Robert Bargmann

Learning-Based Facial Animation

– Ph.D. Thesis –

Dissertation zur Erlangung des Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.)
der Naturwissenschaftlich-Technischen Fakultät I
der Universität des Saarlandes

4. März 2008

max planck institut
informatik

Stuhlsatzenhausweg 85
66123 Saarbrücken
Germany

delivered by
EUROGRAPHICS
DIGITAL LIBRARY
www.eg.org
diglib.eg.org

Eingereicht am 4. März 2008 in Saarbrücken durch

Robert Bargmann
MPI Informatik
Stuhlsatzenhausweg 85
66 123 Saarbrücken

`bargmann@mpi-sb.mpg.de`

Betreuender Hochschullehrer – Supervisor
Prof. Dr. Volker Blanz, Technische Universität Siegen, Germany

Gutachter – Reviewers
Prof. Dr. Hans-Peter Seidel, Max-Planck-Institut für Informatik, Germany
Prof. Dr. Volker Blanz, Technische Universität Siegen, Germany

Dekan – Dean
Prof. Dr. Thorsten Herfet, Universität des Saarlandes, Saarbrücken, Germany

à mes parents

Abstract

This thesis proposes a novel approach for automated 3D speech animation from audio. An end-to-end system is presented which undergoes three principal phases. In the *acquisition phase*, dynamic articulation motions are recorded and amended. The *learning phase* studies the correlation of these motions in their phonetic context in order to understand the visual nature of speech. Finally, for the *synthesis phase*, an algorithm is proposed that carries as much of the natural behavior as possible from the acquired data to the final animation.

The selection of motion segments for the synthesis of animations relies on a novel similarity measure, based on a *Locally Linear Embedding* representation of visemes, which closely relates to viseme categories defined in articulatory phonetics literature. This measure offers a relaxed selection of visemes, without reducing the quality of the animation.

Along with a general hierarchical substitution procedure which can directly be reused in other speech animation systems, our algorithm performs optimum segment concatenation in order to create new utterances with natural coarticulation effects.

Kurzfassung

In dieser Arbeit wird ein neues Verfahren zur automatischen Erzeugung audio-basierter 3D Sprechanimation vorgeschlagen. Ein komplettes System wird vorgestellt, welches in drei Phasen arbeitet. In einer ersten *Phase der Datenaufnahme* werden dynamische Artikulationsbewegungen aufgenommen und ergänzt. In einer zweiten *Lernphase* wird die Korrelation dieser Bewegungen in ihrem phonetischen Kontext untersucht, um die visuelle Natur des Sprechens zu verstehen. Schliesslich wird für die *Phase der Synthese* ein Algorithmus vorgeschlagen, welcher so viel vom natürlichen Verhalten wie möglich aus den aufgenommenen Daten in die endgültige Animation überträgt.

Die Auswahl von Bewegungssegmenten zur Synthese der Animationen beruht auf einem neuartigen Ähnlichkeitsmaß, welches auf einer *Locally Linear Embedding* Repräsentation von Visemen beruht und eng verwandt mit Kategorien von Visemen, wie sie in der Literatur über *Artikulationsphonetik* definiert sind. Dieses Maßermöglicht eine erweiterte Auswahl von Visemen ist, ohne die Güte der Animation zu verringern.

Neben einem allgemeinen Hierarchischen Substitutionsverfahren welches unmittelbar auch in anderen Sprechanimationssystemen verwendet werden kann, führt unser Algorithmus eine optimale Segment-Konkatenation durch, um neue Äusserungsformen mit natürlichen Koartikulationseffekten zu erzeugen.

Summary

Automated speech synthesis based on video footage has shown impressive results. Such high quality results are obtained by the analysis of articulation motions captured at high frame-rates. While these high frame-rates have been provided by video camera setups for quite some time now, the extension of such approaches to 3D has only become possible in recent years with the availability of dynamic 3D scanners. This extension to 3D models is important not only because it relieves the synthesized animation of a single visualization viewpoint constraint, but also because it facilitates the transfer of speech to different identities. The direct transfer of 2D algorithms to 3D remains an open problem that is challenging because the increase of dimensionality presents a more complex process of articulation and has to be considered more thoroughly.

In a learning-based framework, novel animations are synthesized by the concatenation of recorded motion segments available in a database. To this end, an algorithm performs a selection of segments that match a novel audio track and strives at providing smooth continuity over the selected sample sequence. This continuity is essential, as it reduces the necessary interpolation that distorts the behavior in the final outcome. Increasing the size of the database augments the chances of finding smoothly continuous segments but also demands extremely large storage space.

In this thesis, a new behavioral study of articulation is performed which takes advantage of the redundancy in the database: by observing the behavior of the mouth for the different phonemes (phonetic sound units), our system proposes a similarity measure that defines which additional motion segments can be used for phonemes to which these motion segments are not associated in the original corpus.

In a hierarchical substitution procedure, the selection process retrieves an enhanced list of valid segments over a small corpus. Minimal interpolation guarantees the production of expressive and natural speech with rich coarticulation effects. Unlike common rouping of phonemes defined in articulatory phonetics, our method proposes a graded similarity among visemes which relaxes the selection process to larger sets without loss of expressiveness in the produced animation.

Zusammenfassung

Die auf Video basierende automatische Sprechsynthese hat in den vergangenen Jahren beeindruckende Ergebnisse gezeigt. Solche Resultate höchster Qualität werden erhalten durch die Analyse von Artikulationsbewegungen, die mit hohen Bildfrequenzen erfasst werden. Während diese hohen Bildfrequenzen von Videokameras bereits seit einiger Zeit geliefert werden, ist die Erweiterung solcher Verfahren auf 3D erst in den letzten Jahren, mit der Verfügbarkeit dynamischer 3D Scanner, möglich geworden. Diese Erweiterung auf 3D Modelle ist wichtig, nicht nur weil sie die synthetisierte Animation von der Beschränkung auf einen einzigen Visualisierungswinkel befreit, sondern auch weil es die Übertragung der Sprache auf verschiedene Identitäten erleichtert. Die auf Video-Datenbanken durchgeführten Techniken haben sich allerdings auf 3D Modellen als weniger leistungsfähig erwiesen, da die Zunahme an Dimensionalität einen komplexeren Artikulationsprozess impliziert, welcher genauer betrachtet werden muss.

In einem lernbasierten Verfahren werden hier neuartige Animationen synthetisiert durch Aneinanderreihung von in einer Datenbank verfügbaren Bewegungssegmenten. Zu diesem Zweck führt ein Algorithmus eine Auswahl von Segmenten durch, die einen neuen Audiosignal entspricht, und darauf abzielt, eine glatte Kontinuität über die gewählten Segmente zu liefern. Diese Kontinuität ist wesentlich, da sie die notwendige Interpolation reduziert, welche das Verhalten im Endergebnis verzerrt. Die Vergrösserung der Datenbank erhöht zwar die Wahrscheinlichkeit, direkt kontinuierliche Segmente zu finden, verlangt aber auch extrem grossen Speicherplatz.

In dieser Arbeit wird eine Untersuchung des Artikulationsverhaltens durchgeführt, welche die Redundanz in der Datenbank ausnützt. Indem es das Verhalten des Mundes bei den verschiedenen Phonemen (phonetischen Toneinheiten) beobachtet, schlägt unser System ein Ähnlichkeitsmass vor, das definiert, welche Bewegungssegmente für Phoneme verwendet werden können, obwohl sie diesen im Originalkorpus nicht zugeordnet sind. In einem hierarchischen Substitutionsprozess stellt das Auswahlverfahren eine erweiterte Liste gültiger Segmente über einen kleinen Korpus auf. Eine minimale Interpolation garantiert die Erzeugung ausdrucksvollen und natürlichen Sprechens mit differenzierten Koartikulationseffekten. Im Gegensatz zur üblichen Gruppierung der Phoneme, wie sie in der Artikulationsphonetik definiert ist, schlägt unser Verfahren eine graduelle Ähnlichkeit unter den Visemen vor, welche den Auswahlprozess aus grösseren Mengen ermöglicht, ohne einen Verlust an Ausdrucksfähigkeit der erzeugte Animation zu bewirken.

Acknowledgements

I am indebted to my supervisor Prof. Dr. Volker Blanz, Technische Universität Siegen, for having proposed the topic of this Thesis and for his help and advice during the latter years. I am grateful to Prof. Dr. Hans-Peter Seidel, Director of the Computer Graphics Group at the Max-Planck-Institut für Informatik, for the kind reception in his laboratory and for offering me the opportunity to to carry out my PhD dissertation in such a great research environment.

For their professional support, I want to transmit my great appreciations to Sabine Budde, Conny Liegl, and especially Christel Weins who spread my work over the German media.

Special thanks go to Dr. Ivo Ihrke, Prof. Dr. Michael Neff with whom a could share refreshing discussions on countless topics, and who broaden my views on research. Their everlasting trust and support is invaluable to me.

The quality time I spent in- and outside the MPI is equally due to the close friendships that developed in particular Andrei Linţu for taking people as they are, Grzegorz Krawczyk for reminding me that the only way is forward, Lukas Ahrenberg for his enthousiastic moods but also Timo Stich, Kristian Hildebrand and Kaleigh Smith.

Because my time as a PhD student will as well always be associated to the social life I acquired during those years, I am grateful to Christoph Vetter, Monika Haberer, Uli Ludat, Heiko Wanning, Valérie Quilez, Heiko Wanning and David Schuler for the complicity and laugths we shared and will share in the future.

A very special acknowledgment goes to Vlad Tanasescu, because, even with the great distance, we have been through this together and I wish he all the best in the years to come.

Finally, my heart goes to the loving support of my family, Mum, Dad, Sabine et Fabienne and the closeness that we are all able to share.

Contents

Part I Introduction

1 Introduction 3

2 Related Work 9
- 2.1 3D Acquisition Techniques 9
- 2.2 Data Registration 12
- 2.3 Face Models 15
- 2.4 Face and Expression Transfer 18
- 2.5 Speech 20
  - 2.5.1 Simulation of Coarticulation 21
  - 2.5.2 Statistical Methods 23
  - 2.5.3 Segment Concatenation Approaches 24
  - 2.5.4 Viseme Classification 27

Part II Background

3 3D Data Acquisition 35
- 3.1 A Structured-Light Based Scanner 35
- 3.2 Structured-Light Encoding and Phase Shifting 36

4 Data Registration 41
- 4.1 Optical Flow 41

5 Dimension Reduction 45
- 5.1 Linear Methods 46
- 5.2 Singular Value Decomposition (SVD) 46

5.3 Principal Component Analysis (PCA) 48
5.4 PCA in a Multidimensional Morphable Model (MMM) 49
5.5 Multidimensional Scaling (MDS) 50

6 Nonlinear Dimension Reduction 55
6.1 Kernel PCA 57
6.2 Isometric Feature Mapping (Isomap) 57
6.3 Locally Linear Embedding (LLE) 59

Part III Learning Visemes and Articulations

7 Data Acquisition and Preprocessing 65
7.1 3D Acquisition 65
7.2 Hole Filling 69
7.3 Semi-Automatic Teeth and Tongue Removal 71
    7.3.1 Teeth and Tongue Detection 72
    7.3.2 Teeth and Tongue Removal 74

8 Building a Viseme-Space 81
8.1 Tracking the Marker-Points 82
    8.1.1 Rigid Head Alignment and Data Conversion 85
8.2 Optical Flow 85
8.3 Multistage Optical Flow 87
    8.3.1 The Iterative Optical Flow Algorithm 89
8.4 A Morphable Model for Articulation 91
    8.4.1 Weighted PCA (WPCA) 93
    8.4.2 Sound Synchronization 95
    8.4.3 Projecting the Sequences 98
8.5 Expression Vectors 100
8.6 Reanimating Faces 101

9 Face and Articulation Transfer 105
9.1 Head Completion 106
9.2 Importing Identities 108

Part IV Speech Synthesis

10 A Nonlinear Model as a Similarity Measure for Visemes . . . 117
10.1 Locally Linear Embedding . . . 119
10.1.1 Data Simplification . . . 120
10.1.2 The LLE Computation . . . 121
10.2 A Measure for the Similarity of Visemes . . . 124
10.3 Inclusion and Substitution Rules for Visemes . . . 128

11 Finding the Optimal Triphone Concatenation . . . 131
11.1 Concatenation of Triphones . . . 132
11.2 Finding the Optimal Sequence . . . 134

12 Results . . . 137
12.1 Phoneme Similarity . . . 137
12.2 Animation . . . 139

Part V Discussions and Conclusions

13 Discussions . . . 147
13.1 Improving the Data . . . 147
13.2 LLE in a Speech Synthesis Frame Work . . . 148
13.3 Perception Test . . . 149
13.4 Comparison to Other Algorithms . . . 150

14 Future Work . . . 155

15 Conclusions . . . 159

Appendix . . . 161
16.1 The Corpus . . . 161
16.2 Teeth Reconstruction . . . 166
16.3 An Alternative for the Concatenation Cost . . . 167
16.4 Map Distances . . . 169

References . . . 171

Part I

Introduction

1

Introduction

Facial Animation in movie productions has reached an impressive level of realism today. Such great results are usually obtained by directly mapping articulations from a real actor to 3D face models. While this approach generates highly realistic and expressive animations, it is extremely tedious and requires actors to perform complete scenarios inside a motion capture system. In order to simplify and accelerate this process, a number of automated techniques have been proposed. For learning-based approaches, the largest work investment is put into building a knowledge database of accurate and high quality captures in order to maximize the amount of information that can be transferred from the real data to the synthesized animation. The synthesis is then usually produced very rapidly.

Dynamic scanning techniques which today show increasing availability are an excellent choice for capturing face motions. This novel type of acquisition, however, bears specific problems: dynamic approaches generally lose on geometric quality, and automated algorithms can thus easily become error-prone. The goal in this work is to create a robust approach for analyzing such data, and to implement a complete animation system that exploits the advantages of these new dynamic measuring devices, while dealing properly with the lower quality of the provided data.

The quality of the original data can be greatly improved in a preprocessing step. The acquisition system used in this work, provides only partial recovery of the recorded face: holes in the face geometry have to be detected and filled, but also the teeth and the tongue cannot be acquired properly and need to be cut out. A semi-automatic algorithm is implemented that allows to interactively remove teeth and tongue over long data sequences in little time. Manual removal would be too cumbersome, and large corpora would take several days to handle.

For the synthesis of speech, automated approaches for video-based animation have, recently, reached impressive levels of realism. The results are, however, difficult to transfer to other identities and the approaches are usually applied to faces which have little rotation movement. A three-dimensional approach solves these two problems: the viewing direction is free, and 3D face models can be more easily adapted to further identities. Furthermore, speech synthesis procedures for video-based systems have shown to be less suitable in their extensions to 3D; the articulation process has thus to be analyzed more thoroughly.

The process of articulation is a complex one. Defining a unique mouth configuration for each phoneme is not sufficient, as the shape is dependent on many parameters (phonetic context, expression, mood, persons, etc.). Speech animation, in general, is facing two key challenges: (1) produce photo-realistic images in each frame (spatial domain), and (2) produce realistic motion sequences (temporal domain). In order to address both of these challenges, we carry out a learning-based approach and analyze streams of 3D scans of talking faces and facial expressions.

In a *morphable model* framework, the succession of the recorded 3D frame produce high dimensional curves that describe the articulation motions. The frames are labelled with their associated phoneme observed during the recording process. In order to synthesize novel animations, motions are selected according to the targeted phonetic context. In a novel sentence, each phoneme is considered with its two adjacent phonemes (forward and backward), and the phonetic consistency is ensured by picking, from the database, curve segments that are associated with the same phonemes. Two problem arise: first, by selecting segments from the original data, one cannot ensure a smooth continuity in the target animation. Hence, segments must not only consider the phonetic context, but also the segments' suitability for concatenation. Secondly, the number of possible combinations of three consecutive phonemes is too large to be recorded, without even considering that several segments with the same combination would be required to offer better suitability for concatenation. Hence, a database cannot offer all desired segments, and substitution rules have to be defined in order to select segments associated to different phonemes that best match the required one.

For defining substitution rules, the process of articulation has to be analyzed, and the correlation of the mouth deformations with their phonetic context understood. A linear representation of the data does not provide a good representation for the interpretation of the articulation phenomenon. In this work, we propose to interpret the visemes involved in the process of speech in a nonlinear reduced form: using a *locally linear embedding* of the original recorded data, clusters of visemes associated with the different phonemes can be better separated, and we derive a novel similarity measure for visemes.

Hence, our speech animation system relies on two different representations of the data: first, a *principal component analysis* gives a linear low dimensional representation of the data, which enables a pragmatic morphing framework for the synthesis of novel animations. The second representation is a nonlinear reduction, which serves for a behavioral analysis of speech. This analysis derives a general hierarchical procedure for substitutions which designates motion segments which can be used for phonemes which they are not originally associated with. Moreover, the procedure is inherent to the process of articulation and does not rely on the data structure involved in the system; it can, hence, be used for other segment concatenation-based animation systems. The concatenation problem is simultaneously addressed. Our selection procedure offers a list of candidate segments for each query. By then combining the lists, the final animation is generated by selecting the segments that offer the smoothest concatenation, ensuring that little interpolation is required in the synthesis. This ensures that most of the original coarticulation information from the original data is transferred to the final animation.

Aside from the speech synthesis system, this work also proposes an extension that enables the transfer of articulation to novel face identities. Moreover, a set of expressions are included which, added atop the generated animation, increase realism in the final animations.

Outline

The thesis is divided into 5 parts. The present Part I gives the introduction and situates the work in the current state of the art of automatic speech animation. The section on related work covers various topics such as acquisition techniques, face models in relation to their applications, and more thoroughly investigates analyses on articulation.

Part II is devoted to the exposition of the underlying theoretical and technical aspects involved in this work, in order to better separate them from the contributions of this thesis. Chapter 3 describes how dynamic 3D acquisition is obtained by combining structured-light approaches to phase-shifting techniques. Chapter 4 exposes the optical flow algorithm proposed by *Blanz*[14], which integrates a correlation-based approach into a coarse-to-fine framework. The rest of the second part (Chapters 5 and 6) offers a short survey on different dimension reduction paradigms; both linear and nonlinear models are presented and, in particular, Locally Linear Embedding (LLE), which we use for the viseme analysis, is introduced.

In Part III, the knowledge database is constructed. Original articulations are recorded and the provided meshes amended in Chapter 7. After the data is registered, a viseme representation is built in the form of an articulation morphable model (Chapter 8). Chapter 9 sets the model into correspondence with a morphable model of face identities (*Blanz*[14]), which allows both the completion of missing face surface of the dynamic data and the transfer of articulations to novel faces.

In Part IV, the procedure for the synthesis of novel animations is established. The geometrical variations of the face are observed in a nonlinear representation using a LLE (Chapter 10), and a similarity measure for visemes is proposed. This measure indicates which visemes are most suitable for substitutions when the database cannot provide ideal samples. Chapter 11 exposes the procedure used for the construction of novel animations. The database is queried for adequate samples that match a novel audio track, and an optimum combination is sought. Chapter 12 concludes the fourth part by comparing the viseme hierarchy deduced by the similarity measure to popular classifications and by commenting the generated animations.

The last part of the thesis proposes some improvements on the acquisition side and discusses the validity of a nonlinear representation of visemes in a speech modeling system(Chapter 13). Outlines on the directions of future investigations are given in Chapter 14 and the final conclusions are drawn in Chapter 15.

Contributions

In the different fields dealt with in the present work, the main contributions are:

- the dynamic 3D acquisition of speech and the registration of a large corpus of articulations;
- the reliable and efficient amendment of noisy and incomplete data, involving hole filling, smoothing, and semi-automatic mouth detection for removing inconsistent teeth and tongue information. An approach to face completion is also proposed;
- an extension to optical flow techniques that grasps large deformations over multiple long sequences by combining an absolute matching with a relative one;
- a weighted principal component analysis (WPCA) for the generation of a multidimensional morphable model (MMM) for articulations;
- a framework for finding correspondence between multiple MMMs allowing face completion and articulation transfers to novel faces;

- an intuitive LLE representation of the visemes involved in the speech process for large corpus sets;
- an similarity measure for visemes which is inherent to speech thus independent of the structural model of the system;
- a substitution rule for optimum viseme selection, and a substitution graph inherent to speech, thus, reusable for novel segment-based speech animation systems;
- an optimization paradigm for animation synthesis which ensures optimum selection of articulation motion segments providing realistic animations.

2

Related Work

Building a complete speech animation system requires to undergo many different steps that touch different fields such as computer graphics, computer vision or image processing. While these fields relate to how the data or the information held inside it is processed, understanding speech articulation is yet another different field on its own respect. In a learning-based approach, the system presented in this thesis first integrates data acquisition (2.1) and data registration (2.2). Through the registration, a face model can be built (2.3) on which different speech animation techniques (2.5) can be implemented producing different, yet, realistic animations

In this chapter, published work in these different fields is discussed. Due to their different natures, these aspects are investigated separately as each can relate to different domains of research. This chapter is thus separated into four main sections that discuss the aforementioned fields.

Section 2.4 of this chapter discusses work on face and expression transfer. While the goal of this field is different from speech animation, the approach and the necessary setup are in many aspects similar and are thus relevant to the present work.

2.1 3D Acquisition Techniques

While there exist many 3D modeling software systems for creating three-dimensional structures or recreate real object, 3D-scanning remains important because it captures the true nature of the object. The precision offered by acquisition systems reveals details that would be tedious to model or that would not be accurately created by an artist. Also, phenomena that cannot be a priori simulated are required to be recorded first in order to be analyzed.

Basically, the important aspect of data acquisition is to acquire the knowledge of a structure (static) or a behavior (dynamic) that one wants to understand.

Data acquisition is a fastidious and computationally expensive task but is probably the most sensitive part in the setup of a system. The reward one gets from the cost of acquisition, is that this task is performed a single time and once the knowledge is acquired, the observed phenomenon can be recreated or simulated, avoiding further acquisitions. Therefore, when selecting a technology for data acquisition, not only the structural quality has to be observed, but more importantly, the retrieved data has to reflect best the actual information that is intended to be observed.

Many different techniques exist to record three-dimensional data, all are based on a two-dimensional acquisition interface, mostly in the sense of a camera. Some approaches use a single 2D acquisition interface and map deformations to 3D models (*Chuang & Bregler*[34] or *Pighin et al.*[99]). The depth that cannot be recorded is difficult to reproduce on 3D models and has to be simulated. While this approach is suitable for certain applications, it fails to capture all the information that can be acquired. Thus, these approaches somewhat compromise realism. In a static model context, *Pighin et al.*[101], *Goto et al.*[52] or *Georghiades*[50] use multiple photographs to recover true three-dimensional face shapes; *Guenter et al.*[54] in a dynamic context achieve 3D reconstruction from 6 synchronized video cameras by tracking a large set of marker points.

In the 3-dimensional domain and for data-driven approaches, while acquisition can still be done in two dimensions and retargetted to 3D models, recent work shows that the tendency is to move towards 3D acquisition setups. The way the acquisition is performed is heavily dependent on the targeted model. Muscular structured models (with early works from *Waters*[135]) will require tracking many marker points so that the behavior of the distinct muscles can be precisely analyzed (*Sifakis et al.*[115, 116]), while geometry deformation analysis deals with precise data registration. In any case, high quality of the acquisition is required to allow good analyzes. Volumetric approaches of faces can also be performed using *magnetic resonance imaging* (MRI) as used by *Sifakis et al.*[115]. These scans bring good muscular information but are strictly limited to static acquisition.

For dynamic approaches with a high temporal resolution, several acquisition techniques exist such as a multiple cameras setups with stereo algorithms or additional techniques that exploit the temporal coherence (*Zhang et al.*[141]). To reduce the number of cameras, *Pighin et al.*[100] or *Sifakis et al.*[115] record a single view of an animated face, track a set of predefined feature-points on the subject and fit a previously acquired generic face model to the video by applying deformations on it (Fig. 2.2). *Brand*[21] and

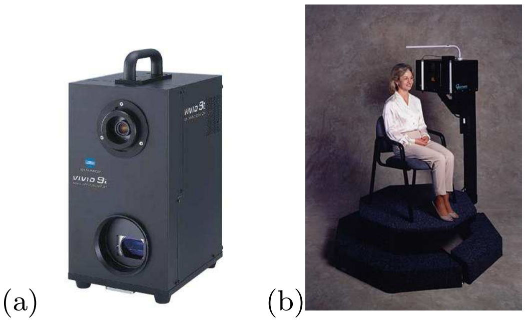


Fig. 2.1. Static 3D scanners The two 3D scanners perform static acquisitions. The Minolta scanner (a) is placed in front of the object; the laser browses the object while a 2D camera records the contact points. The Cyberware scanner (b) rotates around the face of the recorded subject while the laser browses the face vertically at different rotation angles. The Cyberware scanner generates a complete scan around the face whereas the Minolta scanner only records the surface visible from its viewpoint. For both setups, the subject must not move during the acquisition process.

Chai et al.[29] acquire the knowledge of the mouth deformation from a two-dimensional source.

Active approaches to surface acquisition illuminate the recorded object with a light point or a light code that is recorded and triangulated. Most common scanners are based on retrieving surface locations by triangulation (see Section 3.1). Static laser-based acquisition has found applications like in the works from *Blanz et al.*[18, 16] (Fig. 2.1), where face geometries are captured. The same triangulation principle has since been extended to *structured-light* systems (*Wolf*[137]) that allow dynamic acquisition, see *Song Zhang et al.*[143] or *Li Zhang et al.*[141]. While dynamic techniques are usually static techniques at high frame rate, *Zhang et al.*[141] additionally exploit temporal coherence over the recorded mesh to improve the final quality. Figure 2.2 illustrates the basic setup behind triangulation on which many 3D-acquisition approaches are based. The setup for a laser scanner consists of a laser beam and a camera (Fig. 2.2 left). After calibration, the respective distance between the the source of the laser and the eye of the camera is known. The camera observes the contact point of the emitted beam onto the object. If the emission angle is known and the highlighted pixel on the camera's viewport is detected, the exact position of the impact can be computed in three-dimensional space by triangulation (see Section 3.1 for more detail on the triangulation computation).

While this technique is very precise, the acquisition time is long as the laser has to cover the whole surface and the camera has to capture each recored point with a single frame. This approach is thus only suited for static recordings. A much faster method is derived by using structured-light (Fig. 2.2

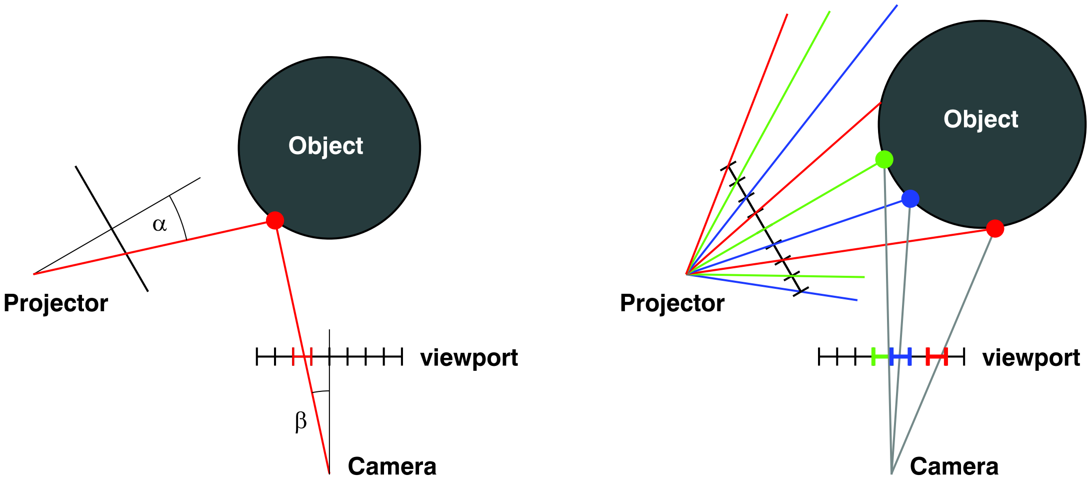


Fig. 2.2. Structured-light triangulation The triangulation technique determines the 3D location of the impact of a laser beam by knowing both the emitted angle and the observation angle (left). Structured-light techniques propose to detect more impacts simultaneously by encoding the emission (right). (Section 3.1 discusses such techniques in more detail.)

right). Instead of using a laser beam, a projector is used that illuminates the whole object. The projector works as a dual component to the camera and thus projects a grid of pixels on the object. For the camera (or the system) to be able to attribute the corresponding projected pixels to the correct impact on the recorded surface, an encoding has to be performed on the projected grid. On that aspects, techniques differ, but usually, a sequence of a few pattern permits an accurate decoding (Section 3.2 discusses this in more detail). In that way, within a few frames, the whole object can be recorded. The acquisition rate lies in the order of several 10s of frames per second depending on the system.

2.2 Data Registration

In the previous section, we described common scanning technologies but omitted to discuss the output format of the recorded data. This format is directly dependent on the technology used for the acquisition and does not reflect the nature of the recorded object. Generally, for scanners with fixed positions like the Minolta laser scanner from Figure 2.1 or dynamic scanners, this format is deduced from the two-dimensional interface. For the Cyberware scanner from Figure 2.1, for instance, the resolution is determined by the number of rotation steps performed by the scanner around the face multiplied by the resolution of the vertical scans performed by the laser. In both cases, the data comes as depth maps, however, the first type of scanners will produce coordinates in Euclidean space while the second generate polar coordinates.

Standard procedures, particularly for static recording, perform remeshing in order to compress, smooth or even to represent the data with a structure that better follows the characteristics of the obtained topology [49]. Other procedures aim at fitting the acquired data to a model [71] or to put them into correspondence [18]. The correspondence process, or registration, is there to match the data to a global mesh in order to find a one to one vertex mapping from the new data to the model.

With dynamic acquisition, the data output consists of a collection of independent meshes taken at several time intervals. This means, that the generated vertices are not attached to the surface over time but rather fluctuate with the underlying deformation in the same way a ping-pong ball behaves on agitated water. *Edge & Hilton*[44] use their acquisition data as a *3D Video*. To recreate animations, 3D video segments are stitched together smoothly, thus circumventing the registration problem. To be able to study actual deformations, the different 3D-frames however have to be put into correspondence so that every location on the recorded surface is tracked and its exact displacement of each of them is known.

For 2D videos, the approach is to align the faces over time and define a segmentation over the face. In such a manner, different parts of the face can be synthesized separately and assembled in the new video. Such techniques have been proposed in *Video Rewrite* by *Bregler et al.*[22] or *Ezzat et al.*[47] who coupled it with optical flow techniques.

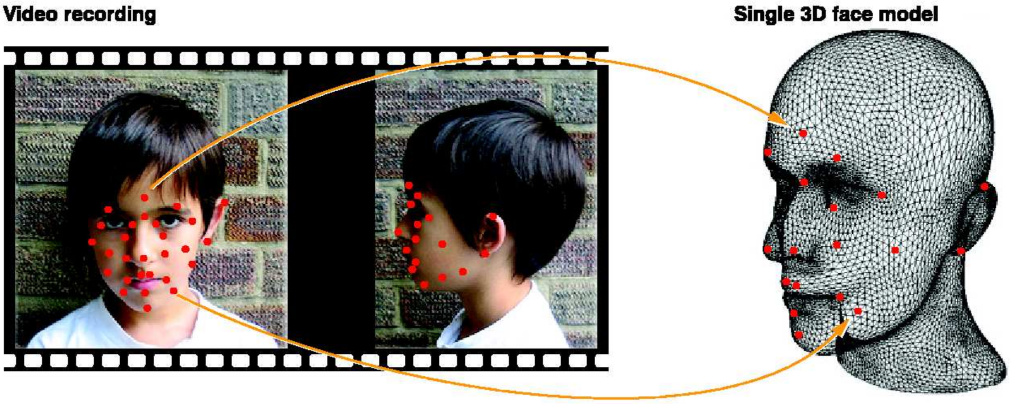


Fig. 2.3. 2D deformations mapped to a 3D model Here, 3D deformations are acquired from multiple camera setups. Stereo algorithms perform correct associations of marker points over the different video streams and deduce their location in the Euclidean space. The displacements of the marker points are then mapped to a 3D face model.

Before high quality 3D scanners were available, a simple solution for tracking face deformation was to track marker points on two or more simultaneous video footage and reapply their displacement to a 3D face model (see Fig. 2.3). To record deformation with a static 3D scanner in combination with 2D videos, the subject is recorded with marker points on specific locations on his face which are detected in the 3D reconstructions. With the use of two or more standard calibrated video cameras, the subject is then recorded performing expressions or speech and the displacement of the maker points are tracked on the video. These deformation can be mapped to the three-dimensional model and the three-dimensional deformation can be reconstructed. This approach, while it performs reliably, lacks in precision as only the maker points are truthfully mapped and a morphing algorithm has to be performed on the rest of the mesh. The advantage with this approach, is that the deformations are obtained directly and the registration process is circumvented: this approach directly records deformation instead of surfaces (*Guenter et al.*[54]).

In a similar manner, and without the need of a 3D scanner, this approach can be used to record a 3D face based on a single 3D face mesh. If the marker points on the recorded face correspond to the ones on the 3D model, measurements on the footage or pictures can be used to deform the 3D model to best match the recorded face. By extension, 3D models can also be deformed to match a set of photographs. This approach was performed by *Pighin et al.*[100, 101] or by *Parke*[97] who directly painted a mesh structure on a face which allowed direct registration.

When it comes to structured-light dynamic 3D scanners, data registration is the central problem in retrieving information after the acquisition process. Some approaches use optical flow techniques (see Section 4.1) which have proven to be reliable for small deformations in an absolute *one-to-all* frame registration. Large deformations can be tracked in a relative *one-to-next* registration but the accuracy diverges over long sequences. There exist several known optical flow techniques which we discuss in more detail in Section 4.1. A new registration approach addresses the divergence problem in Section 8.3.

Optical flow based registration has been performed by *Blanz et al.*[18]. They address the problem that optical flow may produce unreliable results in smooth regions where the similarity of adjacent pixels is large. They couple a *relaxation* technique that produces smooth results in a *coarse-to-fine* approach (see also *Vlasic et al.*[129]). *Zhang et al.*[141] use the temporal coherence of the data to retrieve missing information in a frame from subsequent frames. To address the problem of large deformation which optical flow techniques fail to track, *Blanz et al.* propose to group similar visemes into batches within which optical flow computes the correspondence. The different batches are then registered in a bootstrapping method involving user interaction.

An alternative to optical flow is to use *Radial Basis Functions* (RBF). The RBFs align and deforms the recorded face to match a generic face model. Work from *Cao et al.*[27] or *Kim & Ko*[74] learn deformations by tracking marker points in 3D which they reapply to their 3D face model. Because only the maker points can be tracked accurately, the rest of the surface is deformed by warping vertices accordingly to the surrounding markers (see Section 8.1). *Joshi et al.*[69] proposed a *Blend Shape* technique that addresses this warping problem and was used in *Mood Swings* (*Wang et al.*[133]) or by *Chuang & Bregler*[34]. *Kalberer et al.*[73] avoid the warping by matching the whole 3D face shape instead of maker points. Using RBFs, *Noh & Neumann* are able to transfer expressions by computing mapping functions between different face models. *Zhang et al.*[145] adapt a physically-based model to 3D acquired scans in a hierarchical refinement of surface subdivisions, fitting first globally and then locally.

2.3 Face Models

Face animation covers many research fields in contemporary computer graphics. All these fields have to deal with high quality and accuracy. The reason is that the structure of the face or its behavior is extremely integrated in the sensitivity of people as it is the body part with which humans interact. Therefore, any structure or behavior that deviates from their plausible nature are quickly disturbing to the observer. The way faces are modeled in the different fields of face animation depends on the applications the simulation or the reconstruction are required for. This section concentrates on the behavioral aspect of this reconstruction and discusses the most common models that appear in recent literature.

Parametric Models

The modeling of faces is roughly divided in three general groups: the first group consists of deformable meshes by a given number of parameters. Poses and expressions are then described by these parameters. In 1974, *Parke & al.*[96] developed the first face animation system on a Silicon Graphics machine. Their model, which was originally controlled with less than 10 parameters was further developed and refined to about 60 parameters by *Cohen & Massaro*[35] with a parametrically controlled polygon topology synthesis technique. Their model was used to simulate coarticulation based on dominance functions to determine the mutual influence among visemes. *Cohen & Massaro* additionally implemented a parametric tongue as did *Parent et al.*[95] and *King & Parent*[75] in a similar framework.

In the last few years, the *MPEG-4 standard for facial animation*[94] got popular. It defines a *facial animation parameter* (FAP) set that closely relates to muscular models. The FAP offers a set of parameters that ensures realistic representations of talking heads for applications such as facial expressions, emotions, and speech animation. *Beskow & Nordenberg*[12] and *Eisert et al.*[45] generated a MPEG-4 compatible model for learning-based frameworks.

Muscular Models

A second group of face models, which is related to parametric models, decomposes the face into a muscular structure and analyzes the behavior by retrieving the muscle activations in order to reproduce them. The animation parameters here become the muscle activation triggers. Early work from *Waters*[135] make an analysis of the muscles involved in articulation and their effect on the skin.

Physically-based methods have to defined interaction models between the muscles and the facial tissues in order the provide quality skin animation by only controlling the muscles. *Terzopoulos & Waters*[124] describe a tetrahedral mesh model that deforms under physical constrains. Applied to CG generated face geometries they are able to generate expressive facial emotions.

In a different approach *Head Shop*, a physically-based face model developed by *Kähler et al.*[70], adapts a muscular model to static face scans. Feature points are defined on the scan and the muscular model adapts to the new shape through refinement and warping methods. *Albrecht et al.*[3] employed the Head Shop model to generate speech-synchronized facial animations.

Sifakis et al.[115] extend the muscular model by combining it with a *quasistatic finite element mesh* to produce realistic reactions of the skin under the dependency of the muscles. A great advantage of their model is that the facial behavior can easily be set in interaction with external physical elements. In their following speech simulation paper [116] they are able to deform animated speech by interacting with lollipops of different shapes.

Where *Choe & Ko*[33] circumvent skin simulation by involving *hand-generated muscle actuation basis*, *Zhang & Sung*[144] further extend the mass-spring system by using non-linear springs in order to simulate the viscoelasticity of the skin.

Finally, *Tang et al.*[113] propose a reduced muscle model. The muscles are constructed with B-splines (NURBS) each featuring several control points. Their model learns the activation parameters by observing lip movements from video to which they match the feature-points.

Learning-based Models

A radically different approach to face models consists by representing only the apparent part of the face, namely its surface or the skin. This representation results from scan-based acquisition techniques which only capture the topological aspect of the face. The advantage of learning-based face models is that the faces can be either represented in their three-dimensional form or in a two-dimensional cylindrical projection where the geometry is interpreted as depth maps. This two-dimensional representation permits analyses using image processing techniques. Hence, approaches used for video animation can be directly adapted to three-dimensional models in particular for data registration.

In their speech animation framework, *Edge & Hilton*[44] record 3D video with a face capture rig working with IR cameras for shape acquisition and standard cameras for the texture. They do not register the data as their goal is to synthesize animation with the original subject. Their approach uses *video-textures*[111] (based on a Hidden Markov Model) which consists in concatenating video segments and deformation tracking is unnecessary.

For learning articulation and expression deformations, *Kalberer et al.*[71, 72] tracked marker points with a 3D scanner and mapped the observed displacement to a parametric face. In this case, no shape in acquired but only motions.

A more versatile and now popular shape model is based on a *Multidimensional Morphable Model* (MMM) (*Jones & Poggio*[68]) introduced by *Vetter & Poggio*[102, 128] as *Linear Object Classes*. MMM is a statistical linear compression process that generates a low-dimensional representation of high-dimensional data which Section 5.4 presents in more details. *Blanz*[18] presented a MMM for 3D faces in which 200 faces where registered. The low-dimensional representations allows the controls of shape and texture variations to produce morphings between different heads. There exists a wide range of applications of such models. Facial animation can be included in the model while keeping the facial identify deformation separated from speech or expression deformations. Such techniques have been used by *Vlasic et al.* and *Blanz et al.*[16] to transfer expression from one person to another. In a similar manner, *Wampler et al.*[132] learn shapes and expression from different individuals. While these two latter works acquired the data simultaneously, *Blanz et al.*[16] are able to learn viseme in a second pass by matching the required deformations on the MMM. Further applications to MMM are face identification from photographs by fitting the MMM to two-dimensional images (*Romdhani et al.*[104] and *Huang et al.*[63]) or in a similar scenario, faces can be exchanged in photographs once the MMM is fitted [15].

DeCarlo et al.[37] construct morphologically plausible faces based on *face anthropometry* (the science dedicated to the measurement of human face). These measurements are taken from a set of measurements, which become the constraints for an surface optimization reconstruction problem solved using *variational modeling.* This model offers the advantage that, for instance, if the anthropometric relation between the skull and the face geometry is known, novel face shapes can be generated with automatically adapted physical structures. While such an approach is powerful for modeling faces, its principles are ill-suited for animation purposes.

2.4 Face and Expression Transfer

A field that is closely related to speech animation is *expression animation* or, how it is often associated with, *expression transfer.* Expressions synthesis also requires great accuracy in order to recreate genuine feelings or sensations. The particularity, is that the different regions of the face involved in expressions deform unsynchronized and expressions can be combined but must be done so in non-linear manners (see *Deng and Neumann*[43]). While the aspects observed for expressions against speech synthesis are fundamentally different[1], their general setup for acquisition or face modeling is closely related.

The notion of transfer applies more to expression than to speech. While the ultimate system should combine both aspects, expressions are controlled in a different manner (see Section 8.5), they are rather meant to be applied atop speech. In expressions, the behavior is learned from real persons and is applied directly onto a face model. For instance, *Buck et al.*[23] transfer expression changes from a video to a hand-drawn face; while these expressions might also involve visemes (in the sense of speech) these are not treated as such, but rather just like further expressions. For that reason, this aspect is referred to as *expression transfer* (see also work from *Pyun et al.*[103] for real time applications). Automatic speech generation systems on the other hand regenerate articulations and do not aim at a direct reproduction of recorded articulations but as a synthesis process.

A direct approach to expression transfer is to record expression visemes with a 3D dynamic scanner and to reproduce them directly on to the recorded face model. *Blanz et al.*[16] use such an approach. They project a 3D face model onto a photograph (the face shape is already learned from that photograph) and by morphing through the visemes, they are able to reanimate

---

[1] Investigations considering expressions syntheses try to reproduce recorded expressions on novel faces, whereas speech investigations tend to learn a behavioral process in order to recreate novel animations.

images. In the 3D domain, *Noh & Neumann*[90] propose a registration system to make several face models compatible through an automated heuristic correspondence algorithm. With this process, they are able to *clone* geometry deformations available for a source face to any target face. Note, similar work was done by *Fratarcangeli et al.*[48] in which they made the model compatible to the MPEG-4 standard (see 2.3).

In a dynamic framework, expression movements are recorded with a 3D dynamic scanner and reproduced onto the recorded face model. In *Spacetime faces*, *Zhang et al.*[141] propose a temporal coherent dynamic 3D scanner and learn expression parameters in order to reproduce them. They are able to reproduce expressions with great realism as they observe them over time. This is, expressions do not correspond to a typical face configuration but rather an evolution of deformations over time, e.g. the mouth reaches "happiness" while the eyes are only half way to their targeted shape. For instance, in his early work, *Parke*[97] recorded several expressions as single visemes and was able to morph between them using a cosine interpolation. While this process generates smooth transitions, the natural unsynchronized progression of the face components that generate the face movement is lacking, an aspect that can be understood as a counterpart to coarticulation to expressions.

Expressions and speech can be seen as two statistically independent sources observed simultaneously. Generally, to be separated, the mixture of such signals have to be observed by at least as many captors as there are signals under different conditions (the Cocktail Party example). In facial animation, the approach is usually to record repeated utterances with different expressions. Once these utterances are aligned over time it becomes possible to separate the expression component from the speech component. Mathematical tools for such purposes are for instance *Independent Component Analysis* (ICA) used by *Kalberer et al.*[71]. In their approach, the use a dynamic 3D scanner to record a subject and match the acquired face deformations to a generic face model using *Radial Basis Functions* (RBF). The components are then separated from the model. ICA was also used by *Cao et al.*[26], their systems offers an interface to intuitively edit expressions and speech separately.

Another tool to operate separation is to use a bilinear model (*Tenenbaum & Freeman*[123]). *Chuang & Bregler*[34] use this approach to factorize and control speech and emotion in image-based motion capture data to more effectively retarget facial motions to another 3D face model. They weight subsets of morph target that belong to different facial expressions to convey more emotion than single emotion vectors would produce. The idea is to learn the characteristics of expressions in relation to speech from a set of training data and to retarget these expression models to new animations.

For separating expressions from speech, *Wampler et al.*[132] also use a multilinear model but use tensors in order to learn face variations instead of shapes over different individuals. While their speech synthesis bases on an anime-graph (see Section 2.5.3), learned expressions are transferred to novel animations according to the generated articulations.

Vlasic et al.[129] use a similar multilinear model to separate identities, expressions and visemes. Their statistical model is then the automatically adapted to novel video sequences and enables the control of a 3D face model from from any individual though a two-dimensional interface using optical flow techniques. Earlier work on video controlled face animation was performed by *Chai et al.*[29] where the tracking was performed through feature point detection on the video. Also *Zhang et al.*[142] transfer expressions from images to 3D models but require user input to assign feature points.

In order to control expressions on a synthesized animation, *Deng and Neumann*[43] propose to an isomap-based (see Section 6.2) user interface. While speech and expressions are learned from real data, expressions are described along in a low-dimensional manifold on which all points correspond to valid face expressions. Dynamic programming is used for building a motion-sequence from the database that reflects the user's selection along the manifold and that has a predefined emotion and smooth transitions.

Wang et al.[134] perform a slightly different separation. Their goal is to obtain a generic expression movement from the several individuals and separate it from the style of specific subject. A bilinear model is used but in this case it is performed on a dimensionally reduced representation of the motion curves. For the dimension reduction, they use *Locally Linear Embedding* (LLE) (see Section 6.3) to better visualize the one-dimensional manifold along which expressions deform.

Finally, *Face Poser*[79], an expression modeling system developed by *Lau et al.* offers an interactive way for designers to generate realistic expressions. The system is constructed on a statistical model based on a real data training set. The user is first offered a neutral face shape on which he can sketch deformations; the system deform the specified region and the rest of the face adapts accordingly.

2.5 Speech

Automatic speech animation is the central aspect of this thesis. Natural articulation is a really complex behavior which has been analyzed for over a century. This behavior has to be understood in order to be expressed into algorithms and reconstructed.

