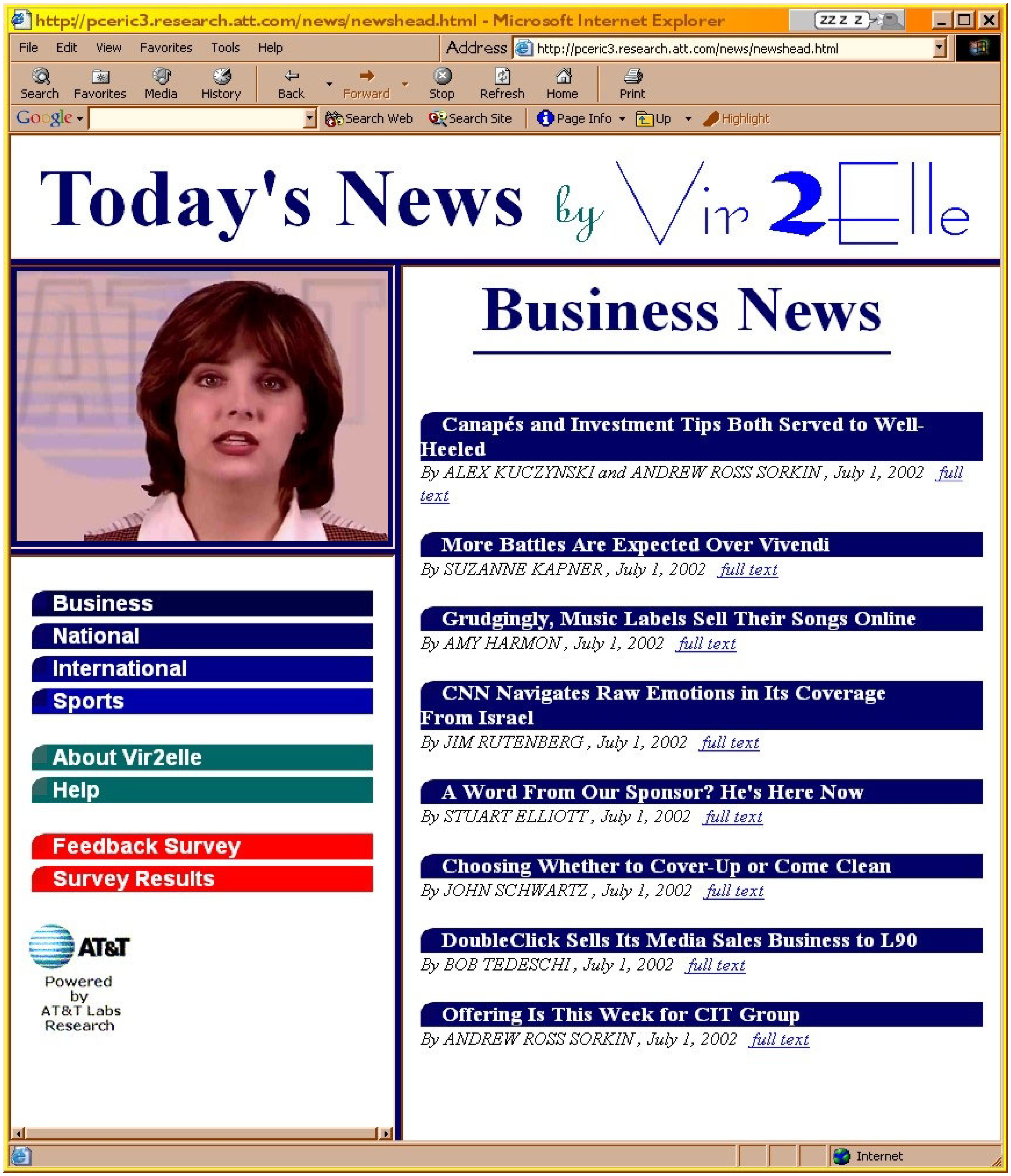
TRAN-Z 0

DOC select: Eye globes (texture map

Object select: HeadRot

```
Object_3Dh::setup_from_oi_file: p
Object_3Dh::setup_from_oi_file: p
Object_3Dh::setup_from_oi_file: p
Object_3Dh::setup_from_oi_file: points=310; parts=524
Object_3Dh::setup_from_oi_file: points=4; parts=1
Object_3Dh::setup_from_oi_file: points=4; parts=1
Object_3Dh::setup_from_oi_file: points=4; parts=1
```

Figure 3.24. Screen shot of the development environment. We develop all our applications using this integrated development environment. It allows scripting and supports display and editing of 2D and 3D objects.


http://pceric3.research.att.com/news/newshead.html - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://pceric3.research.att.com/news/newshead.html
Search Favorites Media History Back Forward Stop Refresh Home Print
Google Search Web Search Site Page Info Up Highlight
Business
National
International
Sports
About Vir2elle
Help
Feedback Survey
Survey Results
AT&T
Powered by AT&T Labs Research
Internet