Speech as a phenomena, is observed from many sides. *Phonetics* focus on the produced sounds and classify *phonemes* according to their spectral properties. The first separation divides phonemes into vocals[2] and consonants. While this separation is related to the classification in the alphabet (vowels and consonants), vocals, as opposed to alphabetical vowels, also include for instance phonemes like $/W/$ (co**w**). Moreover, letters can have several way of pronunciation depending on the lexical context they are in (k**i**d, h**i**gh) or when in combination with another letter (**s**wing, ca**s**h). The phonetic separation thus differs from the one from the alphabet. As a matter of fact, different letter can even be pronounced the same way (rou**gh**, **f**inal). Another aspect in speech is the way sounds are produced. In a similar manner, the physical pronunciation has a direct impact on the produced sound, however, similar movements can produce many distinct phonemes and similar phonemes can be pronounced with different mouth configurations. In reality, many parts of the body are involved in the pronunciation process. The field of *articulatory phonetics*[13] focus on this aspect and reveal that a specific phoneme always requires specific body parts to be active. In some cases, the mouth for instance is not a mandatory component in the process and its shape has thus little effect on the output sound. Vocals for instance are heavily dependent on the shape of the mouth as it modulates the air flow coming straight out through the larynx. Articulatory phonetics divide phonemes according to the components involved in the pronunciation, the unitary articulation elements are referred as *places of articulation* [78]. The perceived appearance of the phonemes can however only be noticed by the shape of the mouth and the movement of the tongue. The configuration of these two elements are known as *visemes* and are understood in some extent as the visual counterparts of phonemes. Hence, while articulatory phonetics give a complex but complete classification of phonemes, visemes can be divided in a less restrictive classification which correspond to a higher level classification of the places of articulations.

2.5.1 Simulation of Coarticulation

A primary approach to generate speech is to select an average viseme for each phoneme and generate speech animation by linearly interpolate between them in synchronization with an audio file. In reality, the visemes that are produced during articulation are strongly dependent in the phonemes that preceded or that are following and particularly if the uttered phoneme does not require the involvement of the mouth. This effect on the mouth of the phonetic context is known as *coarticulation*. Coarticulation actually encapsulates two phenomena: the influence of phonetic context but also the articulation movement between

---

[2] In phonetics, 'vocals' are also called 'vowels'.

two phonemes. This effect was studied on different aspects (Section 2.5.1) and has been modelled statistically (Section 2.5.2).

The two articulation aspects encapsulated in coarticulation are considered separately in order to generate novel animations. First, to generate realistic transition motions between two uttered phonemes motion segments are retrieved from recorded animation and concatenated in order to produce smooth animations (Section 2.5.3). The segmentation approach ensure that the natural transition information is reproduced in the final animation. Secondly, the study of the viseme dependency on its phoneme context can be analyzed in order to classify visemes 2.5.4. This classification allows valid substitutions rules that widen the segments selection in a concatenation framework.

An early analysis on coarticulation was performed by *Cohen & Massaro*[35] where they applied Löfqvist's gestual model [81] as a general framework for visual speech synthesis. Their model goes beyond keyframe interpolation in order to model the interaction between subsequent phonemes. Dominance functions that control how the influence of each phoneme slowly increases before the phoneme is actually heard, and how it decreases slowly afterwards. Due to the overlap of dominance functions, phonemes interact and produce a smooth motion sequences. Recently, this approach was used in muscular-based approaches by *Albrecht et al.*[3] and *Scott & Richard*[75] or in an MPEG-4 framework by *Beskow & Nordenberg*[12] or *Eisert et al.*[45].

A behavioral study of articulation focuses on the visemes interactions and their relative role importance in speech. Visemes are taken from real data in their best matching phonetic context. The concatenation of two successive sequences is performed by following the acquired motion curves and blending accordingly to the respective dominance functions. The dominance function approach has proven to be reliable and so far the best model for simulating coarticulations (*Parent et al.*[95] *Beskow & Nordenberg*[12]) but remains computationally expensive and complex due to the many articulation parameters involved. In their discussion on *Issues with Lip Sync Animation*, *Parent et al.* propose to circumvent the coarticulation computation by using motion segments from real data animation which already contain the coarticulation information. Ideally, the highest quality is obtained by finding segments spanning long sequences of phonemes to ensure realism. This concatenation approach had been already used by *Bregler et al.*[22] where they introduced triphones concatenation as a framework to generate realistic 2D animations. This approach, which we use in this thesis, is discussed in Section 2.5.3.

Pelachaud et al.[98] perform a different analysis of coarticulation. They consider that vowels have a more important impact on articulation than consonants. Their modeling follows a *forward-backward* rule: visemes are defined at phoneme occurring time positions. When the viseme is followed by a vowel, the forward rule applies and adjusts the transition position towards that vowel. If the consonant is preceded by a vowel, a similar but inverse backward rule is applied. This rules are however adapted according to the phonemes involved in the transitions and their mutual relations. These relations are defined by a phoneme grouping determined by the visemes deformability and their context dependency.

In a similar but simplified model *Wang et al.*[133] suggests to produce coarticulation by selecting target phonemes, the ones that have stronger impact on articulation, and adapt preceding phonemes that them.

2.5.2 Statistical Methods

There exist a large diversity of statistical methods for simulating speech animation. This section mentions the four of the most successful methods which use Markov models, machine-learning techniques.

Probabilistic Methods

Several methods model speech as a Hidden Markov Model (HMM). This model is well suited to perception-based phenomena where the actual states in a Markov model have to be probabilistically determined through observations. The parallel with speech is directly obtained: uttered phonemes are the observed phenomena and the visemes are the states to be found. With a sequence of observations, the Viterbi algorithm finds the most probable matching sequences of states.

In *Voice Puppetry*[21] by *Brand*, the observed phenomena is speech and expressions from an audio track. The presented system finds from a database of pose parameters the best pose sequence to match the audio signal and generates a new corresponding animation. Here, the approach is applied to video face model (2D) but can easily be extended to three-dimensional models. The parametric model is learned from real data.

Ma et al.[83] segment a large corpus of markers into syllables and concatenate them for new utterances. They take novel phoneme sequences from audio tracks as observation and find the best matching syllable concatenation from their database. Following the Markov model, this graph-based method ensures to keep the coarticulation information that is contained in the stored motion segments. Differently, *Brand* ensures coarticulation by generating transition movements following the high-dimensional curved manifold described by the acquired data.

Machine-Learning

In video speech animation, great results have been achieved with high realism. In particular, *Ezzat et al.*[47] use a *regularization* based on machine-learning techniques. In a learning-based approach visemes are represented by a Gaussian distribution in a high-dimensional vector space based on principal component analysis (PCA). The distribution is centered around the average appearance of the viseme, and varies according to the variation found in a database of samples. For finding the motion trajectory for a new utterance, a regression algorithm computes a smooth curve that passes as close as possible to the centers of the visemes, relative to the variance of the viseme cluster. In a refinement step of the training of the system, the viseme clusters are shifted and deformed in order to obtain trajectories that are as close as possible to those in a training set. In this model, coarticulation is due to the smoothness of the curve and a statistical representation of the variance of each viseme. On continuing work, *Chang & Ezzat*[30], are able to keep generic speech information and relearn articulation from another person using a mush smaller training corpus.

These approaches were recently extended to 3D in the work of *Kim and Ko*[74]. In that paper, *Kim and Ko* argue that the regularization approach can generate movements that are too mechanical when used with three-dimensional face models. They address that problem by combining a data-driven approach. While this allows a reduction in the size of the database, the authors claim that where the machine learning takes over the data model, the results look less natural.

2.5.3 Segment Concatenation Approaches

As discussed above, producing coarticulation based on dominance functions generates realistic and smooth results; this approach however, is complex and computationally costly. Moreover, it suits best parametric face models and is incompatible with statistically-based face models. While statistical motion approaches try to reinvent coarticulation movements another popular way to attack this problem is to consider that the coarticulation information is directly available in the recorded corpus. The goal is to transport as much as possible of this information from the acquired data to the output animation.

Segments concatenation approaches truncate the original information in motion segments and associate them with the phonemes they correspond to. Under this consideration, several segmentation frameworks have been proposed which divide the original in different motion units: syllables [95, 35], visyllables, animes or triphones.

Visyllables

In the same way visemes are related to phonemes, *visyllables* relate to *syllables*. This segmentation method is well justified as the unit measure considers sequences of phonemes and thus holds coarticulation information. Syllables are islands of phoneme sequences isolated by pauses or silences; this is particularly noticeable in slow speech. On pauses, the mouth tends to return to its neutral position and therefore, the extremities of a visyllable segment hold less coarticulation information.

Visyllables were proposed by *Kshirsagar & Magnenat-Thalmann*[76] in 2003 as visual elements of speech, instead of visemes or triphones (see below). The authors present a system that learns visyllables from marker point data, transfers them to *Facial Movement Parameters* (FMP), and uses these for animation. In speech synthesis, visyllables are stitched together, and boundary mismatches are corrected.

The syllabic approach was used by *Ma et al.*[83] (they don't use the term *visyllable* though) where they segment a large corpus of marker data into syllables and concatenate them in a new graph-based approach.

Animes

Cao et al.[27] proposed an Anime-Graph, which combines visual and speech information in a single data structure, and use a greedy search algorithm for generating new utterance in a real-time capable framework. Unlike the static visemes, animes capture the entire motion during a phoneme and are stored in a database and labeled with their corresponding phoneme. The motion segments are first normalized over time and then compared. Similar motion are clustered and a single prototype for each cluster is kept while it is associated with all the different phoneme labels it includes. This procedure not only reduces the size of the database but offers new concatenation possibilities. The original anime succession information is also preserved and after clustering, each anime is connected to several further animes that ensure smooth transitions (see Fig. 2.4). Finally, the clustering generates an *Anime Graph* that captures the context dependencies of individual instances of phonemes. By selecting animes with an appropriate context from the graph, the algorithm synthesizes animation with coarticulation effect.

The viseme substitution problem is addressed in one direction: an *anime* can be associated to several phonemes. That is, in a new animation synthesis phonemes are matched only to motion segments with which they were associated in the original corpus. In this thesis, this problem is addressed in both ways: for a given phoneme valid substitutions to motion segments are sought that were not attributed to that phoneme in the original recorded corpus.

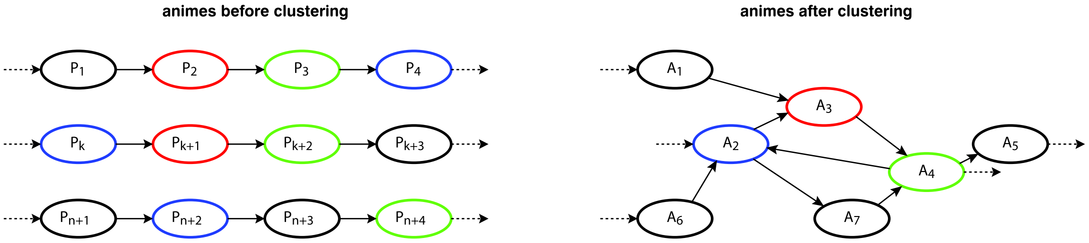


Fig. 2.4. Construction of an *Anime-graph* An *Anime* is a motion segment labeled with its associated phoneme. In a first step, animes are connected according to the original recorded sequences (left). Animes are then compared and clustered by similarity (right); a single motion prototype is kept for each cluster (reducing database size) but the original connections remain, thereby transforming the original sequence in an *anime graph.* (illustration after *Cao et al.*[27])

This is possible because the general "behavior" of the phoneme is similar to the one of the selected curve. In their following work, *Cao & al.*[28] extended their speech model by including expressions that they separated from speech by an Independent Component Analysis.

In recent work, *Kim & Ko*[74] coupled anime-graphs with the regularization techniques from *Ezzat & al.*[47]. Where the anime-graph only guarantees weaker transitions, the coarticulation is obtained through machine-learning techniques. The combination of both approaches ensures realistic animations even with a small data corpus. *Wampler et al.*[132] also use an Anime-based graph algorithm, but rely on a bilinear model for separating expression and speech.

Triphones

In 1997, *Bregler et al.* introduced the usage of *triphones* for speech synthesis. Triphones are sequences of three consecutive phonemes. The term is interchangeably used is the phonetic or in the visual domain. In their paper *Video Rewrite*[22], they present a triphone concatenation framework for the synthesis of two-dimensional (or video) speech animations. Triphones present the great advantage that they hold all the coarticulation information around the their central phoneme that is the transition motion from and to the preceding and the following phoneme.

In a data-driven approach, they address the problem of database size by grouping specific visemes together to simplify the lookup process and higher the matching of segments to a novel sentence. This grouping is based on *Owens & Blazek*[91] viseme classification which the following section 2.5.4 discusses in further details.

For generating novel animations, triphones are stitched together in an overlapping fashion (see Fig. 2.5). To match a audio file, triphones are selected according to two criteria: one is the similarity of the triphone in the database to the triphone required for the animation. If any of the required triphones are not found in the database during speech synthesis, alternative triphones are selected according to a viseme clustering; visemes may be replaced if they belong to the same viseme class. The second criterion controls transition smoothness by minimizing the difference between the connected, overlapping ends of the triphones. In this thesis, this approach is extended by presenting a data-driven rule for triphone selection by a general statistical analysis of visemes and an application to 3D faces (see Chapter 10).

2.5.4 Viseme Classification

The introduction of this section on *Speech Synthesis* briefly presented the fields of *phonetics* and *articulatory phonetics.* As exposed, articulatory phonetics is closely related to visual speech animation while phonetics focuses on the audio aspect of speech. Addressing viseme clustering is the kernel of this thesis; a brief outline of the methods used to obtain such classifications concludes this chapter.

From the several body components involved in speech (see Fig. 2.6), only the lips and the tongue have a visual impact and play a role in visual animations. The teeth are also necessary but their importance is secondary as they work in correlation with the tongue. Typically, for vocals, the air is directly propelled out of the lungs and only the lips are involved in the sound modulation. Each vocal has thus a typical mouth configuration which can be viewed as a viseme with restricted degrees of freedom. Therefore, each vocal is clustered separately. In their 1983 paper, *Montgomery & Jackson*[87] investigate characteristics of vowels. Their results are used by *Kalberer et al.*[73] for building a viseme-space. The rest of the thesis restrains to the classifications of consonants.

For consonants the analysis is more complex. While visemes can be logically classified according to the body components involved in the speech process, *Owens & Blazek*[91] perform a perceptual analysis. On a videotape without sound, they present series of *vocal-consonant-vocal* (VCV) syllables involving four vowels and 23 consonants. Hearing-impaired subjects are asked to guess which consonant is uttered between the vowels. Viseme groups are then determined by the cross-correlation of confusions. Indeed, when two consonants form similar visemes they are likely to be confused in their observation.

By performing this experience on several subjects, a correlation table is drawn and after defining a correlation threshold, the classification is obtained. In their paper, *Owens & Blazek*[91] compare their classification with results

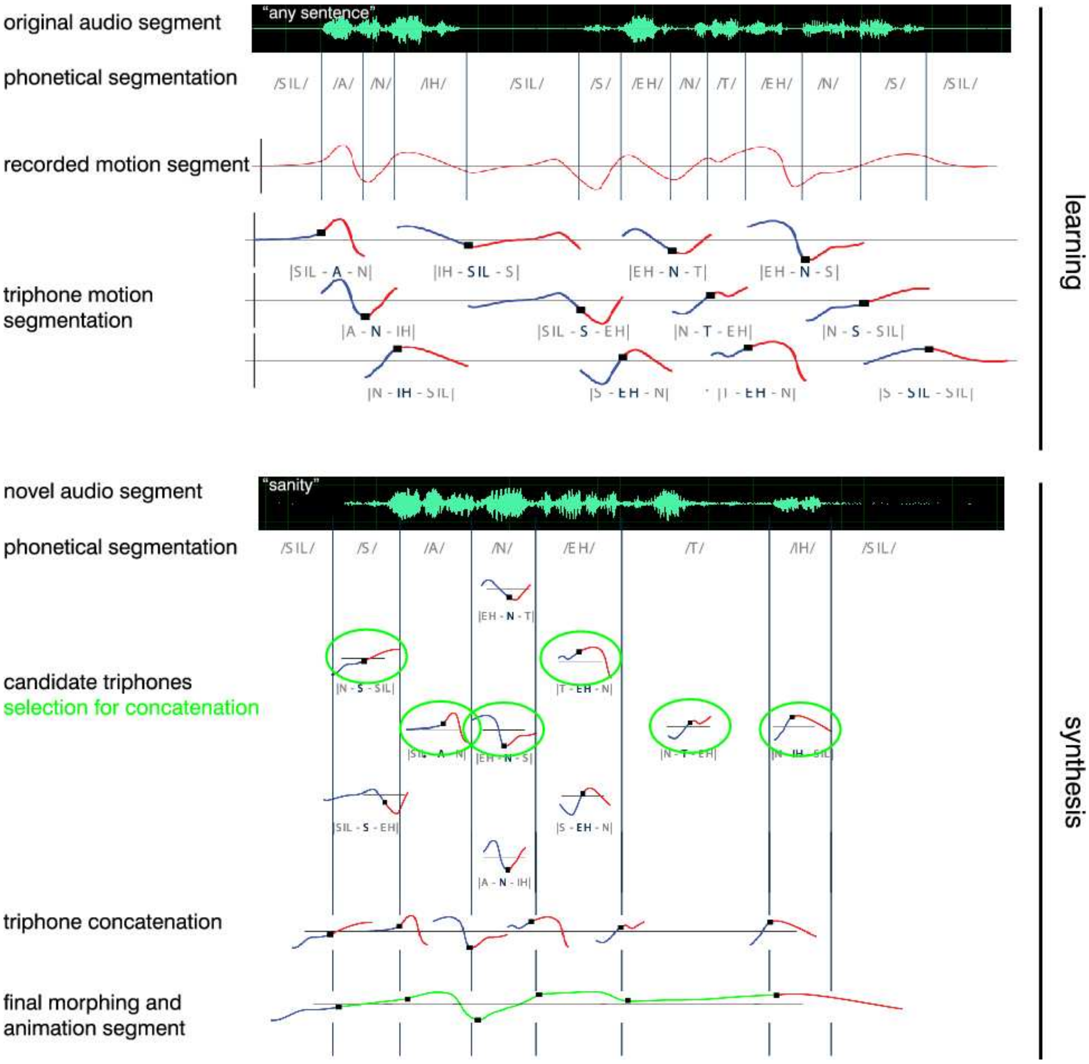


Fig. 2.5. The principle of triphone-based synthesis The triphone-based approaches are divided into two parts: learning phase and synthesis. In the learning phase, 3D data is acquired along with audio speech information. The recorded animation is segmented according to the successive phoneme occurrences. Triphone segments are stored in a database and labelled with their associated phonemes. For the synthesis, the successive phonemes from a novel audio track indicate the sequence of the triphones that have to be selected from the database. From several candidates, the sequence that performs the smoothest concatenation is selected and new animation segments are produced by morphing along the selected triphone.

from *Binnie et al. Walden et al.*. While the resulting clusters slightly differ, the correspondences validate their approach. The obtained classification is given in Section 12.1.

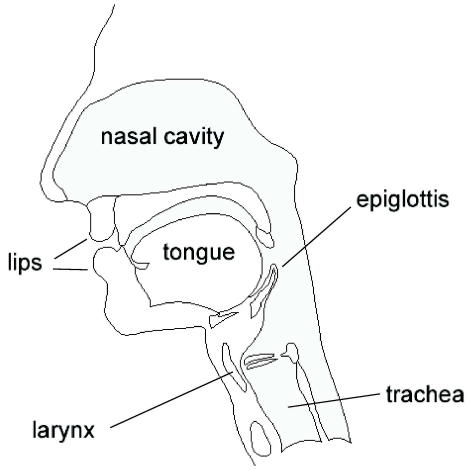


Fig. 2.6. The body components involved during speech From the illustrated components, only lips and tongue are visible for the observer. The lungs are missing in the illustration.

The classification given by *Owens & Blazek* has become a standard classification and is often used for viseme substitutions for speech synthesis. *Bregler et al.*[22] use this classification (see Section 2.5.3). *Kalberer et al.*[73] use the same classification for consonants and *Montgomery & Jackson*'s[87] for *monophthongs* (a "pure" vowel sound, with an almost fixed articulation as opposed to *diphthong*). Their coarticulation model is based on a limited set of phonemes that appear to be visually more important than others, such as vocals and labial consonants.

The MPEG-4 standard viseme classification is derived from it (they grouped visemes that are originally separated). *Yau & al.*[139] use the standard MPEG-4 classification for speech recognition from video sequences; they deviate from standard substitution approaches by focusing on distinguishing articulations for consonants without taking the actual pronounced phoneme into account. *Beskow & Nordenberg*[12] build a system that synchronizes speech and expressions dependently and rely on the MPEG-4 standard. Differently, *Kshirsagar et al.*[77] use motion capture data to estimate facial animation parameters defined on the same standard.

The approach presented in this thesis, derives a rule for viseme substitution starting with no *a priori* knowledge of viseme similarities. These are automatically deduced through a statistical analysis of the data and give a quantitative similarity measure that relaxes the selection rule of viseme grouping.

Part II

Background

In this part, we introduce the technical and theoretical aspects considered throughout the thesis and also discuss some alternatives. We present these aspects in detail here, so that the following Parts III (*Learning Visemes and Articulations*) and IV (*Speech Synthesis*), in which we build the system, can be understood and directly validated. It also allows us to better separate the contributions of our work from the underlying theory.

This part is divided as follows: in Chapter 3, the theoretical aspects of the dynamic 3D scanner we used for recording facial animations is presented. In Chapter 4, we present the optical flow techniques used to register the data. Chapters 5 and 6 give an introduction to dimensionality reduction techniques which we use for the analysis of the data. From these two chapters, the first one presents the linear eigendecomposition performed by *Principal Component Analysis* (PCA) (Section 5.3) and its application to *Multidimensional Morphable Models* (Section 5.4). We then present *Multidimensional Scaling* in Section 5.5, as a connection to nonlinear methods. Section 6.1 generalizes PCA to a nonlinear reduction technique known as *KernelPCA*, and finally, we present two popular reduction techniques used in speech and expression analyses: *Isomap* in Section 6.2 and *Locally Linear Embedding* in Section 6.3.

3

3D Data Acquisition

Section 2.1 already gave an overview on different 3D acquisition techniques. This chapter explains in more detail the techniques involved in the system used in this work for the dynamic acquisitions. The next section describes the triangulation reconstruction theory. Its extension to *structured-light* based scanners is presented in Section 3.2 along with *phase shifting* techniques which increase the resolution of the captured geometry.

The setup of the prototype scanner used in this thesis along with some software characteristics and its structure data is later described in Section 7.1.

3.1 A Structured-Light Based Scanner

Laser scanners use the triangulation process to recover three-dimensional geometries. Referring to Figure 3.1, suppose the distance b between the camera C and the laser P is known (via calibration). The laser beams a ray with an angle α and the impact on the object is observed on the camera's viewport. The location on the viewport indicates the angle β with which the camera observes the impact. Because the laser and the camera are synchronized, the system knows the three parameters b, α and β and can recover the depth h of the object at the impact location:

$$h = b\frac{\sin\alpha\sin\beta}{\sin(\alpha+\beta)} \tag{3.1}$$

Laser scanners retrieve the whole geometry of the object by running the beam over a defined vertical and horizontal angular range. This process requires several seconds and is thus well suited for static acquisitions. The process can be accelerated if, for instance, two laser beams are used; the camera,

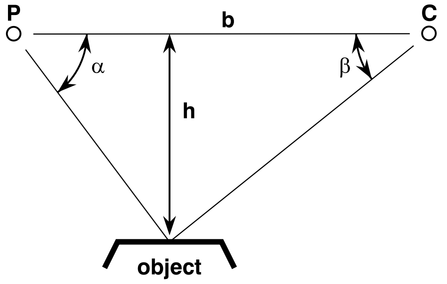


Fig. 3.1. The triangulation process for a typical laser-based 3D scanner Knowing the distance b between the camera C and the laser P, the observation angle β in the camera's viewport and the projection angle α, the depth h of the object at the impact location can be computed. (figure after *Wolf*[137])

however, will not be able to make the distinction between the two observed impacts. By using different laser colors, a light encoding is performed which solves the problem.

On the same principle, structured-light based acquisition technique use an encoding based on the projection of line patterns. *Bouguet & Perona*[19] showed such a technique in its simplest form: using only a desk lamp and a pencil, a video footage of the shadow of the pencil traversing a three-dimensional object is sufficient to recover a geometric structure. By using a more sophisticated setup, better acquisition can be obtained. Structured-light scanners propose to encode light by projecting a set of line patterns over a short sequence (5 in the present case). The lines are projected through a LCD panel at different resolutions (see Figure 3.3). This fast encoding technique allows to acquire geometries at a higher speed; if such patterns are projected at a high frame rate (about 200 Hz), a moving object can be considered to be at rest over a sequences of 5 shots, hence, producing full geometry recovery at a fifth of the camera/projector frame rate, i.e., at 40Hz in the present case.

The rest of this Section describes the theory behind structured-light encoding. The matter concerning the proper acquisition procedure is later discussed in Section 7.1.

3.2 Structured-Light Encoding and Phase Shifting

In this project a structured-light relies dynamic 3D scanner is used which bases on a *phase shifting* technique (see *Wolf*[137]). In accordance with the forementioned paper, in the following the phase does not refer to the complex component of light but to levels of grey in the gradient from black ($-\pi$) to white ($+\pi$) in a sinusoidal intensity pattern.

The encoding for structured-light works as follow: 7 line codes are projected in sequence with different resolution. These codes are shown in Figure 3.2. After the line codes are projected, the camera can attribute a bit sequence to each of the pixels of the viewport and the system knows the association to the projector angles which are thus converted to line codes.

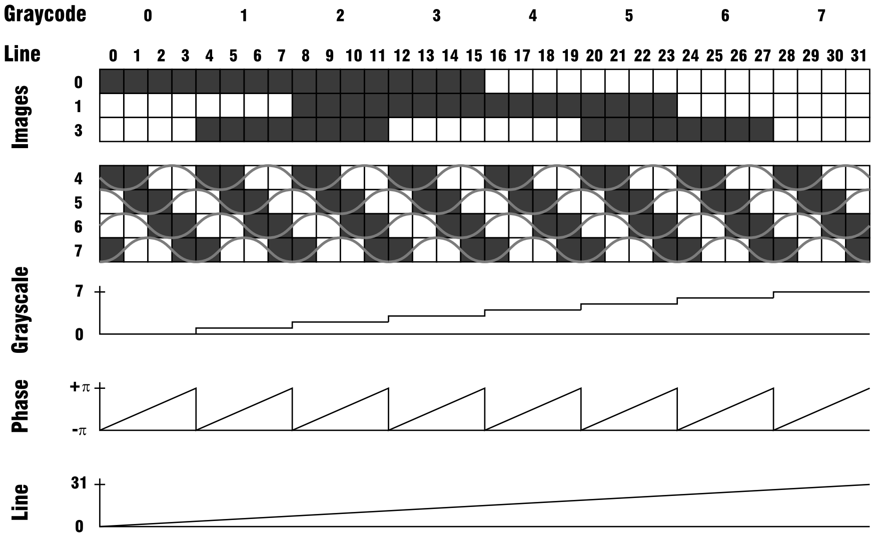


Fig. 3.2. Structured-light and phase-shifting encoding Seven different line patterns are successively projected on the recorded object which provides a binary encoding of the projecting angle. While this encoding divides the recorded surface into 128 segments only, the resolution can be increased by combining a phase shift technique. By projecting sinusoidal patterns, an almost infinitesimal resolution can be obtained. (figure after *Wolf*[137])

By projecting only black and white codes, the sequence shown in Figure 3.2 divides the projected area into $2^7 = 128$ line regions which would provide a low horizontal resolution reconstruction of the object. In order to increase that resolution, the *phase shifting* techniques proposes to project sinusoidal patterns (patterns 4 to 7) so the phase variation can be measured across the lines (see Fig. 3.3 for an illustration of varying line pattern projections) and provides an almost infinitesimal resolution. The sinusoidal pattern is obtained by slightly defocusing the lens of the projector which has a minimal impact on the first three patterns.

While such a step provides full 3D reconstruction for a static object by projecting only 7 patterns, the system can be further accelerated and perform better for deformable objects. If relatively small deformations are assumed, the projection of the three first patterns can be performed only once. The final system demands that the object remains at rest at the beginning of the

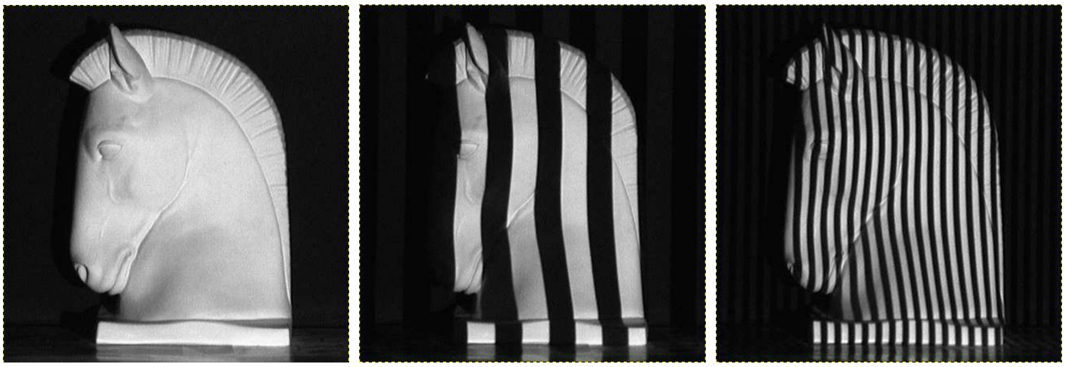


Fig. 3.3. Three light patterns (or line codes) on a static model Here, no phase-shifting is used. (illustration taken from *Pagès*[92])

acquisition in order to first perform a full reconstruction (7 patterns). Once this information is recorded, only the sinusoidal patterns are projected and rather than performing full reconstructions, only depth variations are tracked. Figure 3.4 illustrates such a deformation. In the viewing direction of the camera, a pixel has an original reconstructed depth value of z. A deformation is tracked in parallel to the viewing direction with a velocity v that produces a surface displacement Δz over two successive timesteps n and $n+1$. On the projection plane P, Δz is seen as a Δx displacement along the line pattern to which a shift in phase $\Delta\varphi$ is associated. According to this phase shift, Δz is then found.

The scanner first performs a global reconstruction and then detects variations locally assuming relatively slow deformations. This decreases the necessary recording frames to 5 (4 for the sinusoidal pattern, 1 code-free projection to retrieve the texture). With the camera and the projector running at 200Hz, the reconstruction rate is $40Hz$. This increase of speed, however, comes at a cost. In a speech animation scenario, the opening of the mouth induces abrupt jumps in depth from the lips to the teeth. Such a case produces a non continuous phase shift which prevents a correct association to the line patterns. The results show depth measurement errors which makes it impossible to retrieve the shape of "suddenly appearing" components (see Section 7.1).

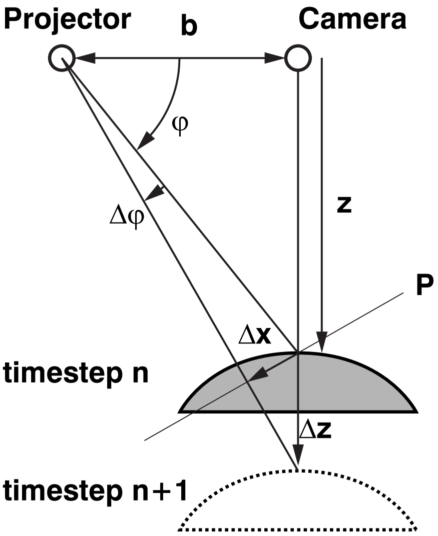


Fig. 3.4. Depth variations detection through phase-shifting Relatively small deformation velocities can be tracked by the phase shift they induce. The depth variation Δz produces a phase shift Δx along the projected pattern, hence the gray value changes on the recorded surface. (figure after *Wolf* [137])

4

Data Registration

Section 2.2 gave a review of popular approaches to data registration and mentioned different optical flow techniques. The optical flow algorithm used in this thesis uses, in combination, the approach proposed by *Lukas and Kanade* in 1981 [82] and the one by *Bergen et al.*[10] in 1992. This combination, used by *Blanz*[14], works with a pyramidal approach and offers two variants: one uses a Gaussian downsampling, the other a Laplacian downsampling. The next section gives the theory behind this approach and serves as the foundation of the extensions we performed to it in order to reduce divergence over long frame sequences with large deformations. This extension is presented in Section 8.3.

4.1 Optical Flow

The concept of *Optical Flow* refers to the displacement map between two images (*Neumann*[89]), typically from a video sequence where the time variation is relatively small; to each pixel in the first image, a velocity vector is attributed that indicates its displacement to the next image. The optical flow is a popular tool for data registration, as displacements or deformations from recordings can be deduced from its interpretation [38].

Consider for example a rotating sphere of uniform color; as no geometry or pattern variation can be perceived, such a phenomenon cannot be captured by optical flow techniques. For scenarios where displacement is perceivable, solving the optical flow is still not trivial and two known problems have to be addressed. Ideally, the displacement of a pixel in an image is recovered upon the postulate that its intensity remains constant from one frame to the next:

$$\frac{dI}{dt} = 0 \tag{4.1}$$

This is, however, in general, not the case for acquired data as illumination conditions change over the scene which affects the visual aspect of objects. Another problem is known as the *aperture problem.* Suppose a vertical homogeneous pole is recorded but its extremities are not visible from the camera. If the pole moves in a plane parallel to the camera's viewport, only the lateral displacement can be detected; optical flow techniques are unable in such a case to recover vertical movements until one of the extremities becomes visible. There are three general ways to address the aperture problem (see *Singh*[117] or *Barron et al.*[5] for a survey):

gradient-based approaches assume the conservation of image intensity. *Nagel*[88] solves the aperture problem, using second order intensity variations; *Horn & Schunck*[60] induce smoothing constraints;

correlation-based approaches (*Marr & Poggio*[85]) use the surrounding information of pixels. *Lucas & Kanade*[82] run a correlation window to look for the best match of a patch around the pixels; *Bergen et al.*[10] propose a downsampling framework that reports displacements measured in a lower resolution image version back to the original. These last two methods are used in this work and are discussed in more detail below;

spatiotemporal energy-based approaches: *Heeger*[59] takes advantage of the relation of the spatial and temporal frequencies with the velocities of a moving stimulus. In the spectral domain, displacements are defined over a plane on which the stimulus is tracked.

The founding optical flow approach used in this work is described by *Blanz*[14]. A short introduction is given here, the reader is, however, invited to refer to the forementioned thesis for a more rigorous description and explanatory illustrations. An extension to this optical flow approach is described in Section 8.3.

Optical flow techniques are usually applied to 2D data. To extend such techniques to 3D face data, face models are projected into a cylindrical representation where the projection axis traverses the head vertically in the middle. With such a projection, the data can be observed as a two-dimensional texture:

$$R(h,\phi), G(h,\phi), B(h,\phi) \tag{4.2}$$

and a two-dimensional depth map

$$r(h,\phi) \tag{4.3}$$

Hence, the optical flow tracks colors intensities for the texture and depth as "intensities" for the geometry. The intensities observed in these conversions to time variant 2D images are described as

$$I(x, y, t), \quad x \in \{1, \ldots, w\}, \quad y \in \{1, \ldots, h\}, \quad t \in \mathbb{R} \tag{4.4}$$

Knowing the intensities in a reference image at t_0, their displacement is found by computing the optical flow with the following constraint:

$$I(x(t), y(t), t) = I(x(t_0), y(t_0), t_0) \tag{4.5}$$

Lucas & Kanade[82] address the aperture problem by observing a square window around the pixels and finding a gradient that best matches the whole square. The displacement of the square is then reported to the central pixel. This approach gives a vector field that provides a first estimate but is subject to local extrema. This is solved by combining this approach in a coarse-to-fine framework as proposed by *Bergen et al.*[10]. This method consists in smoothing the images with a low-pass filter to reduce the amount of high-frequency information. This process corresponds to a downsampling of the images (usually by powers of 2) and by taking a weighted amount of neighboring intensities in the low resolution representation. By performing this downsampling step several times, a pyramidal hierarchy of the images is obtained. A first iteration of the *Lucas & Kanade* flow computation is performed at the bottom of the pyramid (lowest resolution image) and the flow field is reported to the image above. The flow is then refined by running the optical flow a second time. This step is performed until the top of the pyramid is reached.

The approach proposed by *Bergen et al.* solves the aperture problem for edge features, for instance, but is still unsatisfactory for low contrasted surfaces; typical examples for face models are the cheeks of the forehead. *Blanz*[14] proposes a further relaxation step: the vector field obtained by the previously described technique is coupled to a mass-spring system in a grid model matching the pixel distribution of the images. Each displacement vector is connected to its four neighbors. Reliable vectors are fixed (sharp contrasted features) and the others automatically adapt to these constraints. With this process, a smooth optical flow is produced between all images. Note that this smoothing process is particularly desired for scenarios like processing face images when deformations have an impact on large regions. Applications focusing on tracking small displacements (e.g., following a car in a video sequence) would avoid such a step.

A further point that needs to be discussed here is the filter used for the coarse-to-fine downsampling in the approach from *Bergen et al.*. A Gaussian pyramid is obtained by performing a low-pass filter which produces a blurred version of the image. Another option proposed by *Peter J. Burt and Edward H. Adelson*[24] is to use a Laplacian pyramid which corresponds to a high-pass filter. The Laplacian reduction can be directly obtained by computing the Gaussian downsampling and subtracting it to the original image. The

Laplacian pyramid keeps the high frequencies of the images, that is, the optical flow tracks sharp features (lips, face edge) rather than color intensities. Section 8.2 discusses the results after applying both types of filters. It turns out that a Laplacian pyramid works better for matching different face identities with a common expression, whereas a Gaussian pyramid is preferred when matching a single face with different expressions.

5

Dimension Reduction

Performing a dimension reduction on a dataset serves multiple purposes. First, the original data structure of a set might not be well representative of the experiment; if the correlation is great among observed variables, the phenomenon is likely to have a lower inherent dimensionality than the one provided in the observation. These reduction techniques can thus lower the dimensionality of a problem and better reflect the actual behavior that is to be analyzed. A second aspect is the fact that the produced representation gives a compressed form of the original data which facilitates computations and offer better portability.

While reduction methods are all based on eigendecompositions, there exist two major categories. The first concerns *Linear* reductions, they perform linear transformations and projections of the dataset. The basis of the low-dimensional space indicates the orthogonal variation directions of the data which are ordered in accordance to the decreasing magnitude of their associated variances. By taking the most important components, a *cumulative information variation* can be computed which indicates what percentage of the original variation is kept. Generally, the remaining components represent small fluctuations inside the data and by ignoring them, the reduction also performs as a noise filter.

The second category concerns *nonlinear* reduction and is discussed in the following Chapter 6. Measurements of nonlinear phenomena require high-dimensional spaces to be represented but they often evolve on a curved manifold of lower dimensionality which cannot be unfolded in a linear approach. Linear reductions are thus ill-suited for that purpose as they fail at giving an intuitive observation (these aspects are further discussed in Chapter 6).

The remainder of this chapter is dedicated to linear methods, SVD (Section 5.2), PCA (Section 5.3) and their application to Morphable Models (Section 5.4) which is used throughout this thesis. Section 5.5 describes MDS which can perform both linear and nonlinear reduction and give a good introduction to Chapter 6.

5.1 Linear Methods

Linear dimension reduction approaches have been well investigated and and their applications are multiple. This section gives a brief overview of two approaches, namely *Principal Component Analysis* (PCA) (Section 5.3) and *Multidimensional Scaling* (MDS) (Section 5.5).

PCA performs a *Singular Value Decomposition* (SVD) (Section 5.2) on a data matrix X in order to retrieve the principal variance directions of the covariance matrix C of X. *Multidimensional Morphable Models* (Section 5.4), which are based on PCA, are flexible model representations which are used throughout this thesis.

The second linear reduction method, MDS, shows a different approach that provides similar results as PCA but can also perform nonlinear reduction. By an example, nonlinear approaches are introduced which are then discussed in the next chapter.

An important property of linear methods is that they not only offer data compression, but can also serve as noise reduction processes. Indeed, the information component in a signal is generally greater than the noise component; removing the components associated with the lowest weights (singular- or eigenvalues) ensures the removal of unwanted disturbances. Naturally, the remaining components are biased by the acquired noise but capture well the relevant information.

On another level, because these methods are all based on eigenvalue extraction of the data, they provide the inherent dimensionality of an observed phenomenon if the latter has a linear nature. This aspect is discussed in Chapter 6, where a survey of nonlinear methods is given.

5.2 Singular Value Decomposition (SVD)

Suppose a matrix $A \in \mathbb{C}^{m \times n}$ is given that describes a linear transformation in $\mathbb{C}^n$. Such a transformation can be decomposed into three consecutive "simple" transformations: an alignment (rotation), a stretching and a hanging (second rotation) as illustrated in Fig. 5.1. The *Singular Value Decomposition* (SVD) decomposes the transformation matrix into three matrices $A = (hanger)(stretcher)(aligner)$ or

$$A = U\Sigma V^* \quad \text{with} \quad U \in \mathbb{C}^{m \times m}, \Sigma \in \mathbb{C}^{m \times n}, V^* \in \mathbb{C}^{n \times n} \tag{5.1}$$

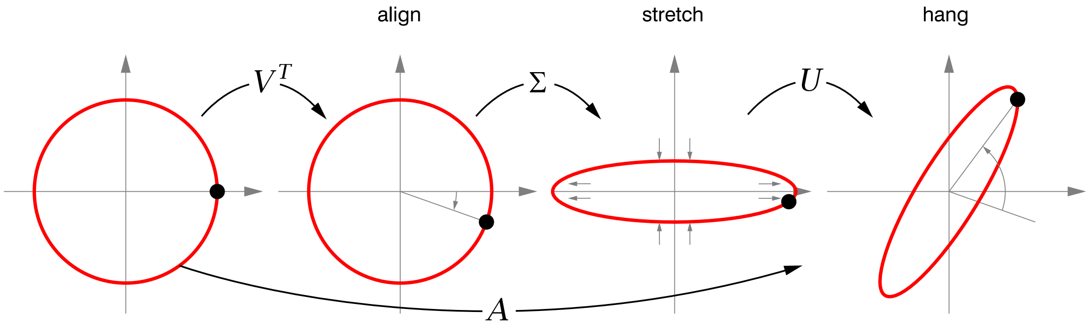


Fig. 5.1. The decomposition of a linear transformation From left: the first rotation performs an alignment. The unit circle is then stretched to becomes a ellipse and a second rotation brings the circle to the final state.

and V^* is the conjugate transpose of V. If only real numbers are considered, Equation 5.1 becomes

$$A = U\Sigma V^T \tag{5.2}$$

In the example of the unit circle, the purpose of the first rotation (the alignment) is not clearly shown. The image of the unit circle through A is a rotated ellipse with its two main axes. In the pre-image, these axes are already orthogonal but they are not necessarily aligned to the basis of the original space. The first rotation precisely aligns these axes to the basis in order to perform the stretching. The stretching is performed along the axis of the basis by a single coefficient for each axis; this is visible in the matrix Σ which is a diagonal matrix. The values $\sigma_i, i \in \{1, \ldots, n\}$ are the *singular values* of A, and the shared square roots of *eigenvalues* of AA^T and A^TA; they are furthermore given in decreasing order. In Equation 5.2, U and V hold the *left* and the *right singular vectors*, respectively, of A or the eigenvectors of AA^T and A^TA. The multiplication by a matrix with its transpose is a symmetric, matrix and therefore, the eigenvectors in U and V form distinct orthonormal basis.

If A is a singular matrix (does not have an inverse), a projection occurs along at least one of the basis axis. It is thus not possible to recompute the pre-image though the transform. This means that at least one of the singular values is zero ($\exists i \in \{1, \ldots, n\} | \sigma_i = 0$). (A pseudo-inverse A^+ matrix can be computed: if A has full column rank, then $A^+ = (A^TA)^{-1}A^T$.)

5.3 Principal Component Analysis (PCA)

The goal of *Principal Component Analysis* (PCA) is to perform a statistical analysis of a distribution of samples in a high-dimensional space by providing a basis that better describes the data.

Suppose an experiment is performed m times and each time a set of n measurements is gathered. The measurements (variables) of each experience form an n-dimensional vector $\mathbf{x}_i$. If n is high and the intrinsic dimensionality of the observed phenomena likely to be much smaller, it becomes interesting to look for a simpler representation. The reduction in dimensionality is possible, when the number of parameters that influence the phenomenon is smaller than the number of variables. By observing the correlation of the n variables over the m experiences, the number of influencing parameters l can be found.

If two observed variables x and y are *correlated* (they vary in accordance to each other and $\mathbf{cov}(x, y) > 0$), they are likely to be described by a single variable. In a similar idea, by analyzing the correlation of all variables against each other (cross-correlation), the dimensionality of the observation space can be reduced to the inherent dimensionality of the phenomenon.

This dimension reduction process is performed by an analysis of the covariance matrix[1]

$$C = \frac{1}{m}\bar{X}\bar{X}^T, \quad C \in \mathbb{R}^{n \times n} \tag{5.3}$$

which contains all the correlation information of the sampled vectors $\mathbf{x}_i$; $\bar{X}$ is the $n \times m$ matrix containing all the sample vectors from which the average vector $\hat{\mathbf{x}}$ is subtracted (centering of the data).

Presently, the samples are observed in a n-dimensional space. If their distributions follow a Gaussian process, a PCA produces a new orthogonal basis along the successive most important variation directions of the data. The axes of such a basis are the eigenvectors of the covariance matrix and the distributions follow the standard deviations $\sigma_i, \in 1, \ldots, n$.

As presented in the previous section, the SVD of a matrix A provides the eigenvectors and the eigenvalues of the matrix $A^T A$. Hence, by performing an SVD on $\bar{X}$

$$\bar{X} = U \varSigma V^T \tag{5.4}$$

the columns of U give, ordered after the decreasing magnitude of their associated eigenvalues, the eigenvectors of $\bar{X}\bar{X}^T$ which up to a coefficient $\frac{1}{m}$ is

---

[1] In general, if the samples represent only a subset of a population, the fraction uses $(m - 1)$ instead of m; the results come closer to the standard deviation measured on the whole population. In this thesis, we consider the acquired data as a representation of a full population of 3D samples and use m in the computation.

the covariance matrix of $\bar{X}$. The eigenvalues on the diagonal of Σ give the distribution of the data along the eigenvectors. By computing the cumulative ratio

$$g = \frac{\sum_{i=1}^{l} diag(\Sigma)_i}{\sum_{i=1}^{N} diag(\Sigma)_i} \tag{5.5}$$

the information gathered along the l first components can be measured.

The l first eigenvectors $\mathbf{u}_i$ and their associated variance σ_i^2 give a representation of the data with a lower dimensionality ($l << n$). It is generally admitted that the remaining $(n - l)$ components reflect the noise gathered during the measurements. Therefore, the PCA not only reduces the dimensionality of a problem but also performs as a noise filter.