Today's News by Vir2Elle

Business News

Canapés and Investment Tips Both Served to Well-Heeled
By ALEX KUCZYNSKI and ANDREW ROSS SORKIN , July 1, 2002 full text

More Battles Are Expected Over Vivendi
By SUZANNE KAPNER , July 1, 2002 full text

Grudgingly, Music Labels Sell Their Songs Online
By AMY HARMON , July 1, 2002 full text

CNN Navigates Raw Emotions in Its Coverage From Israel
By JIM RUTENBERG , July 1, 2002 full text

A Word From Our Sponsor? He's Here Now
By STUART ELLIOTT , July 1, 2002 full text

Choosing Whether to Cover-Up or Come Clean
By JOHN SCHWARTZ , July 1, 2002 full text

DoubleClick Sells Its Media Sales Business to L90
By BOB TEDESCHI , July 1, 2002 full text

Offering Is This Week for CIT Group
By ANDREW ROSS SORKIN , July 1, 2002 full text

Figure 3.25. Screenshot of the Automatic Newscaster application. The web site is organized using page frames. On the left side of the screen our talking-head is speaking a summary of the headlines, while on the right side of the screen, the user can click on any headline and have the talking-head directly skip to it or alternatively access the full text for reading. The talking-head animation is generated entirely automatically from the textual content downloaded from the Internet.

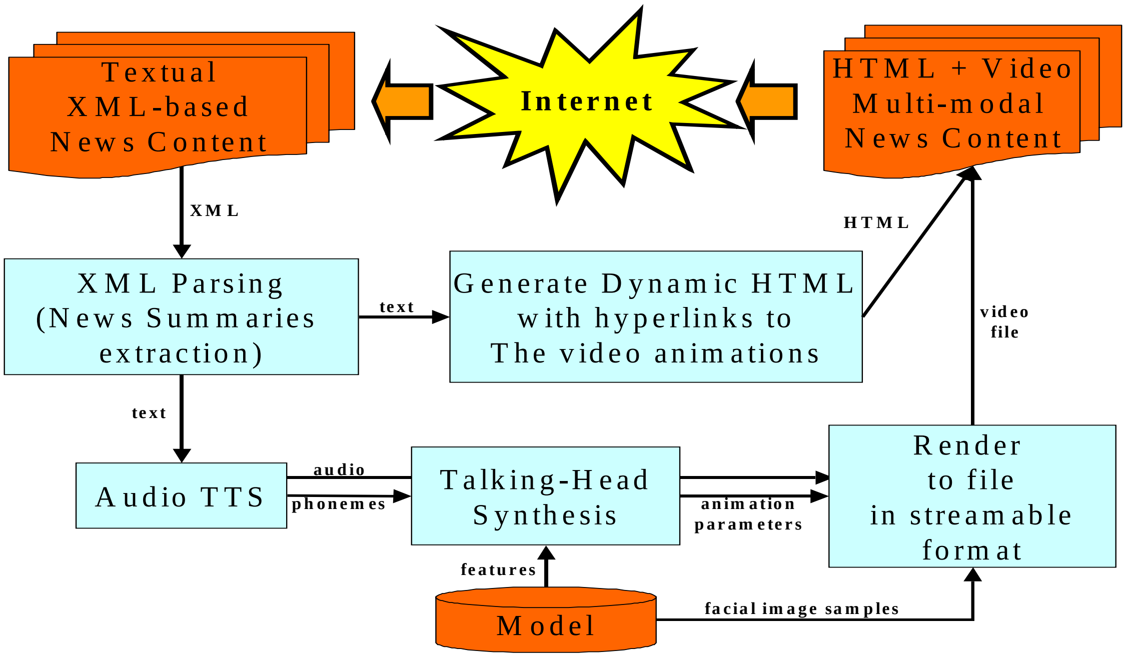

Textual XML-based News Content
Internet
HTML + Video Multi-modal News Content
XML
HTML
XML Parsing (News Summaries extraction)
text
Generate Dynamic HTML with hyperlinks to The video animations
video file
text
Render to file in streamable format
Audio TTS
audio
phonemes
Talking-Head Synthesis
animation parameters
features
Model
facial image samples

Figure 3.26. Architecture of the automated newscaster application. News in XML format are obtained from the web and parsed to extract their structure and content. The raw text is then sent to the text-to-speech module (TTS) that produces the audio and the phonetic string used by the video synthesis module to compute the animation parameters. Finally, the renderer module synthesizes the animation into a streamable video file. The structure extracted from the XML document is used to build the final HTML page with links to the synthesized video files.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE   smil   PUBLIC   "-//W3C//DTD   SMIL   1.0//EN"   "http://www.w3.org/TR/REC-
smil/SMIL10.dtd">
<smil>
   <head>
      <layout>
        <root-layout background-color="white" width="320" height="240"/>
        <region id="videoregion" width="310" height="230" z-index="2"/>
        <region id="borderregion" width="320" height="240" background-color="#006666" />
      </layout>
   </head>
   <body>
             <seq>
                   <video src="rtsp://pceric3.research.att.com/news/b_news/05_b.rm"
                          region="videoregion" type="audio/x-pn-realaudio-plugin" />
                   <video src="rtsp://pceric3.research.att.com/news/b_news/06_b.rm"
                          region="videoregion" type="audio/x-pn-realaudio-plugin" />
             </seq>
   </body>
</smil>
```

Figure 3.27. An example of a SMIL file for the automated newscaster. The body of the SMIL code contains the <seq> ... </seq> codes which defines a suite of sequential multimedia operations. In this case, four video clips (05_b.rm, 06_b.rm, 07_b.rm, 08_b.rm) will be played sequentially. This code is invoked when the user clicks on the 5th story on the right pane of the news page (the agent not only speaks the selected story, but also the following ones). Similar SMIL files are generated for every story.

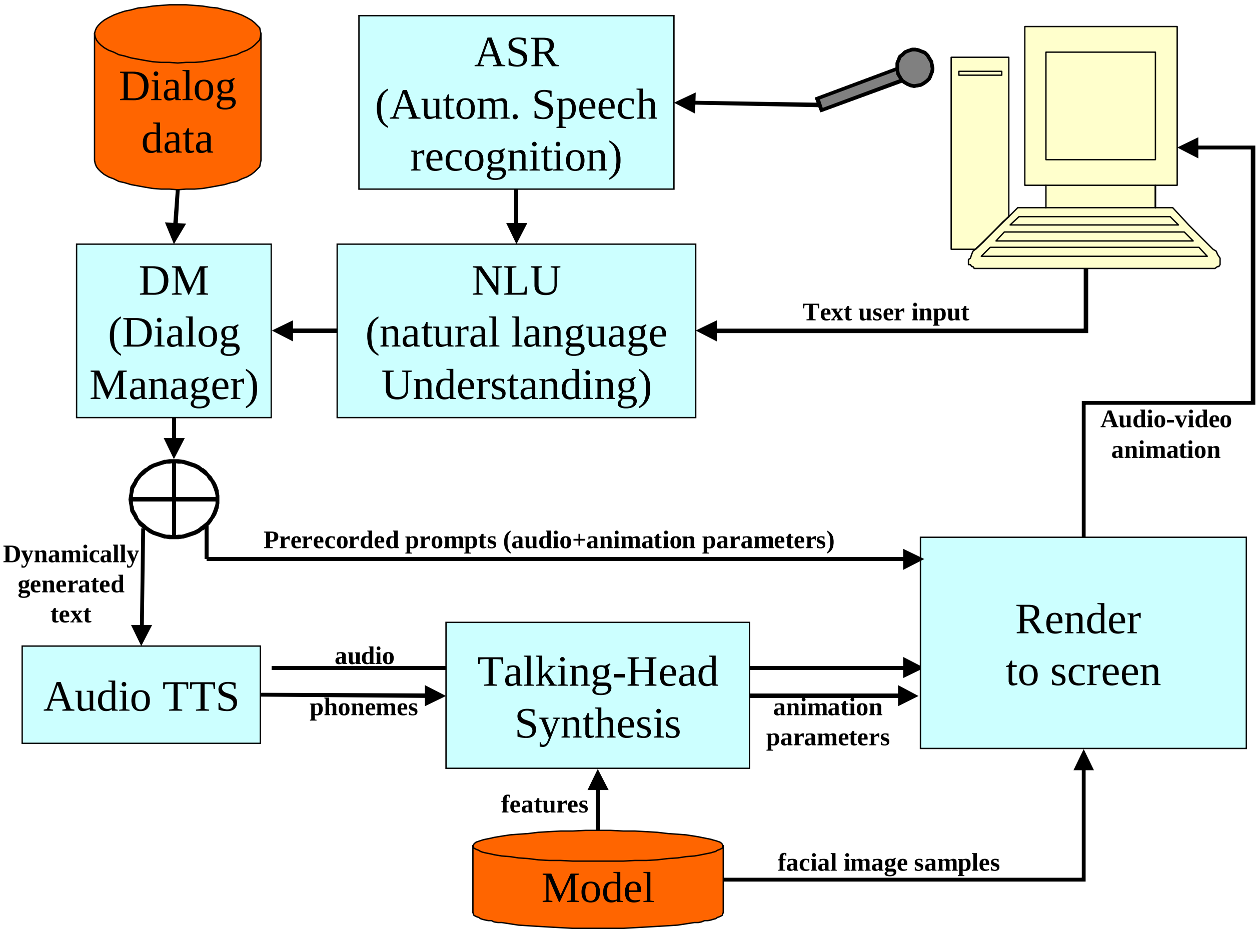


Figure 3.28. Architecture of the “Help Desk” application. The user interfaces with the “Help Desk” application using either textual input (keyboard) or via a microphone (through an automatic speech recognition module (ASR)). The latter has not yet been implemented (but all the pieces are readily available). The NLU module extracts structure and meaning from the raw text and ships the results to the dialog manager, which in turn provides the textual answers for the agent based on dialog data and state information. To reduce the workload on the server, the prompts that are known in advance are pre-computed into animation parameter files and in this case, can be rendered directly. The dynamic content on the other hand has to be synthesized and hence is sent to the TTS and talking-head synthesis module which compute the animation parameters and send them to the real-time renderer.

CHAPTER 4

CONCLUSIONS

Table of Contents

4.1 General Conclusions 95
  4.1.1 Model Creation 95
  4.1.2 Synthesis 96
4.2 Future Work.......... 97
4.3 Technical Contributions 97
4.4 Personal Contributions 98
4.5 Acknowledgements.......... 98
4.6 Curriculum Vitae 99
  4.6.1 Professional experience.......... 99
  4.6.2 Education.......... 99
  4.6.3 Other 99
  4.6.4 Publications 100
  4.6.5 Patents 101

4.1 General Conclusions

In this work, we have presented a system that is capable of synthesizing video-realistic talking-head animations. Our approach to building a head model is sample-based and works in two phases. The ‘off-line’ phase consists of building a model of a person’s head. With the model, the system is then capable of generating ‘on-line’ any speech animation from a phoneme string (either TTS annotations or labeled recorded audio). The synthesis can be achieved in real-time for applications requiring it or in close-to-real-time for optimum video realism. The results are very good in both cases. Many people can’t tell whether our animations are synthesized or recorded.

4.1.1 Model Creation

The model creation process is performed in several steps: recording – digitizing – image analysis – sample extraction – database creation – database scrubbing. Only the first and the last steps are manual. The rest involves running scripts and is mostly automatic with only few parameters to adjust for particular recording conditions. At this stage, we are still working on streamlining this process with the final goal being that a trained operator can create a model in three days: one day for recording, the second day for digitizing and correcting the scripts (so that the recorded audio matches the text of the script for automatic phonetic alignment), the second night for running the analysis (automatic) and the third day to scrub the database. This might seem like a long time to create a head model but we are aiming at creating high quality models for “serious” use, not the “at-home-for-fun” type. In this context, people and businesses are ready to invest into creating these models because the high up-front cost is quickly offset by the savings incurred in replacing expensive manual video shoots by automatic synthesis.