Recall that the low-dimensional representation only describes the variations of the data. In order to reproduce the original samples, the variations must be added to the average sample $\hat{\mathbf{x}}$ that was previously subtracted from the $\mathbf{x}_i$.

5.4 PCA in a Multidimensional Morphable Model (MMM)

A Multidimensional Morphable Model (MMM) is a flexible model for representing and synthesizing objects of a common class. The model is based on a set of m acquired prototypes. In a class, objects must offer similar characteristics or features (usually shape and/or texture); for instance, a set of cars [68], of faces [18], of teeth [17] and so on. The principle is based on first matching the common features of a set of objects in a *registration* step. For each of the acquired prototype i, a vector $\mathbf{s}_i$ describes the list of n selected features.

The *construction* step of the morphable model performs a *Principal Component Analysis* (Section 5.3) on the registered data. The average vector $\hat{s}$ is subtracted from the samples $\mathbf{x} = \mathbf{s} - \hat{\mathbf{s}}$.

In order to analyze the variance of the data, all vectors $\mathbf{x}_i, i \in \{1, \ldots, n\}$, are combined to a data matrix X and the diagonalization of the covariance matrix

$$C = \frac{1}{m} X X^T \tag{5.6}$$

is computed by a Singular Value Decomposition (Section 5.2) $X = U \Sigma V^T$. The principal components $\mathbf{u}_i$ are the columns of the matrix U.

The PCA gives a representation of lower dimensionality l. In order to get the coordinates of the original prototypes in the PCA space, each vector $\mathbf{x}_i$ is projected onto the components $\mathbf{u}_k, k \in \{1, \ldots, l\}$. The shape is then recovered by adding the average vector $\hat{\mathbf{s}}$.

Besides the fact that morphable models offer a low-dimensional model that performs a data compression of large datasets, their applications are multiple. For instance, as it is going to be the case in this thesis, a morphable model permits the synthesis of novel shapes: an infinite set of shapes can be generated by morphing along the components $\mathbf{u}_i$:

$$\mathbf{s}' = \sum_{k}^{l} a_k \mathbf{u}_k \tag{5.7}$$

where the coefficients $a_k \in \mathbb{R}$ can take any values.

Blanz et al. have also proposed a fitting technique that allows the reconstruction of shapes. By using the available knowledge in the database, the optimum coefficients a_k can be found to matches shapes that are incomplete. This has applications for reconstructing teeth geometry [17] or to generate three-dimensional face shapes from 2D images [16].

5.5 Multidimensional Scaling (MDS)

In 1952, *Torgerson*[125] introduced *Multidimensional Scaling (MDS)* as a tool for retrieving low-dimensional representations of high-dimensional data. This method has the advantage that is can apply as a linear as well as nonlinear reduction.

MDS is based on relative measures among the samples (distances); the samples are thus reference free and analyzed by their (dis)similarities. This dissimilarity measure is described by distances between all sample pairs, which can be a metric (satisfying the triangular inequality) or a subjective judgment mapped to a grading scale. The dissimilarity measures δ_{ij}, the composing elements of the distance matrix D, between two samples i and j must follow the three following rules:

$$\delta_{ij} \geq 0 \tag{5.8}$$

$$\delta_{ii} = 0 \tag{5.9}$$

$$\delta_{ij} = \delta_{ji} \tag{5.10}$$

and a dissimilarity is a metric if in addition

$$\delta_{ij} \leq \delta_{ik} + \delta_{kj} \tag{5.11}$$

This last rule is typically lost when the perceptual measures are performed. The distances are then combined in a matrix D which is symmetric, positive, and the diagonal is null (distance matrix).

In Section 5.3, the Principal Component Analysis is described by performing a Singular Value Decomposition (SVD) on the centered data metric $\bar{X}$ to find the eigenvectors of the covariance matrix C (Eq. 5.3). The principle does not absolutely require a covariance matrix but can also be performed on a correlation matrix (normalized covariance matrix) or a dot-product matrix (neither centered nor normalized). Because the distance matrix D is not positive semi-definite (see *Abdi*[1]) it is not suited for an eigendecomposition. The first step in MDS is to transform D to an equivalent dot-product matrix (see *Yang et al.*[138]).

The dot-product matrix B is computed as a Gramian matrix: a matrix H is constructed with its elements

$$h_{ij} = \delta_{ij} - \frac{1}{n}. \tag{5.12}$$

H subtracts the average of each row i and each column j from δ_{ij}; the average distance over the whole matrix D is again added. B is then given by

$$B = -\frac{1}{2} H D^2 H. \tag{5.13}$$

Finally an SVD decomposition is performed on B as for the PCA. In analogy to the cumulative ratio computed for the PCA (see Eq. 5.5), MDS computes the residual variance r by summing up the remaining $(n-l)$ eigenvalues. The "MDS *stress*" r quantifies the information left along the remaining components.

If the distance matrix D is based on Euclidean distances, the decomposition produces exactly the same results as the PCA does. We next introduce by an example how MDS can perform a nonlinear reduction.

Connecting Cities

We give here an example of an application of MDS using a Matlab implementation from *Slaney & Covell*[118]. Depending on the nature of the distances (similarities) in the matrix D, MDS performs either a linear or a nonlinear reduction.

Consider the the table of aerial (straight) distances between several cities, in this example Switzerland (see Appendix 16.4 top of table). MDS can recover the relative positions of the cities as a two-dimensional set of coordinates. Figure 5.2 illustrates the output (white crosses) mapped back on the real locations (green circles). The given distances fit naturally in a two-dimensional plane and the locations are thus recovered perfectly. The output is a relative reconstruction; the axes in the novel representation are meaningless and the

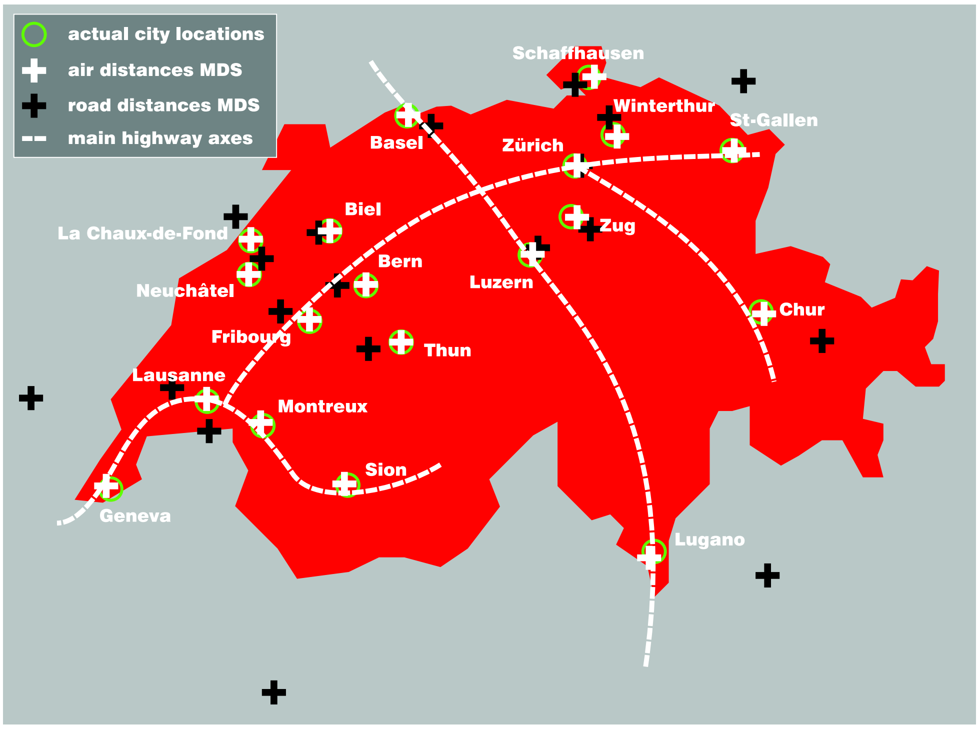


Fig. 5.2. An example of a linear and a nonlinear application of MDS The figure gives a map of Switzerland and its principal cities (green dots). MDS computes a distribution of data based on a distance matrix. Here, the results for two matrices are obtained; first, with a distance matrix composed of real geographic distances, where MDS places the cities exactly on their real locations (white dots). The second scenario takes road distances in the matrix which reflect perceived distances for a driver. Due to the mountainous topology of the country, some cities require longer travel distances and are thus perceived further away than they really are. With that matrix, MDS places these cities accordingly (black dots). The second output is centered on Zurich. (figure by the author, MDS Matlab implementation by *Slaney & Covell*[118])

locations are correct up to a Euclidean transformation. A linear transformation is necessary, in order to match the output to the original map.

Consider now the measuring of road distances connecting the cities (see Appendix 16.4 bottom of table). Typically, these would be the perceived distances between the cities for a driver. By the topological nature of Switzerland, some cities require a large detour to be reached by car and are thus *perceived* further away than they actually are. The triangular inequality is lost and the graph connecting the distances can only be represented in a higher-dimensional space. The driver, however, still likes to see the distances in a plane, and we thus look for a mapping of the graph to a lower-dimensional space where the distances between the cities reflect as much as possible the relative perception.

MDS performs exactly the same way for this kind of problem, only the desired output dimensionality needs to be specified. Again, Figure 5.2 illustrates the new mapping (black crosses). With the output centered on Zurich, one can see how cities like Sion or Lugano are perceived much further away than they really are, but such a representation much better reflects the perceived distances for a driver. The perceived topology of Switzerland is thus significantly different from the geography and thus follows a nonlinear deformation in order to match the perception.

This extension to MDS is the key to the growing interest in nonlinear reductions. In a linear representation, data can be really hard to interpret, and its intrinsic structure can be left unnoticed. If the nature of the data lies on a curved manifold, a nonlinear approach can produce a more intuitive representation.

6

Nonlinear Dimension Reduction

Algorithms for nonlinear dimension reduction are currently an extremely hot topic: while these approaches enjoy great success, they still are often disputed as to the nature of the structure they produce; indeed, these structures can vary from one technique to the other on a same set of samples. The other point of controversy is based on the fact that these techniques always assume that the samples lie on a low-dimensional curved manifold which is difficult to visualize in a linear reduction approach. While these reduction techniques always provide a reconstruction, the assumption of a manifold cannot always be proven. However, nonlinear reduction techniques have proven to be a successful tool in several applications and often offer a better alternative to linear techniques.

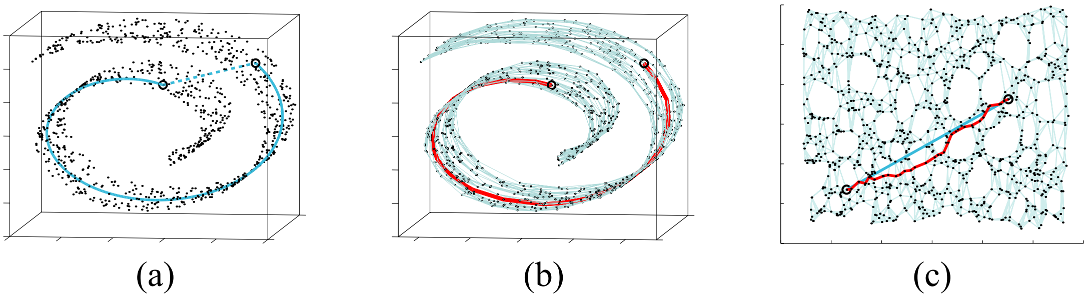


Fig. 6.1. Unfolding The Swiss Roll This example shows the nonlinear reduction process on points sampled along a curved manifold. In a Euclidean representation (a) two points that lie far apart on the manifold are located close to each other, which does not well represent the nature of the data. *Isomap* generates a graph based on sample adjacency (b) which is unfolded in a lower-dimensional representation (c). This representation better reflects the nature of the original data. (Taken from *Tenenbaum et al.*[121]

The idea of performing a nonlinear dimension reduction on a set of data, is to detect the curved manifold on which the data is distributed and to *unfold* it to a lower-dimensional representation that better reflects its inherent global coordinate system. Figure 6.1 illustrates such a process. In this example, the sampling points along a “Swiss roll” are shown in a Euclidean representation. With such a visualization, two points that lie far apart on the manifold are located close to each other, which does not reflect the real nature of the data (a). Here, using *Isomap* (see below), an adjacency graph is built by connecting each sample to its closest neighbors (b). The graph is then unfolded to get the two-dimensional representation of the manifold (c). As one can see, this representation better reflects the actual distance between the two samples. (In general, the path connecting the samples if found by seeking the shortest path on the graph.)

This chapter discusses some of the existing nonlinear techniques. In the previous chapter, PCA was presented as a linear reduction technique that performs an eigendecomposition of a covariance matrix. In fact, the decomposition can be performed on different matrices (the requirement is that these matrices are positive semi-definite). The first section of this chapter (Section 6.1) discusses this aspect by introducing *KernelPCA* a nonlinear generalization of PCA. In fact, most nonlinear techniques are variants of KernelPCA.

Two important techniques are then discussed. Section 6.2 presents *Isometric Feature Mapping* (Isomap) and Section 6.3 *Locally Linear Embedding* (LLE). These two approaches differ by how the samples are considered with respect to the manifold. Isomap performs a *global* reduction that reflects the original distances between all the samples, whereas LLE performs a *local* reduction which only guarantees local invariance.

In this thesis, speech articulation samples are analyzed using LLE. We give here a brief summary of both of the latter approaches as they become popular tools for expression and speech analysis [67, 140]. Further techniques can be found in a good survey on most of the linear and nonlinear techniques by *van der Maaten*[127]. In particular, *Brand*[20] recently introduced *Manifold Charting*: this techniques, which is quite different from the forementioned ones, consists in computing linear models locally on the manifold and realign them in a lower representation.

An important point to mention about these low dimensional nonlinear representations, is that the space they describe is only defined on the location of the samples, intermediate positions can only be approximated by a combination of adjacent samples. On that aspect, *Bengio et al.*[7] offer an extension to most of the nonlinear reduction techniques which allows to locate new samples in the manifold representations without having to recompute the eigendecomposition.

6.1 Kernel PCA

Section 5.3 presented the PCA eigendecomposition of a covariance matrix in order to perform a linear reduction. As mentioned in Section 5.5, such decompositions can be performed on matrices of different nature (correlation or dot-product matrices). *KernelPCA* [112, 8] proposes to use a *kernel matrix*, the nature of which is defined according to the desired analysis [127]. By defining a kernel function

$$k_{i,j} = \kappa(x_i, x_j) \tag{6.1}$$

the kernel matrix K is constructed and centered according to Equation 5.12. The reduction then follows the steps of MDS.

If k is a linear kernel, the reduction is equivalent to PCA; a dot-product kernel performs just like MDS. In fact, KernelPCA is a nonlinear generalization of PCA (see also *Williams*[136]). In the appendix of their paper, *Schölkopf et al.*[112] present a set of kernel selections according to the applications.

Dimensional Augmentation

KernelPCA can also be used to separate data in a higher-dimensional space, it is thus not always referred to as a dimension reduction technique. For instance, by taking

$$\begin{aligned} k &: \mathbb{R}^2 \to \mathbb{R}^3 \\ (x_1, x_2) &\mapsto (x_1, x_2, x_1^2 + x_2^2) \end{aligned} \tag{6.2}$$

the samples distant from the origin are mapped higher in the third dimension and can be separated by a plane from the samples close to the origin.

6.2 Isometric Feature Mapping (Isomap)

Isomap (*Tenenbaum et al.*[121]) is another eigendecomposition based reduction technique; but instead of using measured distances, it estimates the geodesic distances along the low dimensional manifold. The dimensionality of the manifold needs, however, not to be known a priori and is automatically found by the algorithm, which make this approach really powerful.

Isomap is based, like MDS (Section 5.5), on relative measurements and employs a distance matrix. In order to retrieve the geodesic distances, the distance matrix is transformed in two steps: first, a binary adjacency matrix is generated that gives the connections of each sample to its closest neighbors. The *adjacency condition* can be defined either by a k-number rule (the

k closest samples are selected) or by an ϵ-distance threshold (all neighbors lying closer than ϵ are selected). The second step computes the geodesic distances between all samples by cumulating the distances traversed on the graph derived from the adjacency matrix (*Dijkstra*'s shortest path). A second distance matrix is generated which contains all the pairwise geodesic distances. If the adjacency condition is robust, no shortcuts occur in the graph[1] and the manifold can be unfolded by performing MDS on the geodesic distances matrix.

Isomap presents two problems. First, because MDS works with pairwise distances between all n samples, computing the Dijkstra shortest path between all of them is very expensive ($O(n^3)$). The problem can be reduced by choosing a subset of m samples (landmarks) and perform the MDS reduction on them [36]. The shortest distances between them are, however, still computed on the dense graph but the complexity reduces to $O(m^2n)$.

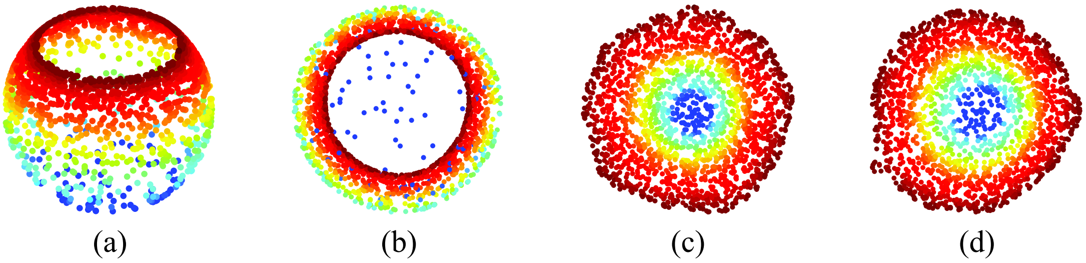


Fig. 6.2. Conformal mappings: data generated on a plane and conformally warped to a fish-bowl shape (a); note the dense sampling around the rim. Isomap fails to recover the geometry due to its violated assumptions (b); Conformal Isomap (c) and LLE (d) both recover the original data. (figure and caption taken from *Ihler*[65].)

Isomap retains the local distance between the samples, but also the angles and is thus well suited for unfolding manifold like the Swiss-Roll example[65]. However, the second problem is a direct consequence to this isometric nature of the Isomap reduction. The fish-bowl example illustrated in Figure 6.2 shows the conformal mapping to a fish-bowl of data generated with homogeneous distribution on a plane. The mapping produces a denser distribution along the rim (a) which constrains the expandability of that region for the Isomap (b). *de Silva & Tenenbaum*[36, 126] propose *conformal Isomap* (or *C-Isomap*) that adds invariance to local scale by only considering the relative angles. The

---

[1] If the low-dimensional manifold folds together, a weak adjacency criterion could connect two samples that lie far apart on the manifold. See *Saxena et al.*[110] for a variant to Isomap that increases robustness to shortcuts.

behavior of conformal Isomap (c) is very similar to LLE (d) which is presented in the next section.

Note: the “Connecting Cities” example given in 5.5 is similar to the performance of Isomap. Taking the road distances between the cities is similar to considering geodesic distances.

6.3 Locally Linear Embedding (LLE)

Locally Linear Embedding (LLE) proposed by *Roweis and L. Saul*[105, 108] is a different approach to dimensionality reduction than the techniques discussed so far. LLE is a *local* nonlinear technique as it is based on local reconstructions. In the algorithm, only the connections between adjacent samples are considered, inducing a sparse matrix which accelerates the eigendecomposition.

The Algorithm

The algorithm takes m n-dimensional input vectors $\mathbf{x}_i$. The assumption is that a sample $\mathbf{x}_i$ and its k neighbors[2] $\mathbf{x}_j$ form a linear patch in the low-dimensional manifold so that $\mathbf{x}_i$ can be approximated by a linear combination of its neighbors: $\mathbf{x}_i \sim \sum_j^k w_{i,j}\mathbf{x}_i$. The weights $w_{i,j}$ are chosen so that for each patch they sum up to one: $\sum_j^k w_{i,j} = 1$. They are found by minimizing the reconstruction errors $\mathcal{E}(W)$ as a constrained least square problem [109]

$$\mathcal{E}(W) = \sum_i |\mathbf{x}_i - \sum_j w_{i,j}\mathbf{x}_j|^2 \tag{6.3}$$

$\mathbf{x}_i$ is hence projected in a hyperplane defined by its neighbors[3]. The $n \times n$ weighting matrix W is a sparse matrix, as $w_{i,j} = 0$ when $\mathbf{x}_j$ is not adjacent to $\mathbf{x}_i$.

The last step of the algorithm is to find $\mathbf{y}_i$ the image of $\mathbf{x}_i$ in the global internal coordinate system of the manifold. In the image space, the weights should remain the same in order to preserve local invariance of the samples relatively to their neighbors. This is solved by minimizing a second error

---

[2] Similarly to Isomap, the neighbors are selected according to either a k-number rule or an ϵ-distance criterion.

[3] With k neighbors selected and assuming $k < n$, the dimensionality d of the hyperplane on which $\mathbf{x}_i$ is projected is smaller than k ($d \leq k - 1$); this condition is extended to the manifold, as LLE performs an unfolding of that projection space. When LLE is computed, setting d above k generates additional dimensions in which show no information.

function $\Phi(Y)$ similar to Equation 6.3 but this time the weights are fixed and the embedding coordinates are optimized:

$$\Phi(Y) = \sum_i |\mathbf{y}_i - \sum_j w_{i,j}\mathbf{y}_j|^2 \tag{6.4}$$

In the minimization, a target dimensionality l is attributed to the size of the embedded vectors $\mathbf{y}_i$. Moreover, this minimization, which has a unique global minimum, is performed without the knowledge of the input vectors and the embedding is thus found only from the weight information.

The minimization is performed by solving an $n \times n$ eigendecomposition: Equation 6.4 is rewritten in a quadratic form $\Phi(Y) = \sum_{ij} m_{ij}(\mathbf{y}_i \cdot \mathbf{y}_j)$ with m_{ij} defining a cost matrix M. By constraining the translation, the rotation, and the scaling degrees, the low-dimensional embedding is found by performing an eigendecomposition of the matrix M. The global internal coordinate system of the manifold is then defined by the $l + 1$ eigenvectors associated with the lowest eigenvalues (*Rayleitz-Ritz* theorem in *Horn & Johnson*[61]). In fact, the lowest eigenvector is also discarded as it correspond to the free translation mode of the embedding associated with the eigenvalue 0; the dimensionality of the embedding is finally l.

LLE versus Isomap

LLE is similar to Isomap as both are based on a graph representation of the low-dimensional manifold. In both cases, the graphs are derived from neighborhood connectivity. Isomap, however, only uses the graph representation in an intermediate step and recomputes geodesic distances between all pairs of samples. The LLE thus performs the eigendecomposition on a sparse matrix and, as a consequence, the embedding is retrieved much quicker. The fish-bowl example in Figure 6.2 also shows that because LLE retains only local scaling of the samples, the variation of sample density over the manifold does not prevent it from retrieving a correct embedding.

However, LLE reconstructs samples as a weighted linear combination of their neighbors and, unlike Isomap[122], the embedding does no reflect the original distances between the samples. This last aspect is one of the main differences in the outputs generated by *local* and *global* nonlinear techniques. Moreover, Isomap is able to detect the inherent dimensionality of the manifold, whereas the LLE requires the user to take guesses. Usually, this implies that the process has to be run several times with different parameters until a correct embedding is found.

Part III

Learning Visemes and Articulations

In this part of the thesis, we set up an articulation model based on real data acquisitions. This model will permit the analysis of speech articulation in Part IV in order to recreate natural animations.

In his thesis, *Blanz*[14] proposed a morphable model for face identities. This model, based on the acquisition of 200 faces, allows to morph between original faces, but also permits to define novel faces in a high dimensional face-space. In our work, a similar approach is undertaken; but instead of considering identities, our model bases on face deformations associated to speech. While the face acquisitions in Blanz' model are performed with a static scanner, articulation motions are, here, captured using a high-speed dynamic scanner. The articulation model will thus not only permit the generation of face shapes (visemes), but also, by extension, the generation of face animations.

The different steps involved in the construction of such a model are detailed in the next chapter. In Section 7.1 of the first Chapter, an actor is recorded with the dynamic 3D scanner which produces, with a frame rate of 40Hz, a collection of 3D face meshes or a “3D video” sequences. However, the face geometries delivered by the scanner cannot be directly studied. Indeed, the face coverage of the recordings is only partial and varies over the sequences. Also, the presence of holes and noise makes the data difficult to be analyzed. For instance, the geometry of teeth and tongue cannot be captured, resulting in erroneous reconstructions inside the mouth cavity. In Sections 7.2 and 7.3, we describe how the original data is amended by removing the structure of teeth and the tongue, how we smoothly filled the holes and reduced the acquisition noise (see Figure 6.3 for an example).

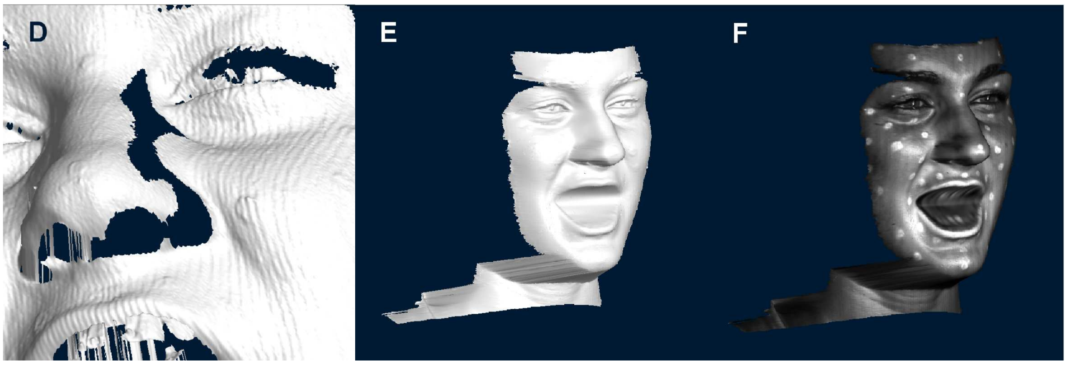


Fig. 6.3. Data preprocessing Detail of the mesh obtained from the scanner with holes and residues of the projected line pattern (D), reconstruction with tongue and teeth information removed, holes filled and geometry smoothed (E), reconstruction with average texture (F).

Once the data is cleaned, we register all recorded sequences to obtain a general mesh consistency over all recorded 3D frames. Indeed, 3D frames generated by the 3D scanner are sequences of independent meshes where vertices do not correspond to the same location on the face over the time. Two approaches are undertaken: the first one (Section 8.1) tracks marker-points in order to perform a pre-warp over the data so that large deformations relative to a reference frame can still be captured by the optical flow (Section 8.2). In the second approach (Section 8.3), an adapted optical flow technique is used which combines an absolute with a sequential matching. The second method proves to give better accuracy for the registration.

After registration, we analyze the variations in the geometry over the set of the 3D frames and build a *Multidimensional Morphable Model* (MMM) for articulation (Section 8.4) which offers a spatial representation of visemes.

We have already recorded sound simultaneously with the data. By performing a phonetic alignment (Section 8.4.2), we label each frame with its associated phoneme. By projecting the 3D frames onto the viseme-space, we can then define clusters of visemes and already generate simple animations (Section 8.6).

7

Data Acquisition and Preprocessing

7.1 3D Acquisition

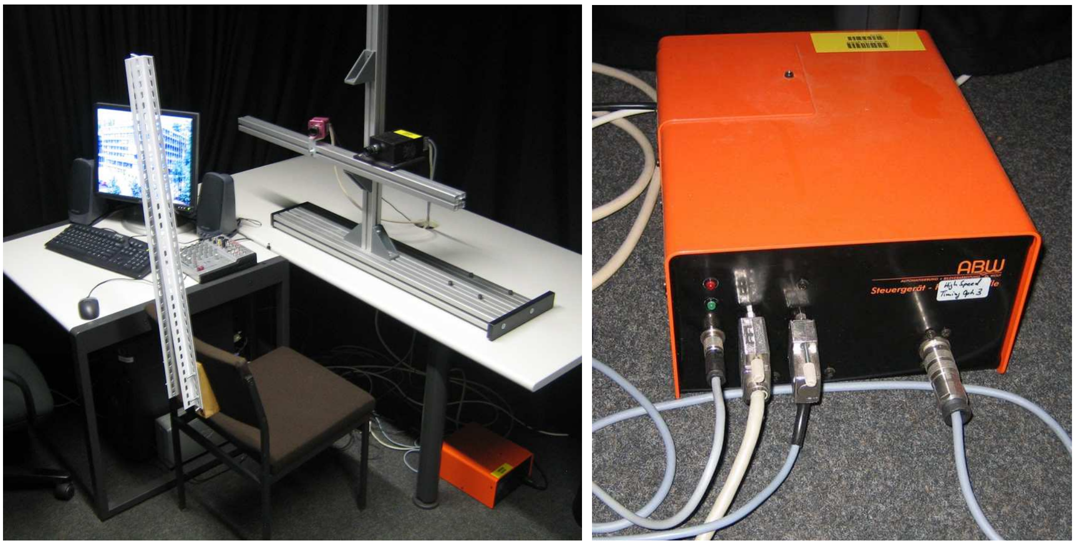


Fig. 7.1. The dynamic 3D scanner setup Both the high-speed camera (pink) and the projector (black) are fixed on a horizontal bar. The control device (orange box) keeps these two elements synchronized, and the data is acquired by the computer on a frame grabber directly from the camera. The chair is equipped with two metallic bars which reduce head motion during the acquisition.

The dynamic 3D scanner setup by ABW[1] consists in a high speed greyscale camera with a resolution of 640 × 480 pixels and a projector that emits the structured-light onto the model (see Section 3.1 for details on structured-light based scanners). As depicted in Figure 7.1, the camera (pink) and the projector (black) are fixed on a horizontal bar and both components aim at a point which is about 40cm from the bar and corresponds to the center of the recorded frustum. This frustum is about the size of an average face, about a 30cm cube. The control device (orange box) ensures synchronization between both elements and is controlled by a computer. Frames captured by the camera are directly sent to the computer's frame-grabber with a frame-rate of 200 Hz.

The acquisition interface on the computer offers a live camera view in order to ensure the right positioning of the subject. A test 3D frame can be captured to verify the quality of acquisition. The user can then define the recording length in frames (at 40 Hz) of a dynamic sequence. While the camera records with 200 Hz, 5 consecutive frames are necessary to generate a 3D model; hence, the output 3D frame rate is of $40Hz$. To proceed with acquisition, the user enables recording and a first high pitched beep is heard. From this moment on, the recorded subject has to remain still for a full second before it can start speaking. This pause is required in order for the system to build a first reliable depth image which serves as reference for constructing the following ones. The ending of the recording generates a second beep. The two emitted beeps are later necessary to synchronize the audio track which is recorded on a separate system.

In this thesis two corpora were recorded. In a first attempt to learn only the degrees of freedom of the mouth, a relatively small corpus was recorded. For undertaking a triphone-based speech synthesis system, a larger corpus became necessary and a second corpus was recorded. The first corpus consists of the 11 long sentences (1933 frames, about 48 seconds) given in Appendix 16.1; the second of 116 sentences (17142 frames, about 7 minutes 8 seconds) consisting of the sentences from the first session plus a second set of long sentences and a set of short sentences (see Appendix 16.1). The second session contains also separately recorded expressions. All sequences are recorded beginning with a neutral face to facilitate the registration over the different sequences. Regarding the processing time, the reconstruction algorithm needs about 2 hours to generate 10 seconds of 3D animation, that is 9,6 hours for the first corpus and about half a week for the second. The following table gives these values:

---

[1] ABW GmbH, Siemensstraße 3, D-72636 Frickenhausen, `http://www.abw-3d.de`

	Session 1	Session 2
nb sentences	11	116
nb frames	1933	17142
animation time	48 sec	7 min 8 sec
reconstruction time	9.6 hours	3.5 days

After reconstruction, each 3D frame captured by the scanner is a depth field parameterized by the camera pixel coordinates (u, v). More specifically, we obtain 3D coordinates $\mathbf{x}(u, v)$ at discrete steps of (u, v), along with a greyscale texture and a binary array for the mask, indicating which pixel is a valid depth estimate and which is not. With this parameterization, the texture information (8 bits) reflects the actual image taken by the camera. Each pixel is associated with a 3D vertex where the *mask* (8 bits) gives it as valid (0 for valid and 1 otherwise). If the pixel is valid, the X, Y and Z coordinates are stored in 3 further arrays (32 bit floats). Hence, for a single 3D frame, the scanner delivers 5 arrays. In Figure 7.2, two 3D frame examples are shown by their texture, mask and Z arrays.

Two problems arise from this acquisition technique. First, the mask indicates regions inside the recorded area which are not valid: holes lie with the recored geometry. This problem arises because both camera and projector do not have the same alignment. The projector casts shadows onto the face that are visible by the camera and symmetrically the camera cannot capture all the lit surface as occlusion occurs. Such a scenario is typically seen around the nose: one side is invisible to the camera while the other side is shadowed by the nose. The second problem is the fact that the geometry of teeth and of tongue cannot be reconstructed correctly. Indeed, the structured-light projects lines onto the surface which are tracked during the reconstruction process. When the mouth is open, the lips cast a circular shadow around the teeth which prevents the algorithm from making the correct association between the lines inside the mouth and the ones on the skin. Also, the phase-shifting technique (Section 3.2) is best suited for capturing undulating depth variations such as deforming skin surface. When the mouth opens, the inside of the mouth appears abruptly which results in a sudden jump in depth which can be wrongly interpreted. The effect of these two problems generates extremely noisy reconstructions of the teeth and the tongue, as Figure 7.3 illustrates. Sections 7.2 and 7.3 address these two issues: holes are filled according to the surrounding geometry and the teeth and the tongue are removed in a semi-automatic framework and filled with a concave surface matching the lip contour.

An attempt to reconstruct the teeth and the tongue was made by matching the recorded model to the face morphable model from *Blanz et al.* [14]. Since this model considers full faces, it does not offer the same resolution on the area of the mouth. By converting the recorded frames to that morphable model, the

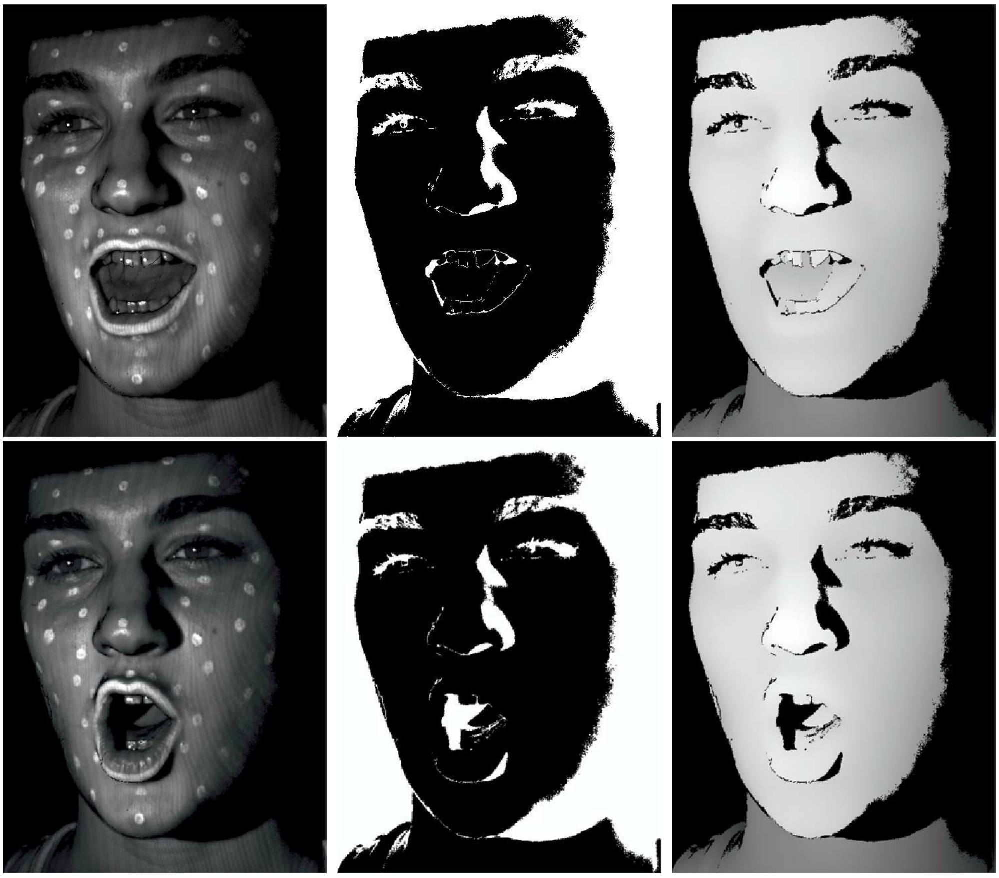


Fig. 7.2. The data structure The 3D scanner delivers a total of 5 arrays of data for each frame. This figure shows three of these arrays (columns) for two different frames (rows), namely the texture, the mask and the Z coordinates. Note that the mask gives invalid pixels inside the surface of the recorded face.

shape of the mouth cannot be reliably restituted (see Section 9.1). The path is thus not further investigated. Appendix 16.2 presents some of the obtained results.

Finally, for the recording, the subject wears 50 white marker points (Figs. 7.2 and 7.3). These markers serve two purposes. First, they help the optical flow correspondence in the registration process, particularly for low contrasted areas like the cheeks or the forehead. Secondly, before the registration, a rigid alignment of the faces is performed which is based on a subset of these markers.

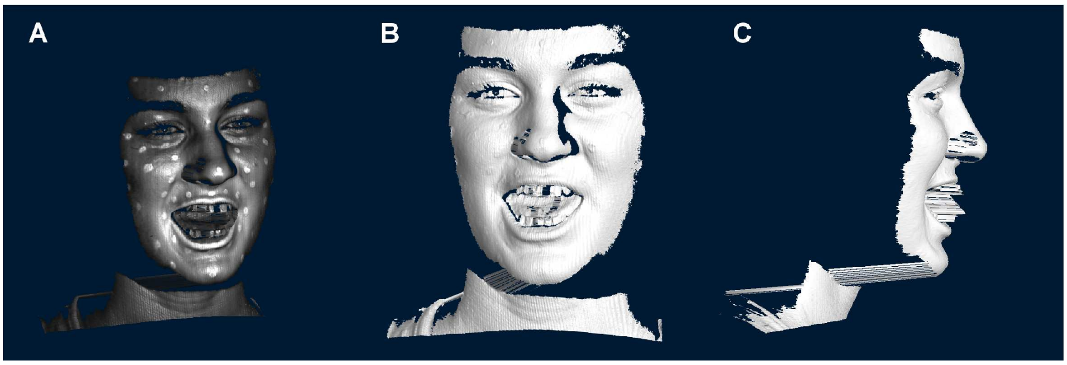


Fig. 7.3. The recorded 3D shape The face geometry delivered by the scanner presents two problems: holes are present over the recorded surface, and the geometry of both teeth and tongue cannot be accurately retrieved.

7.2 Hole Filling

Because some regions cannot be reconstructed by the dynamic scanner, the given face geometries carry holes which have to be filled according to the surrounding topology. Doing so improves the quality of the original data and is required for the statistical analysis of the face deformations.

The approach undertaken is similar to *binomial kernel smoothing* (*Marchand & Marmet*[84]). The process fills the holes linearly and the geometry is adapted iteratively in the same manner one would fill a hole with clay and smooth out the geometry with the thumbs. The algorithm first fills a hole horizontally with respect to the array representation of the data. The filling is performed through a linear interpolation between two opposite vertices relatively to the hole.

The smoothing behaves like a Binomial filter. It convolves the data with normalized coefficients with a kernel size defined by the user (9 in our case). The convolution is performed on the first derivative of the vertical and horizontal geometry curves and affects only the filled vertices. As a result, at each iteration, the flat filling is "pushed" out until it composes a smooth reconstruction (see Fig. 7.4). The filter is applied to the derivative because, in that way, the geometry adapts much faster to the topology; also, filtering around a local maximum of a function does not produce a smooth continuation which would be visible if applied directly on the surface. Using the derivative circumvents that problem.

Figure 7.4 illustrates such a local reconstruction on a sinusoidal curve. The reconstructed geometry is represented by a red curve and the derivative by a green curve. At the beginning the derivative of the linear filling is zero. Each iteration smoothes the derivative from which the geometry is reconstructed.

10 iterations are sufficient to produce decent smoothing and are performed horizontally and vertically, alternatively.

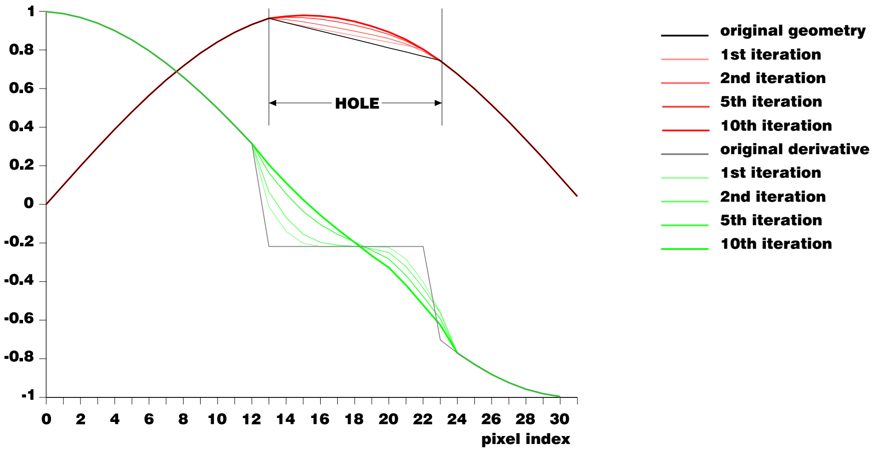


Fig. 7.4. Hole filling and smoothing Holes inside the geometry are filled and smoothed with a *binomial-kernel*-based filter. In an iterative process, the linearly filled geometry is adapted to the surrounding topology upon reconstruction of a smooth first derivative along the vertical and horizontal geometry curves with respect to the array representation of the data.

The same type of filter is used to smooth out the noise from the acquisition. Indeed, residues of the projected pattern from the structured-light based scanner produce thin undulations on the surface of the skin, also visible in Figures 6.3-D and 7.5-left. Because these undulations are direct residues of light encoding of structured-light based acquisition, they propagate along the horizontal axis in the data arrays. Hence, a horizontal smoothing is performed with a smaller kernel size (5 in our case) and two filter passes are sufficient to reduce that artifact and do not affect the overall face topology: notice, in Figures 6.3-E and 7.5-right, how the details of the eyes and eyebrows are conserved.

The following section shows how the data inside the mouth cavity is removed. After removal, the hole inside the mouth is filled and smoothed by the same process as described above. However, the topology adaptation discussed in the beginning of this section is not applied to that region so that the mouth cavity is modelled with a purely concave surface matching the inside border of the lips.

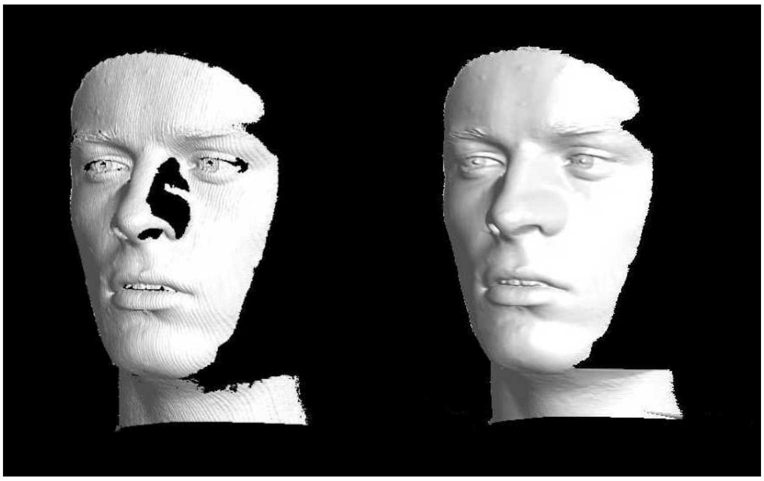


Fig. 7.5. Hole filling and scanning residues smoothing This figure illustrates how the holes in the original recorded data (left) can be filled and adapted to the surrounding topology (right). This adaptation is obtained by smoothing the filling with a binomial kernel. The same procedure is performed on the full geometry in order to remove the residual noise from the acquisition. The smoothing is able to keep the details of the eyebrows and of the eyelids.

7.3 Semi-Automatic Teeth and Tongue Removal

The geometry of the reconstructed teeth and tongue is chaotical and inconsistent from one frame to another. Hence, this region is removed and is filled with a concave surface like the one illustrated in Figure 6.3-F. Because of the size of the data, removing the inside of the mouth by hand would be too tedious and inconsistent from one frame to the other. A semi-automatic algorithm is proposed which simplifies the extraction of the mouth cavity.

The idea behind the algorithm is to detect the edges of the lips by using the information given by the texture, the depth map (Z coordinates) and the mask in order to detect and isolate small continuous portions of the surface and to distinguish them from the larger face and lips surface. The algorithm segments the recorded surface and keeps only the largest area. In order to ensure that small segments, particularly around the eyes and the eyebrows, remain intact, the user specifies in the first frame a region in the data array that encloses the mouth over the whole sequence. Restricting the area of processing also accelerates the detection process.

The approach is decomposed into two main steps. First, the *mask* and the *depth* information are used in combination to setup the data for edge detection. In a *dilate-erode* approach, the edges are then transformed into a closed form which is necessary for the segmentation. Section 7.3.1 presents the edge detection principle and Section 7.3.2 describes how the segmentation takes place.

7.3.1 Teeth and Tongue Detection

In order to be able to properly detect small regions on the recorded mesh, the mask and the depth information is used in combination. As can be seen in Figure 7.2-center and -left, the mask already holds a lot of information regarding the edges. The Z-array brings, however, a good complement: the upper teeth are well connected to the lips, hence the mask shows no invalid vertex between them. However, the depth map has the information of abrupt discontinuities on the geometry. These discontinuities are marked according to a user given threshold which, beside the restricted processing area, is the only parameter to control the detection algorithm.

Once the edges inside the mask are merged with the discontinuities of the depth information (see Fig. 7.9-top), a dilation/erosion technique (see [32, 51, 55, 58, 66, 120] for related literature) is implemented in order to connect the edges and segment the data. First, dilation thickens the edges until they connect, and then erosion makes them thinner again while keeping the connections. These two complementary techniques have to be applied carefully in order not to be degenerative (see Fig. 7.7). In the following paragraphs, both techniques are presented separately.

Dilating

Dilation works on two image arrays: the source m_s and the destination m_d. In the present case, these image arrays are defined by a rectangular region on the data to process.

A square patch of size $n \times n$ (n odd, usually $n \in \{3, 5\}$) traverses m_s; every time the center pixel of the patch falls onto an edge pixel, all the pixels from m_d that lie under the patch at the corresponding coordinates are marked as edges. This process is performed on two separate arrays so as to avoid newly marked edge pixels to be recognized as original edge pixels.

Depending on the needs and the data, the size of the patch can be adjusted and the process can be performed iteratively. Figure 7.6 (red), shows an example of such a process on a simple curve. Figure 7.7 (red), shows a similar example on a slightly smaller loop-shaped curve where the dilation process fills the loop; this particular case is to be avoided, as it leads to a degenerative erosion as described below. The size of the patch and the number of passes have thus to be carefully selected to get optimal results. In the present case, a patch of size 5×5 was used with a single pass.

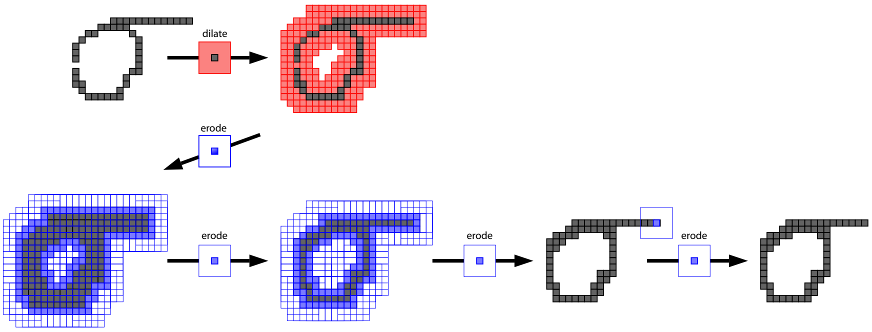


Fig. 7.6. Ideal *dilation/erosion* scenario The missing pixel to close the σ-shaped edge is retrieved by first expanding the edges with a single pass and shrink back in two passes. Note that performing more erosion passes would make the tail disappear which leads to information loss. By superimposing the original edge map with the one obtained after this process, no information loss occurs.

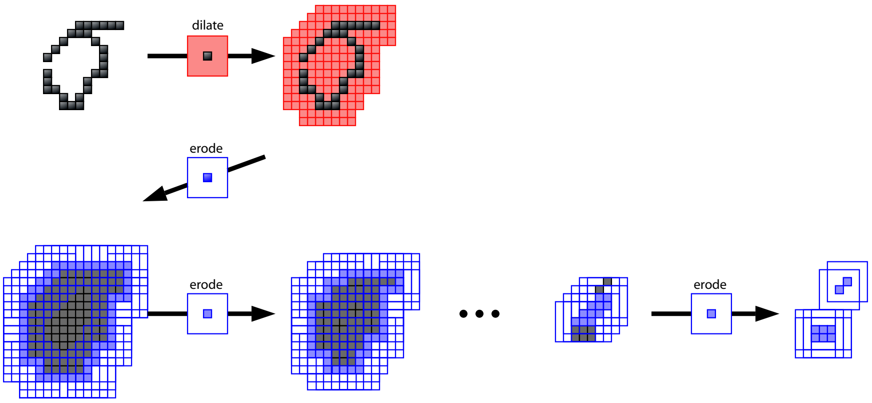


Fig. 7.7. The degenerative *dilation/erosion* scenario If a loop is filled by the dilation process, eroding will fail at retrieving the original shape of the edges and the trace disappears.

Eroding

The eroding process can be intuitively seen as the inverse procedure to dilation. The algorithm starts with the image array m_d from the dilation process and generates an eroded image array m_e. Again, a square patch of an odd pixel size traverses m_d. The most straitforward approach would be to mark a pixel as an edge in m_e whenever the patch in m_d is fully covered with edge pixels. This scenario is however degenerative for multiple passes: edges get thinner and finally disappear as the patch can no longer be fully covered (see Fig.7.7).

To prevent the algorithm from removing thin edges, a different consideration over the patch is conducted: an edge pixel in the center of the patch is reported to m_e, if inside the patch in m_d, edge pixels divide the patch into 2 or more zones (see Fig. 7.8).

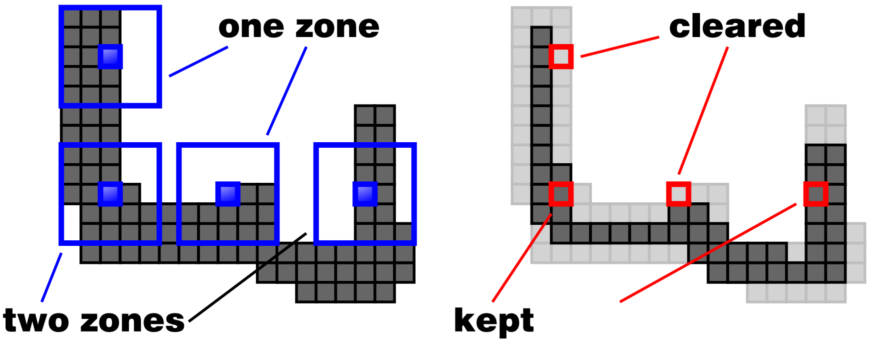


Fig. 7.8. The non-degenerative erosion scenario For the erosion process to be non-degenerative, an edge pixel is cleared only if the remaining edge pixels inside the patch fail to divide the patch in at least two distinct zones.

With a minimum size of 5 × 5 pixels, it ensures that an edge line cannot be cancelled and the edges converge to a thickness of one or two pixels (Figure 7.8). However, extremities of edges still regress. To circumvent this regression, in a final step the resulting edges are merged with m_s to recover lost edges during the dilate/erode process.

As mention before, a typical degenerative scenario is depicted in Figure 7.7. When dilation is performed with a patch of large dimension, loops get filled and after a few iterations the edge shape disappears. Still, by performing only a few eroding steps, the degenerative effect can be controlled.

As an illustration to the *dilation/erosion* process on the mouth region, Figure 7.9 shows the results for a small region on a frame of the original data. One can see how small segments of edges get connected to form a continuous edge.

7.3.2 Teeth and Tongue Removal

In the previous section, we showed how closed edges can be drawn onto the data arrays to delimit the teeth and the tongue from the rest of the face. The edges separate the surface and the different segments need yet to be identified. Only the largest segment is kept as it corresponds to the skin and the lips but the remaining segments are removed. The algorithm works in four steps:

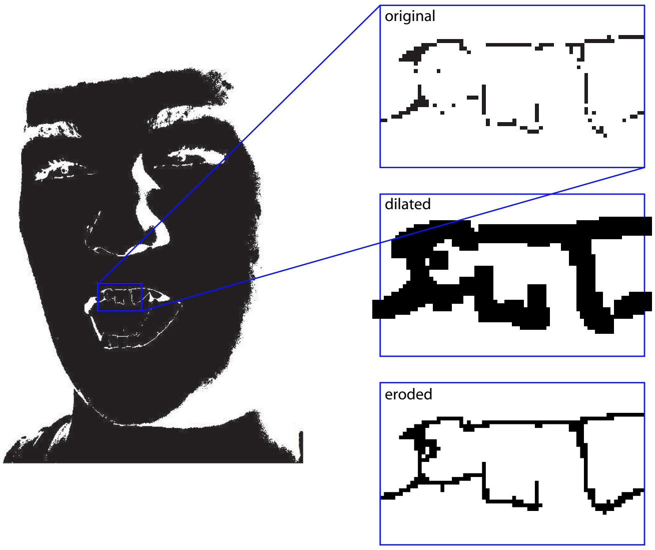


Fig. 7.9. Edge completion on face data The process of *dilating* and *eroding* is performed on the edge information obtained by the combination of the mask information and the depth information. Edges are expanded until they connect and then shrunk in a non-degenerative fashion to produce a continuous edge.

1. the expanded edge image $(m_e + m_s)$ is run line by line, attributing a unique number (ID) to continuous line segments;
2. column-wise connected line segments are merged and get a common ID;
3. for each ID associated with a segment, the number of vertices is summed up to get its area in pixels;
4. the segments with the smallest area are discarded.

In the following, these four steps are described in more detail.

1. Detect Line-Wise Neighbors

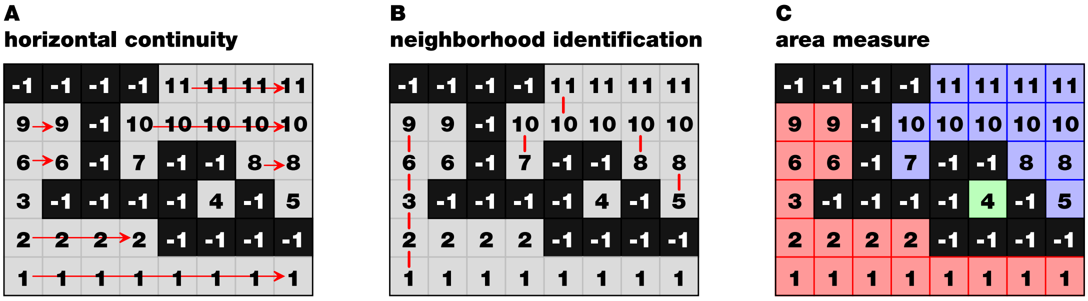
Referring to Algorithm 1 (see below), the array is traversed line by line; a counter *count* is set to 0. The counter is increased every time a new line is started or a sequence of edge pixels is crossed. Edge pixels are marked as −1. Valid pixels are marked with the current value of the counter (Fig. 7.10-A). A table *area* keeps track of the number of pixels marked for each count index. Each time a pixel is attributed to an index, the status of the neighbor pixel on the previous line is verified. If not an edge pixel, a *link* is created between the index of the current pixel and the index of this neighboring one in the lower line (Figure 7.10-B). This information is necessary to later group the indices that correspond to vertically adjacent pixels to attribute them to a common segment. Note that every index is also linked to itself; this is useful for the second step.

Algorithm 1: island removal - PHASE 1: pixel counting

```
global: : segments[$w \times h$]
local:  : pixels[$w \times h$], area[]
edge ← true;
count ← 0;
init(area, 0);
for pix←0 to w×h do
    are we on an edge pixel?;
    if is_edge_pixel(segments[pix]) then
        pixels[pix] ← -1;
        edge ← true;
        continue;
    end
    if not, do we come from an edge pixel?;
    if edge = true then
        we start with count incremented by 1;
        edge ← false;
        count++;
        pixels[pix] ← count;
        area[count]++;
    else
        the value of the pixel is the same as the preceding;
        pixels[pix] ← count;
        area[count]++;
    end
    If the neighbor pixel on the previous line is not an edge, the region
    attributed to the current count is the same as the one attributed to
    the upper pixel, and we link the two indices;
    if is_edge_pixel(pixels[pix-w]) then
        create_link(pixels[pix], pixels[pix-w]);
end
```

2. Join Connected Line Segments

The second step of the segmentation process is to identify the different regions (segments). For visual verification of the segment detection, a different colors



A
horizontal continuity

-1	-1	-1	-1	11	11	11	11
9	9	-1	10	10	10	10	10
6	6	-1	7	-1	-1	8	8
3	-1	-1	-1	-1	4	-1	5
2	2	2	2	-1	-1	-1	-1
1	1	1	1	1	1	1	1

B
neighborhood identification

-1	-1	-1	-1	11	11	11	11
9	9	-1	10	10	10	10	10
6	6	-1	7	-1	-1	8	8
3	-1	-1	-1	-1	4	-1	5
2	2	2	2	-1	-1	-1	-1
1	1	1	1	1	1	1	1

C
area measure

-1	-1	-1	-1	11	11	11	11
9	9	-1	10	10	10	10	10
6	6	-1	7	-1	-1	8	8
3	-1	-1	-1	-1	4	-1	5
2	2	2	2	-1	-1	-1	-1
1	1	1	1	1	1	1	1

Fig. 7.10. The first three steps of the segmentation algorithm Common indices are attributed to neighbor pixels horizontally (A); indices of vertically connected line segments are linked (B); linked line segments receive a common color and their area is computed (C).

is attributed to the different regions, and a color map is generated for the different indices. A common color is attributed to pixels with a common index or is they are vertically adjacent (Fig. 7.10-C). Referring to Algorithm 2, a color table of the size of *count+1* that was reached in the Algorithm 1 is created, all values initialized to -1; the value *colors*[0] holds a color counter. The list of *Links* generated by the Algorithm 1 is then traversed, and for each link the color attributed to the two connected indices (*source* and *pit*) are compared. Four scenarios can occur:

- **for both indices, no colors are attributed**: give the color value *colors*[0] to both entries in the color table and increment the color value;
- **both indices have the same color value attributed**: do nothing;
- **only one of the indices has a color value attributed**: set the color entry of the unattributed index to the value of the color for the attributed one;
- **the color value entries for the indices differ**: set the color value for all entries with the same color value as the second index in the link to the color value of the first index in the link.

Once the color table is generated, a different color is attributed to the detected segments (Figure 7.10-C). The segmentation performed over the whole face for a small sequence is illustrated in Figure 7.11.

3. & 4. Getting the Size of the Segments

Finally, the different *areas* are summed up according to whether they share the same color, and the smallest regions are discarded, i.e. they are marked as invalid in the mask array (see Algorithm 3, below).

Algorithm 2: island removal - PHASE 2: color map generation

```
alloc(colors, count+1);
init(colors, -1);
for i←0 , nbLink do
    if ( then ¬is_valid(link[i])) continue ;
    source -1, pit -1 ⇒ set both ids to a common new color;
    if colors[link[i].source] = colors[link[i].pit]and colors[link[i].source]
    == -1 then
        colors[link[i].source] ← colors[0];
        colors[link[i].pit ] ← colors[0];
        colors[0]++;
        continue;
    source A, pit A, ⇒ do nothing;
    if colors[link[i].source] = colors[link[i].pit]and colors[link[i].source]
    != -1 then
        continue ;
    source A, pit B, ⇒ set all ids with color B to color A;
    if colors[link[i].source] != -1 and colors[link[i].pit ] != -1 then
        b ← colors[link[i].pit];
        for  id←0, count do
            if colors[id]=b then  colors[id] ← colors[link[i].source];
        end
        continue ;
    source -1, pit A, ⇒ set source color to A;
    if colors[link[i].source] = -1 then
        colors[link[i].source] ← colors[link[i].pit];
        continue ;
    source A, pit -1, ⇒ set pit color to A;
    if colors[link[i].pit] == -1  then
        colors[link[i].pit] ← colors[link[i].source];
        continue;
end
```

The segmentation results are demonstrated in Figures Figures 7.11 and 7.12. The first figure illustrates the segmentation effect on the data arrays for two different frames. In the second figure, the precision and the consistency of the segmentation along the lips can be observed.

Note: in the second corpus, the white lipstick on the recorded subject was made thicker which results in a better contrast. Upon the definition of a color threshold, the edge information from the texture turned out to be richer

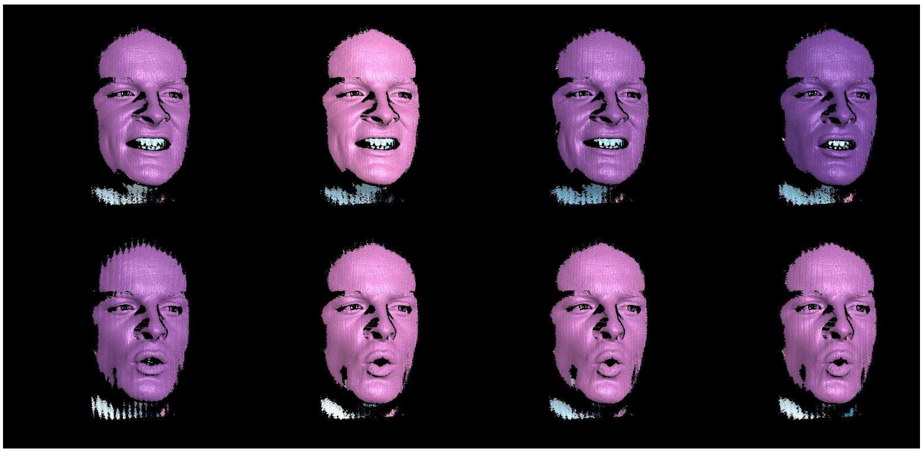


Fig. 7.11. Face segmentation for teeth detection In this animation sequence, different color are attributed to the different segments. The tone variation from one frame to the other is due to the different numbers of segments detected and is of no consequence; segments are removed according to their relative area size.

Algorithm 3: island removal - PHASE 3: summing up area and delete small ones

```
alloc(area_sum, colors[0]);
init(area_sum, 0);
sum up areas by colors;
for i←1, count+1 do  area_sum[colors[i]] + ← area[i];
for pix←0, w×h do
    if is_edge_pixel(mark[pix]) then
        if area_sum[colors[pixels[pix]]] < areathresh then
            mask[pix] ← 1;
end
```

than the one from the depth map. The data from the second session was thus processed with a combination of the texture and the mask information which gave much better results and even accelerated the mouth segmentation process.

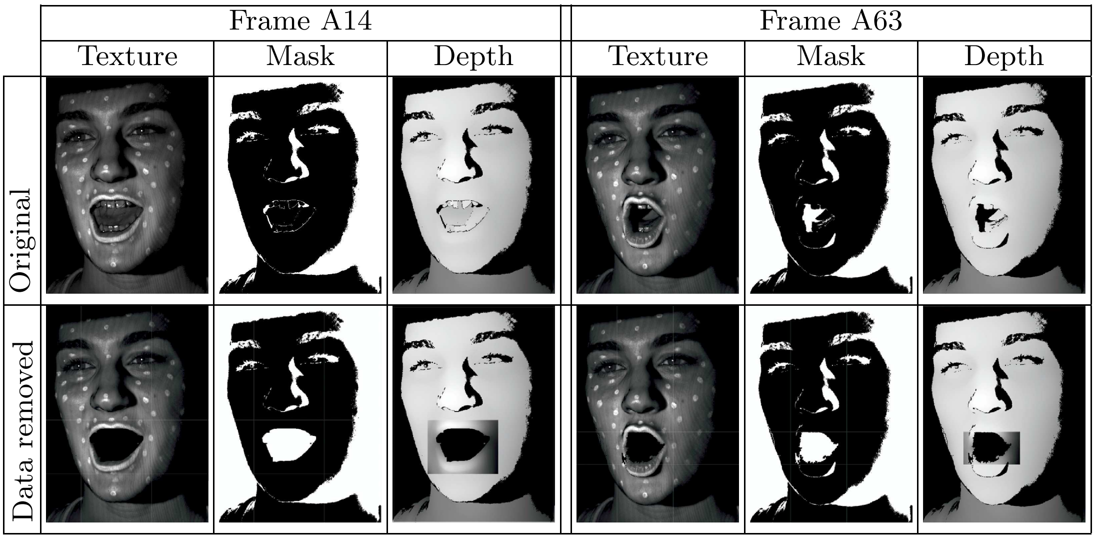


Fig. 7.12. Performance of the semi-automatic segmentation and removal of the teeth and the tongue The lower line shows the restricted area on which the algorithm removes data from texture, mask and Z coordinates arrays.

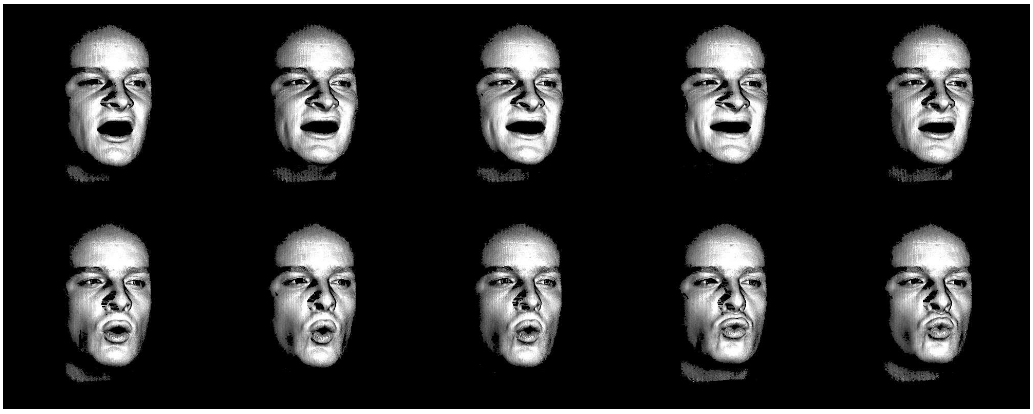


Fig. 7.13. Sample frames of the final sequence after automatic teeth and tongue removal

8

Building a Viseme-Space

In this section we show how speech information is retrieved from the recorded corpus. Up to this point, the data is stored as a sequence of independent face meshes, the number of vertices varies from one mesh to the other. The 3D meshes are first geometrically aligned to cancel head movements. In order to understand articulation, one has to study the actual deformation of the face during that process. Hence, the statistical analysis needs to focus on the deformation of the face rather than on its geometry. For that purpose, a correspondence between the 3D frames needs to be found. The registration process is based on optical flow techniques to remesh each sample in accordance with a reference frame by aligning topological and textural features of the meshes.

In a first attempt, optical flow is run on both the texture and the depth image (Z-coordinates) form the original data arrays (Section 8.2). A *pre-warping* (Section 8.1) is performed by tracking marker-points in order to improve the outcome of the optical flow. However, the registration shows imperfections along the face borders as these vary along with the head movements in the recording phase. To circumvent this border effect, the 3D meshes are projected onto a cylindrical representation (Section 8.1.1). This representation extracts a two-dimensional texture map and a depth map on which the optical flow is run. In this representation, the face geometry is flattened which solves the border problems. Moreover, the registration on the face topology turns out to be more reliable.

In the recoding sessions, a chair was provided with a metallic bar system to reduce head movement during articulation (Fig. 7.1). However, these sessions lasted several hours and several sequences were recorded. The position of the face from one sequence to the next is thus significantly different which is problematic when the registration is performed over all sequences. Also, in the second corpus, articulations are emphasized and the original optical flow implementation is not able to track the deformations accurately anymore. The optical flow is thus modified and a *multi-stage* approach is presented (Section 8.3).

Once the original meshes are correctly registered and the head movements canceled, the data is understood as a long single sequence of vertices displacements or face deformations. Because of the high dimensionality of this data, a *Principal Component Analysis* (PCA) is performed over a representative subset of the whole sequence in order to capture the most important deformations of the face (Section 8.4). This reduction gives 50 components along which the reference model can be deformed and, by combination, all original face shapes can be reconstructed with high accuracy. This PCA transformation serves also the purpose of data compression. The 50 principal components describe a low-dimensional face space onto which all registered frames can be projected and represented by a 50-coordinates point.

During the acquisition sessions, speech was recorded and synchronized with animation. This audio track is decomposed into a sequence of phonemes with their associated position in time (Section 8.4.2); hence, each 3D frame has a correspondence to a phoneme. With that information, a statistical model of visemes can be associated for each of the different phonemes from which a first animation framework is derived (Section 8.6).

Aside from speech acquisition, a set of expressions is recorded which can be represented as single deformation vectors outside the morphable model (Section 8.5). Later, when animations are synthesized, the vectors bring expressiveness to the speaking character.

From now on, the following notation is used: phonemes are written with slashes, e.g. $/AH/$, $/B/$ or $/K/$, and visemes are written with vertical bars, e.g. $|AH|$, $|B|$ or $|K|$. $/SIL/$ corresponds to the "silence phoneme".

The work presented in this part has been published in the Proceedings of Pacific Graphics 2006 [4].

8.1 Tracking the Marker-Points

About 50 white marker-points are painted with make-up on the face of the subject. These markers serve two purposes: first, they help the optical flow in performing a more reliable registration in low-contrast surfaces like the cheeks or the forehead. White lipstick also helps for a better registration but, in general, it produces a better segmentation of the lips for removing the teeth and the tongue (Section 7.3). The second purpose of the marker points is that specific markers offer precise information of the position of the head. Section 8.1.1 shows how these markers are used in order to perform a global rigid alignment of the faces over the whole sequences. This alignment is necessary for an optimum outcome of the optical flow.

For the tracking of the marker points, a template-matching algorithm is implemented. The user places square patches of 10×10 pixels over each marker. The algorithm stores the texture information inside those patches and superimposes them on the next frame. It then matches this pattern in a neighborhood of 20×20 pixels around the location in the previous frame. The matching is performed by minimizing the squared distance of the pixel color value over the neighborhood. This approach gives a precise and reliable way to track the markers over each sequence (see Fig. 8.1-A).

As low-contrast markers are more difficult to track a spring system is incorporated to the tracking algorithm: in a first run, the algorithm tracks all markers; the user then selects the erroneous patches and links them by a spring to surrounding correct markers (Fig. 8.1-B). The tracking is performed a second time. This combination of a spring system with a pattern-matching technique shows sufficient accuracy. The tracked motions are then smoothed out over each sequence to remove remaining jittering artifacts.

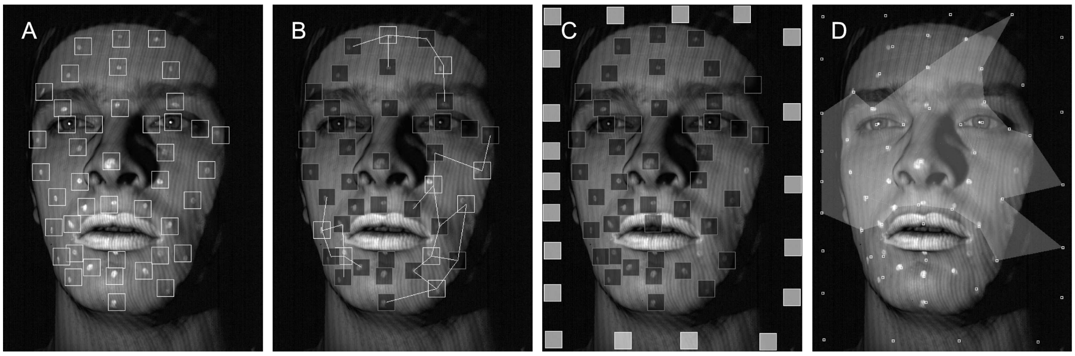


Fig. 8.1. The different steps for computing the correspondence from the first mesh of a sequence to the rest Square patches are placed over marker points. The software then tries to track all markers over the sequence (A). Patches which cannot accurately track markers are then linked to neighboring patches by a spring, so that their relative position also influences the tracking (B). Fixed points around the face are added in order to perform the pre-warp over the whole face (C). Finally, the patches are connected to form a triangle mesh (here only partially completed) which defines the triangular morphing area over the rest of the vertices (D).

For matching the sequences among them, a *reference frame* is selected from the corpus. The first frame of each sequence is then superimposed to that reference frame and corresponding marker-points are manually aligned (see Figure 8.2).

Once all marker-points are tracked and their correspondence is determined over the different sequences, the data images are morphed by triangulation so that the marker points match the locations in the data arrays of the reference frame. For that purpose, additional markers are placed along the image border which remain at constant position over the sequences (Fig. 8.1-C) and the triangle web on which the morphing performs is defined by hand (Fig. 8.1-D).

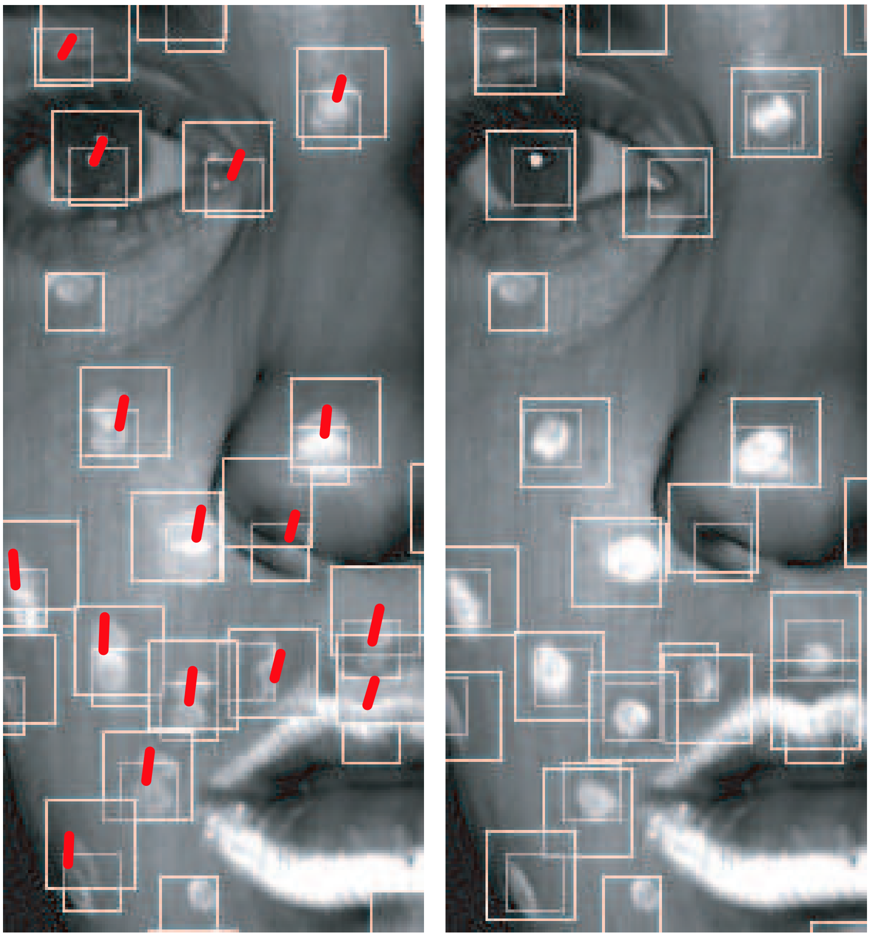


Fig. 8.2. Pre-warp initialization by matching marker points Within a stream of scans, the marker-points are tracked automatically by template-matching. To initialize the correspondence of these markers over different sequences, the first frame of each sequence is superimposed to a selected reference frame and the markers are aligned manually (red lines).

8.1.1 Rigid Head Alignment and Data Conversion

For rigid alignment, a set of marker-points is selected from the forehead, from the nose and from feature-points around the eyes, so that the movements of the jaw have no influence on the final orientation of the head. These marker points are selected on regions where the skin does not slide over the bone structure and which thus give a correct information of the orientation of the skull. Based on these marker-point coordinates, the faces are aligned in orientation and scaling via *3D-3D Absolute Orientation* (see *Haralick & Shapiro*[56]).

The rigid alignment has no impact on the data structure. In the array representation, to each pixel, 3 coordinates and a texture value are attributed. The rotation affects directly the coordinate values but does not require any resampling of the acquired data.

8.2 Optical Flow

In the data registration process, a dense point-to-point correspondence on the recorded meshes is established in order to transform the stream of independent face meshes into a stream of deformations, relative to a reference mesh. A reference head is selected with a slightly opened mouth from one of the sequences.

For the correspondence algorithm, the definition of shape and texture vectors are expressed in terms of image coordinates u, v of the scanner camera (Section 7.1). The algorithm relies both on structures in shape ($z(u,v)$) and in texture ($g(u,v)$). In shape, the depth coordinate $z(u,v)$ would vary along the cheeks towards the left and right edge of the scan, due to the curved structure of the cheeks. However, the curvature is quite uniform, and so it is more appropriate to use a quantity for shape matching that does not vary along the cheeks. Experiments have shown that curvature as defined in differential geometry does not improve the quality of the correspondence (*Blanz*[18]). On our data, the quality is best if the z-coordinate is replaced by the radius relative to the vertical axis of the head, which varies little along the cheek, and is less sensitive to the noise of the scanner than differential quantities:

$$r(u,v) = \sqrt{x^2(u,v) + y^2(u,v)}. \tag{8.1}$$

With this, we represent scans as combined arrays

$$\mathbf{I}(u,v) = \left(r(u,v), g(u,v)\right)^T . \tag{8.2}$$

In an extension of the optical flow algorithm of *Bergen & Hingorani*[11], the minimum of the cost function

$$E(\mathit{\Delta} u, \mathit{\Delta} v) = \sum_{u,v \in R} \left\| \mathit{\Delta} u \frac{\partial \mathbf{I}(u,v)}{\partial u} + \mathit{\Delta} v \frac{\partial \mathbf{I}(u,v)}{\partial v} + \mathit{\Delta} \mathbf{I} \right\|^2 , \tag{8.3}$$

$$\text{with a norm } \|\mathbf{I}\|^2 = r^2 + g^2 \tag{8.4}$$

is found in each point u, v, where R is a 5 by 5 neighborhood of u, v. This optimization is performed in a coarse-to-fine sequence on a Gaussian or a Laplacian pyramid (see blow) of $r(u, v)$ and $g(u, v)$. After each step, a relaxation algorithm smoothes the displacement field $\mathit{\Delta} u(u, v)$, $\mathit{\Delta} v(u, v)$ in regions with low contrast [18].

The displacement field $\mathit{\Delta} u(u, v)$, $\mathit{\Delta} v(u, v)$ allows to sample the surface of each scan in correspondence with the reference head, and to form shape and texture vectors by concatenating the 3D coordinates and grey values of all sampled vertices $i = 1...n$:

$$\mathbf{s} = (x_1, y_1, z_1, x_2, \ldots, x_n, y_n, z_n)^T, \tag{8.5}$$

$$\mathbf{t} = (g_1, g_2, \ldots, g_n)^T. \tag{8.6}$$

The optical flow algorithm is strongly affected by the edges of the recorded surfaces. Due to the natural head movements of the recorded model, the edges vary over time. For this purpose, after the the faces are aligned rigidly and before optical flow is computed, all raw data is cropped automatically: first, a common mask is defined as the intersection of all the individual valid surface points in the cylindrical representation. This mask is then used for cropping the scans.

Gaussian versus Laplacian Pyramid Types

As described above, in order for the registration process to generate a consistent optical flow on regions with low contrast, a pyramidal sequence of down-sampling steps is performed. This down-sampling follows a Gaussian smoothing by weighting neighbor pixels values into the down-sampling. The Gaussian smoothing performs a low-pass filtering on the intensity maps. An alternative is to use a Laplacian pyramid which performs a high-pass filtering. The Laplacian down-sampling is directly obtained by subtracting the low-pass filtered version of a map to the original map.

For computing the correspondence of faces of different individuals *Blanz*[14] shows that a Laplacian pyramid performs better; indeed, the Laplacian pyramid highlights sharp features of the face, as high-pass filtering performs an edge detection over the shape and texture vectors. The variation of skin color, for instance, is thus ignored. In the present case, the correspondence is performed on the same individual with varying expressions. The use of a Gaussian pyramid shows to produce slightly more robust tracking of the deformations.

Two further parameters are involved in the optical flow process. The computed flow is a combination of both the flows computed from both the shape vectors and the texture vectors. With the attribution of respective weight coefficients w_r and w_g to these flows, their respective influence in the final flow is controlled; however, best correspondence is found by attributing equal weights with respect to the typical varaition (amplitude) found in r and g respectively. Finally, the depth of the pyramidal process can be chosen. Increasing the depth improves the robustness of the flows but induces additional processing time. For the small corpus, a single down-sampling iteration shows to be sufficient. For the second corpus which contains larger deformations, 3 iterations are necessary.

8.3 Multistage Optical Flow

The optical flow process presented in Section 8.2 performs an *absolute* matching: a reference frame is selected from which a flow is computed against all other frames. A *relative* approach computes the flow between successive frames, the matching from any frame to the reference frame is then obtained by the inverse composition of the flows. The relative approach has the advantage that it can easier track deformations over long sequences, as these are small from one frame to the next. However, while the composition of several relative flows over long sequences does not imply an important increase of computation, it induces problematic accumulating errors (measurement and rounding errors).

While the absolute approach is sufficient for tracking deformations in the first corpus, it fails for the second one in which articulations are more pronounced. In his thesis, *Blanz*[14] proposes to group similar images in batches and to compute absolute correspondence within each of these. In a bootstrapping framework, these batches are then connected involving user interaction. This approach works well, but is only well suited for small corpora of frames. In the present case, it is difficult to define a distance threshold to separate the data into separate clusters.

The problem of computing reliable optical flow correspondence over the corpus is multifold:

1. the frame sequences are too large for a relative matching approach;
2. absolute matching is too weak to grasp large mouth deformations;
3. face positions are different from one sequence to another (there are 116 sequences);

The method proposed in the next section takes advantage of the independence on the data scale of absolute matching while retaining the flexibility of relative matching. Also, it requires little user interaction for computing precise flows over several large sequences.

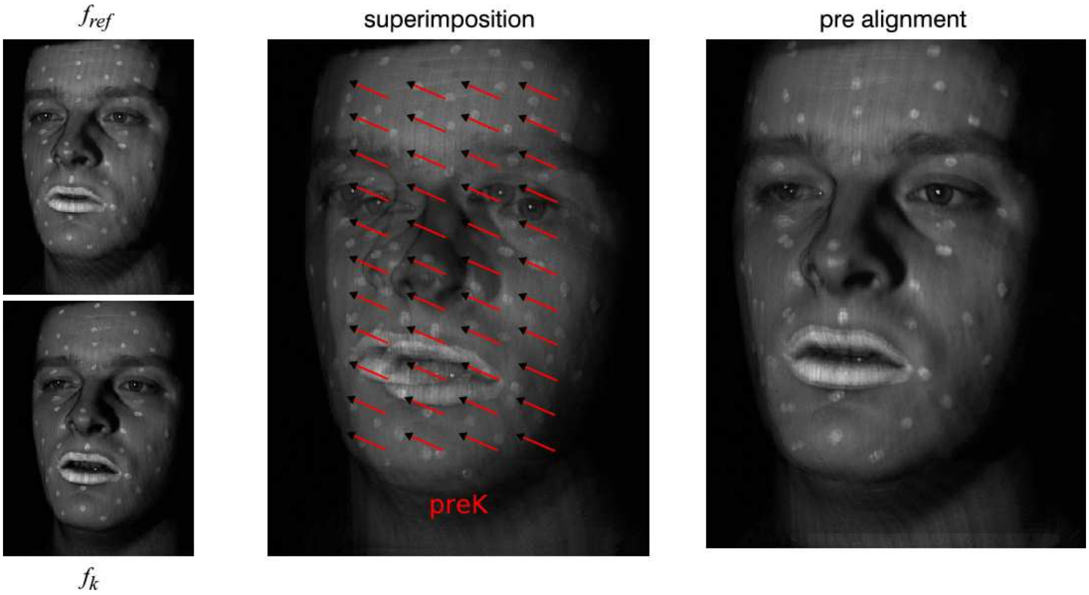


Fig. 8.3. Pre-alignment of different 3D sequences The head position varies significantly among the different sequences which demands an adaptation step for the optical flow in order to find good correspondence between them. First, a neutral reference face is chosen from the first sequence, and from each of the remaining sequences, a similar sample is selected. These samples are then superimposed on the reference face and manually pre-aligned. The displacement vector serves as a constant flow direction which provides a first estimate for the deformation.

An absolute flow matching any frame f_i to the reference frame f_{ref} can be found by induction (see next section). For any frame f_i, the absolute flow $F_{i \to ref}$ has to be found which matches f_i to the reference frame f_{ref}. If $F_{i \to ref}$ is known, the absolute flow matching the next frame f_{i+1} to f_{ref} can be approximated by the composition of $F_{i \to ref}$ with the relative flow $F_{(i+1) \to i}$. The approximation is then corrected in a second pass. The absolute flow matching f_{ref+1} to f_{ref} is the relative flow $F_{1 \to 0}$.

The problem for connecting the different sequences is solved as follows: for the reference frame f_{ref}, a face configuration is selected which is the most likely to occur in each sequence: mouth slightly open, neutral expression. The user selects from each sequence k the most similar frame f_k. Each f_k is then visually superimposed on f_{ref} (see Fig. 8.3) and the user performs a *pre-alignment* of the faces. From the displacement given by the alignment a constant flow F_{preK} is deduced which is combined with the flow approximation discussed above. The next section presents the iterative approach in more details.

8.3.1 The Iterative Optical Flow Algorithm

The *multi-stage* optical flow proposes a *sequential pre-warping* that improves optical flow registration by using the *temporal coherence* of the data; an iterative algorithm (Algorithm 4, below) takes for each frame the absolute flow computed for the previous frame, and pre-warps the current frame with it. The pre-warp is thus much closer to the reference frame and the absolute flow can be computed.

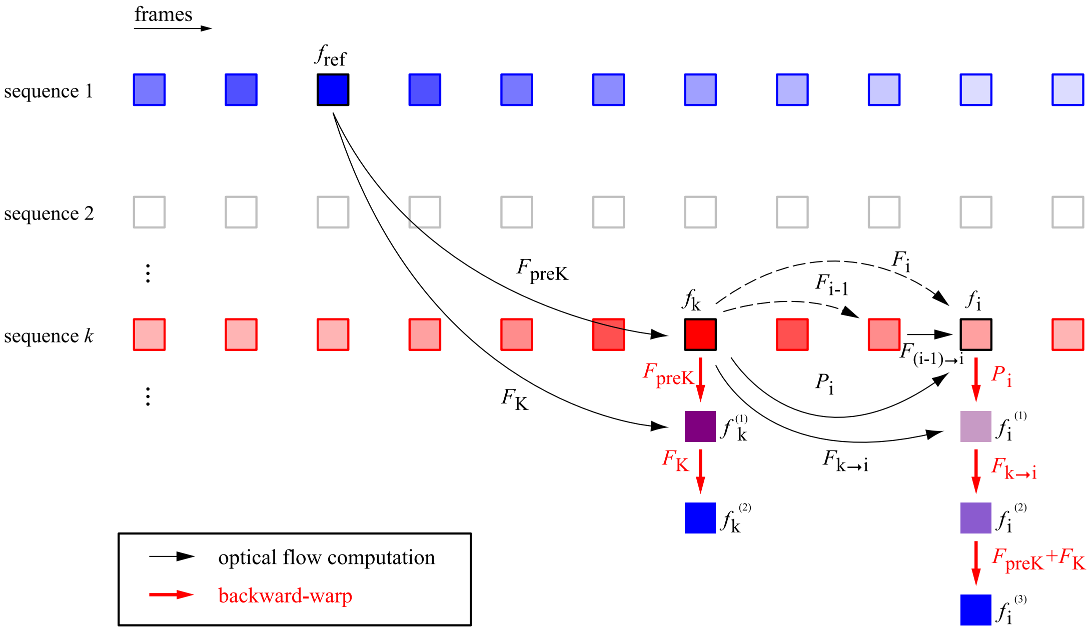


Fig. 8.4. The multistage optical flow The different flows computed in the *multi-stage* procedure. Absolute and relative warping are combined in order to track important face deformations over long and disconnected sequences. Black arrows show the computed flow between two frames which are then applied for back-warping (red arrows). Each frame f_i is first warped to f_k, the local reference frame of a sequence, before it is finally warped to f_{ref} the global reference frame.

The difficulty in matching sequences with substantial face variations of face orientations and carrying important mouth deformations to a single reference frame resides in three aspects:

1. the head position varies from one sequence to the other; hence, two frames with similar mouth shape have a different information distribution inside the data array structure;
2. large mouth deformations cannot be tracked with the original optical flow approach;
3. a relative frame-to-frame flow approach introduces cummulated rounding errors.

The multi-stage algorithm addresses these three aspects:

1. a global constant flow is generated for each sequence as a raw alignment to the reference frame (see Fig. 8.3);
2. a relative flow is used in order to track large mouth deformations;
3. an approximation is used that combines the advantages of an absolute (two steps) flow to the reference frame while using the temporal coherence of successive frames.

Figure 8.4 illustrates the algorithm: from the first sequence, a reference frame f_{ref} is chosen with a slightly open mouth. This shape is preferred to a closed mouth shape as the mouth cavity is already visible. Also, it is preferable to select a shape where the teeth are not visible, in order to avoid mis-registration between the teeth and the lips for frames where teeth are not visible. A sample frame f_k from the $K = 116$ sequences is then selected which is visually similar to the reference.

To compute the flow for matching f_k to f_{ref}, we proceed in two steps: first, a hand-generated constant flow F_{preK} is used that describes a translation in the data array (see Fig. 8.3) and back-warps f_k to $f_k^{(1)}$. $f_k^{(1)}$ is now a first approximation of f_{ref} and we compute the flow F_K matching $f_k^{(1)}$ to f_{ref}, the product of which is $f_k^{(2)}$.

Note that for the local reference frame f_k, two warping processes are necessary to compute F_K. Nevertheless, even though warping processes are not additive in general[1], one is constant here (and we are neglecting border effects), so the previous warping processes can be performed in a single pass by taking the warping function $F_{preK} + F_K$. Thus, f_k is warped to $f_k^{(2)}$ and we obtain the correspondence of f_k to f_{ref}.

For the remaining frames of the sequences a flow $F_{k \to i}$ is sought that first warps frame f_i to f_k; this is however difficult if f_i differs too much from f_k. The temporal coherence (strong similarity of subsequent frames) comes here in handy: a flow $F_{(i-1) \to i}$ is computed from frame f_{i-1} to f_i, and by combining all relative flows in F_i, we can reach back to f_k from any frame. However, the

---

[1] for two flows R and S and an image a, $(S \circ R)(a) \not\equiv (R + S)(a)$

farther f_i gets from f_k in the sequence, the more error is cummulated in F_i. To circumvent this problem the following approximation is used:

$$F_i \sim P_i = F_{i-1} + F_{(i-1)\to i}. \tag{8.7}$$

P_i is an approximation because of the addition performed to circumvent flow combinations. This approximation allows a direct warping of f_i to f_k. With P_i, f_i is warped to $f_i^{(1)}$. To correct the error introduced by the approximation, a new flow $F_{k\to i}$ is computed to match $f_i^{(1)}$ to f_k as $f_i^{(2)}$.

With the condition

$$F_k = \mathbf{0} \tag{8.8}$$

we can compute

$$\begin{aligned} P_1 &= F_0 + F_{0\to 1} && (8.9)\\ F_1 &= F_{k\to 1} \circ P_1 && (8.10)\\ &\vdots \\ P_i &= F_{i-1} + F_{(i-1)\to i} \\ F_i &= F_{k\to i} \circ P_i \end{aligned}$$

Back-warping $f_i^{(1)}$ with $F_{k\to i}$ gives $f_i^{(2)}$, the correspondence of f_i to f_k. Finally $f_i^{(2)}$ is warped to f_{ref} as $f_i^{(3)}$. The final algorithm is given in Algorithm 4.

The successive iteration of the approximation given in Eq. 8.7 generates local rotational errors along the edges of the mesh. This is corrected by slightly smoothing $f_i^{(1)}$ with a Gaussian blur. The smoothing cancels the error and the relaxation step from the original optical flow (Section 4.1) reproduces consistent warping when computing $F_{k\to i}$.

Figure 8.5 presents and discusses the results obtained by the multistage optical flow over a set of selected frames.

8.4 A Morphable Model for Articulation

The core of the learning-based approach is to find statistical correlations in the training dataset that distinguish natural motion from random noise. The morphable model approach (see Section 5.4) captures such correlations automatically. The goal in our data analysis is to define and analyze clusters of visemes. This is done in a low-dimensional representation which is obtained by a *principal component analysis* (PCA - Section 5.3).