The automated part of model creation process starts with a hierarchical 3D model of the head where facial parts such as mouth eyes and eyebrows are modeled separately from the "base-head" (surrounding skin, nose, ears, hair, etc.) We obtain the 3D model by scanning the talent with a Cyberscanner, but simpler (and cheaper) techniques can be used because the modeling of the facial parts need not be of high precision. In fact we only recently acquired the Cyberscanner, mostly to produce a full 3D model of the head for real-time rendering. The raw Cyberware data is only a cloud of 3D points with color information. We recently developed a technique to fit our hierarchical, animatable head model to the scanned data. The method is semi-automatic and requires the user to enter a set of corresponding pairs of points on the existing head model and the scanned data. This is easily done using a 3D graphical interface. The matching of the remaining points is performed automatically by interpolation on a cylindrical coordinates system. The system uses the 3D head shape of each facial part both for analysis and for synthesis, allowing head movement both on the recordings (no constraints placed on the talent) and on the synthesized animation (needed for realistic appearance).

Image analysis techniques are then used to process the raw audio-visual data into a normalized database of facial parts. The low-level processing of each image to identify feature points has been perfected over the years and is very robust. While fully automatic, it is not completely user independent and some parameters may need to be tuned to adapt to particular skin tone and recording setup. To make the system robust to varying conditions, we had initially trained the low-level image processing on speech recordings of over 50 different people. With adjustments for individual recordings, the system essentially performs without errors on ‘good’ segments (an occasional laugh or wild head movement can throw the recognizer off). We are working on making these adjustments simple and intuitive to allow trained operators to perform this task reliably. Overall, the current procedure works reliably and has been successfully used on 5 different talents. There are very few errors the system makes in creating the databases and they are mostly due to the talents turning their head beyond the allowed range or producing spontaneous laughs. These outliers are simply discarded during the manual process of scrubbing the database.

A difficulty in sample-based methods is that the model being made of samples tend to be very large. The size of a talking-head model is about 100MB and is too large for direct installation over the Internet. On the other hand it is not prohibitive for distribution on a CD-ROM and, once installed, it can fit entirely in memory on most of today's machine, enabling real-time or close-to-real-time synthesis.

4.1.2 Synthesis

The synthesis process uses a graph search to find the optimal animation of the mouth from a target phoneme string (either labeled recorded audio, or TTS phonetic annotation). The graph search is directed by a set of cost functions that ensure both the synchronization of the lips with the target audio and a smooth visual animation. A smart pruning mechanism reduces the complexity of the search to allow real-time implementation of the algorithm. This module is the centerpiece of this work and has been refined and optimized considerably. An MMX implementation of critical routines has even been written to reduce latency to about 1/20th of real-time. We have found this algorithm to provide reliable results on hundreds of synthesis clips we have produced over time. The only errors come from erroneous phonetic labeling of the database. For example, the audio labeler has difficulty to align closures such as 'm', 'b' and 'p' with the images of a closed mouth (the acoustic features during closures are not well marked, making it difficult for the labeler to precisely find the boundaries of theses phonemes). Such labeling errors result in missed closures during synthesis. Closures being a key visual cue for lip-reading, we set out to manually check and, when necessary, correct all closures in the database. This is an issue that needs to be resolved in the future.

A visual prosody module drives the movements of the head and the eyes based on textual analysis. A set of probabilistic state machines add random mechanical movements such as eye blinks, lip parting and small head movements. The prosody module has produced animations that have been found more pleasing than the previous version where random motions were used for the head movements. At the moment, the prosody module is still in its infancy and further work is needed to provide for flexible synthesis of prosodic elements such as head and eyebrows movements. This is an area for further research.

An expression module allows for insertion of smiles, frowns, etc. The length and intensity of expressions are parameterized, allowing easy insertion of these emotions during silences within a speech animation. Expressions are key in the success of talking-heads as a whole. They do make the heads be perceived as more human, providing comfort and liveliness. However, the capture and digitization of these expressions is mostly manual and it is something of an art to elicit these expressions from the talent. This will probably remain a part of the capturing process that can't be automated.

Two "base head" rendering engines have been developed. One uses recorded video of head movements over which mouth and eyes are overlaid. This model is coined 2.5D, referring to the well-known animation technique. The 2.5D renderer produces animations that are often indistinguishable from real recorded videos. It is, however, limited by a lack of flexibility, large footprint and slower rendering (about 10 frames per second). At this point it is not parameterized and hence has only limited capability to produce on-demand head movements (where head movements are synchronized with the textual content, for example). It requires the storage of full size video clips which need to be loaded for rendering. The 2.5D renderer is well suited for applications that can afford an off-line synthesis such as the Automated Newscaster.

The second renderer is a full 3D renderer. Head rotations and translations as well as eye globe movements are fully parameterized on that renderer, making it well suited to produce on demand animations such as visual prosody and idle motions. The 3D renderer can take advantage of hardware acceleration for graphics operations such as texture mapping that are now standard on most video cards, making it ideal for applications that demand real-time rendering. Coupled with fast unit selection algorithms, this renderer was key in implementing the Help Desk real-time application.

4.2 Future Work

This is an application-oriented work, aimed at solving the specific problems involved in creating video-realistic talking-head animations. It is not very likely that it will spark research in other directions than that of talking-heads. Hence the areas for future work are targeted at improving the system:

- Prosodic unit selection.
- Reduce the footprint of the model.
- Streamline the model creation process.
- Improve modeling and realism of 3D model.
- Target phonetic audio labeling specifically to the synthesis.
- Integrate voice creation with visual part.

Some of these items include developmental work. For example, the model creation process should be streamlined, simplified, documented and made intuitive and accessible to trained technicians. This is a necessary step for this approach to be productized.

A promising area for research is prosodic unit selection. At the moment we are using prosodic information to drive head movements by simply concatenating recorded trajectories. Eventually the right approach is to use a trajectory graph built from recorded head movements and find the optimal trajectory with a graph search using cost functions based on smoothness and prosodic features. A similar approach could be used with the 2.5D renderer by using recorded head movements videos and build an optimal animation based on visual costs and prosodic features.

Progress in reducing the database footprint should articulate around two axes. First, the number of recorded frames being inserted into the database should be lowered to a minimum. Clustering techniques can be used to identify key frames and discard similar instances. Second, compression techniques should be developed to take advantage of the similar nature of the frames in the database (normalized facial parts from a single person).

Another area of research includes the adaptation of the talking-head model to new data from a 3D scanner. The talking-head model contains a set of 3D shapes representing facial parts. The construction of the model has involved manual intervention. When new talents are recorded we will automatically adapt the existing head model to the scanned 3D data of the new talent. This model also needs to be improved in terms of visual pleasantness. In particular eyes and hair need to be modeled more carefully.