PCA finds the directions of the greatest variance of the vectors, and provides an orthogonal basis of the linear span of examples. With this basis, every

Algorithm 4: multistage optical flow on a sequence k

```
global: : $F_{preK}$, $F_K$
local:  : $F_i$
$f_{ref} \leftarrow$ load_ref_frame();
init($F_i$, 0);
Frames are equalized due to luminosity variations in the scans;
get_equalize($f_{ref}$);
$f_k \leftarrow$ load_frame($k$);
equalize($f_k$);
we omit the equalization step in the following;
The loop has to be performed a second time for $i \leftarrow k-1$ to 0;
for $i \leftarrow k+1$ to nbFrames do
    $f_{i-1} \leftarrow$ load_frame($i-1$);
    $f_i \leftarrow$ load_frame($i$);
    $F_{(i-1)\rightarrow i} \leftarrow$ Lucas-Kanade($f_{i-1}$, $f_i$);
    $P_i \leftarrow F_i + F_{(i-1)\rightarrow i}$;
    $f_i^{(1)} \leftarrow$ Bilinear-warp ($P_i$, $f_i$);
    $F_{k\rightarrow i} \leftarrow$ Lucas-Kanade( $f_k$, $f_i^{(1)}$);
    $f_i^{(2)} \leftarrow$ Bilinear-warp ($F_{k\rightarrow i}$, $f_i^{(1)}$);
    $f_i^{(3)} \leftarrow$ Bilinear-warp ($F_{preK} + F_K$, $f_i^{(2)}$);
    $F_i \leftarrow F_{(i-1)\rightarrow i)} \circ P_i$;
end
```

face can be transformed into a low-dimensional vector. For numerical reasons, the PCA is calculated only on about 200 frames, that is, for the corpus every tenth frame was taken, for the second every 100th. From the PCA the first 50 PCs are retained each time. It turns out that for mouth movements, the most important deformations are already defined by the 10 to 15 first components, the next ones are necessary for bringing realism to the movement, whereas after the 50th component, mainly measurement noise is reflected. On a set, PCA is performed once on the shape information and once on the texture, though the latter is not necessary as the texture does not change over time. However, the average texture given by the PCA is taken for animations, as the remaining artefacts from the scanner (vertical lines) are cleared out. Moreover, mapping the texture of the average face onto the rest of the data is a good way to verify the quality of the registration by testing whether the texture of the eye or of the lips always fit to the topology of the shape.

By applying a PCA on the whole shape of the face model, eye movements are also captured and bias the generated components which should only reflect mouth deformations. The following section addresses that problem.

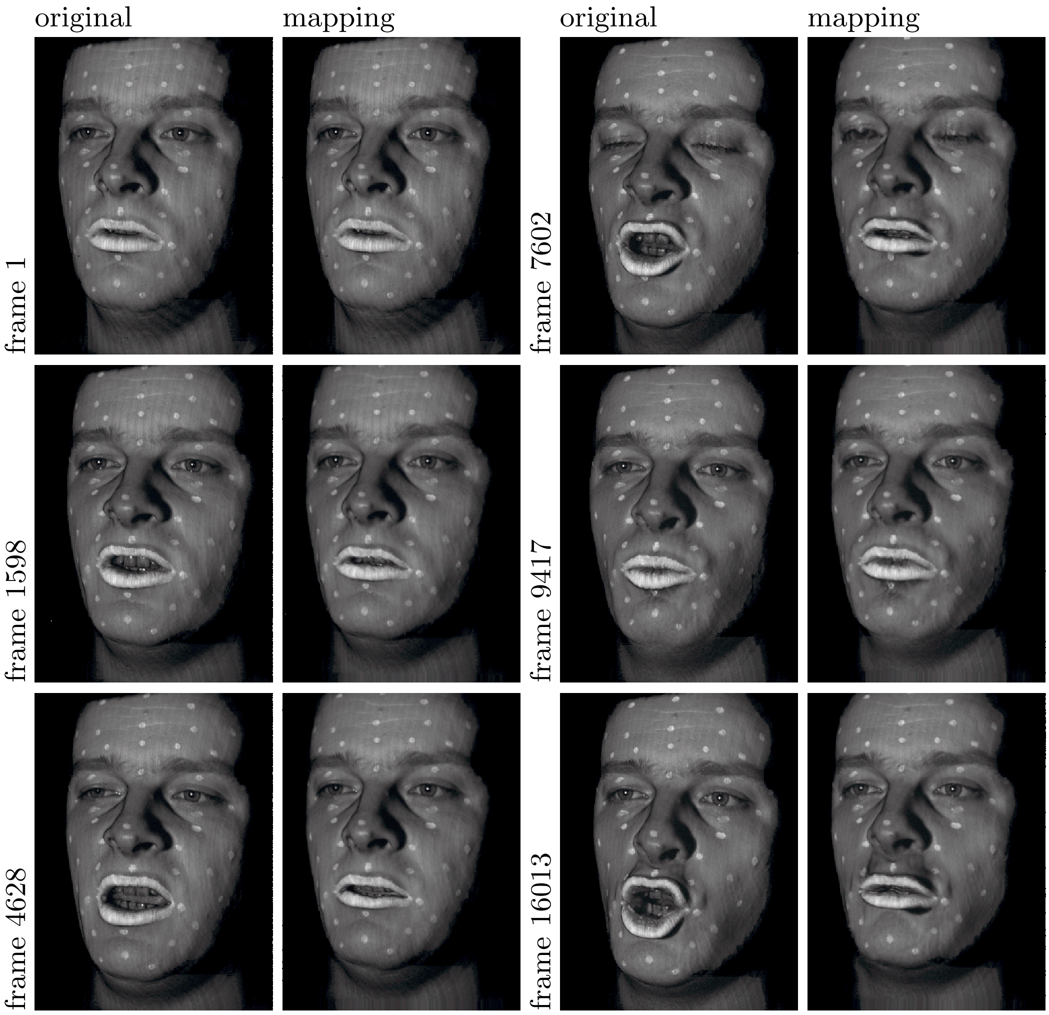


Fig. 8.5. The Optical Flow Mapping of Faces to the Reference Face The multistage optical flow algorithm matches all frames from multiple and large sequences to the reference frame (frame 1). Columns *original* show the face as it is recorded with each pixel associated to a vertex. To get the correct vertex associations with the reference frame, the original frames are morphed to the reference frame with the flow found by the algorithm. The *mapping* columns show the result of the morphing. Ideally, all frames in the *mapping* columns look alike. Note how the face orientations vary from one frame to the other, and the importance of the mouth deformations. The algorithm finds a correct matching for every frame. Frame 7602: when the eyes are completely closed, the algorithm cannot find a correct mapping.

8.4.1 Weighted PCA (WPCA)

In the principal components at this point, there is a significant contribution of the movement of the eyes because the actors blinked during the recording process. To solve this problem, a modified version of PCA is developed which weights regions on the surface differently according to their relevance as to the phenomenon that has to be analyzed. This *Weighted PCA* (WPCA) is performed only on a chosen segment of the surface, but still produces principal component vectors that apply to the entire face. Unlike a trivial version that would set the displacements of the eye region to zero in all frames, this method retains all degrees of freedom of the morphable model, and retains correlations across face regions that may be present in the data (see Figure 8.6).

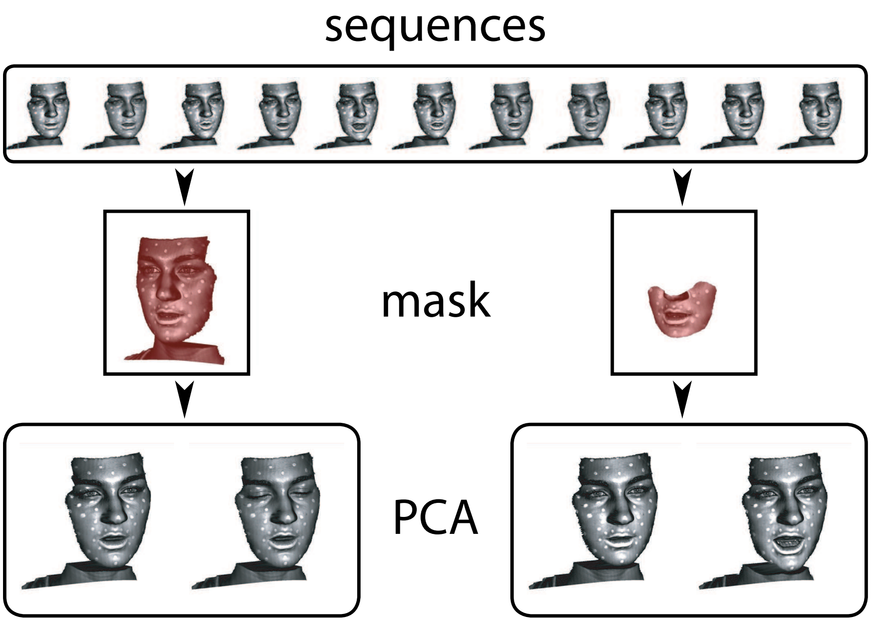


Fig. 8.6. Weighted PCA computation When the *Principal Component Analysis* (PCA) is computed over the whole face, important components may be biased by eye-blinking. To circumvent this, a *wheigted* approach is developed which produces principal components on the whole face but considers only the variance of the mouth. Note that if the eye movements were correlated with the movements of the mouth, this method would have retained it automatically.

Throughout the rest of this thesis, shape vectors are considered after subtracting their average, $\mathbf{x} = \mathbf{s} - \bar{\mathbf{s}}$. In order to analyze the variance on the mouth, each data vector $\mathbf{x}_i$, $i = 1...m$, is transformed to a reduced vector $\tilde{\mathbf{x}}_i$ that contains only the vertex coordinates of the mouth region. Combining these to a data matrix $\tilde{\mathbf{X}}$, the diagonalization of the covariance matrix can be computed

$$\tilde{\mathbf{C}} = \frac{1}{m}\tilde{\mathbf{X}}\tilde{\mathbf{X}}^T = \frac{1}{m}\tilde{\mathbf{U}}\tilde{\mathbf{W}}^2\tilde{\mathbf{U}}^T \tag{8.11}$$

by a Singular Value Decomposition of the data matrix: $\tilde{\mathbf{X}} = \tilde{\mathbf{U}}\tilde{\mathbf{W}}\tilde{\mathbf{V}}^T$. The principal components $\tilde{\mathbf{u}}_i$ are the columns of the matrix $\tilde{\mathbf{U}}$, and these can be rewritten in terms of the data vectors, using the matrices $\tilde{\mathbf{V}}$ and $\tilde{\mathbf{W}}$:

$$\tilde{\mathbf{u}}_i = \sum_j \frac{1}{\tilde{w}_i} \tilde{v}_{ji} \tilde{\mathbf{x}}_j. \tag{8.12}$$

With the same linear weights, the principal components are mapped to the original vector space:

$$\mathbf{u}_i = \sum_j \frac{1}{\tilde{w}_i} \tilde{v}_{ji} \mathbf{x}_j. \tag{8.13}$$

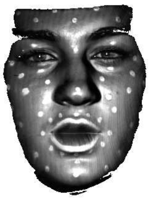
Unlike $\tilde{\mathbf{u}}_i$, the vectors $\mathbf{u}_i$ do not form an orthogonal basis, and they differ from a PCA computed on the original data directly, even though they span the same subspace.

Figures 8.7 and 8.8 illustrate for both corpora the average head or the origin of the PCA-space and the deformations along the first four PCs. The deformations are amplified by a factor 3, so that the consistency can be reviewed and the deformations better visualized.

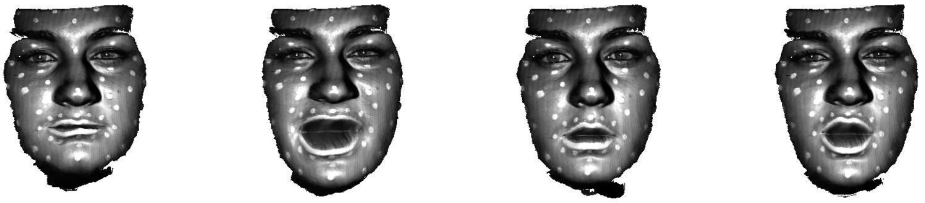
To get the representation in the viseme-space, all recorded faces are projected onto the basis defined by the 50 first PCs (Section 8.4.3). By doing so, the projection of all recorded sequences produce a cloud of points around the origin of the viseme-space. Clusters of visemes can be defined according to the phoneme they are associated to, i.e., the phoneme that was uttered during each of the frames. By synchronizing the sequences with the recorded audio (Section 8.4.2), all frames can be labelled and the clusters can be drawn (Figure 10.1 shows such a representation along the first three components).

8.4.2 Sound Synchronization

To associate audio speech to articulation movements, audio is recorded along with the 3D data. As described in Section 7.1, when the scanner initiates the acquisition, the computer generates a beep tone which is captured on the same audio track as the speech. The same happens when the acquisition terminates. These audio signals are important, as the audio recording is initiated before the actual 3D recording, and they allow the alignment of the 3D sequences with the audio track. The length of a recorded sequence is of about 10 seconds but we only want to retain the sequence where the actual articulation takes place. With the sound aligned, frames containing speech information are retained interactively (see Fig. 8.9) by cropping out the sequences where no movement is performed. Our interface allows a visual exploration of the data



Average head



PC1 $(-3\sigma^2, 3\sigma^2)$ PC2 $(-3\sigma^2, 3\sigma^2)$ PC3 $(-3\sigma^2, 3\sigma^2)$ PC4 $(-3\sigma^2, 3\sigma^2)$

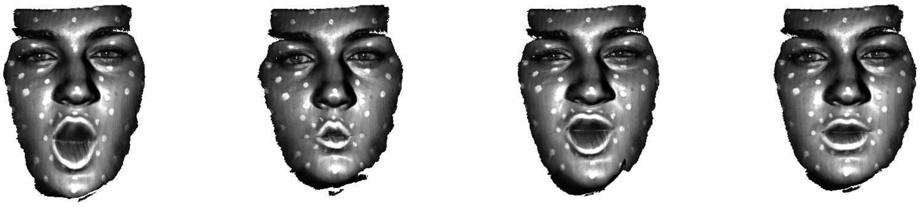
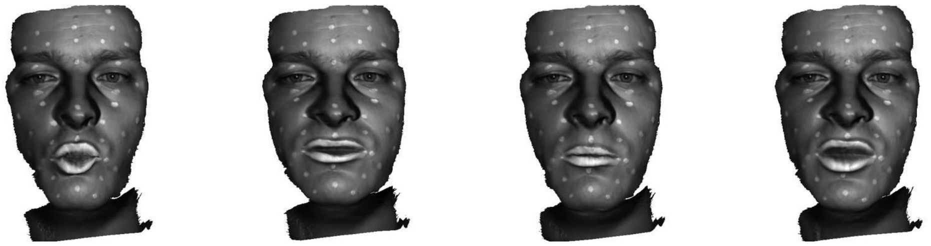


Fig. 8.7. PCA for the small corpus The top picture shows the average head. The four columns then show the result of shifting the average face by +3 or -3 standard deviations along the first four principal components.

by showing the audio curve along with the animation. By moving the mouse over the audio curve, the animation window is updated with the corresponding frame which allows a precise trimming of the data of interest, from where the mouth starts to open before the phoneme is actually heard to when it finally closes.

In order to identify visemes in the PCA-space, each frame has to be labelled by the phoneme that was uttered when it was recorded. To make the phoneme association with the recorded frames, the phonemes from the audio track are detected and a textual representation of the phoneme sequences along with their time location is required. The *Sphinx Group* [106] at Carnegie Mellon University has developed a web-based client for speech recognition that offers such a speech-to-phoneme time decomposition. A piece of software is installed on the local computer which decomposes the audio based on a dictionary that is obtained from the web interface. The generated data comes in the form of

Average head



PC1 ($-3\sigma^2$, $3\sigma^2$) PC2 ($-3\sigma^2$, $3\sigma^2$) PC3 ($-3\sigma^2$, $3\sigma^2$) PC4 ($-3\sigma^2$, $3\sigma^2$)

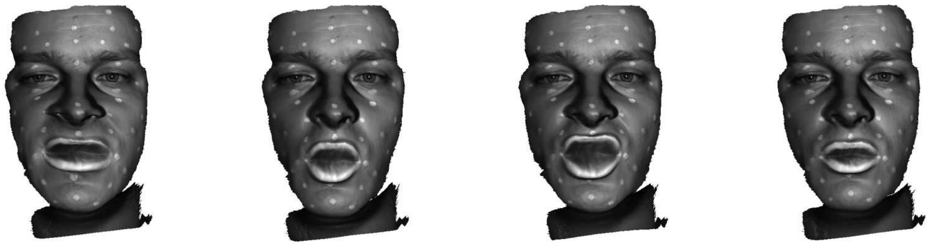
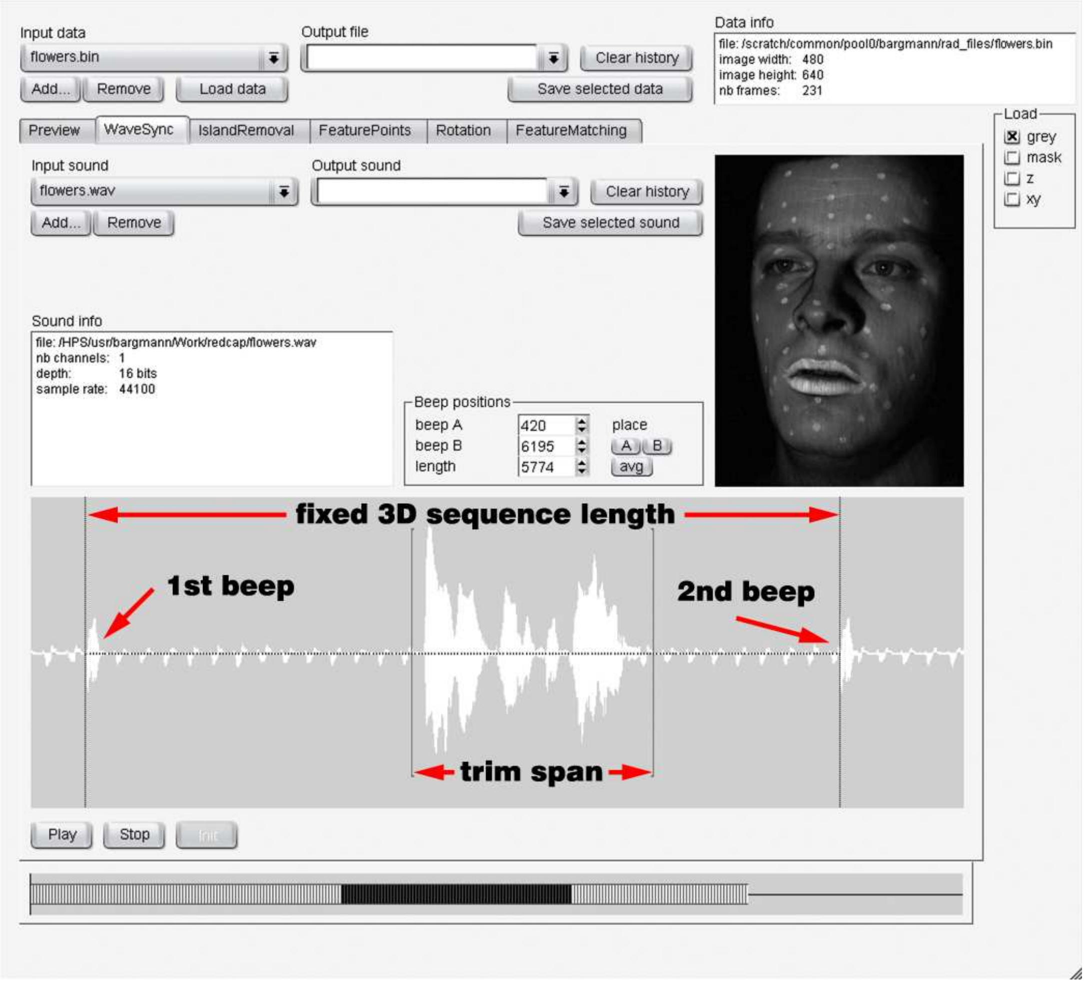


Fig. 8.8. PCA for the large corpus The top picture shows the average head. The four columns then column show the result of shifting the average face by +3 or -3 standard deviations along the first four principal components

a list of phonemes, each associated with a time duration interval (beginning and ending) as 0.01 sec.

The time unit is converted to the proper frame rate of the recorded corpus (40Hz), which shows that phonemes span between two to eight consecutive frames. This frame span is important, as it reflects one part of the coarticulation information: the transition movement from one viseme to the other. Hence, longer spans hold more information. However, the second half of the coarticulation information, namely the viseme itself or the mouth shape within its context, is available within every frame. In the rest of this thesis, visemes are considered as the first shape of such a span. This is discussed later in Chapter 10.



Input data: flowers.bin

Data info
file: /scratch/common/pool0/bargmann/rad_files/flowers.bin
image width: 480
image height: 640
nb frames: 231

Load
☒ grey
☐ mask
☐ z
☐ xy

Input sound: flowers.wav

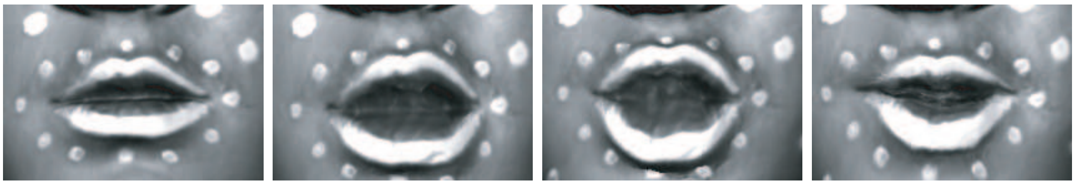
Sound info
file: /HPS/usr/bargmann/Work/redcap/flowers.wav
nb channels: 1
depth: 16 bits
sample rate: 44100

Beep positions
beep A: 420
beep B: 6195
length: 5774

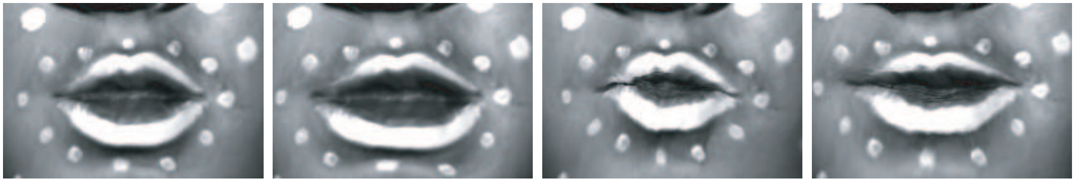
Fig. 8.9. Our interface for sound synchronization The original 3D sequence is aligned according to the the beep tones which insures an exact matching with the audio track. The user then selects the subsequence of frames to be retained (bottom panel): the waveform shows the position of the speech information, the visualization window is updated as the user moves the cursors so that these can be accurately placed before the start and after the end of the articulation movements. The lower panel highlights in black the frames which are retained.

8.4.3 Projecting the Sequences

This section presents the last step in building the viseme-space. Each 3D frame is labelled with its associated phoneme and is projected onto the basis of the viseme-space constructed in Section 8.4.1. While the projection is performed following the Gram-Schmidt projection algorithm, one has to recall that the basis was computed on a sub-region of the face shape, and that the basis of the viseme-space is not orthogonal. The projection has thus to be performed



silence d**A**rk (*/AA/*) cat**CH** (*/CH/*) li**F**t (*/F/*)



chi**CK**en (*/K/*) **S**erve (*/S/*) y**OU** (*/UW/*) do**V**e (*/V/*)

Fig. 8.10. A subset of visemes The audio track allows to associate each 3D frame to a phoneme. Here, the average shape is shown for a set of phonemes. Note that the white lipstick is only drawn on the outer edge of the lips and can give the impression that the mouth is not completely closed even when it actually is.

along the considered mouth region. Hence, the projection of a face vector $\mathbf{x}$ is achieved by retrieving the reduced vector $\tilde{\mathbf{x}}$, and calculating the scalar product

$$a_k = \tilde{\mathbf{u}}_k \cdot \tilde{\mathbf{x}} \tag{8.14}$$

Then,

$$\mathbf{x} = \sum_k a_k \mathbf{u}_k \tag{8.15}$$

By projecting the whole corpus onto the basis, the positions for all the faces in the space are obtained. A table is built in which each phoneme is listed along with its different corresponding frame numbers; each list then describes a cluster of visemes for a specific phoneme[2]. A first approach to generate new animations is proposed in Section 8.6 which takes a single viseme shape for

---

[2] We obtained 44 different visemes corresponding to the phonemes */AA/* (48), */AE/* (45), */AH/* (62), */AO/* (34), */AW/* (21), */AX/* (155), */AXR/* (31), */AY/* (38), */B/* (38), */CH/* (17), */D/* (114), */DH/* (71), */DX/* (8), */EH/* (63), */ER/* (26), */EY/* (60), */F/* (41), */G/* (30), */HH/* (41), */IH/* (92), */IX/* (66), */IY/* (104), */JH/* (21), */K/* (89), */L/* (85), */M/* (68), */N/* (174), */NG/* (25), */OW/* (46), */OY/* (9), */P/* (56), */R/* (114), */S/* (98), */SH/* (14), */T/* (196), */TH/* (18), */UH/* (20), */UW/* (43), */V/* (38), */W/* (62), */Y/* (23), */Z/* (89), */ZH/* (1) and */SIL/* (559), for the silent viseme. The numbers in parenthesis indicate their occurrences in the corpus.

each phoneme. These visemes are taken as the average face inside each cluster. Figure 8.10 illustrates the average viseme in some of the clusters for the first corpus.

8.5 Expression Vectors

The registration of the data gives a dense point-to-point matching over the recorded sequences. By selecting a reference frame and subtracting it to the other frames, the shape sequence becomes a deformation sequence which dictates how the reference shape varies over the time.

The PCA generates the viseme-space basis by successively selecting the variance of these deformations. By combining the deformations along the generated components, each of the shapes can be accurately reproduced. Because the data on which the viseme-space is built does not include face expressions they are hardly available in that context; this means that recorded face expressions cannot be projected into the viseme-space.

During the recoding of the second corpus, a separate set of faces was acquired which represents typical face expressions such as *happiness*, *disgust*, *fear*, or *sadness*. While these shapes are not included in the PCA, they can be registered to match the reference face of the speech sequences. Hence, by subtracting the reference frame, a displacement vector field is deduced for each of these expressions which can be added to any 3D frame and affect the expression during the speech. In order to be able to register the expression shapes which have strong dissimilarities with the reference frame, the motion from a neutral face to the desired expression was recorded. The registration of the most expressive frame is then obtained by the method proposed in Section 8.3.

Because of their nature, the expression shapes are dissociated from the articulation morphable model, they are referred to as deformation vectors or *expression vectors*. Figure 8.11 illustrates three of the four recorded expressions; the *disgust* expression was left out as it is similar to *sadness*.

These expression vectors are used atop the speech synthesis process. This means, when a new articulation animation is created, the vectors are progressively applied onto the generated 3D frames, thereby adding realism and expressiveness to the animation. However, this approach to face expression does not compete with expression transfer frameworks like *Spacetime faces* by *Zhang et al.* [141] which learn expressions and their evolution over time. Indeed, natural expression evolves; the expression of the eyes for instance appears usually before the mouth expression.

There are two ways the expression vectors can be applied to animation. The first one is to use the expression frame as a fixed face shape (a viseme). A

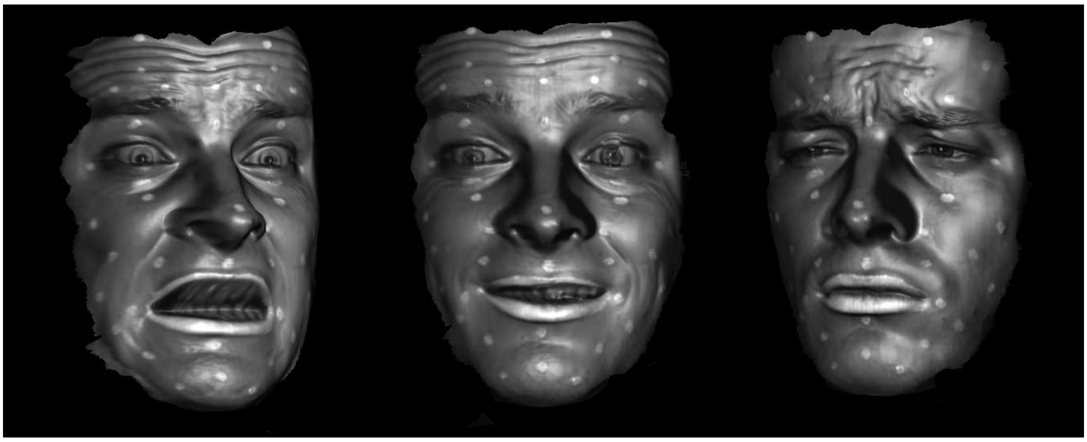


Fig. 8.11. **Expression vectors** Along with the speech corpus, expressions are recorded. The articulation space computed by the PCA does, however, not span the space of such expressions. The expressions vectors are still registered according to the rest of the data, but are taken as deformation vectors relative to the reference face. For new animations, expressions are added atop the generated faces to add realism and expressiveness.

synthesized 3D frame vector is then weighted with the expression which lets the face point glide towards the expression. This approach, however, quickly cancels the articulation component in animations. The second approach, the one used in this thesis, is to take the expression vector flow as a relative deformation to the reference frame that is added atop the synthesized 3D frames. With this approach, the expressions have to be added carefully, as strong articulation, combined with strong expressions, can generate exaggerated face configurations which lack realism. However, soft application of expressions turns out to produce nice and realistic results.

8.6 Reanimating Faces

With the current state of the system as built up to now, a simple animation synthesis can already be performed which shows reasonable results (*Bargmann et al.*[4]). The goal is to generate an animation from a novel audio speech. The new animation has first to be synchronized to the audio which is thus first decomposed into a textual form as a phoneme list associated with their frame occurrence. To this end, the *Sphinx* software [106] from Section 8.4.2 is used. Again, each phoneme spans several consecutive frame (a slot). Here, to the first frame of each phoneme slot, the average face of the associated cluster is selected to ensure correct phoneme alignment. By performing this over the whole sentence, a series of succession of points is defined in the viseme-space.

The animation is then produced by morphing between these points to fill the remaining frame and simulate the transition effect of coarticulation. Instead of a linear morph between the average faces, these visemes are first weighted (or influenced) by the neighbor visemes in order to get a context influence of the visemes.

Let $\mathbf{a}_i$ denote the coordinates of the original average viseme corresponding to the phoneme attributed to the ith slot of the phoneme list. An adapted context dependent viseme $\mathbf{a}'_i$ is created as a weighted linear combination of $\mathbf{a}_i$ and the neighbors in the sequence. The weights are attributed as follows:

$$\mathbf{a}'_i = 0.1 \cdot \mathbf{a}_{i-1} + 0.5 \cdot \mathbf{a}_i + 0.3 \cdot \mathbf{a}_{i+1} + 0.1 \cdot \mathbf{a}_{i+3} \tag{8.16}$$

The weights sum up to 1 and the heaviest weight is put on the succeeding frame while keeping some influence from the previous frame $i-1$. An influence of the $i+3$ frame is included to increase influence on future phonemes as the lips tend to aim at the next phoneme to be uttered. Equation 8.16 simulates the context dependence of visemes, i.e., the shape taken in a phonetic context. The transition aspect is considered next.

According to this process, all $\mathbf{a}'_i$, $0 \leq i < N$, of the N slots are interpolated linearly. The synthesis is generated at a frame rate of 100Hz, the frame rate generated by the *Sphinx* software. As animations run at 25 fps, 4 subsequent frames are combined to produce a motion-blur effect that gives a more realistic touch to the animation, where strong lip displacement during each camera shot becomes visible. Figure 8.12 shows a sequence of snapshots from a synthesized animation. The results are discussed in Section 13.4.

From here, several approaches can be taken for speech synthesis. The approach presented by *Ezzat & al.* [47] defines smooth curves over the viseme clusters that produce realistic animations in 2D. The clusters are modelled by a Gaussian distribution in a PCA-space. The distribution is centered around the average appearance of the viseme, and varies according to the variation found in a database of samples. For finding the motion trajectory for a new utterance, a regression algorithm computes a smooth curve that passes as close as possible to the centers of the visemes, relative to the variance of the viseme cluster. *Kim and Ko* [74] applied this principle to 3D data. They argue, however, that this approach sometimes generates movements that are too mechanical when used with three-dimensional face models. Better results are obtained by combining *Ezzat*'s method with a data-driven approach.

The following part of this thesis attacks the speech synthesis problem on a new front. It bases on a *triphone*-based approach that was proposed by *Bregler et al.* [22], to simulate realistic coarticulation transitions, but the visemes are selected according to their context in a new selection technique.

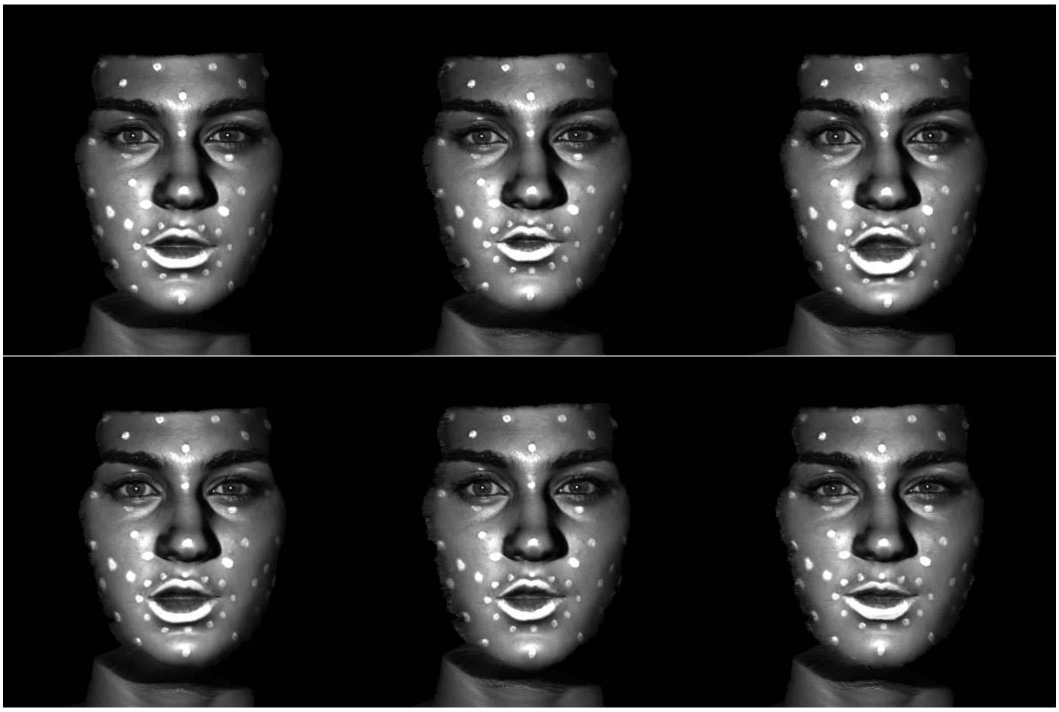


Fig. 8.12. Simple Animation Synthesis By taking the average face shape of every viseme cluster, simple animations can already be synthesized. The visemes are selected and synchronized according to phonemes occurrences in an audio track. The articulation is achieved by a weighted interpolation between the visemes. On this face model, the white lipstick is only applied on the border of the lips, which gives the impression that the mouth is not fully closed when it actually is. (See Section 13.4 for further discussions.)

9

Face and Articulation Transfer

Multilinear Morphable Models (MMM) give a low-dimensional representation of high-dimensional structures. For geometric shapes, the space associated with the MMM spans all possible combinations of elements in a set of registered data. In this thesis, such a MMM was built on a set of articulating face shapes (Section 8.4.1). *Blanz*[14] has built a MMM which spans a space of individual neutral faces (identity space) based on a set of 200 different heads. These 200 samples are sufficient to acquire the knowledge of characteristic shapes of most Caucasian individuals which then permits, in a fitting approach (*Blanz & Vetter*[18]), to approximate novel faces from 3D scans or even photographs *Blanz et al.*[16].

This Chapter puts the viseme and the identity MMMs in relation to address two problems:

1. the 3D scans of the visemes show only partial coverage of the face but the identity MMM builds full head shapes. Putting both MMM in relation can complete the missing information in the viseme MMM;
2. the viseme MMM performs on a single identity, the one of the recorded actor. A correspondence between both MMMs permits to transfer the articulations to different face shapes.

These two aspects are addressed in the two following Sections 9.1 and 9.2, respectively. The two aspects work in opposite directions: for completing the viseme shapes, the model is converted to the identity MMM and the data structure is altered. For transferring articulation, identities are imported to the viseme MMM. Theoretically, both problems can be solved both ways, however, due to technical constraints, we opted for these approaches.

9.1 Head Completion

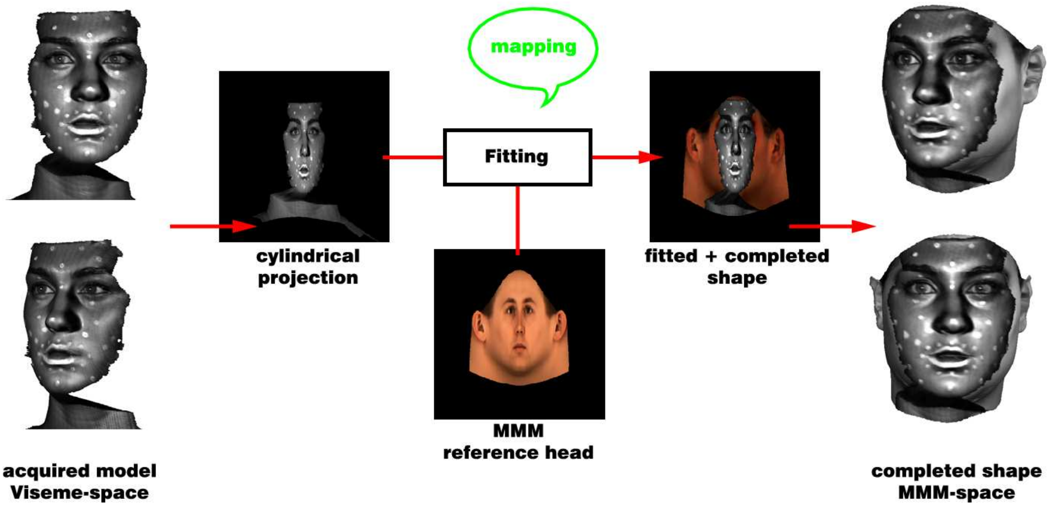


Fig. 9.1. Face completion of the viseme model The viseme face model shows only partial coverage of the face. By fitting the data to a MMM of identities (*Blanz*[14]), the missing information can be reconstructed. For that purpose, the viseme model is converted into the identity model in a cylindrical projection (left) and the fitting process finds a best match *mapping*. However, the identity MMM cannot reconstruct mouth shapes, as it spans a face space of neutral expressions. The converted viseme face is thus pasted onto the reconstruction, and a hybrid shape is obtained (right). The geometry shows smooth transitions at the edges, and the texture can easily be replaced. This procedure has to be performed over all frames of the viseme model and is thus computationally costly.

The acquired face shape from the dynamic 3D scanner shows only a partial coverage of the face. To obtain a full head completion of the geometry, the missing data can be learned from an identity MMM developed by *Blanz*[14]. In this model, shapes and textures are described in a cylindrical data structure. The first step is to transform the viseme data to correspond to the identity data. By defining corresponding feature points on selected single samples from both MMMs, the viseme sample is rigidly aligned (orientation and scale) to the identity sample. This alignment ensures a compatible cylindrical projection and the viseme data is transformed from a $640x480$ pixels to a $512x512$ pixels data structure (refer to Fig. 9.1-left). An iterative optimization algorithm (*Blanz & Vetter*[18]) fits the viseme sample to the neutral sample from the identity MMM. The fitting produces a best approximation to the viseme sample, given the information acquired from different identities. Because the identity MMM is based on neutral faces, the mouth shape from the visemes cannot be reconstructed and the viseme shape is thus pasted onto the reconstructed model. The generated head is a hybrid combination of the the identity model and the available viseme shape with the structure parameterization of the identity MMM (Fig. 9.1-right).

The results show that the transition along the border of the viseme shape to the full head is smooth and the original viseme sample sees its geometry reliably completed. However, the underlying head model fitting a specific viseme is not suited for all the viseme database as the shape along the viseme edges varies with the articulation. This fitting process, which requires a few minutes, has thus to be performed for each of the viseme samples which has shown to be an extremely time consuming task for more than 17’000 samples.

Another aspect has to be considered: the fitting process generates a novel head shape for each of the viseme samples. By the correlation of the face features, in order to match different face shapes, different heads are produced with different ear shapes, for instance. This problem is addressed by progressively smoothing the geometry of the generated head towards the back to the neural face of the identity MMM which guarantees to keep the smooth transition between the two superimposed face geometries while providing consistency over the completion.

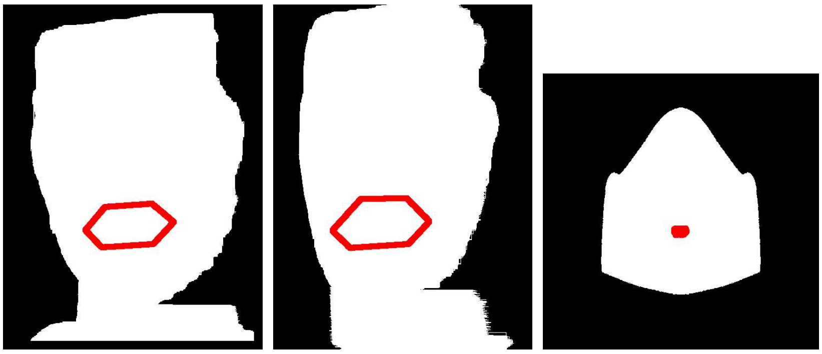


Fig. 9.2. Relative data structure size of the different MMMs and lip coverage The data arrays of both viseme corpora have a size of 640 × 480 pixels (left and center), against 512 × 512 for the identity MMM (right). While these resolutions are comparable, the identity MMM has a full face coverage, and less pixels are available for lip information (red region). Hence, by converting the viseme MMM to the identity MMM, articulation information is lost.

Transforming the data structure of the viseme model shows another issue. Figure 9.2 illustrates the disposition of the data in the array structure of the corpus of the first recording session, the second corpus, and the array structure for the identity MMM, respectively. While the size of the structures are comparable, the identity MMM focuses on the full surface of the face, hence reducing the mouth region to a small set of pixels and losing important information about its shape. As is discussed in the following part of this thesis, it is important that the mouth structure keeps as much information as possible in order to perform an optimal analysis. Hence, this direction of conversion of the data was abandoned.

9.2 Importing Identities

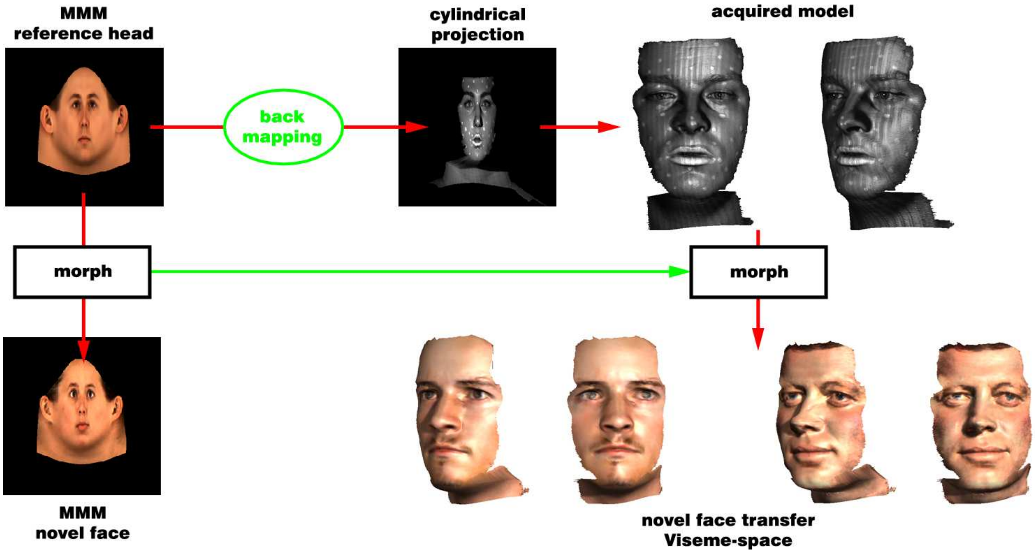


Fig. 9.3. Importation of novel identities to the viseme model Novel identities can be imported from the identity MMM. The mapping obtained in Figure 9.1 is reused backwards and applied to the vertices available in the viseme model. Knowing the mapping function and the deformation vector matching the neutral faces of both MMMs, a morphing in the identity- space can be reproduced in the viseme MMM and novel faces can be reconstructed (here the actor Orlando Bloom and President J.F.Kennedy).

Section 8.6 already exposed a first approach to generate articulation animations. As has been shown, the animation is performed on the original recorded face. The model can, however, be extended to different faces by defining a mapping between the viseme and the identity MMM. In the previous Section, such a mapping has been performed in one direction (see Fig. 9.1). This mapping can be reused backwards in order to import faces from the identity MMM to the present model (see Fig.9.3). However, the mapping is only performed partially as only the face surface captured by the dynamic scanner is available[1].

$$\begin{tikzcd}
\underline{\text{identity MMM}} & & \underline{\text{viseme MMM}} \\
\mathbf{S}_{ref} \arrow[r, "F_{transfer}"] \arrow[dr, "F_{novel}"'] & \mathbf{S}_{viseme} \arrow[d, "F_{morph}"] & \mathbf{V}_{ref} \arrow[l, "\text{mapping}"'] \arrow[d, "F'_{morph}"] \\
 & \mathbf{S}_{novel} \arrow[r, "\text{back mapping}"] & \mathbf{V}_{novel}
\end{tikzcd}$$

Fig. 9.4. Transferring a morphing function from the identity MMM to the viseme MMM The reference face from the viseme MMM is mapped to the identity MMM where the morphing function F_{morph} is first computed. F_{morph} is then mapped back to the viseme MMM in order the get F'_{morph} which performs the desired morphing from $\mathbf{V}_{ref}$ to $\mathbf{V}_{novel}$.

In order to perform the correct morphing in the viseme MMM and obtain the novel face, the neutral face from the viseme MMM is transferred to the identity MMM where using the mapping function from the previous section and the morphing function F_{morph} is computed. Using the back-mapping function, the morphing function F'_{morph} for the viseme MMM is retrieved. F'_{morph} is similar to F_{morph} but performs only on the partial geometry available in the viseme MMM. The procedure works as follows (refer to Fig. 9.4): in the identity MMM the morphing function which transforms the neutral face shape $\mathbf{S}_{ref}$ to a novel shape $\mathbf{S}_{novel}$ is known:

$$\mathbf{S}_{novel} = F_{novel}(\mathbf{S}_{ref}) = \mathbf{S}_{ref} + \Delta\mathbf{S}_{novel} \tag{9.1}$$

The neutral shape of the viseme-space $\mathbf{V}_{ref}$ is mapped to the identity-space as $\mathbf{S}_{viseme}$ using the fitting algorithm developed by *Blanz*[14]. The morphing function from $\mathbf{S}_{ref}$ to $\mathbf{S}_{viseme}$ performs

---

[1] An approach to reconstruct the face entirely is to export the faces to the identity MMM where they would be completed and to transfer them back to the viseme model. However, as Figure 9.2 illustrates, the completion information cannot fit in the actual data structure which would require to be reconstructed. While this is possible, this path was not pursued.

$$\mathbf{S}_{viseme} = F_{transfer}(\mathbf{S}_{ref}) = \mathbf{S}_{ref} + \varDelta\mathbf{S}_{transfer} \tag{9.2}$$

with

$$\varDelta\mathbf{S}_{transfer} = \mathbf{S}_{viseme} - \mathbf{S}_{ref} \tag{9.3}$$

and the direct morphing function is found:

$$\begin{aligned}\mathbf{S}_{novel} &= F_{morph}(\mathbf{S}_{viseme}) = (F_{novel} \circ F^{-1}_{transfer})(\mathbf{S}_{viseme}) \\ &= \mathbf{S}_{viseme} + \varDelta\mathbf{S}_{morph}\end{aligned} \tag{9.4}$$

with

$$\varDelta\mathbf{S}_{morph} = \varDelta\mathbf{S}_{novel} - \varDelta\mathbf{S}_{transfer} \tag{9.5}$$

To get the novel face shape $\mathbf{V}_{novel}$ in the viseme-space, F_{morph} is adapted as F'_{morph} which only retains the transformation for the vertices on the viseme shape:

$$\mathbf{V}_{novel} = F'_{morph}(\mathbf{V}_{viseme}) = \mathbf{V}_{viseme} + \varDelta\mathbf{V}_{morph} \tag{9.6}$$

with $\varDelta\mathbf{V}_{morph}$ deduced from $\varDelta\mathbf{S}_{morph}$ through the back-mapping.

The transfer shows modest results. Rounding errors in the mapping and residual noise in the 3D scans affect substantially the novel face. Animation also shows that speech cannot be transferred to a novel face directly; indeed, the shape of the mouth differs from one person to the other, and affects the lips in unnatural manners. Figure 9.5 shows the resulting transfer on the first small viseme corpus. Inconsistencies arise at the edge of the face shape where the registration is less precise. Indeed, the edges correspond to where the acquisition direction is tangential to the skin and depth measurements are more sensitive. These measurement errors are visible on the imported face shape.

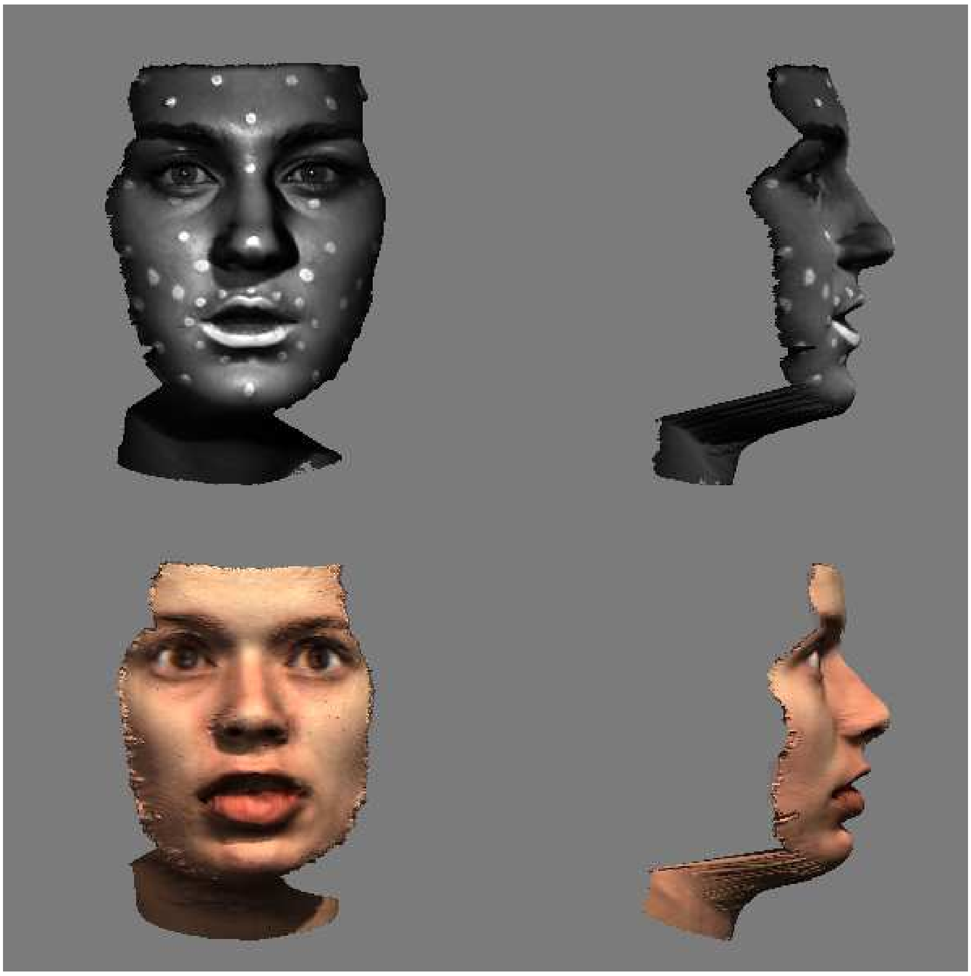


Fig. 9.5. Identity importation example A face shape from the identity MMM is imported to the first viseme model (small corpus). Only the coverage of the viseme model can be imported and acquisition errors on the edge are reproduced on the imported model.

Part IV

Speech Synthesis

Articulation in natural speech is a process which is difficult to model as it depends of many parameters. Speech is the most widely used means of communication for human interaction, and therefore, the brain is trained and highly skilled for detecting correct articulation movements. Hence, while being a complex process, in the eye of the observer any behavior that deviates from natural speech is disturbing and immediately noticed.

The movements performed by the mouth depend on many parameters. Different persons have different ways of speaking which, combined with different moods, span a large catalogue of manners of speech. The speech information we gathered in the previous part focuses on a neutral but well articulated manner of speech of a single person. The speech model we presented in that fourth part makes abstraction of the manners of speech (*global influences*) but focuses on the interaction of visemes taken in their phonetic context. From this point of view, *local influences* depend less on the actor and allow the development of a general model for the synthesis of neutral speech.

In Section 2.5 we explained how phonemes are produced by the modulations of the air stream propelled out of the lungs. Several components of the body perform these modulations, but their involvement is not always visible from the outside of the body. Some phonemes require also more effort to be produced than others, in consequence of what, the shape the mouth takes during the production of a phoneme is not only dependent on that phoneme, but also, and actually at a high level, on the surrounding ones. In short, the face shape or the viseme is highly dependent on the phonetic context it is taken from.

The viseme context dependency induces the concept of *coarticulation* (see Section 2.5.1) which is twofold: (1) for a given phoneme, the associated viseme is not unique, and (2) for natural articulation, the movement of the mouth should be observed, rather than a single viseme (see *Triphones* on page 26). The movement which we have to learn from the dynamics of the mouth starts at a point in time before the phoneme begins to be heard, and ends at a point in time after it ceased to be heard, when it leaves already towards the next shape.

We already stored these two aspects of coarticulation in the speech database. The duration of a phoneme spans several frames, and for each phoneme, the viseme sequence from the previous phoneme to the next, which is referred to as *triphones*, is kept. Triphones are composed of a succession of three phonemes (or visemes) and the two motion segments in between. A triphone composed of the phonemes $/A/$, $/B/$ and $/C/$ is written $|A, B, C|$ and represents the animation segments rather than the visemes $|A|$, $|B|$ and $|C|$ which are three single 3D face shapes.

The use of triphones is a common approach in speech synthesis and has been used in works from *Bregler et al.*[22] where it has been applied to 2D animation segments. The triphone approach produces a novel animation by picking motion segments from a database which matches the phoneme sequence of a novel audio file. While there may exist several candidates in the database for a desired phoneme sequence, the triphones are selected in a manner that guarantees the smoothest transitions in the new sentence. If good concatenation is found, the interpolation process alters the original motion only slightly and ensures that the original coarticulation is reproduced in the synthesis, thus giving natural movements.

The difficulty resides in the selection process. In the English language, there are 44 different phonemes[2] (see footnote on page 99). If a database held all possible phoneme combinations for triphones, the minimum number of necessary triphone samples would be $44^2 \cdot 43 = 83'248$, still offering only a single sample of each. The number of triphone samples available in our database is, however, 2484 and includes repetitions.

The key to the selection problem resides in the fact that several phonemes can be associated to similar visual behaviors. In the filed of *articulatory phonetics* such viseme classifications have been defined for quite some time now: one of these classification was presented in the work of *Owens & Blazek* [91] in 1985 and became a standard classification. These classifications, however, resolve the problem only partially: while the number of necessary triphone samples is reduced, it still occurs that combinations are missing or that only a few samples are offered. If the number of samples to pick from is too small, chances are that heavy interpolation is required to smooth trajectories which cancels the natural coarticulation information.

In this part, we directly address the selection problem: by performing a nonlinear analysis of the articulation data, we derive a measure of similarity between visemes which gives a graded degree of substitution preferences; when the database is queried for a specific triphone and this one is not available, we provide triphone candidates with are the most similar to the requested one. Moreover, combined with an interpolation cost, our measure permits the selection of motion segments originally associated with different phonemes, if they offer a better concatenation, but natural articulation is still preserved.

In the following, most of the illustrations refer to the analysis or the results obtained from the second recording corpus with the larger dataset. Comparisons with the smaller corpus are explicitly mentioned.

---

[2] For the French language, for instance, only 39 phonemes exist, from which *Benoît & Le Goff*[9] consider only 32.

10

A Nonlinear Model as a Similarity Measure for Visemes

The viseme-space generated by the PCA in Section 8.4.1 gives a first representation of the data. Visemes are represented by points in a 50-dimensional space which makes it difficult to visualize. It is, however, possible to observe the distribution of the visemes in a three-dimensional representation, as illustrated in Figure 10.1. An interactive user interface was developed for that purpose: the user can freely select which component of the viseme-space basis should be represented and observe the clusters separately. In this Figure, the clusters for a few phonemes are represented along the three first components. Visemes associated to the cluster (the first frame of the span of the phoneme) are depicted by a black dot, the curves show the associated triphone segments: blue segments indicate the coarticulation curve before the actual viseme, red segments, the curve afterwards.

This representation is interesting, as it shows the actual behavior of the mouth during speech. Indeed, subsequent frames in the recorded corpus have only little differences; in the viseme-space, their locations lie close together. An articulation movement, hence, describes a smooth curve.

These curves, however, show that the mouth moves freely and that for a specific phoneme, the coarticulation curves can be highly uncorrelated. Moreover, by observing the central visemes of these triphones in the space, their location cannot be restricted to specific regions, and the different clusters strongly overlap. As it is going to be presented later, the PCA space is a good representation for generating curves to produce animations; the analysis of the behavior or the characteristics of the different visemes is here, however, hardly possible.

In this chapter, a nonlinear data analysis is performed which uses *Locally Linear Embedding* (LLE) and generates an intuitive representation of the data (Section 10.1). The LLE representation allows to better study the behavior of the mouth in relation to phonemes by defining regions in which it evolves and others to which it is dissociated. Depending on the distributions of these clusters, a hierarchical model is derived which gauges degrees of similarity among the visemes (Section 10.2) and which finally dictates a substitution rule for picking optimal triphones in the database (Section 10.3).

Note that the LLE reduction is only used for determining similarities among visemes and their extension to triphones. All processes related to the animation are performed in the PCA space and are discussed in the following Chapter 11.

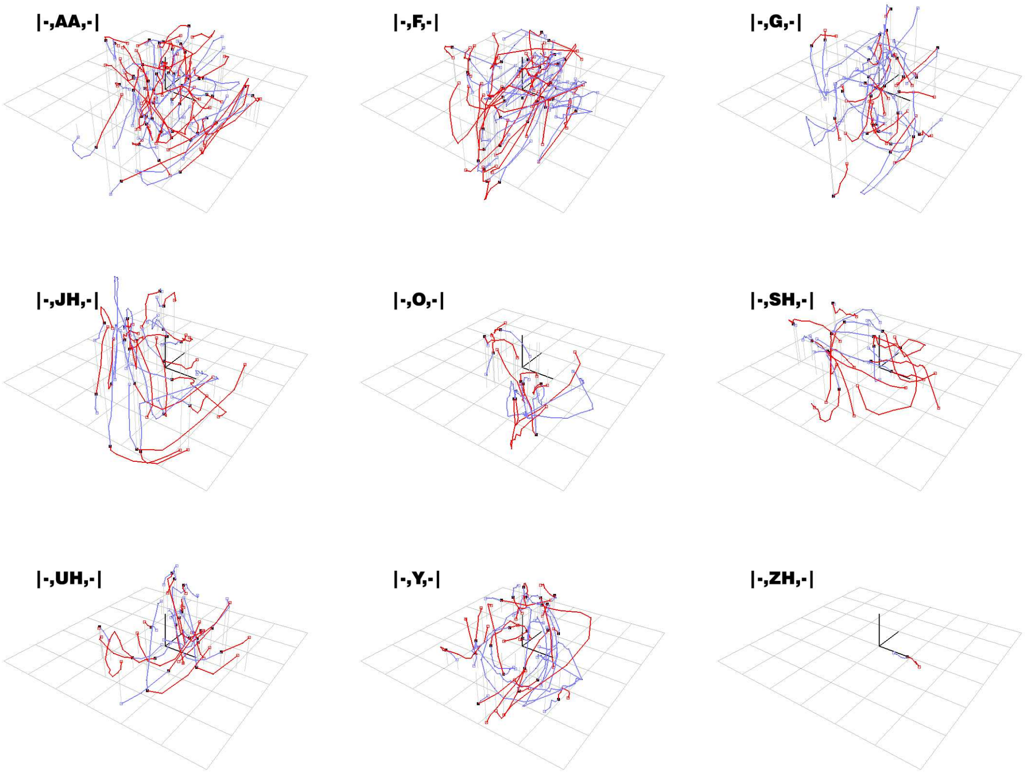


Fig. 10.1. The representation in the PCA space of the recorded triphone curves around different phonemes The first frame of the occurrence of a phoneme is marked as a black point. The blue and the red segments represent the articulation movements from the preceding and to the succeeding phoneme, respectively. The behavior of the mouth (curves) and the shape of the visemes (locations of black dots) show no coherent structure and this representation is thus ill suited for the behavioral analysis on speech.

10.1 Locally Linear Embedding

The use of a reduced linear vector representation of mouth shapes is a logical approach for the generation of a viseme model; morphing along the different axes within a delimited span generates plausible shapes. However, the way the mouth behaves during articulation draws a low-dimensional manifold which is not perceivable in a linear representation. From the many physically realizable visemes, many are simply not produced during speech, hence, the space spanned by the visemes involved in the articulation process has a lower dimension.

Chapter 6 presented different well-known nonlinear dimension reduction techniques. The usage of *Locally Linear Embedding* (LLE) in this thesis is inspired by the representations of viseme-spaces in the works of *Saul & Roweis* [109] or *Graf & Cosatto* [53] (see Fig. 10.2). These representations show that the low-dimensional manifold for speech has the property of a star shaped distribution with the edges spreading towards extreme mouth configurations. In this thesis, a study on viseme behavior enables us to characterize visemes clusters according to their distribution on these manifolds.

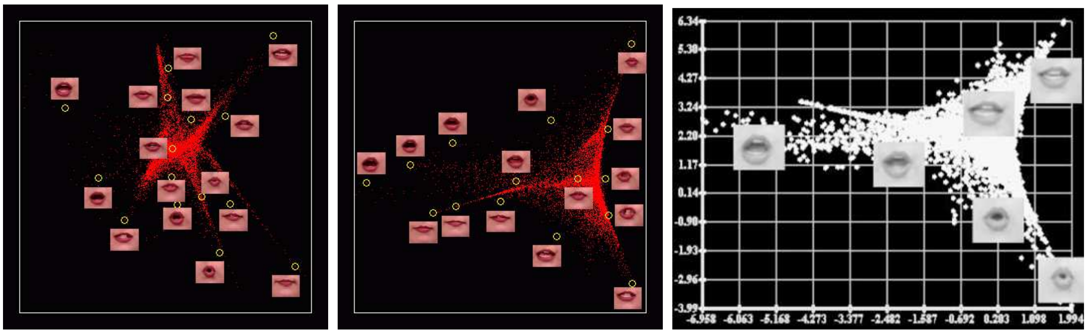


Fig. 10.2. LLE use for speech animation in contemporary literature *Saul & Roweis* [109] (left and center) or *Graf & Cosatto* [53] (right) have already associated LLE reduction with speech data. The representations always show a star shaped distribution of the low-dimensional manifold even when different parameters to the reduction are given (left and center). This characteristic distribution is also produced for the three-dimensional data studied in this thesis (see Fig. 10.5).

LLE unfolds the manifold to a low-dimensional space by keeping linearity between adjacent samples (see Section 6.3). The degree of sample adjacency considered in the process has a great effect on the outcome and has to be given by the user. Moreover, LLE demands the desired dimension of the space which the manifold should be projected to. These two parameters are of course not available *a priori*. The only way to find them out, is to perform several trials until a convincing representation is produced. The problem arises that these tests are computationally expensive for high-dimensional vector sets, as the LLE reduction performs a matrix inversion and the processing time becomes problematically long. The next Section addresses this problem by considering only a small set of points along the lips to reduce the dimension of the problem; the intrinsic dimensionality of the manifold is kept, as all the necessary information of the mouth deformations is encapsulated in this set of points. Section 10.1.2 then performs a viseme selection to reduce the size of the data in order to process and optimize the output of the LLE algorithm.

10.1.1 Data Simplification

The problem caused by the high dimensionality of the data of the LLE process is addressed by considering only a series of vertices along the lips. This series of 9 vertices is selected by hand and illustrated in Figure 10.3. The three coordinates of each of the vertices are considered which still ensures a spatial tracking of the lips over the whole dataset and reduces the dimensions to 27.

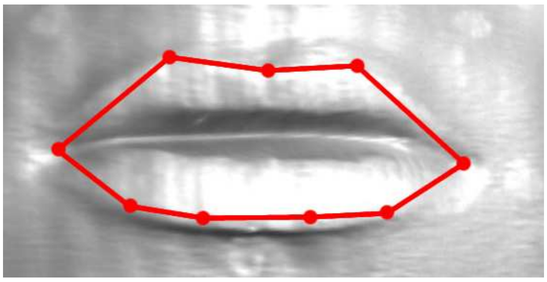


Fig. 10.3. The 9 selected vertices retain all movement information of the lips To address the problem of high-dimensional data processing in the LLE algorithm, only a series of 9 hand selected vertices along the lips is considered. The spatial deformation information of the lips is thus kept, while the data is reduced to 27 dimensions, each vertex being represented by its three coordinates.

These selected point vectors $\mathbf{v} = (x_1, y_1, z_1, ..., z_9) \in \mathbb{R}^{27}$ taken from the registered corpus data lie in a “dangerous” region of the face: by the very nature of the registration, vertices can slightly slide on the surface of the face which becomes problematic for vertices that lie on bumpy regions such as the lips. By observing the evolution of the series over the frames, a flickering motion is detected when these vertices slip towards the mouth cavity. This flickering has a direct impact on the outcome of the LLE reduction and has thus to be removed. With the aim of reducing the noise, a PCA is performed over these vertices which gives the principal deformation components. By removing the last components (the noise) this flickering is eliminated. After that process, each vector $\mathbf{v} \in \mathbb{R}^{27}$ is further reduced to a vector $\mathbf{v}' \in \mathbb{R}^{8}$ which still holds the important deformations of the lips. We note that in this reduced PCA space, still no simple pattern is found in the trajectories.

The last claim is defended as follows: when one thinks of the possible mouth configurations, several parameters have an influence on the resulting viseme, in particular the face muscles in the region of the mouth or the tongue. However, although all these components play an essential role, it can be observed that, for a given configuration of the mouth, they lose all degrees of freedom (at least for the ones that have a visual impact). This means that, for a given viseme, all these parameters are defined and are thus strongly correlated (for a given shape of the lips no muscle can be activated without having a visual consequence). Because of this strong correlation, a small selection of points can almost completely retrieve the state of all the components.

10.1.2 The LLE Computation

The problem related to the dimensionality of the original data has been addressed in the previous Section, and each of the 17142 recorded frames is represented by a vector point $\mathbf{v}' \in \mathbb{R}^{8}$.

Further tests performed with the LLE also showed that the data distribution over the manifold affects its outcome; a better representation is obtained if the distribution is homogeneous. An important characteristic of the acquired data is, that, because the samples are recorded dynamically, the distribution of the data is strictly speaking not homogeneous. The reason is that the samples follow articulation curves and, most often, their closest neighbors are lying on the same curve. LLE constructs a representation that is based on the spatial relation among the samples, and to get a better representation of the underlying manifold inside the data, one must first select a sub-set of samples that is evenly distributed in the original space. To achieve this, the distances between all pairs of samples are measured (Euclidean distance in the reduced space of the vector points). By running through all the samples successively, the algorithm checks whether a sample is closer than a given threshold k to another sample; in that case, the sample is removed from the database. If a sample is removed, it is not going to be present in the LLE representation. To keep track of the removed samples, their phoneme label is associated with their closest sample which thus has multiple labels. With increasing k, neighbors lying on a common curve are slowly removed, until the threshold reaches the average distance that separates different curves, and samples start to disappear more quickly. Drawing the number of remaining samples in function of k shows that when k reaches this critical distance, the function curve decreases rapidly. Figure 10.4 illustrates that behavior for both corpora. For the

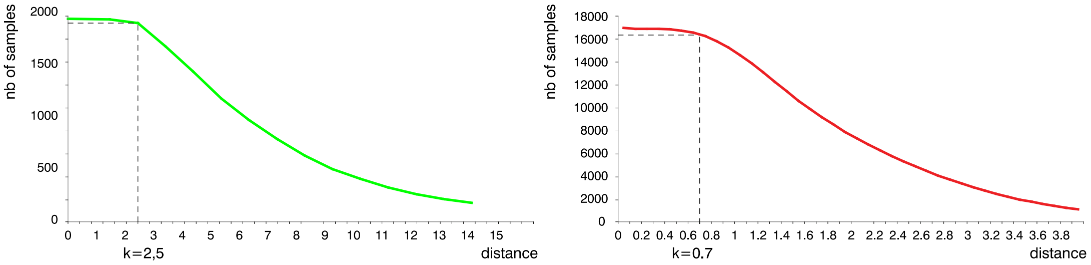


Fig. 10.4. Sample reduction curves To get a homogeneous distribution of the data, samples that lie too close to each other are removed according to a distance threshold k. The number of remaining samples drops steeply when the threshold reaches the critical distance that separates the articulation curves, which is around $k = 2.5$ for the small corpus (left) and $k = 0,7$ for the large one (right).

small corpus, the curve breaks at $k = 2,5$, for the large corpus at $k = 0,7$, as the manifold is much more dense (17142 samples against 1933). This procedure makes sure that LLE is not dominated by the individual curves of closely adjacent vectors, but by the intrinsic structure of the set of different mouth shapes involved in articulation.

As shown in Figure 10.4, for the small corpus, the number of samples is reduced to about 1800 and to about 16'000 for the large one. Another problem arises: the LLE is performed on a Matlab platform which shows memory allocation problems for datasets larger than 8000 samples. While this is no problem for the first corpus, the second one, on the other hand, has to be drastically reduced. The threshold is thus further increased until the remaining number of samples reaches 7000 ($k = 2,5$).

The LLE is finally computed on the reduced dataset. The best separation of the data is obtained by selecting a neighborhood connectivity of six samples and a target dimensionality of six in the case of the large corpus; for the smaller corpus, these parameters are set to 4 and 4. The fact that the neighborhood degree equals the output dimension is a known property of the LLE reduction, as discussed in Section 6.3.

The reduction generates the star shaped representations shown in Figure 10.5; there is a clear similarity with the representations obtained by *Saul & Roweis* [109] or *Graf & Cosatto* [53] in Figure 10.2 which demonstrated that the nonlinear reduction produces a low-dimensional manifold that is intrinsic to the process of articulation. This information can be retrieved independently of the data structure used for the analysis. Section 10.2 discusses the distribution in detail, it is, however, interesting to already point out some of the characteristics: for the smaller corpus, a three directional star shape is obtained, while the larger corpus gives a four directional shape. In fact, a fifth peak can be observed but it turns out that it corresponds to a small

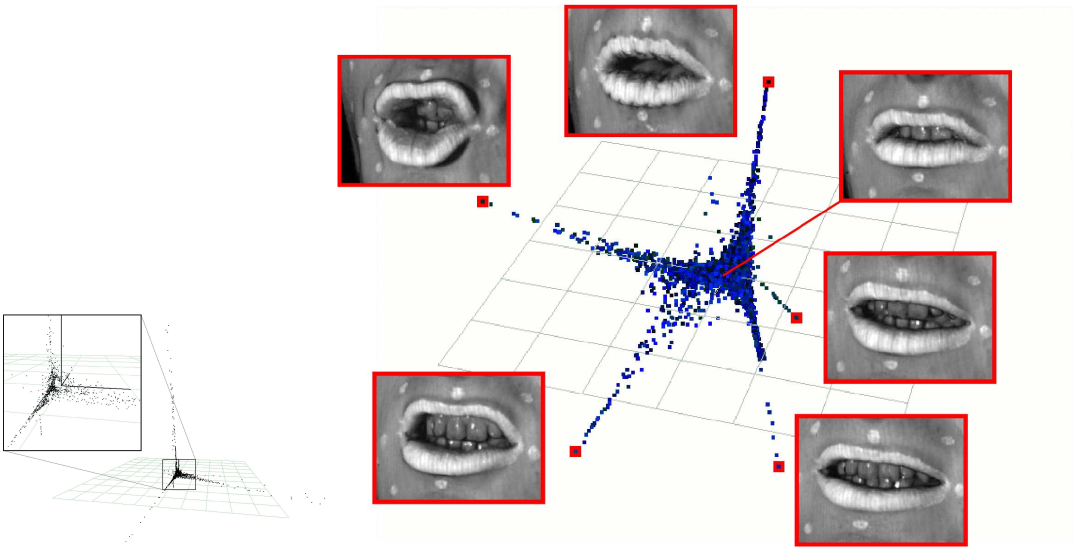
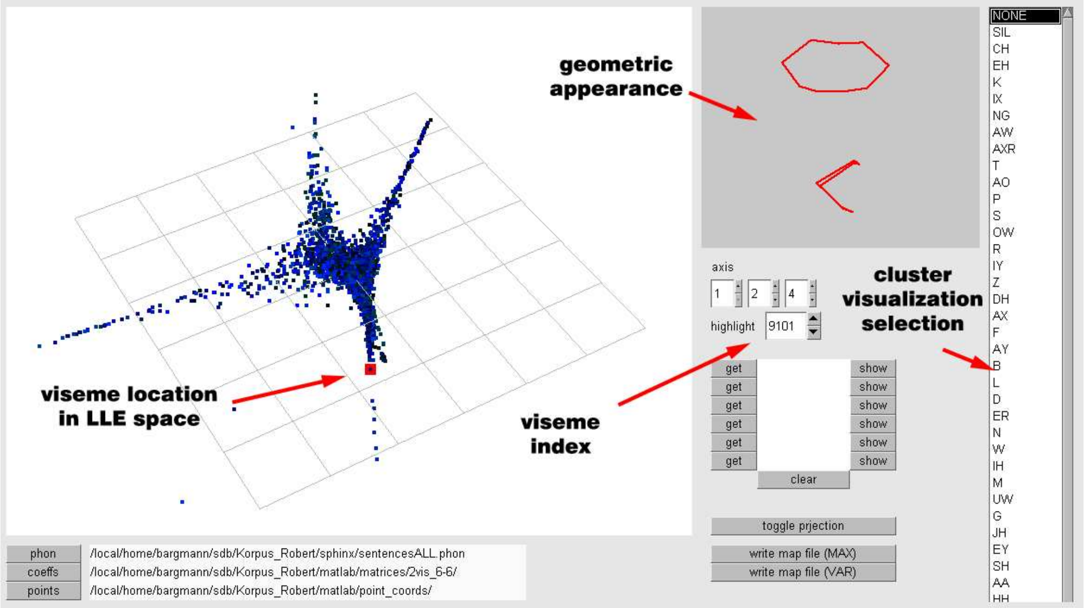


Fig. 10.5. The articulation manifold representation in the LLE-space A Locally Linear Embedding (LLE) over the recorded data shows that a low-dimensional representation is not only intuitive but also intrinsic to mouth movements. The central region of the star-shaped manifold pertains to neutral mouth configurations. The three directions correspond to mouth shapes that are typical for $|EH|$, $|W|$, $|AW|$ and $|I|$ for the larger corpus (right). For the smaller corpus (left), the LLE representation presents only three directions since $|W|$, $|AW|$ coincide. These representations show great similarities to previous work illustrated in Figure 10.2.

continuous segment on which the registration performed with less accuracy. This segment is thus ignored in the following. Regarding the distributions, the central region of the stars holds neutral mouth configurations; the four directions (three for the small corpus) go towards mouth shapes that typically correspond to $|EH|$, $|W|$, $|AW|$ and $|I|$. For the smaller corpus, the $|W|$ and $|AW|$ are merged.

One aspect still remains to be discussed: for the large corpus, the output dimension is 6 which a priori makes it still difficult to visualize. The interface developed for the analysis of LLE allows to visualize three dimensions simultaneously (see Fig. 10.6) which are specified by the user. In the present case, the third dimension emphasizes the erroneous segment which spreads quite far, while the fourth dimension better separates the four main edges of the star. On the fifth dimension, all coordinates are zero and on the sixth, all are equal to 1. The shape presented in Figure 10.5, hence, shows dimensions 1, 2 and 4, a good visualization of the low-dimensional manifold.



phon /local/home/bargmann/sdb/Korpus_Robert/sphinx/sentencesALL.phon
coeffs /local/home/bargmann/sdb/Korpus_Robert/matlab/matrices/2vis_6-6/
points /local/home/bargmann/sdb/Korpus_Robert/matlab/point_coords/

Fig. 10.6. The user interface developed to visualize the LLE The main panel shows the three-dimensional representation of the LLE in which all reported samples are color coded in order to facilitate user selection. A selected viseme is highlighted in red and the corresponding lips shape can be viewed in the the upper panel and its frame index is displayed. On the right, the user can select a phoneme from a list to highlight all associated frames. This option allows the visualization of the viseme clusters (see Section 10.2).

10.2 A Measure for the Similarity of Visemes

The audio from the recorded corpus is decomposed into a sequence of phonemes using the CMU-SPHINX algorithm [64] and gives the time interval (or a sequence of frames) during which the phonemes are heard.

From each interval, the first frame is taken as a viseme sample and is labelled with the associated the phoneme. Then, for each phoneme, the distribution in the LLE representation of its visemes is investigated. (Note that while all recorded samples are represented, only the distribution of these visemes are considered.) The remaining frames capture the transition movements, and are used in Section 11.1 to synthesize new sentences with realistic transition velocities and accelerations.

The difficulty in a learning-based approach is that the natural number of possible phoneme combinations forming triphones is too large to be possibly recorded, not considering that several samples of each triphone would be appreciated. However, different triphones may have very similar motions: by identifying similar triphones, a robust substitution option ensuring good

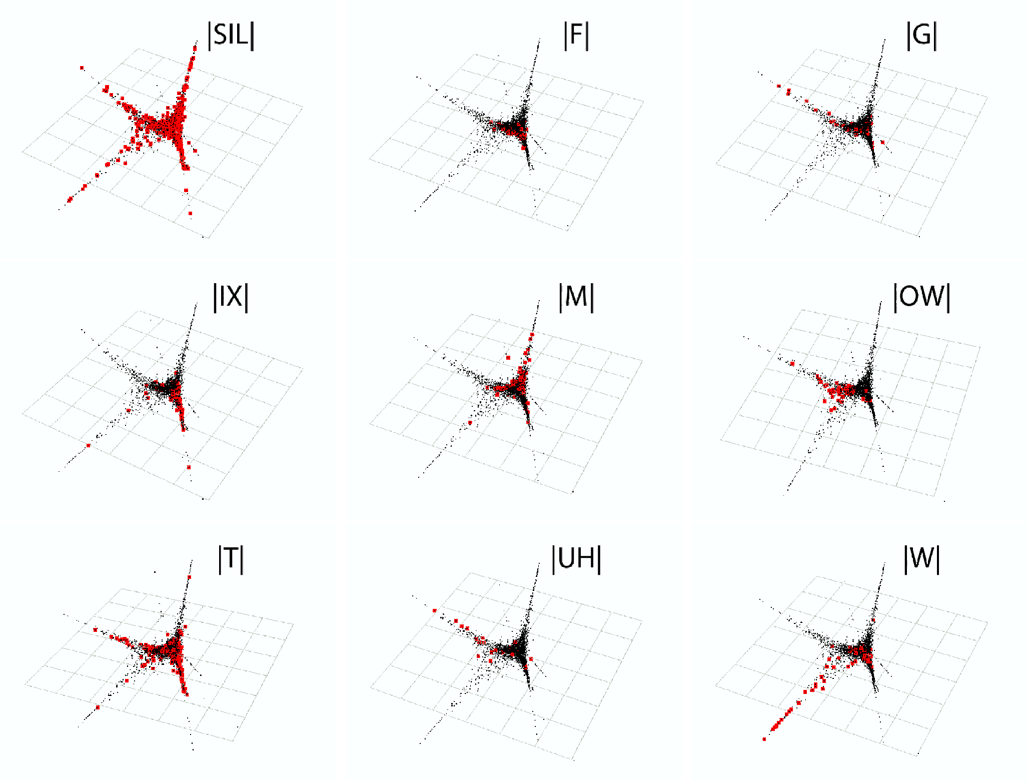


Fig. 10.7. The LLE representation of the recorded viseme candidates for the different phonemes The black points show all the recorded frames from the corpus, the red squares illustrate the location of the visemes associated with the phonemes. Their distribution defines different viseme clusters for each of the phonemes. Note how $|SIL|$ typically can happen with any mouth shape, unlike $|W|$, for which the lips have to take a particular shape.

transitions between visemes during coarticulation is offered. Moreover, such substitutions become even more necessary when desired triphones are not directly available.

From the original corpus, all visemes (first occurrences of phoneme sequences) are mapped to the LLE representation, where their distributions form viseme clusters. Figure 10.7 illustrates some of the different viseme clusters. The different clusters vary not only in how they spread along the branches of the star-shaped manifold, but also how they are distributed along them. These two criteria are used to distinguish the clusters and to define a *measure of similarity*.

The idea is to measure the spread of the clusters along each branch after a chosen quantization. By taking the branches in a fixed order, the distribution of a cluster can be described by a quadruplet that indicates how far the cluster extends on each branch. For instance, [2411] means that the cluster of this viseme spreads to a quantized distance of 2 on the first branch, 4 on the second, 1 on the third and on the fourth branch of the manifold.

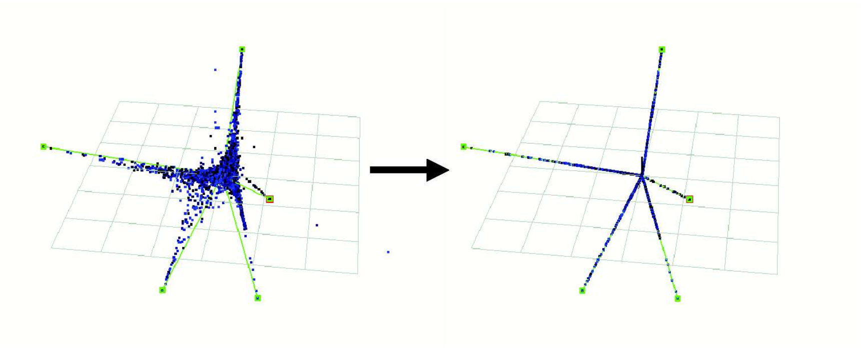


Fig. 10.8. The projection process involved in the quantization of the cluster distributions Five axes are defined by connecting the extremities of branches to the origin. All samples are then projected onto their closest axis on which their distribution is measured and a quantization is retrieved which describes the nonlinear model for (or the behavior of) each viseme cluster. As discussed in Section 10.1, the smallest axis is ignored, as it represents only a few erroneous samples and also relaxes the substitution rule.

This model is implemented the following way: the four visemes at the extremities of the branches are selected, they define four main axes connecting these visemes to the origin of the LLE space (Fig. 10.8-left). All samples are then projected onto the closest axis (Fig. 10.8-right). The quantization is then given by the spread of the samples along the axis while weighting them according to their distributions over the different axes (see equations below). Weighting the spreads allows to get a more accurate description of the behavior of the different viseme clusters: when a cluster spreads far along several axes, their quantization is lowered to favor other clusters which spread on less axes and let the final quantization better characterize them: the length of each axis i is normalized to 1 and the average squared distance d_i of the samples to the origin is taken. d_i is then weighted proportionally to the amount of samples on that axis in proportion to the total count of visemes for the considered phoneme.

Thus, for a cluster, if X_i is a random process describing the position of the samples along the axis $i \in \{1, \ldots, 4\}$:

$$d_{X_i} = \frac{1}{n_i} \sum_{j=0}^{n_i} x_{j,i}^2 \tag{10.1}$$

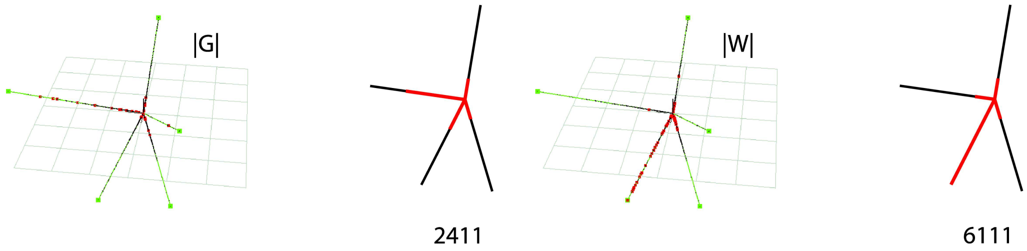


Fig. 10.9. Each viseme cluster defines a distribution on the projected LLE representation Projected along the main axes, the quantization of the spread along each axis is computed, which, taken in a fixed order, describes the distribution by simple quadruplets and characterizes the clusters.

where n_i is the number of visemes on axis i. With N being the total number of visemes for a specific phoneme p, the spread along the axis i becomes:

$$q_p(i) = \frac{n_i}{N} d_{X_i} = \frac{1}{N} \sum_{j=0}^{n_i} x_{j,i}^2 \tag{10.2}$$

The maximum $q_p(i)$ over all phonemes in P is then normalized to $Q = 6$ (the desired quantization steps[1]) and the remaining $q_p(i)$ are scaled accordingly. Finally, rounding to the closest integer gives the quantization of the spreadings along the axes. Figure 10.9 illustrates the process for the viseme clusters $|G|$ and $|W|$.

Table 10.1 gives the conversion to the quadruplet notation (triplet for the small corpus) for each of the viseme clusters and for the two corpora. The categorization in this table is an intrinsic statistical property of speech. Suppose a new animation system is built: when the registration process and the phonetic segmentation of the data are performed, this table can be directly taken to perform the segment selection for novel animations (see Section 10.3). The LLE analysis has not to be performed again, as these quadruplet values are intrinsic to the process of articulation.

A first look at Table 10.1 shows correspondence between visemes[2] that are intuitively similar. For instance, the visemes for $|P|$ have the same quadruplet representation as the visemes for $|B|$. Indeed, the utterance of their associated

---

[1] $|W|$ is particularly concentrated on the first axis. The quantization Q is set to 6 to in order to keep a fine enough resolution to distinguish and better separate the other viseme clusters.

[2] In the following, because the distribution of a cluster reflects the "behavior" of the visemes, the term *viseme* sometimes refers to the cluster as a whole rather than to a single sample.

Large corpus, quadruplet notation, 7 quantization steps ($Q = 6$)

PIT	0000	ER	1213	UW	2221
ZH	0002	JH	1312	K, T	2222
SH	0212	OY	2010	SIL	2223
CH	1011	AW	2013	G, UH	2411
AY	1022	NG	2022	IX	3031
DX	1023	EH	2023	HH	3032
Z	1032	AXR	2111	V	3112
F	1111	AA, AO	2112	OW	3211
AH	1112	P,B	2113	R	3212
AE	1122	TH, L, N, S, IY, DH	2122	EY	3331
D	1123	M	2123	Y	3312
IH	1221	AX	2212	W	6111

Small corpus, triplet notation, 4 quantization steps ($Q = 3$)

PIT	000	N	102	B, AA, W, L	121
IX	001	JH	103	HH	130
AXR, AW,	010	M, SH, IH	110	AX	131
TH	011	P, D, DH, DX, F,		IY	210
OW	021	G, K, AE, EH, R	111	T	211
ZH, AO, ER, EY, UH, NG	100	V, CH	112	S	312
Z, AY, OY, Y	101	AH, UW	113	SIL	333

Table 10.1. The quadruplet and triplet notations for the different visemes All phonemes are mapped to a quadruplet representation with 6 quantization steps. PIT is defined for the neutral head (a slightly open mouth) as a general substitute ot all visemes (0000) to prevent the substitution rule from being degenerative.

phonemes is visually identical. As opposed to $/P/$, $/B/$ has a voiced component which has no visual impact on the face. In the table, for the smaller corpus, $|P|$ is a first degree substitution to $|B|$; large corpora insure convergence of these quadruplets. Section 12.1 discusses such aspects in more detail.

10.3 Inclusion and Substitution Rules for Visemes

With the quadruplets defined for each viseme cluster, a rule for viseme substitution can be formulated. Consider two clusters $|A|$ and $|B|$ with a smaller distribution along the axes for the visemes of $|B|$. In the quadruplet notation, this implies that all digits of the quadruplet $[B]$ are pairwise smaller or equal to digits in the quadruplet $[A]$. This *inclusion rule* writes: $|B| \subset |A|$.

In the animation, unnatural movements are to be avoided. Therefore, in this example, an instance from $|B|$ (the smaller cluster) can replace instances from $|A|$ but not vice versa. In the quadruplet notation, this means that for a given quadruplet, a lower sum of its digits as compared to another quadruplet, is not a sufficient criterion for being a substitution candidate.

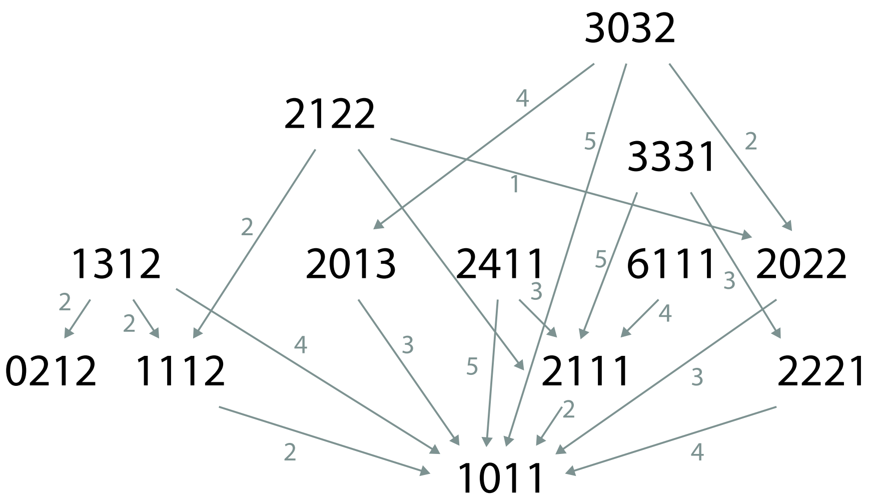


Fig. 10.10. The acyclic directed substitution graph (here only a sub-set of the total graph) The graph defines the valid visemes substitutions and their respective substitution cost.

While the database holds candidates for every viseme, substitutions become necessary when looking for triphone candidates. The number of possible triple-viseme combinations is far too large to be fully available, hence, best matching substitutes have to be selected. Ideally, a triphone matching the phoneme combination is found; if this is not the case, one has to look for similar visemes. In that aspect, visemes with the most similar distributions are preferred and a *similarity measure* is defined according to a *substitution cost* θ. This cost is measured as the sum of the digit differences of the desired viseme to its possible substitute (e.g. [1111] has a substitution cost of 3 over [2131]). By extension, the distance of a triphone to its substitute, is the sum of its three visemes' distances to their substitutes. The substitution of the central viseme in the triphone is penalized by taking twice its distance in that sum. This substitution cost is later combined with another *concatenation cost* (which favors natural transitions) to select the best suited substitution sequences (Section 11.1).

The inclusion rule and the substitution cost describe a graph of valid substitutions (Fig. 10.10 shows a sub-graph example). The nodes hold the quadruplet values, the edges indicate valid substitutions according to the inclusion rule, and are weighted with their respective substitution cost. Following the edges of the graph, all nodes are valid candidates. At every jump, at least one of the digits decreases until the quadruplet $|PIT|$ [0000] is reached. This quadruplet is a non-viseme cluster that doesn't spread over any axis. If PIT is reached without having found a fitting quadruplet, the neutral face is used in the substitution. However, experiments show that the rule is generous enough for the substitution algorithm to never reach that state.

While the inclusion rule is transitive

$$|C| \subset |B| \quad and \quad |B| \subset |A| \quad \Rightarrow \quad |C| \subset |A| \tag{10.3}$$

all edges in the graph cannot be deduced from the 1st degree edges (substitution cost $= 1$):

$$|B| \subset |A| \quad and \quad |C| \subset |A| \quad \nRightarrow \quad |B| \subset |C| \quad or \quad |C| \subset |B| \tag{10.4}$$

Hence, all edge degrees have to be sought. A typical example, in Figure 10.10, shows that the second degree edge between [2122] and [1112] cannot be deduced from single degree edges of the graph.

The graph now allows to find substitutes to triphones that are not available. For instance, suppose an instance for $|AA, P, UH|$ is sought: the triphone is converted into the quadruplet notation: $|2112, 2113, 2411|$. This triphone can then be compared to all available triphones and the distance between all three corresponding clusters of the two quadruplet is computed. The next chapter exposes precisely how this selection is performed.

11

Finding the Optimal Triphone Concatenation

The synthesis of speech animation starts with an audio track of a new sentence. This track is chopped into overlapping triplets of phonemes (Section 8.4.2) which give the sequence of triphones required for the novel animation. In order to account for coarticulation and to generate the most natural transitions between the visemes, the triphones are selected following the *substitution rule* (Section 10.3), while an optimal overlapping is provided that guarantees smooth transitions and extracts the most out of the original movements.