The phonetic audio labeling need to be tuned specifically for accurate visual labeling. At the moment, most of the errors in synthesis come from bad audio labeling. For example closures (‘m’,’b’,’p’) need to be marked where the lips are actually closed otherwise closures might be missed during synthesis (a very visible mistake). We are planning on using analysis results from the visual channel to refine the audio labeling to solve this issue in an automated fashion.

Even further along, we will have to include audio modeling in our work so that a complete talking-head model (visual + voice) can be created in one shot.

4.3 Technical Contributions

The sampled-based talking-head project was started in AT&T Labs in 1996. It originated from early work by Tsuhan Chen and Hans-Peter Graf on low bitrate video telephony. The facial recognition aspect of the work then was developed by Hans-Peter and I later started the synthesis effort.

One technical contribution in this thesis is the audio-visual unit-selection. Originally inspired from concatenative speech synthesis, we adapted this technique to perform variable length visual concatenation with smoothness and synchronization constraints. A second contribution is the original combination of 2D and 3D graphics. We successfully demonstrate that the photo-realism of 2D graphics can be given a significant degree of flexibility in terms of animation. Another area where we provided original contributions is in the facial parts normalization module, which includes facial feature location, head pose estimation and normalized bitmaps extraction. While other groups have shown systems with similar functionality, our approach is original, robust and provides both location and measurement of facial parts. The area of visual prosody is currently emerging and we are providing early contributions by combining modeling and motion capture results. Finally, the demonstrated applications represent a substantial effort in terms of implementation and while contributing only marginally in the technical domain, they certainly contribute substantially in evaluating and exploring the use of talking-head agents.

4.4 Personal Contributions

Overall, the AT&T VTTS (visual text-to-speech) project has been a team effort with both Hans-Peter and myself contributing equally. I have been the lead researcher on the modeling and synthesis aspect of talking-heads, while Hans-Peter Graf has been leading the facial analysis aspects of the project. Within the analysis part, I contributed the 3D pose estimation and sample normalization. Hence, in chapter 2, only section 2.6.2 stems mostly from Hans-Peter's work but all figures (except Figure 2.23) and text of that section are original of this thesis. In Chapter 3, the work covered in section 3.4.10 and 3.5.2 was conducted mostly by Hans-Peter. The rest, including the design and implementation of the development environment and the two applications are my personal contributions.

4.5 Acknowledgements

I started writing this Thesis ten years after finishing my Masters. This "interval" turned out to be both an advantage and an inconvenient. An advantage because, having had the good fortune to find a position at AT&T Bell Labs, I had plenty of time to research the field, write papers and get good results, all the while being paid for it. An inconvenient though because, as the interval grew, the willingness, energy and motivation to embark on writing a PhD thesis diminished. This is where I'd like to acknowledge several people who have been instrumental in getting me started and helped in keeping me motivated. First my dear friend and esteemed colleague Hans-Peter Graf with whom I've been fortunate to work for these past years. His vision and sheer determination have largely contributed to make this project a success. I also gratefully acknowledge my bosses at AT&T: Yann Lecun, Juergen Schroeter, Rich Cox and especially Larry Rabiner, who, by believing in the project, have made it possible and secured the necessary funding for it. I want to thank Professor Murat Kunt for his leadership, encouragements and for being so flexible as to let me work on the thesis remotely (including a video-conferenced exam!). And last but not least, I want to recognize my family, friends and colleagues for encouragements, littles pushes and sometimes big shoves along the way: my wife Isabelle, my parents, my friends: Simon, Elyes, Carina, Michael, Patrice, Hillary, Léon, Andrea, Daniela, Tony, Marwan, Vladimir, Joern… Finally I dedicate this thesis to my little monster Alexia who is 2 years, 7 months, 25 days, 3 hours and 50 minutes old as I write this line.