The database stores motion segments of natural articulation as *triphones*: these segments consist of three phoneme occurrences with two consecutive transition movements in between. The information held in such a triphone contains both aspects of coarticulation: the first aspect is the shape of the central viseme in its phonetic context, the second is both, the transition from the previous to the central viseme and the transition from the central viseme to the next viseme (in this thesis, only the first shape over a span of frames associated to a phoneme is considered as a viseme). For generating a new animation, a collection of triphones is selected, and the concatenation is performed by overlapping the segments (see Fig. 2.5). Because these triphones are pulled out of their phrasal context, their combination is likely to produce incompatible concatenations. This chapter addresses the problem of selecting suitable triphones for maximizing the transfer of the original coarticulation information by minimizing the necessary interpolation between the overlaps.

From a defined sequence of triphones, a list of valid substitution candidates is picked from the database along with their associated *substitution cost* θ. A *concatenation cost* φ is introduced (Section 11.1) which measures the overlap compatibility between triphones. Minimizing both costs over the whole triphone sequence provides the right combination of triphones to be selected from which a novel animation can be constructed in the PCA viseme-space (Section 11.2).

11.1 Concatenation of Triphones

A triphone $|v_1, v_2, v_3|$, with v_i as PCA vectors, is composed of a central viseme $|v_2|$ and two neighbors $|v_1|$ and $|v_3|$. It holds the position of the three visemes plus the frames for the transitions $|v_1, v_2|$ and $|v_2, v_3|$. When two successive triphones $|v_1, v_2, v_3|$ and $|w_1, w_2, w_3|$ are concatenated, a curve is generated that traverses v_2 and w_2, which keeps the tangentiality and the direction of the original curve at these points (see Fig. 11.1).

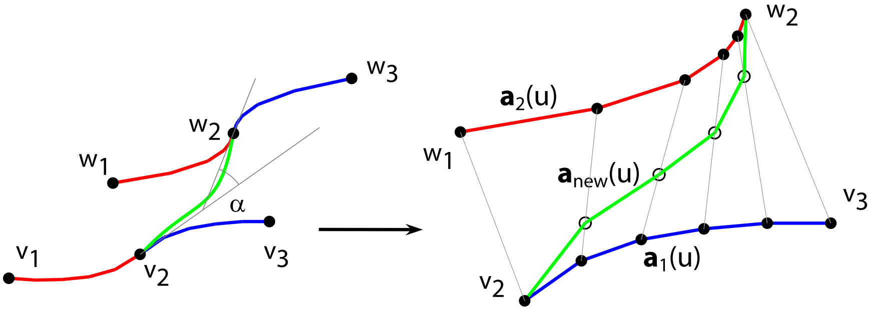


Fig. 11.1. Triphone morphing The interpolated curve passes through the center visemes v_2 and w_2 of both triphones $|v1, v2, v3|$ and $|w1, w2, w3|$ (left) while a weighted combination of both velocities and accelerations along the original curves $\mathbf{a}_1(u)$ and $\mathbf{a}_2(u)$ is provided in order to preserve the coarticulation information (right). The normalized time is denoted by the variable $u \in [0, 1]$.

First, a “quadriphone” $|v_1, v_2, w_2, w_3|$ is generated from which only the $|v_2, w_2|$ segment is retained (Fig. 11.1-left). In the morphing algorithm, it is important that the central visemes of the two triphones remain unaltered regarding their position, tangent and acceleration. The transition $|v_2, w_2|$ has to be computed knowing that

- the number of frames available in $|v_2, v_3|$ and in $|w_1, w_2|$ is different;
- the number of frames to be generated for the new animation is dictated by the phoneme sequence in the target audio file;
- variations in the speed of articulation between the two transitions have to be taken into account;
- the tangent of the curve at the central visemes should remain unchanged.

For each frame t of a candidate triphone, the original shape vector $\mathbf{s}$ is initially transformed into 50 PCA coefficients $\mathbf{a} = (a_1, a_2, ..., a_{50})^T$. The curved trajectory $\mathbf{a}(t)$ in the coefficient space can be mapped back to shape vectors $\mathbf{s}$ and be rendered on the screen for reproducing the original motion.

A piecewise linear function is then used to interpolate between the discrete time steps of the original frames. The argument $t \in [t_0, t_1]$ of this function can be substituted by a variable $u \in [0, 1]$, $u = \frac{t-t_0}{t_1-t_0}$ to obtain a time-normalized function $\mathbf{a}(u)$ (see Fig. 11.2). If the lengths of the transition phases in two subsequent triphones differ, this will be corrected by the substitution of t by u.

Let the transitions $|v_2, v_3|$ and $|w_1, w_2|$ be parameterized by $\mathbf{a}_1(u)$ and $\mathbf{a}_2(u)$, respectively. The new transition function $\mathbf{a}_{new}(u)$ for $|v_2, v_3|$ (Fig. 11.1-right) is an interpolation between $\mathbf{a}_1(u)$ and $\mathbf{a}_2(u)$. A linear interpolation between the two curves is given by:

$$\mathbf{a}_{new}(u) = (1 - u) \cdot \mathbf{a}_1(u) + u \cdot \mathbf{a}_2(u) \tag{11.1}$$

However, the tangents at the central visemes would be altered. To retrieve the tangents of the curves' extremities, the interpolation coefficient is weighted over time using the function

$$s(u) = \frac{u^2}{u^2 + (1 - u)^2} \tag{11.2}$$

and the weighted transition becomes

$$\mathbf{a}_{new}(u) = (1 - s(u))\mathbf{a}_1(u) + s(u)\mathbf{a}_2(u) \tag{11.3}$$

$\mathbf{a}_{new}(u)$ is then sampled with constant Δu to generate the desired number of frames for the new transition. Figure 11.2 illustrates the result of such a morphing.

This section already presented the approach of the morphing even though it has not been discussed how the selection is performed among several valid triphone candidates (see next Section). Indeed, this selection requires a *concatenation cost* to be defined which quantifies the compatibility of two successive triphones in terms of smoothness. By clarifying the morphing process first, it becomes clear what that cost should represent.

A first guess would suggests that the morphing energy should be minimized and the *concatenation cost* φ would then be defined as the integral of the distance between the two segments:

$$\varphi = \int_0^1 \|\mathbf{a}_2(u) - \mathbf{a}_1(u)\| du \tag{11.4}$$

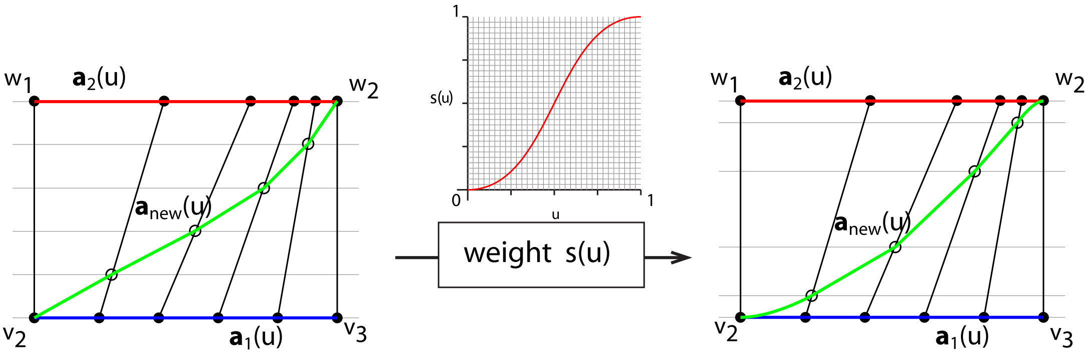


Fig. 11.2. Preserving original tangentiality at the central visemes When two transitions of two different triphones are morphed, the tangentiality is preserved by weighting the interpolation coefficient using a S-shaped weight factor $s(u)$ in order to ensure an optimal transfer of the coarticulation information.

It turns out that this consideration is not ideal, but rather the directions and the location of the curves in the PCA space should be taken into account. An attempt was performed by defining a cost function of these two parameters which did not provide optimal concatenations (Appendix 16.3 discusses that attempt). In fact, the best results are obtained by only considering the angle α between the curve tangents at $|v_2|$ and $|w_2|$ and defining the cost function as (see Fig. 11.1):

$$\varphi(\alpha) = -\cos\alpha \tag{11.5}$$

11.2 Finding the Optimal Sequence

Given the sequence of required triphones for a new sentence, a sequence of triphones from the database has to be found which minimizes both the substitution cost θ and the concatenation cost φ.

The new sentence is decomposed into a sequence of N triphones $i \in 1, .., N$. To compute the solution, the system provides a list of all M_i possible candidate substitutions $C_{i,j}$ in the database for each triphone i in the sentence. A graph is constructed whose edges connect every triphone from these lists, to all candidates in the preceding and the next list (see Fig. 11.3). Each one of the candidates j holds a *substitution cost* $\theta_{i,j}$ (the distance to the requested target triphone) and the *concatenation costs* $\varphi_{i,j,k}$ for the connection of $C_{i,j}$ to each of the preceding candidate k nodes $C_{i-1,k}$. The path that minimizes the total cost is then given by

$$\Theta_{min} = \sum_{i=0}^{N} \min_{j \in \{0,\ldots,M_i\}} \min_{k \in \{0,\ldots,M_{i-1}\}} (\theta_{i,j} + \varphi_{i,j,k}) \tag{11.6}$$

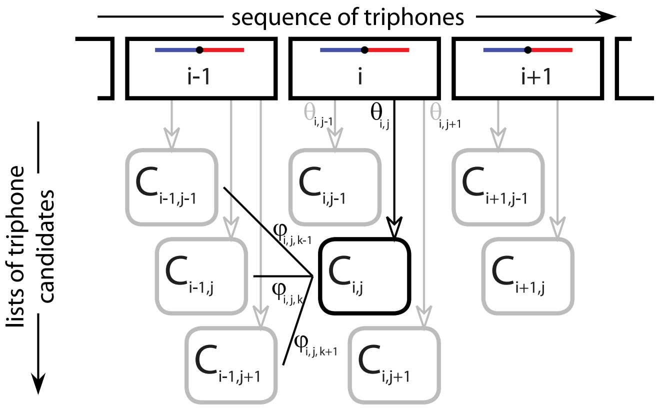


Fig. 11.3. Finding the optimal triphone sequence The optimal sequence of triphone substitutions is the one that minimizes the substitution cost θ and the concatenation cost φ. This optimal selection is found by computing the shortest path on a graph generated over the triphone candidates.

A shortest path algorithm is used over the graph to find the optimal solution. Note that Θ_{min} can be divided by the number of triphones in the sentence to defines a measure for evaluating the quality of the animation. The morphing is then performed between the selected triphones in order to generate the final articulation curve by mapping the coefficients $\mathbf{a}(t)$ to shape vectors $\mathbf{s}$.

Finally, after the triphones are selected and interpolated, the animation curve in the PCA space is passed through a low-pass filter which automatically smoothes the transitions and removes the noise present in the measured data (see Fig. 13.1).

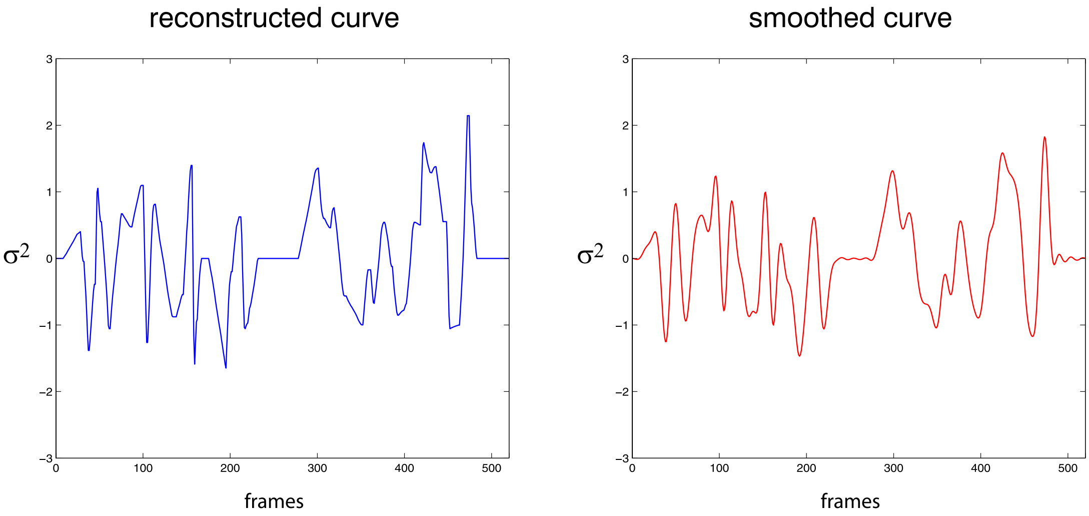


Fig. 11.4. Smoothing the reconstructed animation The synthesized animation obtained by the concatenation of triphones is described by a curve in the high dimensional PCA space. A low-pass filter is applied to the curve in order to remove the high frequencies. This figure illustrates the coefficient variation along the first principal component before the smoothing (left) and after (right). The filtering keeps the global deformation and provides a smoother animation.

12

Results

This chapter concludes the part dedicated to speech synthesis by discussing the obtained results. Besides the generation of synthetic visual speech from audio data, which is discussed in Section 12.2, the nonlinear model also gives some insight into the intrinsic structure of natural articulation. The next section therefore discusses and compares the empirical taxonomy of phonemes to the standard viseme grouping in the field of *articulatory phonetics* (Section 12.1).

12.1 Phoneme Similarity

According to phonetics, phonemes are generated by combined activations of body components, mainly of the lips, the tongue, the epiglottis and the larynx (the nasal cavity is only phonetically noticeable as a consequence of the air conducted to the nose). From these four elements, the first two have a direct influence on the shape of the mouth, whereas the last two occur at the back of the mouth. The epiglottis closes the aperture to the trachea to produce phonemes like $/K/$ or $/T/$ (unvoiced), whereas the larynx has the control over producing a voiced or an unvoiced phoneme; $/K/$ or $/T/$ become $/G/$ or $/D/$.

Vocals $/AA, AE, AH, AO, AX, AXR, AW, AY, EH, ER, EY, IH, IX, IY, OW, OY, UH, UW/$ are all voiced phonemes with no influence of the tongue while some air can be conducted through the nose like for $/ER/$ (the nasal phonemes changes when the nose is closed); vocals only depend on the shape of the lips and have a direct influence over the produced viseme; it is thus valid to represent them all as distinct visemes. Regarding the consonants, some can be visually grouped in pairs, if they are uttered in exactly the same way but differ only by their voiced or unvoiced nature. These groups are the $/B, P/$, $/G, K/$, $/V, F/$, $/ZH, SH/$, $/JH, CH/$, $/Z, S/$, $/D, DH, DX, T/$. Regarding the rest of the consonants, $/M/$, $/W/$, $/Y/$, $/L/$, $/N/$, $/R/$, $/NG/$, $/TH/$ or $/HH/$ have different tongue positions and lip configurations; voiced or not, they should fall into separate categories, as they produce slight differences on the behavior of the mouth. It is one of the strengths of our system that it determines, based on the recorded corpus, which viseme can be approximated by which other, or how similar a phoneme is to another.

In the articulation process, some phonemes have more influence than others. For instance, $/UH/$ forces the mouth to a round shape, whereas $/K/$ can be articulated with almost any shape, because the sound is produced at the back of the mouth. This aspect is important to consider when generating animations, because it dictates which visemes are to be more accurately approximated. This information is available in our database and is automatically taken into account in our pipeline. $|K|$ or $|N|$ are typical examples of visemes that can be substituted by more clusters (their associated quadruplets show that they spread along all branches of the manifold).

Considering that we have a relatively small corpus of viseme samples (from 17'000 frames, around a fifth are taken as viseme candidates) and that the number of occurrences of the different visemes varies massively (from a single candidate for $|ZH|$ to 17 for $|CH|$ and 196 for $|T|$, see footnote on page 99), both our similarity measure and viseme categories can further converge with a larger corpus.

In this thesis, no *a priori* assumption of phoneme similarity is taken. It is interesting to compare the obtained viseme groups with those found in the literature on *articulatory phonetics* [91, 130]. Table 12.1, compares popular viseme associations to our nonlinear model. Numerical closeness in the quadruplet notation indicates similar distributions or similar behavior of the visemes. *Yau et al.* [139] use the MPEG-4 standard that actually comes close to the model proposed by *Binnie et al.* which is discussed in *Owens and Blazek* [91]. We highlight in bold face where our viseme encoding matches closely the standard grouping. Our model matches well both classifications from *Owens and Blazek* and from *Walden et al.* (refer again to [91]). While a greater corpus size would make our model converge, these results provide a rigorous empirical methodology for verifying the classification schemes in the *articulatory phonetics* literature.

The advantage of our model against the standard viseme grouping is that our viseme associations are less restrictive for substitutions: visemes which have little correlation can still be considered for substitutions according to the phonetic context they are taken into.

Note that the literature mentioned in this section considers only consonants, while vowels are mostly considered as separate visemes. Further reading on vowel study can be found in the work by *Montgomery and Jackson*[87].

MPEG-4 (*Binnie et al.*)	*Owens and Blazek*	*Walden et al.*	Nonlinear Model
/P,B,M/	**/P,B,M/**	**/P,B,M/**	**2113 2113 2123**
/F,V/	**/F,V/**	**/F,V/**	**1111 3112** $\lvert F\rvert \subset \lvert V\rvert$
/TH,DH/	**/TH,DH/**	**/TH,DH/**	**2122 2122**
/T,D/	-	-	2222 1123
/S,Z/	-	-	2122 1032
-	/T,D,S,Z/*	-	2222 1123 2122 1032
/K,G/	-	-	2222 2411
/N,L/	-		**2122 2122**
-	**/K,G,N,L/***	-	**2222 2411 2122 2122**
-	-	/T,D,S,Z, K,G,N,L/*	2222 1123 2122 1032 2222 2411 2122 2122
/CH,JH,SH/	-	-	**1011 1312 0212** $\lvert CH\rvert + \lvert SH\rvert \subset \lvert JH\rvert$
-	**/CH,JH,SH,ZH/***	**/CH,JH,SH,ZH/***	**1011 1312 0212 0002** $\lvert CH\rvert + \lvert SH\rvert + \lvert ZH\rvert \subset \lvert JH\rvert$
/W/	-	-	**6111** $\lvert W\rvert$ has a specific distribution that matched no other viseme
/R/	-	-	3212
-	/W,R/*	/W,R/*	6111 3212

$*$ are vowel context dependent and can thus be further divided. [91]

Table 12.1. Comparison of the present nonlinear viseme model with standard viseme grouping The groups where the nonlinear viseme model closely matches the standard grouping are highlighted in bold face. The last column also gives clarification as to the relation of the nonlinear model to the other groups: numerical closeness in the quadruplet notation indicates similar distributions or similar behavior of the visemes. Pairwise smaller digits indicate an inclusion, hence, valid substitutions.

12.2 Animation

To add realism to our animations, we added a few effects, some of them are learned from our recorded data. All are key-frame controlled.

Expressions are added in the lines of Section 8.5 in order to make the results look more lively. The recorded expressions have open mouth configurations, which prevents, the mouth from closing perfectly, when they are applied in the additive combination scheme (see Fig. 12.1).

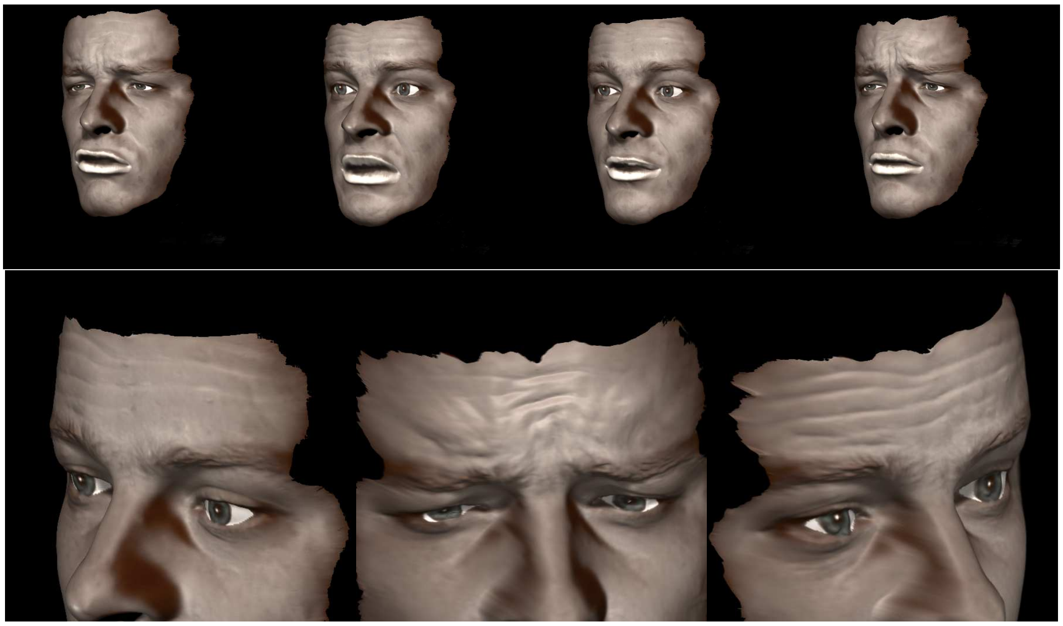


Fig. 12.1. Animating expressions The top row shows the reproduction of the four recorded expression vectors: disgust, fear, happiness and sadness, respectively. The bottom row shows a close-up: the texture is unaltered over the animation, all shadow effects are produced from the deforming geometry of the forehead.

Furthermore, eyecandies learned from the acquired data are added. The eye-blinking movement cannot be tracked by the optical flow in the registration step, as the eye disappears during that process. Instead, the eyelid movement was recorded from its motion associated to the direction, the actor is looking in. This movement, besides being reassociated to the movement of the iris (when looking up or down, Fig. 12.2-bottom), is exaggerated and reused for simulating eye blinking (see Fig. 12.2-top). Eye movements are produced by a simple, yet effective texture displacement in the specified region of the head mesh (*Deng et al.*[41]). All effects are controlled in a keyframe architecture.

During the generation of a video (Fig. 12.3), high frequency motions are smoothed out by a low-pass filter; we make sure to keep the cutoff frequency above the spectral range of natural mouth movements.

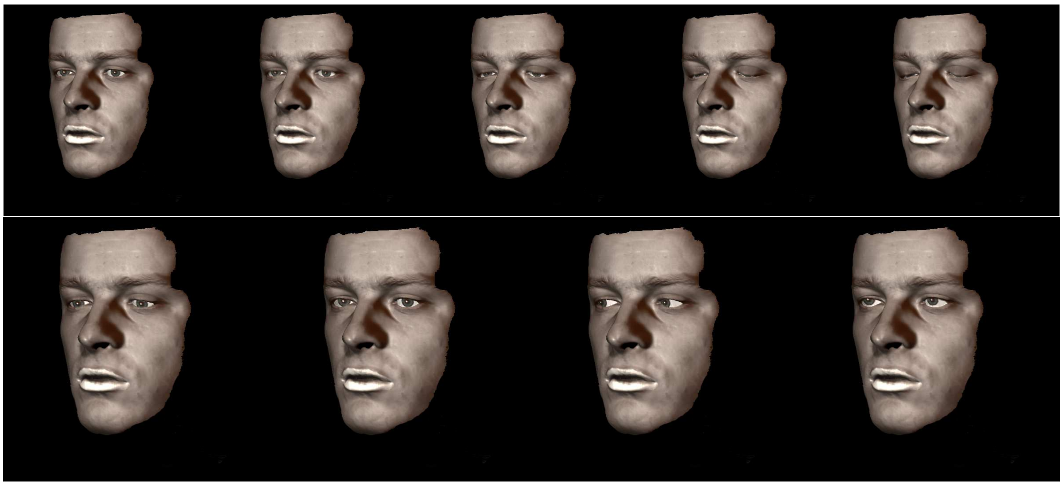


Fig. 12.2. Eye and eyelid motions The reanimation of the eyes is performed by a texture displacement over a region defined on the mesh (top). The motion of the eyelid is recorded from the actor and is applied for eye blinking and follows vertical eye movements (bottom).

When a long pause occurs in the novel sentence, the pause is synthesized by a triphone that spans many frames. The coarticulation is still learned from short triphones and is stretched over time, and the transition during the pause translates to a slow morphing from the last viseme of the previous sentence to the first viseme of the next sentence. This effect is neutralized by defining a time span limit over which the articulation should go to the neutral mouth configuration (PIT) and restart just before the next word.

Phonemes like $/P/$, $/B/$ and $/M/$ are often problematic in speech synthesis and dealt with separately. Indeed, their visual impact is a movement starting from a closed mouth. The problem is that the sound is perceived only once the mouth opens which puts the characteristic viseme before the frame sequence associated to the phoneme. This problem is here solved by the fact that a triphone-based approach is used and ensures the reproduction of the tangentiality at the viseme occurrence (see Section 11.1). Furthermore if we can guarantee smooth concatenation (minimal interpolation) in the selection process, the natural curve will be only slightly, altered and the typical movement for these phonemes remains.

Finally, the original recorded texture is edited to remove the marker points and add colors. The remaining white lipstick (which helped the optical flow algorithm) is still visible on the edited texture.

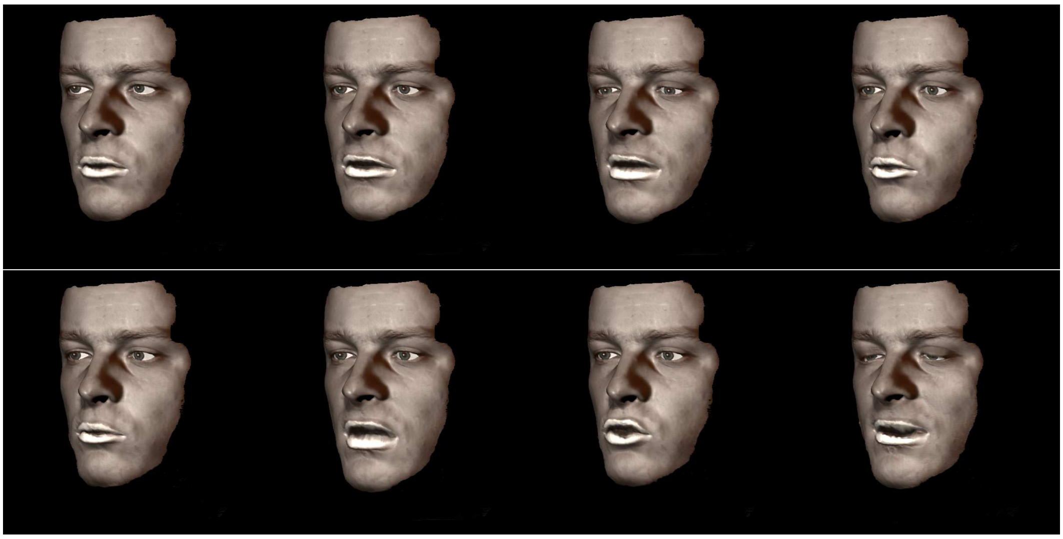


Fig. 12.3. Speech animation snapshots This figure gives snapshots of a speech animation synthesized from an audio file exerpt from the movie *Blade Runner*. Speech, expressions, and eye movements progress in parallel.

Part V

Discussions and Conclusions

In this thesis we introduced a new end-to-end setup for three-dimensional speech synthesis. Part I settled this work in the state-of-the-art context of speech animation regarding the different aspects involved in the building of such a system. In Part II, we clarified the theoretical concepts necessary to understand the approaches undertaken throughout the thesis, so that in Parts III and IV we could focus on exposing the contributions of this work.

In this last part, Section 13 discusses the key components of the system regarding their contributions to the field of articulation synthesis and how they can be improved. Moreover, we comment some of the results. Section 14 presents the directions for our furture investigations which would be based on our current system. Finally, we draw the general conclusions in Section 15.

13

Discussions

This chapter first discusses problems associated with the acquisition system and how they can be addressed if the system is rebuilt (Section 13.1). Section 13.2 intuitively validates the use of Local Linear Embedding as an analysis tool for speech synthesis. Nonlinear reduction techniques have brought up a controversy regarding their validity: by comparing our manifold representation with previous work, the observed consistency of the low-dimensional manifold structure validates the nonlinear viseme model.

While the viseme model is one part of the contributions of this thesis, the realism of the synthesis of animation, which indirectly validates the model, is observed in Sections 13.3 and 13.4. First, a perception test is performed in order to quantify the global quality of the animations while the second section compares the results against a more trivial synthesis approach (Section 8.6).

13.1 Improving the Data

The speech analysis throughout this thesis is heavily dependent on the substance the dynamic 3D scanner is capable to offer on both aspects, face coverage and mesh quality. A considerable amount of work was put into improving the raw data in order to extract a maximum of the available information residing inside the data (Sections 7.2 and 7.3). Since the time this work has been initiated, several new dynamic scanner prototypes have emerged showing an increasing popularity. For instance, the data on which the research of *Edge & Hilton*[44] is based shows a full colored coverage of the face integrating structure of the teeth.

An important drawback of the present 3D scanner is the fact that the geometry of the teeth and the tongue could not be acquired, while both have an important impact on the visual recognition of speech. Therefore, in our study, a statistical analysis of the behavior of the lips has been carried out. The strong correlation of the lips with the tongue and the teeth ensures the validity of the modelling undertaken. However, the availability of these components would still increase realism in the animations considerably. Moreover, there is another actor in the visual speech process, namely the Adam's Apple, specific to male subjects, which would further increase naturalness. Ideal scanners should hence be able to include the throat in a face model.

Chapter 9 presented the way, how our face model can be completed by combining the viseme morphable model to a full face identity morphable model. A dynamic scanner that delivers full facial surface would drastically improve the face transfer process to such a morphable model. Indeed, the face completion process would be circumvented and the problems of both importing novel face and exporting articulations could be performed directly, as soon as the mapping function is defined.

13.2 LLE in a Speech Synthesis Frame Work

In Chapter 6, we presented several nonlinear dimension reduction techniques. While these have found many applications, controversial discussions arose in regard to their validity as a means of representing underlying data structures. First, while these techniques always generate a new representation, they cannot certify the actual existence of a low dimensional manifold. Their usage is hence performed under such an assumption. Secondly, which approach is best suited for what application remains n a way a "black art", and the selection is often based on what history has already proven as to be.

The use of an LLE approach was inspired by the works of *Saul & Roweis*[109] or *Graf & Cosatto*[53] which show similar structures on 2D mouth configurations (see Fig. 10.2). As it is shown in Section 10.1.2, similar results are obtained for our 3D data, the representation always forms a star-shaped manifold with varying numbers of directions. This star-shaped manifold does, however, not reflect general mouth shapes but rather a sub-set containing only shapes involved in the speech. The extremities of the star illustrate strong articulated visemes, while the center is populated by neutral shapes. When the mouth goes towards an extreme configuration such as the viseme $|O|$, it looses degrees of freedom due to the physical constraints such as tissue strain. Moreover, clear acoustic signals are likely to imply optimal mouth shapes in many phonemes. That means that the space around these extreme configurations is not densely populated, and connected to the neutral configurations only by a single, locally one-dimensional distribution.

The manifold for the smaller corpus differs from the one for the large corpus by its number of branches (3 against 4). The similarity measure presented in this work was developed by keeping that aspect in mind: the notation can be adapted to the number of branches and it works as long as such typical manifolds are produced. The star manifold was obtained for the two data collections but also shows up for two-dimensional data. While it cannot be theoretically guaranteed that such an outcome can always be produced, previous and present works have shown such assumptions to be valid. The fact that the similarity model presented in this thesis is based on such an assumption could hence be another source of controversy.

13.3 Perception Test

For evaluating the overall quality of our animations, a lip-reading test is performed on 5 sentences (each with 6 syllables). A group of 14 *naive* persons (none of the subjects was trained in lip-reading) is asked to associate the 5 sentences

A: “No time for coffee breaks”
B: “This is the solution.”
C: “A brand new computer”
D: “He’s usually on time.”
E: “There is no Santa Claus.”

given as text and audio, to 5 silent articulation sequences. Each subject is allowed to listen to the audio as many times as he/she wishes in order to get accustomed to the speech rhythm. The animations are then visualized with

subject no	01	02	03	04	05	06	07	08	09	10	11	12	13	14
sentence A	**A**	E	**A**	B	C	**A**	**A**	**A**	D	E	**A**	**A**	**A**	D
sentence B	**B**	**B**	**B**	C	A	**B**	**B**	**B**	**B**	**B**	**B**	**B**	C	**B**
sentence C	**C**	**C**	**C**	A	B	**C**	**C**	**C**	A	D	**C**	**C**	B	**C**
sentence D	**D**	A	**D**	**D**	**D**	**D**	E	**D**	C	A	**D**	**D**	**D**	E
sentence E	**E**	D	**E**	**E**	**E**	**E**	D	**E**	**E**	C	**E**	**E**	**E**	A
correct	5	2	5	2	2	5	3	5	2	1	5	5	3	2

Table 13.1. 14 *naive* subjects are asked to associate 5 sentences (A to E) in their text form to 5 animation sequences. The subjects are allowed to get accustomed to the speech rhythm by listening the the sentences in their audio form *prior* to the visualization.

only the written sentences at disposal. The subjects can repeat the visualizations until they find all associations. While visualizing the animations, the subject is not allowed to listen to the audio tracks again. Table 13.1 show the associations done by the subject and for each of them, the number of correct matches.

The results do not show perfect associations. However, from the 14 subjects, six had 5 correct matches, two of them had 3, five had 2 and one had 1 (note that 4 correct matches cannot occur, as a single permutation already induces 2 errors). This gives us an average of 3.36 correct associations per person with the median at 3 (see Table 13.2). A combinatorial analysis[1] of the experimental task reveals that the expected value of correct responses per person would be 1.0 if the stimuli would convey no information at all, i.e. if subjects would merely guess. Our experimental results are, therefore, well above chance level, indicating that even for non-trained observers, much of the characteristic details of visemes can be perceived and distinguished.

	experimental case 14 experiments	probabilistic case 140 permutations
5 correct:	6	1
4 correct:	n.a.	n.a.
3 correct:	2	10
2 correct:	5	20
1 correct:	1	45
0 correct:	0	44
mean:	47/14 = 3.36	1.0
median	3	1

Table 13.2. While results do not show perfect association for all of the subjects, the average and mean value of the experiment are above those that random associations would produce.

13.4 Comparison to Other Algorithms

We performed two algorithms comparisons. The first one compares the final animation algorithm to the one proposed in Section 8.6, which uses the LLE-based substitution principle on a smaller corpus. The second comparison observes the improvement that the LLE-based similarity measure over one that compares the viseme clusters in the PCA representation.

---

[1] The possible combinations were found by implementing a simulation.

LLE on the Smaller Corpus

In Section 8.6, a simple algorithm for generating animations was proposed. At that point, the viseme model was built, but no analysis had yet been performed of the behavioral aspect of speech. A naive approach was undertaken which proposes to take the average of each cluster as a unique viseme sample for each of the phonemes. The resulting animations (see Fig. 8.12) show coherent articulations but they lack expressiveness. The synchronization of visemes to phoneme occurrences already produces realistic effects with the articulation movements matching the phonetics of the audio. However, by only performing a weighted interpolation between the visemes, the coarticulation effect is missing. In particular, the *plosives* $/P/$ and $/B/$ of the *nasal* $/M/$ which all start with a closed mouth, are not well simulated; because the sound produced by these consonants occurs only once the mouth is already open, the visemes associated to them do not illustrate the the typical burst of the lips. This movement is thus missing in the final animations. The PCA representation of the visemes also shows that the samples of a cluster are distributed sparsely (Fig. 10.1), hence their average shape is likely to be close to a neutral face shape.

The speech synthesis model presented in Part IV integrates original motion segments in the final animation. For the particular case of the plosives and the nasal mentioned above, their characteristic articulation is included in the motion segments they are associated to. With the large selection that the similarity measure offers, minimal interpolation is guaranteed and these movements are recreated in the final animations. The honest reproduction of original segments also guarantees that strong articulations are selected, which increases the expressiveness of the synthesized animations (see Fig. 12.3).

The animations produced from the small corpus present similar aspects (Fig.10.3). However, because it offers less samples, the algorithm reaches the $|PIT|$ state in the substitution hierarchy (see Section 10.3). The consequence is that the mouth sometimes comes to a neutral position when a phoneme is expected to be uttered. While the animations based on the small corpus show realistic behavior, the large corpus allows a considerable improvement in coherence and expressiveness.

A Similarity Measure in PCA Space

As we have described in Section 10.1, the distribution of the different viseme clusters in PCA space are difficult to classify. We propose here a simple algorithm, which, instead of using the LLE-based similarity measure, compares the clusters in their PCA representation. For that purpose, for each cluster we consider its average viseme and measure, for each cluster pair, the Euclidean distance between them. Substitutions are then taken from the closest clusters according to these distances. Unlike the LLE-based method, the substitutions possibilities are here symmetric. While the concatenation cost remains the same to the LLE-based synthesis, the substitution cost is given by the distances between the average visemes. The algorithm performs as follows:

1. once the triphones required for the new sentence are given, the database is queried for suitable samples;
2. if no sample is found, the algorithm looks for triphones that combine visemes that minimize the substitution cost;
3. like in the original algorithm, for each required triphone, lists of samples are selected for each queried triphone and the optimal sequence is found by minimizing both costs over the whole sentence.

Due to the symmetric substitution rule, the PCA-based triphone selection has to be restricted. Indeed, with their associated cost, all triphone segments in the corpus are potential substitute to each other and they all appear in the substitution lists for each triphone (columns in Figure 11.3). With such long lists, finding the optimal sequence becomes extremely time consuming. The restriction can be settled in two ways: for each cluster, (1) a maximum number K of substitution clusters or (2) a maximum substitution cost D can be defined. As we will show, both parameters are involved in the synthesis process.

In a first run, the PCA-based synthesis is performed by limiting only the substitution cost D. Aside from a few artefacts, the generated animations show comparable results with the LLE-based approach. By observing the triphone segments involved in both syntheses, it turns out that the majority of the segments have their central viseme taken from the desired cluster. In fact, the two methods occasionally happen to select the exact same segments. The reason to this, is that the substitution of the central viseme in a triphone is penalized by a higher cost that the *left* and *right* visemes. The database has sufficient samples for each cluster to insure that a good concatenation can be found by selecting, in most cases, segments from the requested cluster. Judging from the animations, while both look realistic, the PCA-based video shows, at some points, movements that are more questionable.

We pushed the comparison further by reducing the size of the corpus in order to increase the number of occurrences where substitutions are necessary. Three reduction iterations are performed, each time dividing the size of the corpus by two. At each iteration, an animation is generated for both the LLE- and the PCA-based methods taking the substitution rules defined on the large corpus for both methods, i.e., respectively, the quartet values are kept and the average viseme distances are the ones computed on the full corpus.

Reducing the size of the corpus demands an adaptation in the PCA-based method of both parameters K and D. In the query process, if K is too small, finding a triphone candidate is not anymore guaranteed and K must be increased. However, this increase has the consequence that the lists for other queries become again extremely large. Again, this can be corrected by reducing D. For each reduction iteration of the corpus, both parameters have to be adjusted. The LLE-based method, on the other hand, finds substitutes without further adjustment.

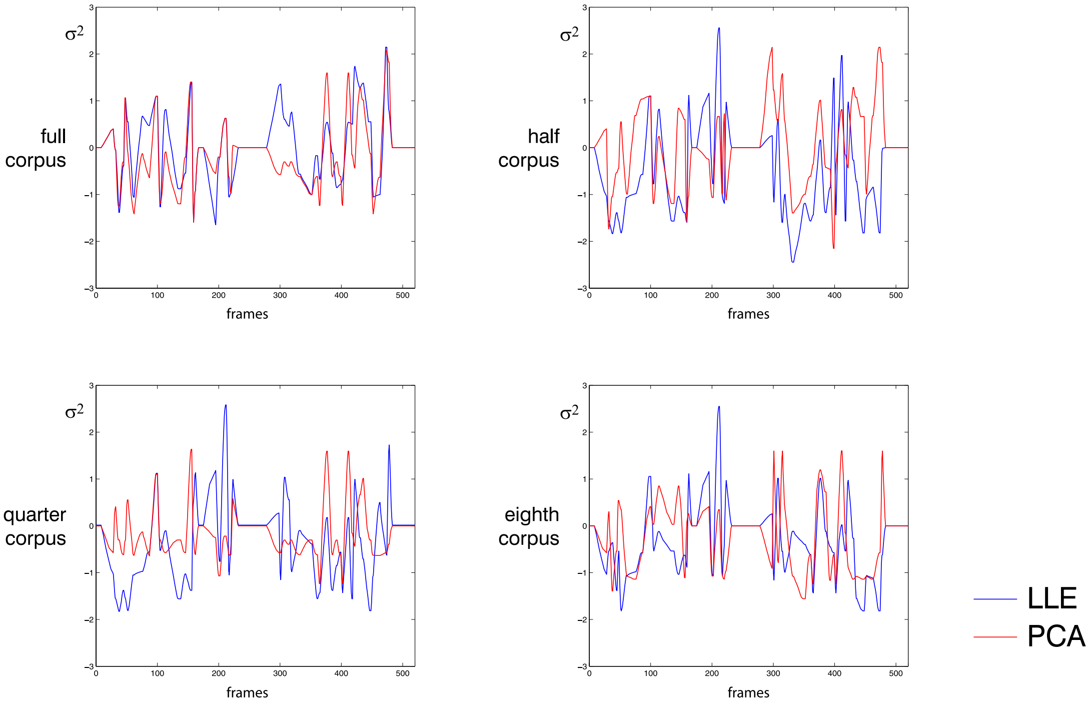


Fig. 13.1. Animation curves along the first principal component for both the LLE- and the PCA-based substitution rules with decreasing corpus sizes Both selection method provide similar animations with the full corpus as substitutions occur less often. By decreasing the size of the corpus, the two methods perform different substitutions which is reflected in the synthesized animations. The perceptual correctness in the animations is higher when the LLE-based substitution rule is used. The reason to that is that the PCA-based rule selects triphone segments that do not visually fit the targeted sentence.

The generated animations after each iteration, show a decrease in realism for both methods. However, the perceptual correctness is higher for the LLE-based approach. Figure 13.1 illustrates the variations of the first coefficient along the first principal component in an animation sequence. For each corpus size, the animation curves for both methods are superimposed. As it can be seen, already with half the corpus size, both methods select different segment samples. In the first plot, however, one can see where the two methods used the same segments.

Figure 13.1 only shows that the selections for substitutions differ for both methods when the size of the corpus deceases. The correctness of the substitutions can be judged by visualizing the generated animations. An important point is that by its nature, a triphone-based synthesis approach generates animations where the mouth moves in rhythm with the phoneme changes in the audio track, which gives a first realistic impression. However, the credibility of the animation is further increased when the selection of animation segments is correct. A large corpus increases the chances that substitutions are found with the central viseme corresponding to the one requested. The synthesized animation produced by both methods show realistic behavior with slight artefacts for the PCA-based method. By decreasing the size of the corpus, we force substitutions, and the two methods are more likely to produce different animations. The obtained videos show that the behavioral correctness remains better with the LLE-based substitution rule than with the one based on the distances between the average visemes of the clusters in the PCA space. The LLE-based substitution is thus better suited for small corpus.

14

Future Work

The focus of this thesis was to find a novel approach for automatic generation of realistic articulation animations. The applications for such a system are multiple, to cite only a few: web-based avatar animation [31], character animations in video games [39] or even educative software [83]. The most targeted application, however, remains the movie industry and the emerging animation films that come to theatres, which use the highest technological standards. While the quality achieved for speech animation in such productions show extremely realistic movements, these are directly captured and transferred via motion capture techniques. Hence, all dialogues have to be performed by actors which is an expensive and fastidious, yet reliable process. Automated techniques can offer a more practical alternative, though they remain illusory as long as the quality is compromised. Owing to the sensibility of the spectator to mouth articulation, the targeted realism is extremely high and automated approaches have still a long way to go.

Adding Realism

This thesis proposed an approach to understand and simulate neutral speech from a single actor. There is, however, not a unique manner of speech as it differs form one individual to the other. Also, expressions alter the articulation model as do speech intonations (*Brand*[21]). A complete automated system should encapsulate all these aspects in order to achieve the natural polyvalence of speech. This polyvalence comes at the cost of extremely large acquisition sets; several actors are to be recorded performing large sets of sentences with different expressions. While up to this, only manners of articulations are considered, several more elements come into play which are correlated to face animation. An animated head requires head movements [42, 25], eye motions [40], eye expressions [41], non verbal expressions and so on.

Nevertheless, from the the present system several paths can be followed. First, improvements on the shape quality should be undertaken. Novel scanning technologies would greatly improve the system, but would require rebuilding the viseme model. Alternatives can be considered such as adding a teeth and a tongue model. Adding teeth can be done in a straightforward approach, as they are attached to the skull. Upon determination of locations on the face shape which remain at rest relative to the skull, the position of the upper and the lower jaws can by evaluated.

Regarding the separation of expression and speech, several techniques have been briefly described in Section 2.4 (Related Work). An interesting approach that has not yet been used in conjunction with facial animation, separates human motion styles: *Hsu et al.*[62] use a linear time-invariant (LTI) [80] model together with an iterative motion warping (IMW) in order to perform *style translations*. Their model relaxes the constraints of motion alignment. For facial animation, the recorded corpus would only require utterances of sentences with varying expressions instead of repeating the same sentences with different expressions, thus, reducing the size of the corpus.

A Tongue Model

A tongue model, on the other hand, requires an extension of the model. If the correlation between the behavior of the lips and the tongue is determined, the model can be incorporated to our viseme model associating each viseme to a specific tongue position. However, a more thorough study would be preferred. In our model, visemes with similar lips shape but different tongue position are associated. An analysis capable of tracking the tongue along with the lips could typically generate further separation in an LLE representation, producing a star shaped manifold with more branches. It would be interesting to see how the viseme model performs in such cases.

Speech in Expressive Context

Another way is to record the same corpus with varying expressions. Having a neutral corpus, the expressive component can be extracted. Previous work (*Deng et al.*[41]) has shown how to extract the expression components from an audio track. In a new framework, these components could steer expression vectors over a synthesized articulation, hence offering a system which automatically generates expressive speech animations only from audio.

A Proof of Concept

As just discussed, the options for further work are multiple. However, the thesis exposed the dilemma of the corpus size to construct the viseme model.