4.6 Curriculum Vitae

4.6.1 Professional experience

1997 – now: ***Principal Scientist*** *AT&T Labs - Research, Middletown, NJ*

Leading research projects on multi-modal user interfaces, computer vision and image synthesis.

- Photo-realistic talking-heads synthesis from image samples, including applications in multi-modal user interfaces.
- Automatic video enhancement (virtual lighting) with application in video-conferencing.
- Sample-based Image Synthesis. Facial animation using morphing and model-based sample animations.
- Face tracking, Face pose detection, and lip-reading.
- Developed a Web-based automatic newsreader. A photo-realistic talking head constantly presents the latest world, business and technology news in a novel multi-modal fashion.
- Developed an integrated platform for prototyping image analysis/synthesis algorithms and applications. Entirely written in C++ with MFC, featuring scripting capabilities, graphical I/O and optimized libraries (PCA, Wavelets, LLE, Nearest-Neighbors, Object recognition, N-grams searches, Graphs optimization)

1992 - 1996: ***Member of Technical Staff*** *AT&T Bell Laboratories, Holmdel, NJ*

Leading research projects on hardware accelerated image analysis.

- Board-level system design of accelerator boards for ultra fast image convolutions on several platforms VME bus, SCSI bus, and PCI bus.
- Design of low-level drivers and libraries to implement hardware-accelerated standard image processing functions.
- Design of high-speed address block location for mail pieces, bill reading and check reading using hardware-accelerated libraries.

1991: ***Assistant Researcher*** *Swiss Federal Inst. Of Technology, Lausanne, Switzerland*

Research in image processing for robotics control using special neural network hardware and field programmable gate array chips.

1989: ***IAESTE-Sponsored Summer Internship*** *ISEL Corp. Barcelona, Spain*

Software development for satellite telecontrol (C, UNIX). Evaluation of graphical data display software.

4.6.2 Education

1986-1991: ***Master Degree in Computer Science*** *Swiss Federal Inst. Of Technology,*
Lausanne, Switzerland

1999-now: *PhD Thesis* *Signal Processing Lab**,** Swiss Federal Inst. Of Technology,*
Lausanne, Switzerland

4.6.3 Other

- Several interviews featuring the Photo-Realistic Talking-Heads were broadcast on TV and Radio (CBS 3/11/01, CNBC 3/9/01, Fox News 1/1/00, Swiss National Radio 10/3/95).
- 20 publications in journals and conference proceedings; 5 patents issued, several pending.
- Several commercial products impacted by my research, including:

- Field-tested an address block reader for the USPS (postal Service) capable of processing 20K second/second. (1995)
- Telephone bill Reader used commercially by AT&T (1996)
- Check Reader Module in NCR ATM (automatic teller machines). (1996)

4.6.4 Publications

1. Graf, H.P., Cosatto E., Strom, V., Huang, F.J., "**Visual Prosody; Facial Movements Accompanying Speech**" International Conference on Automatic Face and Gesture Recognition (FG02), Holiday Inn Capital, 550 C Street N.W., Washington, DC, USA, 20-21 May 2002.
2. Huang, F.J., Graf, H.P., Cosatto E., "**Triphone-Based Unit Selection for Concatenative Visual Speech Synthesis**", ICASSP 2002, Orlando, Fl, May 13-17, 2002.
3. Basso, A., Cosatto, E., Graf, H. P., Gibbon, D. and Liu, S., "**Virtual light: Digitally-generated lighting for video conferencing applications**", *2001 International Conference on Image Processing*, IEEE Signal Processing Society, Thessalonica, Greece, 7-10 Oct. 2001.
4. Graf, H.P., Cosatto, E., "**Sample-Based Synthesis of Talking-Heads**", *ICCV-RATFG-RTS '01*, pp. 3-7.
5. Cosatto, E., Graf, H.P., "**Synthetic Talking Heads with Sample-Based Graphics**", Col. of abstracts, *Learning workshop*, Snowbird, April 2001.
6. Cosatto, E. and Graf, H. P., "**Photo-realistic talking-heads from image samples**", *IEEE Trans. on Multimedia*, vol. 2, no. 3, Sept. 2000, pp. 152-163.
7. Cosatto, E. and Graf, H. P., "**Audio-visual unit selection for the synthesis of photo-realistic talking-heads**", *ICME 2000*, New York, NY, 2000.
8. Graf, H. P., Cosatto, E. and Ezzat, T., "**Face analysis for the synthesis of photo-realistic talking heads**", *Fourth IEEE Int. Conf. Automatic Face and Gesture Recognition*, Grenoble, France, 2000, pp. 189-194.
9. Cosatto, E. and Graf, H. P., "**Sample-based synthesis of photo-realistic talking heads**", *Computer Animation 98*, IEEE Computer Society Press, 1998, pp. 103-110.
10. Graf, H. P., Cosatto, E. and Potamianos, G., "**Machine vision of faces and facial features**", *Proc. of the Second RIEC International Symp. on Design and Architecture of Information Processing Systems Based on the Brain Information Principle*, Sendai, 1998, pp. 48-53.
11. Potamianos, G., Graf, H. P. and Cosatto, E., "**An image transform approach for HMM based automatic lipreading**", *Proc. International Conf. on Image Processing*, vol. III, Chicago, 1998, pp. 173-177.
12. Graf, H. P., Cosatto, E. and Potamianos, G., "**Robust recognition of faces and facial features with a multi-modal system**", *Proc. of the International Conf. on Sys., Man, and Cybernetics*, Orlando, Florida, October 1997, pp. 2034-2039.
13. Potamianos, G., Cosatto, E., Graf, H. P. and Roe, D. B., "**Speaker independent audio-visual database for bimodal ASR**", *Proceedings of the ESCA/ESCOP Workshop on Audio-Visual Speech Processing*, Rhodes, Greece, September 1997, pp. 65-68.
14. Cloutier, J., Cosatto, E., Pigeon, S., Boyer F. and Simard S., "**VIP: An FPGA-based Processor for Image Processing and Neural Networks**", *Fifth International Conference on Microelectronics for Neural Networks and Fuzzy Systems (MicroNeuro'96)*, 1996.
15. Cosatto, E. and Graf, H.P., "**A high speed image understanding system**", in *Adaptive Analog VLSI Neural Systems*, Jabri, M. A., Coggins, R. J. and Flower, B. G. (Ed.), Chapman & Hall, 1996, pp. 201-222.
16. Graf, H. P., Cosatto, E., Gibbon, D., Kocheisen, M. and Petajan, E., "**Multi-modal system for locating heads and faces**", *Proc. Second Int. Conf. Automatic Face & Gesture Recognition*, IEEE Computer Soc. Press, Los Alamitos, California, 1996, pp. 88 - 93.
17. Cosatto, E. and Graf, H.P., "**A neural network accelerator for image analysis**", *IEEE Micro*, vol. 15, no. 3, IEEE Computer Society Press, June 1995, pp. 32-38.
18. Graf, H. P., Chen, T., Petajan, E. and Cosatto, E., "**Locating faces and facial parts**", *Intl. Workshop on Automatic Face and Gesture Recognition*, Bichsel, M. (Ed.), June 1995, pp. 41-46.

19. Graf, H.P., Burges, C., Cosatto, E. and Nohl, C., “**Analysis of complex and noisy check images**”, *Proceedings of IEEE International Conference on Image Processing (ICIP-95)*, IEEE Computer Society Press, 1995, pp. 316-319.
20. Jackel, L., Battista, M., Baird, H., Ben, J., Bromley, J., Burges, C., Cosatto, E., Denker, J., Graf, H., Katseff, H., LeCun, Y., Noh, C., Sackinger, E., Shamilian, J., Shoemaker, T., Stenard, C., Strom, I., Ting, R., Wood, T. and Zuraw, C., “**Neural-net applications in character recognition and document analysis**”, in *Neural-Net Applications in Telecommunications*, Kluwer Academic, 1995.
21. Cosatto, E. and Graf, H.P., “**NET32K high speed image understanding system**”, *Proceedings of the Fourth International Conference on Microelectronics for Neural Networks and Fuzzy Systems*, IEEE Computer Society Press, 1994, pp. 413-421.
22. Graf, H.P. and Cosatto, E., “**Address block location with a neural net system**”, *Advances in Neural Information Processing Systems*, vol. 6, Morgan

4.6.5 Patents

1. Cosatto, E., Graf, H. P., Potamianos, G. and Schroeter, J., **“Audio-visual selection process for the synthesis of photo-realistic talking-head animations”,** *U. S. Patent Filed,* March 2001.
2. Bottou, L., Cosatto, E., LeCun, Y., Mueller, U., “**Virtual Light: Digitally-Generated Lighting For Video Conferencing Applications**”, *U. S. Patent Filed*, February 2001.
3. Cosatto, E., Graf, H. P. and Potamianos, G., **“Robust multi-modal method for recognizing objects”,** *U. S. Patent #6,118,887*, 2001.
4. Cosatto, E., Graf, H. P. and Schroeter, J., **“Coarticulation method for audio-visual text-to-speech synthesis”,** *U. S. Patent #6,112,177*, 2001.
5. Basso, A., Cosatto, E., Greenspan, S. L. and Weimer, D. M., **“System and method for generating coded video sequences from still media”,** *U. S. Patent Filed*, 29 August 2000.
6. Cosatto, E. and Graf, H. P., **“Method of modeling objects to synthesize three-dimensional, photo-realistic animations”,** *U. S. Patent Filed*, 8 Feb. 2000.
7. Bottou, L., Cosatto, E., LeCun, Y. and Muller, U., **“System and method for controlling the delivery of mass messages”,** *U. S. Patent Filed*, 12 Dec. 1999.
8. Cosatto, E. and Graf, H. P., **“Method for generating photo-realistic animated characters”,** *U. S. Patent #5,995,119*, 30 November 1999.
9. Graf, H. P. and Cosatto, E., **“Multi-modal system for locating objects in images”,** *U. S. Patent #5,864,630*, 26 January 1999.
10. Potamianos, G., Graf, H. P. and Cosatto, E., **“Speaker independent, image sequence transform based automatic speechreading”,** *US Patent submitted*, July 1998.
11. Burges, C., Cosatto, E. and Graf, H. P., **“Method of image enhancements using convolution kernels”,** *U. S. Patent #5,647,027*, 8 July 1997.