Section 10.2 presented the quadruplet notation which describes viseme clusters and that identified in Table 10.1. We showed that by increasing the corpus, these models would further converge and give a more exact description of the visemes. The cluster $|ZH|$, for instance, consists of only a single sample which has a direct repercussion on its quadruplet: [0002]. While this is the only viseme with so few samples, it shows that 2500 visemes are still too few. However, the corpus is large enough to build a similarity graph which is precise enough to generate natural animations. The substitution rule, here, provides a graded similarity measure which offers substitution possibilities over a small corpus. Hence, the dilemma lies in the fact that as soon as the similarity table is known, the corpus can remain rather small due to the high redundancy of articulation segments, but in order to build the table, a large corpus is needed. It would be desirable to construct a much larger corpus which ensures at least 20 to 30 samples for each cluster and to derive a definitive table. Once this table is available, the similarity graph can be reused as-is for much smaller corpora. As a proof of concept, we suggest to use the given table and build a small 2D system based on a few video sequences recording only mouth movements. The registration can model a set of displacement fields relative to a reference frame and derive a small database of triphone deformation (along the lines of *Ezzat & Poggio*[46]). The selection for novel animation should follow the graph given in Section 10.3.

15

Conclusions

This work proposes a new data-driven end-to-end system for automatic speech animation. Based on three-dimensional face data recorded from emerging dynamic acquisition technologies, the system builds a Morphable Model for articulation. The geometric correspondence over the recorded frames is obtained by a robust multistage optical flow technique, which guarantees accurate tracking over indefinitely long sequences with large deformations. The precision of the correspondence is demonstrated by applying a unique texture over the registered faces and verifying that no flickering occurs, when the original animation is recomposed. At each timestep, the texture perfectly matches the geometry which guarantees that every vertex position is consistent over the reference.

The recoded 3D frames are projected onto the viseme-space described by the Morphable Model. Labelled with their associated phonemes, the frames describe smooth curves that are segmented into triphone units. This triphone database describes articulation motions as curve segments in a 50-dimensional viseme-space in a Multilinear Morphable Model framework. The morphable model offers a simple, yet effective, model for generating novel animations by morphing between selected triphones and guarantees the transfer of original coarticulation to novel animation syntheses.

While synthetic sentences are generated in a triphone concatenation framework, the novelty resides in the way they are selected. We take full advantage of the dual association between phonemes and visemes: not only can a phoneme take the visual appearance of several visemes, but visemes can be attributed to phonemes with which there is no *a priori* association. The difficulty is to determine, whether a viseme associated to a single phoneme in the original corpus can be used for simulating the utterance of a different phoneme. For that purpose, a similarity measure among visemes was developed which goes beyond the use of standard viseme groupings defined in *articulatory phonetics*: we propose a measure to gradualy distinguish viseme clusters and to build a hierarchical substitution rule. Furthermore, the measure is directly extended to triphone similarities. The benefit of this approach lies in the tradeoff between database size and realism: while a rather small corpus of real data information is kept, realistic articulation motions are generated by finding the optimal combination of triphones.

Another important contribution lies the use of a *Locally Linear Embedding* (LLE) for a statistical analysis of the data which, by its nonlinear nature, generates an intuitive representation for the study of viseme behavior. The justification for using a nonlinear representation for a viseme model has also been discussed.

Finally, the hierarchical structure of our selection method, which we derived from the data, is intrinsic to the nature of mouth articulation and can thus be reused as-is for other speech animation systems. While a larger data corpus would increase the accuracy of the similarity measure, the system is able to recreate realistic and expressive articulations from a rather small amount of data.

16

Appendix

16.1 The Corpus

In what follows, we give the words and sentences used for recording the corpora. For the first corpus, only the set *Long Sentences A* was used. The second corpus further included the *Short Sentences* and the *Long Sentences B* sets. The *Syllable Words* sets were not recorded.

The sentences in these sets have the particularity that they elicit no emotions from the subject. In addition, consonants occur in numerous vocal contexts in order to generate a wide spectrum of articulated syllables. The authors are thankful to Tony Ezzat, Gadi Geiger and Tomaso Poggio for providing this corpus of sentences, which they used in their work on *Trainable Videorealistic Speech Animation*[47].

One Syllable Words

leaf
cheat
ask
plant
vest
jaw
round
back
cave
cell
life
mall
mate
zoo
find
brunt
nest
bump
mail
roof
hide
plow
shell
all
ball
hand
get
cash
art
park
hook
curl
scar
good
top
bathe
with
fair
dark
boost
and
legs
bird
yes
cube
tea
check
deep
bee
spoil
loan
tall
lounge
wrong
pod
grudge
dwarf
want
kid
deaf
bed
risk
sting
knife
death
web
play
you
lift
arch
sag
glare
hair
faith
lab
zoom
bold
jam
coin
huge
scale
this
lack
feet

zip
bead
bud
noise
cap
name
gift
light
noun
shark
who
gain
flesh
cat
pane
foot
dirt
cow
cook
shock
air
dam
rule
man
was
blur
yacht
view
tail
chain
boil
hole
veil
gene
bang
rob
done
quail
wade
bid
beak
ring
mild
growth
tribe
paint
school
hill
brunch
tug
health
nap
trim
zone
badge
point
howl
safe
then
boats
once
veal
call
out
bounce
dive
fell
fad

Two Syllable Words

foible
afraid
daisy
saddle
warship
tattoo
assume
never
depend
version
labor
around
canon
other
annoy
mother
earring
ambush
fiber
dealer
midwife
keyboard
mammal
arrow
treasure
vision
alive
palace
earring
gunshot
heater
magnet
hybrid
shortfall
intrude
dagger
railway
cancer
cable
backward
boiler
themselves
finger
butcher
giant
endorse
fusion
lady
oilfield
obey
gateway
motive
amiss
amaze
loosen
downward
ramble
carving
pleasure
pillow
dermal
accept
thereby
dancer
pothole
fairest
baggage
caress
issue
asian
browser
agent
iceberg
onion
simmer
pamper
mismatch
weatheralways
garlic
abduct
cushion
helmet
tabloid
danger
accept
insure
haggle
knick-knack
obscure
flower
wander
haircut
halfway
decade
machine
measure
compound
baggage
legal
yearly
measure
upper
orchard
bazaar
monkey
angle
landmark
thicken
porridge
altar
teacher
nightmare
author
massive
landslide
rather
kidney
often
adopt
lavish
thirteen
aboard
garbage
thursday
hammer
anchor
barren
tasty
approve
zebra
waffle
nourish
showcase
leader
travel
volume
vibrate
bracket
youngest
feedback
narrow
wealthy
chuckle
prolong
watcher
limit
withdrew
backup
boycott
upscale
mishap
closure
canal
something
toothless

Short Sentences

01. I'm not happy with them.
02. We'll come back to you later.
03. The meeting was frank.
04. You can't stop science.
05. Thank you, Diane.
06. What can he do?
07. Not at all, Tom.
08. More news in a moment.
09. We're going to go back.
10. Another note about crime.
11. It was not the same.
12. Back in just a moment.
13. That's all for tonight.
14. I just don't understand it.
15. Time and again, he endures.
16. Good night.
17. Americans spend too much.
18. His name is Morgan.
19. Clear something up for me.
20. It gained twenty-seven points.
21. Good evening.
22. We have a big story.
23. Thank you very much.
24. There is no access.
25. One juror cried.
26. Others looked sad.
27. He opposes the Americans.
28. Have a good evening.
29. Eat the fish later.
30. Fifty-three percent.
31. Ken is at the courthouse.
32. It's really a jam.
33. They didn't do it.
34. That is our report.
35. It's a matter of money.
36. Who will they believe?
37. It was an easy job to do.
38. It was a stunning defeat.
39. But now we're going to New York City.
40. It could take months.
41. It's impossible.
42. We'll tell you about that later.
43. The trials are the same.
44. There's much more to this.
45. They are two leaders.
46. It was a clear job.
47. Just a moment.
48. They didn't do their job.
49. They put up a giant tent today.
50. We had a good meeting.

Long Sentences A

01. The birch canoe slid on the smooth planks.
02. Glue the sheet to the dark blue background.
03. It's easy to tell the depth of a well.
04. These days a chicken leg is a rare dish.
05. Rice is often served in round bowls.
06. The juice of lemons makes fine punch.
07. The box was thrown beside the parked truck.
08. The hogs were fed chopped corn and garbage.
09. A rod is used to catch pink salmon.
10. Read verse out loud for pleasure.
11. Hoist the load to your left shoulder.

Long Sentences B

01. Checking our top stories.
02. The fire is being allowed to burn itself out.
03. The probe will orbit the moon.
04. Temperatures will reach the single digits today.
05. Windy and rainy conditions are expected this week.
06. Snow showers are possible across the western Rockies.
07. He rushed for 112 yards.
08. He tossed in 31 points and 11 assists.
09. The government is giving workers the day off.
11. The quake was felt 150 miles away.
11. Warmer temperatures are predicted in some areas.
12. They say it would require more money.
13. This will be her second try.
14. Tokyo's stocks fell once again.
15. The Dow is now at fifty-five points.
16. Oil prices hit two year lows yesterday.
17. Both companies declined to comment.
18. Bond prices are up slightly.
19. It flew for the first time Sunday.
20. It cost an estimated 90 million dollars.
21. It was intended to fight the Cold War.
22. Monday morning could be a major headache for commuters.
23. No new negotiations are scheduled.
24. Traffic tie-ups are expected tomorrow.
25. The statue was closed to tourists Sunday.
26. Yesterday there was a lot of rain.
27. President Clinton is scheduled to speak Monday.
28. All these challenges will require cooperation.
29. We've learned how to operate more efficiently.
30. Millions of people participated.
31. The party featured lots of music.
32. There was a defect in the design.
33. Mini-vans sold from 1984 through 1995.
34. The vote may be delayed.
35. He is a 44 year old lawyer.
36. Sale and transport of weapons is banned.
37. It is not clear how effective they will be.
38. The economy is now on a good track.
39. They were not directly involved in what happened.
40. I don't think there are going to be any changes.

41. He’s not even giving public speeches.
42. They’ve got thousands of dollars.
43. He never spoke with John about the case.
44. The Dow Jones Industrial Average lost 125 points, to close at 73.
45. At this point we have to take action.
46. Oil may be running out.
47. Doctors recommend a healthy die.
48. The 30-year bond was up more than a point.
49. Both countries will have to cooperate together.
50. It looks to be a chaotic day.
51. The fundamentals are very good.
52. They wanted a foothold in Europe.
53. He wanted it translated into his native language.
54. They assured me it wasn’t going to happen again.
55. I think she was a trustworthy person.

16.2 Teeth Reconstruction

The dynamic 3D scanner used in this work cannot reconstruct teeth and tongue. The identity Multilinear Morphable Model (MMM), developed by *Blanz*[14], offers a reconstruction tool to recover such parts.

In a similar procedure to the one performed in Section 9.1, a 3D scan is transferred to the identity MMM in a fitting process. The scan is first aligned and scaled to the average head of the MMM and projected to a cylindrical representation. Optical flow techniques find the deformations that map the novel face shape to the model.

An interactive tool allows then to remove unwanted parts in the reconstructed face which is used to trim the inside of the mouth cavity. In a next step, feature points around the lips are given for the algorithm to detect the lip borders, and the teeth and the tongue can be reconstructed. Three scenarios are available in this software:

1. the software guesses the location of the mouth which produces different results for each scan;
2. in addition the mouth cavity is interactively emptied to improve detection;
3. if results are not satisfactory, feature points around the lips can be selected.

This process was abandoned, as we describe in Section 9.1, because the shape of the lips is not reconstructed accurately. Figure 16.1 shows resulting examples for the three scenarios.

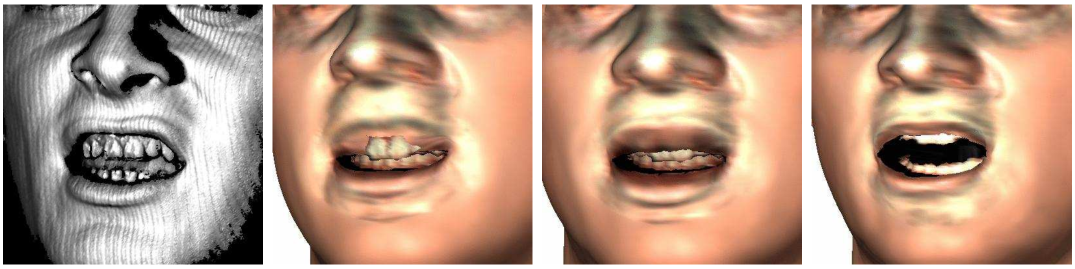


Fig. 16.1. Teeth reconstruction From left: the mouth as obtained from the scanner, reconstruction without mouth indication, reconstruction with tongue and teeth information removed by hand, and reconstruction with lips indication, respectively.

16.3 An Alternative for the Concatenation Cost

Section 11.1 introduces the *concatenation cost* φ that is used to measure the interpolation compatibility of triphones. In that section, we gave a first proposition for the cost which consisted in using the interpolation energy that measures the area between the two curves that need to be interpolated (Equation 11.4). We present here, a different approach that was undertaken before we opted for the error function in Equation 11.5.

This approach takes into account that, in the PCA-space, two successive ideal curves evolve in the same direction with the second one located in the continuity of the first one (in the front). In this regard, two parameters are observed: the relative directions of the curves and their relative position. These aspects are parameterized by two angles (Fig. 16.2-top center): the angle α measures the continuity ($\cos\alpha = 1$: good continuity, $\cos\alpha = -1$: bad continuity) and the angle β masures the relative directions ($\cos\beta = 1$: parallel, $\cos\beta = -1$: opposite directions). The table in the figure (Fig. 16.2-top right) evaluates the outcome of the desied cost function for four scenarios (Fig. 16.2-top left):

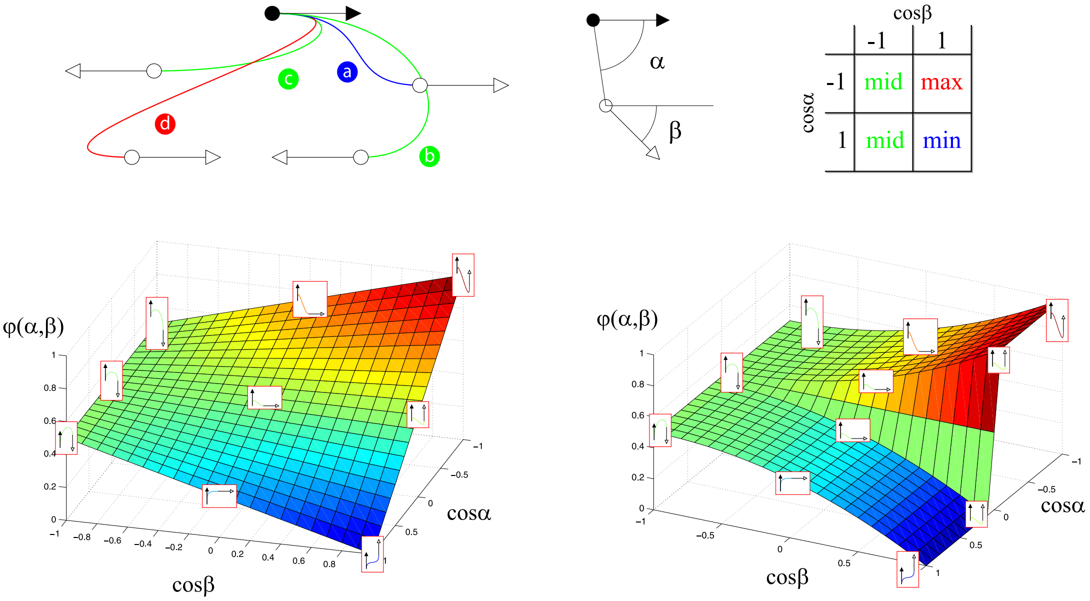
(a) ideally, the cosine of both angles is maximum; the curves evolve in the same direction and the second curve lies in the continuity of the first;
(b) the second curve lies in the continuity of the first, but a change of direction is necessary
(c) same as (b), but the second curve lies behind the first one which bears similar interpolation cost;
d the worst case, both cosines are negative, the interpolation is associated to the highest cost as the generated curve needs to go back to regain the same direction.

We are thus looking for a cost function that characterizes these different cases. According to the table, we are looking for a cost function that maps $(x, y) : [-1, 1]^2 \to [0, 1]$, with $x = \cos\alpha$ and $y = \cos\beta$. The function should have a minimum at $(x = 1, y = 1)$ and a maximum at $(x = 1, y = -1)$. For this purpose, we tested two functions:

$$\begin{aligned}\varphi_1 : [-1,1]^2 &\to [0,1] \\ (x,y) &\mapsto -\frac{1}{4}x(y+1)+\frac{1}{2}\end{aligned} \tag{16.1}$$

$$\begin{aligned}\varphi_2 : [-1,1]^2 &\to [0,1] \\ (x,y) &\mapsto -\frac{1}{8}\mathrm{sign}(x)(y+1)^2+\frac{1}{2}\end{aligned} \tag{16.2}$$

which are plotted in Figure 16.2-bottom.



	cosβ	
cosα	-1	1
-1	mid	max
1	mid	min

Fig. 16.2. The concatenation cost of two successive triphones looks for curves that offer best transition. This property is described by considering the angles α and β between the direction differences and their relative position. A quick table study tells us when the cost function should be maximal and minimal. We opted for the depicted function φ.

In the end, both functions offered similar results. However, the function that offered the best concatenations was the one that considered only the angle α. Hence, as already given in Section 11.1, the cost function that was used in the process of triphone selection is:

$$\begin{aligned} \varphi : [-1, 1] &\to [0, 1] \\ x &\mapsto \frac{x+1}{2} \end{aligned} \tag{16.3}$$

with $x = \cos\alpha$.

16.4 Map Distances

The example "Connecting Cities" on page 51 gives an introduction to nonlinear representation of data. The example shows how distances between cities in Switzerland are perceived differently from their actual geographical nature. In that case, a nonlinear representation gives a more intuitive visualization. The following table gives the geographical distance values used in the example. The road distances are taken from `http://travelguide.all-about-switzerland.info/major-swiss-cities-population-map-distances.html`, the air distances were measured by the author on a map.

air distances [km]

	Basel	Bern	Biel	La Chaux-de-Fond	Chur	Friburg	Genève	Lausanne	Luzern	Lugano	Montreux	Neuchâtel	St. Gallen	Schaffhausen	Sion	Thun	Winterthur	Zug	Zürich
Basel	-	68	53	78	159	89	187	136	72	197	133	88	130	74	146	89	82	78	69
Bern	68	-	24	48	156	24	128	78	67	153	68	46	156	119	78	26	114	86	94
Biel	53	24	-	31	172	34	131	81	79	178	79	36	161	119	99	51	118	96	99
La C-d-F	78	48	31	-	202	39	111	66	110	199	72	14	193	149	102	71	148	127	130
Chur	159	156	172	202	-	178	264	220	93	106	201	201	63	113	177	142	90	82	92
Fribourg	89	24	34	39	178	-	102	51	90	161	44	29	179	146	66	37	139	111	120
Genève	187	128	131	111	264	102	-	51	189	213	64	99	279	248	93	128	241	211	222
Lausanne	136	78	81	66	220	51	51	-	139	183	22	51	229	197	63	80	190	161	171
Luzern	72	67	79	110	93	90	189	139	-	127	126	111	89	73	116	61	58	22	39
Lugano	197	153	178	199	106	161	213	183	127	-	161	191	162	189	122	128	164	136	156
Montreux	133	68	79	72	201	44	64	22	126	161	-	57	214	188	41	64	179	148	159
Neuchâtel	88	46	36	14	201	29	99	51	111	191	57	-	197	156	90	64	153	129	134
St. Gallen	130	156	161	193	63	179	279	229	89	162	214	197	-	64	201	150	48	68	63
Schaffhausen	74	119	119	149	113	146	248	197	73	189	188	156	64	-	187	128	24	56	36
Sion	146	78	99	102	177	66	93	63	116	122	41	90	201	187	-	60	172	138	153
Thun	89	26	51	71	142	37	128	80	61	128	64	64	150	128	60	-	117	83	96
Winterthur	82	114	118	148	90	139	241	190	58	164	179	153	48	24	172	117	-	37	20
Zug	78	86	96	127	82	111	211	161	22	136	148	129	68	56	138	83	37	-	20
Zürich	69	94	99	130	92	120	222	171	39	156	159	134	63	36	153	96	20	20	-

road distances [km]

	Basel	Bern	Biel	La Chaux-de-Fond	Chur	Friburg	Genève	Lausanne	Luzern	Lugano	Montreux	Neuchâtel	St. Gallen	Schaffhausen	Sion	Thun	Winterthur	Zug	Zürich
Basel	-	94	89	98	207	125	255	188	98	306	189	118	160	98	264	120	104	115	85
Bern	94	-	33	68	244	31	161	94	90	262	95	46	207	164	170	28	147	114	122
Biel	89	33	-	44	240	49	157	97	108	295	113	29	203	142	188	61	143	132	118
La C-d-F	98	68	44	-	284	65	150	90	158	330	113	22	247	186	188	96	187	182	162
Chur	207	244	240	284	-	264	390	323	142	154	300	269	92	161	245	216	130	119	122
Fribourg	125	31	49	65	264	-	128	61	121	282	64	43	238	177	139	48	178	145	153
Genève	255	161	157	150	390	128	-	67	251	371	80	128	368	307	165	176	308	275	283
Lausanne	188	94	97	90	323	61	67	-	184	304	23	68	301	240	98	109	241	208	216
Luzern	98	90	108	158	142	121	251	184	-	208	185	136	121	100	245	92	79	24	54
Lugano	306	262	295	330	154	282	371	304	208	-	281	308	245	285	207	234	254	199	229
Montreux	189	95	113	113	300	64	80	23	185	281	-	91	302	241	79	112	242	209	217
Neuchâtel	118	46	29	22	269	43	128	68	136	308	91	-	232	171	166	74	172	160	147
St. Gallen	160	207	203	247	92	238	368	301	121	245	302	232	-	81	377	235	56	97	85
Schaffhausen	98	164	142	186	161	177	307	240	100	285	241	171	81	-	316	174	31	76	46
Sion	264	170	188	205	245	139	165	98	245	262	79	166	377	317	-	159	316	269	292
Thun	120	28	61	96	216	48	176	109	92	234	112	74	235	174	159	-	171	116	146
Winterthur	104	147	143	187	130	178	308	241	79	254	242	172	56	31	317	171	-	55	25
Zug	115	114	132	182	119	145	275	208	24	199	209	160	97	76	269	116	55	-	30
Zürich	85	122	118	162	122	153	283	216	54	229	217	147	85	46	292	146	25	30	-

References

1. H. Abdi. *Encyclopedia of Measurement and Statistics*, chapter Metric multidimensional scaling (MDS): Analyzing Distance Matrices, pages 598–605. Thousand Oaks (CA): Sage, 2007.
2. Alkiviadis G. Akritas and Gennadi I. Malaschonok. Applications of singular-value decomposition (svd). *Math. Comput. Simul.*, 67(1-2):15–31, 2004.
3. Irene Albrecht, Jörg Haber, and Hans-Peter Seidel. Speech synchronization for physics-based facial animation. In Václav Skala, editor, *Proc. 10th Int. Conf. on Computer Graphics, Visualization and Computer Vision (WSCG 2002)*, pages 9–16. UNION Agency, February 2002.
4. Robert Bargmann, Volker Blanz, and Hans-Peter Seidel. Learning-based facial rearticulation using streams of 3D scans. In *Proceedings of Pacific Graphics 2006*, page to appear, Taipei, Taiwan, October 2006.
5. J.L. Barron, D.J. Fleet, and S.S. Beauchemin. Performance of optical flow techniques. *Int. Journal of Computer Vision*, pages 43–77, 1994.
6. Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors. *Advances in Neural Information Processing Systems*, volume 15, Vancouver, BC, Canada, December 2003. MIT Press.
7. Yoshua Bengio, Jean-François Paiement, and Pascal Vincent. Out-of-sample extensions for LLE, Isomap, MDS, Eigenmaps, and spectral clustering. Technical Report 1238, Département d'informatique et recherche opérationnelle, Université de Montréal, 2003.
8. Yoshua Bengio, Pascal Vincent, Jean-François Paiement, Olivier Delalleau, Marie Ouimet, and Nicolas Le Roux. Spectral clustering and kernel PCA are learning eigenfunctions. Technical Report 1239, Département d'informatique et recherche opérationnelle, Université de Montréal, 2003.
9. Christian Benoît and Bertrand Le Goff. Audio-visual speech synthesis from french text: eight years of models, designs and evaluation at the icp. *Speech Commun.*, 26(1-2):117–129, 1998.
10. James R. Bergen, P. Anandan, Keith J. Hanna, and Rajesh Hingorani. Hierarchical model-based motion estimation. In *ECCV '92: Proceedings of the Second*

European Conference on Computer Vision, pages 237–252, London, UK, 1992. Springer-Verlag.

11. J.R. Bergen and R. Hingorani. Hierarchical motion-based frame rate conversion. Technical report, David Sarnoff Research Center Princeton NJ 08540, 1990.
12. J. Beskow and M. Nordenberg. Data-driven synthesis of expressive visual speech using an MPEG-4 talking head. In *Proceedings of Interspeech 2005*, Lisbon, 2005.
13. Anita Bickford and Rick Floyd. *Articulatory Phonetics: Tools for Analyzing the World's Languages*. SIL International, 4th edition, 2006.
14. V. Blanz. *Automatische Rekonstruktion der dreidimensionalen Form von Gesichtern aus einem Einzelbild*. PhD thesis, Universität Tübingen, Germany, 2000.
15. V. Blanz, K. Scherbaum, T. Vetter, and H.-P. Seidel. Exchanging faces in images. In M.-P. Cani and M. Slater, editors, *Computer Graphics Forum, Vol. 23, No. 3 EUROGRAPHICS 2004*, pages 669–676, Grenoble, France, 2004.
16. Volker Blanz, Curzio Basso, Thomas Vetter, and Tomaso Poggio. Reanimating faces in images and video. In Pere Brunet and Dieter W. Fellner, editors, *EUROGRAPHICS 2003 (EUROGRAPHICS-03) : the European Association for Computer Graphics, 24th Annual Conference*, volume 22 of *Computer Graphics Forum*, pages 641–650, Granada, Spain, 2003. The Eurographics Association, Blackwell.
17. Volker Blanz, Albert Mehl, Thomas Vetter, and Hans-Peter Seidel. A statistical method for robust 3D surface reconstruction from sparse data. In Yiannis Aloimonos and Gabriel Taubin, editors, *2nd International Symposium on 3D Data Processing, Visualization, and Transmission, 3DPVT 2004*, pages 293–300, Thessaloniki, Greece, 2004. IEEE.
18. Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 187–194, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.
19. Jean-Yves Bouguet and Pietro Perona. 3D photography on your desk. In *ICCV'98*, pages 43–52, 1998.
20. M. Brand. Charting a manifold. *In Advances in Neural Information Processing Systems*, 15, 2003. Cambridge, MA.MIT Press.
21. Matthew Brand. Voice puppetry. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 21–28, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.
22. Christoph Bregler, Michele Covell, and Malcolm Slaney. Video rewrite: driving visual speech with audio. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 353–360, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.
23. I. Buck, A. Finkelstein, C. Jacobs, A. Klein, D. Salesin, J. Seims, R. Szeliski, and K. Toyama. Performance-driven hand-drawn animation. In *Performance-driven hand-drawn animation. Proceedings of NPAR 2000*, June 2000.

24. Peter J. Burt and Edward H. Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, COM-31,4:532–540, 1983.
25. Carlos Busso, Zhigang Deng, Ulrich Neumann, and Shrikanth Narayanan. Natural head motion synthesis driven by acoustic prosodic features: Virtual humans and social agents. *Comput. Animat. Virtual Worlds*, 16(3-4):283–290, 2005.
26. Y. Cao, P. Faloutsos, and F. Pighin. Unsupervised learning for speech motion editing. In *SCA '03*, pages 225–231. Eurographics Association, 2003.
27. Yong Cao, Petros Faloutsos, Eddie Kohler, and Frédéric Pighin. Real-time speech motion synthesis from recorded motions. In *SCA '04*, pages 345–353. Eurographics Association, 2004.
28. Yong Cao, Wen C. Tien, Petros Faloutsos, and Frédéric Pighin. Expressive speech-driven facial animation. *ACM Trans. Graph.*, 24(4):1283–1302, 2005.
29. Jin-Xiang Chai, Jing Xiao, and Jessica Hodgins. Vision-based control of 3D facial animation. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 193–206. Eurographics Association, 2003.
30. Yao-Jen Chang and Tony Ezzat. Transferable videorealistic speech animation. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 143–151, New York, NY, USA, 2005. ACM Press.
31. Herng-Yow Chen and Sheng-Wei Li. Exploring many-to-one speech-to-text correlation for web-based language learning. *ACM Trans. Multimedia Comput. Commun. Appl.*, 3(3):13, 2007.
32. Su Chen and R.M. Haralick. Recursive erosion, dilation, opening, and closing transforms. *IEEE Transactions on Image Processing*, 4(3):335–345, March 1995.
33. Byoungwon Choe and Hyeong-Seok Ko. Analysis and synthesis of facial expressions with hand-generated muscle actuation basis. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Courses*, page 21, 2006.
34. Erika Chuang and Christoph Bregler. Mood swings: expressive speech animation. *ACM Trans. Graph.*, 24(2):331–347, 2005.
35. Michael M. Cohen and Dominic W. Massaro. Modeling coarticulation in synthetic visual speech. In D. Thalmann and ed. N. Magnenat-Thalmann, editors, *In Computer Animation '93*. Springer-Verlag, 1993.
36. Vin de Silva and Joshua B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In Becker et al. [6], pages 705–712.
37. Douglas DeCarlo, Dimitris Metaxas, and Matthew Stone. An anthropometric face model using variational techniques. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 67–74, New York, NY, USA, 1998. ACM.
38. Douglas DeCarlo and Dimitris N. Metaxas. The integration of optical flow and deformable models with applications to human face shape and motion estimation. In *CVPR*, pages 189–195, 1996.
39. Z. Deng, J. P. Lewis, and U. Neumann. Synthesizing speech animation by learning compact speech co-articulation models. In *CGI '05: Proceedings of the*

Computer Graphics International 2005, pages 19–25, Washington, DC, USA, 2005. IEEE Computer Society.

40. Zhigang Deng, J. P. Lewis, and Ulrich Neumann. Practical eye movement model using texture synthesis. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Sketches & Applications*, pages 1–1, New York, NY, USA, 2003. ACM.
41. Zhigang Deng, J. P. Lewis, and Ulrich Neumann. Automated eye motion using texture synthesis. *IEEE Comput. Graph. Appl.*, 25(2):24–30, 2005.
42. Zhigang Deng, Shri Narayanan, Carlos Busso, and Ulrich Neumann. Audio-based head motion synthesis for avatar-based telepresence systems. In *ETP '04: Proceedings of the 2004 ACM SIGMM workshop on Effective telepresence*, pages 24–30, New York, NY, USA, 2004. ACM.
43. Zhigang Deng and Ulrich Neumann. eFASE: expressive facial animation synthesis and editing with phoneme-isomap controls. In *SCA '06*, pages 251–260. Eurographics Association, 2006.
44. J Edge and A Hilton. Visual speech synthesis from 3D video. In *Proceedings of the 3rd European Conference on Visual Media Production*, page 174, November 2006.
45. P. Eisert, S. Chaudhuri, and B. Girod. Speech driven synthesis of talking head sequences. In Seidel, Girod, and Niemann, editors, *Proc. 3D Image Analysis and Synthesis '97 – Erlangen*, pages 51–56, infix, Sankt Augustin, 1997.
46. T. Ezzat and T. Poggio. Visual speech synthesis by morphing visemes. *International Journal of Computer Vision*, 38, 1:45–57, 2000.
47. Tony Ezzat, Gadi Geiger, and Tomaso Poggio. Trainable videorealistic speech animation. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 388–398, New York, NY, USA, 2002. ACM Press.
48. Marco Fratarcangeli, Marco Schaerf, and Robert Forchheimer. Facial motion cloning with radial basis functions in MPEG-4 FBA. *Graphical Models*, 69(2):106–118, 2007.
49. Emmanuel Garcia. Low cost 3D face acquisition and modeling. In *ITCC '01: Proceedings of the International Conference on Information Technology: Coding and Computing*, page 657, Washington, DC, USA, 2001. IEEE Computer Society.
50. Athinodoros S. Georghiades. Recovering 3-D shape and reflectance from a small number of photographs. In *EGRW '03: Proceedings of the 14th Eurographics workshop on Rendering*, pages 230–240. Eurographics Association, 2003.
51. Joseph (Yossi) Gil and Ron Kimmel. Efficient dilation, erosion, opening, and closing algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(12):1606–1617, 2002.
52. T. Goto, S. Kshirsagar, and N. Magnenat-Thalmann. Automatic face cloning and animation. *IEEE Signal Processing Magazine*, 18, 3:17–25, 2001.
53. Hans Peter Graf and Eric Cosatto. Sample-based synthesis of talking heads. In *RATFG-RTS '01: Proceedings of the IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (RATFG-RTS'01)*, page 3, Washington, DC, USA, 2001. IEEE Computer Society.

54. Brian Guenter, Cindy Grimm, Daniel Wood, Henrique Malvar, and Fredric Pighin. Making faces. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 55–66, New York, NY, USA, 1998. ACM Press.
55. R.M. Haralick, S. Chen, and T. Kanungo. Recursive opening transform. In *Computer Vision and Pattern Recognition, CVPR 1992*, pages 560–565, Champaign, IL, USA, June 1992.
56. R.M. Haralick and L.G. Shapiro. *Computer and robot vision*, volume 2. Addison-Wesley, Reading, Ma, 1992.
57. W.J. Hardcastle and A. Marchal. *Speech Production and Speech Modeling.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 1990.
58. R. Hashimoto. An extension of an algorithm for finding sequential decomposition of erosions and dilations. In *SIBGRAPHI '98: Proceedings of the International Symposium on Computer Graphics, Image Processing, and Vision*, page 443. IEEE Computer Society, 1998.
59. David J. Heeger. Model for the extraction of image flow. *Journal of the Optical Society of America A: Optics, Image Science, and Vision*, 4(8):1455–1471, 1987.
60. Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203, August 1981.
61. Roger A. Horn and Charles R. Johnson. *Matrix analysis.* Cambridge University Press, New York, NY, USA, 1986.
62. Eugene Hsu, Kari Pulli, and Jovan Popović. Style translation for human motion. *ACM Trans. Graph.*, 24(3):1082–1089, 2005.
63. Jennifer Huang, Bernd Heisele, and Volker Blanz. Component-based face recognition with 3D morphable models. In *Proc. of the 4th Int. Conf. on Audio- and Video-Based Biometric Person Authenticitation*, Surrey, UK, 2003.
64. X. Huang, F. Alleva, H.-W. Hon, M.-Y. Hwang, K.-F. Lee, and R. Rosenfeld. The SPHINX-II speech recognition system: an overview (http://sourceforge.net/projects/cmusphinx/). *Computer Speech and Language*, 7, 2:137–148, 1993.
65. Alexander Ihler. Nonlinear manifold learning. MIT lecture, October 2003.
66. Paul T. Jackway and Mohamed Deriche. Scale-space properties of the multiscale morphological dilation-erosion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(1):38–51, 1996.
67. Odest Chadwicke Jenkins and Maja J. Mataric. Automated derivation of behavior vocabularies for autonomous humanoid motion. In *AAMAS '03: Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 225–232, New York, NY, USA, 2003. ACM Press.
68. Michael J. Jones and Tomaso Poggio. Multidimensional morphable models. In *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, page 683, Washington, DC, USA, 1998. IEEE Computer Society.
69. Pushkar Joshi, Wen C. Tien, Mathieu Desbrun, and Frédéric Pighin. Learning controls for blend shape based realistic facial animation. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Courses*, page 8, New York, NY, USA, 2005. ACM Press.

70. Kolja Kähler, Jörg Haber, Hitoshi Yamauchi, and Hans-Peter Seidel. Head shop: generating animated head models with anatomical structure. In *SCA '02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 55–63, New York, NY, USA, 2002. ACM Press.
71. G. A. Kalberer, P. Mueller, and L. Van Gool. A visual speech generator. In S. F. El-Hakim, A. Gruen, and J. S. Walton, editors, *Videometrics VII 2003 IS&T/SPIE*, volume 5013, pages 46–53. IS&T, SPIE, January 2003.
72. G. A. Kalberer, P. Mueller, and L. Van Gool. *3D Modeling and Animation: Synthesis and Analysis Techniques for the Human Body*, chapter Modeling and Synthesis of Realistic Visual Speech in 3D, pages 266–294. In [107], 2004.
73. Gregor A. Kalberer, Pascal Müller, and Luc J. Van Gool. Speech animation using viseme space. In *VMV*, pages 463–470, 2002.
74. Ig-Jae Kim and Hyeong-Seok Ko. 3D lip-synch generation with data-faithful machine learning. In *Computer Graphics Forum*, volume 26, pages 295–301. Blackwell Publishing, September 2007.
75. Scott A. King and Richard E. Parent. Creating speech-synchronized animation. *IEEE Transactions on Visualization and Computer Graphics*, 11(3):341–352, 2005.
76. Sumedha Kshirsagar and Nadia Magnenat-Thalmann. Visyllable based speech animation. In P. Brunet and D. Fellner, editors, *Computer Graphics Forum, Vol. 22, No. 3 EUROGRAPHICS 2003*, pages 631–639, Granada, Spain, 2003.
77. Sumedha Kshirsagar, Tom Molet, and Nadia Magnenat-Thalmann. Principal components of expressive speech animation. In *CGI '01: Computer Graphics International 2001*, pages 38–46, Washington, DC, USA, 2001. IEEE Computer Society.
78. Peter Ladefoged. *A course in phonetics.* Heinle & Heinle, Boston, 4th edition, 2001.
79. Manfred Lau, Jin-Xiang Chai, Ying-Qing Xu, and Heung-Yeung Shum. Face poser: Interactive modeling of 3D facial expressions using model priors. In *2007 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, August 2007.
80. Lennart Ljung. *System identification: theory for the user.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1986.
81. A Löfqvist. *Speech production and speech modeling*, chapter Speech as audible gesture, pages 289–322. In [57], 1990.
82. B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI81*, pages 674–679, 1981.
83. Jiyong Ma, Ron Cole, Bryan Pellom, Wayne Ward, and Barbara Wise. Accurate visible speech synthesis based on concatenating variable length motion capture data. *IEEE Transactions on Visualization and Computer Graphics*, 12(2):266–276, 2006.
84. P. Marchand and Marmet L. Binomial smoothing filter: A way to avoid some pitfalls of least square polynomial smoothing. *Rev. Sci. Instrum.*, 54(1034-41), 1983.
85. D. Marr and T. Poggio. A Computational Theory of Human Stereo Vision. *Royal Society of London Proceedings Series B*, 204:301–328, May 1979.

86. Roger E. Millsap, editor. *Psychometrika*, volume 17. Springer, 1952.
87. A. A. Montgomery and P. L. Jackson. Physical characteristics of the lips underlying vowel lipreading performance . *Acoustical Society of America Journal*, 73:2134–2144, June 1983.
88. H.H. Nagel. Displacement vectors derived from second-order intensity variations in image sequences. *CVGIP*, 21(1):85–117, January 1983.
89. Bernd Neumann. Optical flow. *SIGGRAPH Comput. Graph.*, 18(1):17–19, 1984.
90. Jun-Yong Noh and Ulrich Neumann. Expression cloning. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 277–288, New York, NY, USA, 2001. ACM Press.
91. Elmer Owens and Barbara Blazek. Visemes observed by hearing-impaired and normal-hearing adult viewers. *Journal of Speech and Hearing Research*, 28:381–393, September 1985.
92. J. Pagès. *Assisted visual servoing by means of structured light.* PhD thesis, Universities of Girona and Rennes I, 2005.
93. Nikhil R. Pal, Nikola Kasabov, Rajani K. Mudi, Srimanta Pal, and Swapan K. Parui, editors. *Neural Information Processing, 11th International Conference, ICONIP 2004, Calcutta, India, November 22-25, 2004, Proceedings*, volume 3316 of *Lecture Notes in Computer Science*. Springer, 2004.
94. Igor S. Pandzic and Robert Forchheimer, editors. *MPEG-4 Facial Animation: The Standard, Implementation and Applications*. Wiley), 2002.
95. Richard E. Parent, Scott A. King, and Osamu Fujimura. Issues with lip sync animation: Can you read my lips? In *CA*, pages 3–10, 2002.
96. Frederic I. Parke. *A parametric model for human faces.* PhD thesis, The University of Utah, 1974.
97. Frederick I. Parke. Computer generated animation of faces. In *ACM'72: Proceedings of the ACM annual conference*, pages 451–457. ACM Press, 1972.
98. Catherine Pelachaud, Norman I. Badler, and Mark Steedman. Generating facial expressions for speech. *Cognitive Science*, 20(1):1–46, 1996.
99. F. Pighin, R. Szeliski, and D. Salesin. Modeling and animating realistic faces from images. *Int. Journal of Computer Vision*, 50, 2:143–169, 2002.
100. Frédéric Pighin, Jamie Hecker, Dani Lischinski, Richard Szeliski, and David H. Salesin. Synthesizing realistic facial expressions from photographs. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 75–84, New York, NY, USA, 1998. ACM Press.
101. Frederic H. Pighin, Richard Szeliski, and David Salesin. Resynthesizing facial animation through 3D model-based tracking. In *ICCV (1)*, pages 143–150, 1999.
102. Tomaso Poggio and Thomas Vetter. Recognition and structure from one 2D model view: Observations on prototypes, object classes and symmetries. Technical report, Cambridge, MA, USA, 1992.
103. Hyewon Pyun, Yejin Kim, Wonseok Chae, Hyung Woo Kang, and Sung Yong Shin. An example-based approach for facial expression cloning. In *SCA '03:*

Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, pages 167–176. Eurographics Association, 2003.

104. S. Romdhani, V. Blanz, and T. Vetter. Face identification by fitting a 3D morphable model using linear shape and texture error functions. In *Computer Vision - ECCV 2002, LNCS 2353*, pages 3–19, 2002.
105. S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, December 2000.
106. Alex Rudnicky. Sphinx knowledge base tools, 1999.
107. Nikos Sarris, Michael G. Strintzis, and Strintzis G. Michael. *3D Modeling and Animation: Synthesis and Analysis Techniques for the Human Body.* IRM Press, 2004.
108. Lawrence K. Saul and Sam T. Roweis. An introduction to locally linear embedding. Technical report, AT&T Labs - Research, 2001.
109. Lawrence K. Saul and Sam T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *J. Mach. Learn. Res.*, 4:119–155, 2003.
110. Ashutosh Saxena, Abhinav Gupta, and Amitabha Mukerjee. Non-linear dimensionality reduction by locally linear isomaps. In Pal et al. [93], pages 1038–1043.
111. Arno Schödl, Richard Szeliski, David H. Salesin, and Irfan Essa. Video textures. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 489–498, 2000.
112. Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.*, 10(5):1299–1319, 1998.
113. Sy sen Tang, Alan Wee-Chung Liew, and Hong Yan. Lip-sync in human face animation based on video analysis and spline models. In *MMM*, pages 102–108, 2004.
114. Jon Shlens. *Tutorial on Principal Component Analysis*, Dec 2005. http://www.snl.salk.edu/ shlens/notes.html.
115. Eftychios Sifakis, Igor Neverov, and Ronald Fedkiw. Automatic determination of facial muscle activations from sparse motion capture marker data. *ACM Trans. Graph.*, 24(3):417–425, 2005.
116. Eftychios Sifakis, Andrew Selle, Avram Robinson-Mosher, and Ronald Fedkiw. Simulating speech with a physics-based facial muscle model. In *SCA '06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 261–270. Eurographics Association, 2006.
117. A. Singh. *Optic Flow Computation: A Unified Perspective.* IEEE Computer Society Press, Washington, 1991.
118. Malcolm Slaney and Michele Covell. Matlab multidimensional scaling tools, May 1998. http://web.interval.com/papers/2000-025/.
119. Lindsay I. Smith. *A Tutorial on Principal Components Analysis.* Cornell University, Feb 2002.
120. Pierre Soille, Edmond J. Breen, and Ronald Jones. Recursive implementation of erosions and dilations along discrete lines at arbitrary angles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(5):562–567, 1996.

121. J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, December 2000.
122. Joshua B. Tenenbaum. Mapping a manifold of perceptual observations. In *NIPS '97: Proceedings of the 1997 conference on Advances in neural information processing systems 10*, pages 682–688, Cambridge, MA, USA, 1998. MIT Press.
123. Joshua B. Tenenbaum and William T. Freeman. Separating style and content with bilinear models. *Neural Comput.*, 12(6):1247–1283, 2000.
124. D. Terzopoulos and K. Waters. Physically-based facial modeling, analysis, and animation. *Journal of Visualization and Computer Animation*, 1(2):73–80, 1990.
125. W. S. Torgerson. Multidimensional scaling: Theory and method. In *Psychometrika* [86], pages 401–419.
126. J.B. Tenenbaum V. de Silva. *Nonlinear Estimation and Classification*, chapter Unsupervised learning of curved manifolds, pages 453–466. Springer-Verlag, New York, 2002.
127. L.J.P. van der Maaten. An introduction to dimensionality reduction using matlab. Technical report, Maastricht University, Maastricht, The Netherlands, 2007. http://www.cs.unimaas.nl/l.vandermaaten/Laurens_van_der_Maaten.
128. Thomas Vetter and Tomaso Poggio. Linear object classes and image synthesis from a single example image. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(7):733–742, 1997.
129. Daniel Vlasic, Matthew Brand, Hanspeter Pfister, and Jovan Popović. Face transfer with multilinear models. *ACM Trans. Graph.*, 24(3):426–433, 2005.
130. B. E. Walden, S. A. Erdman, A. A. Montgomery, D. M. Schwartz, and R. A. Prosek. Some effects of training on speech recognition by hearing-impaired adults. *Journal of speech and hearing research*, 24:207–16, June 1981.
131. Michael E. Wall, Andreas Rechtsteiner, and Luis M. Rocha. *Singular Value Decomposition and Principal Component Analysis*, chapter 5, pages 91–109. Kluwel, Norwell, MA, Mar 2003.
132. Kevin Wampler, Daichi Sasaki, Li Zhang, and Zoran Popović. Dynamic, expressive speech animation from a single mesh. In *SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 53–62. Eurographics Association, 2007.
133. Alice Wang, Michael Emmi, and Petros Faloutsos. Assembling an expressive facial animation system. In *Sandbox '07: Proceedings of the 2007 ACM SIGGRAPH symposium on Video games*, pages 21–26. ACM Press, 2007.
134. Y. Wang, X. Huang, C. Lee, S. Zhang, Z. Li, S. Samaras, D. Metaxas, A. Elgammal, and P. Huang. High resolution acquisition, learning and transfer of dynamic 3-D facial expressions. In M.-P. Cani and M. Slater, editors, *Computer Graphics Forum, Vol. 23, No. 3 EUROGRAPHICS 2004*, Grenoble, France, 2004.
135. Keith Waters. A muscle model for animation three-dimensional facial expression. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Com-*

puter graphics and interactive techniques, pages 17–24, New York, NY, USA, 1987. ACM Press.

136. Christopher K. I. Williams. On a connection between kernel pca and metric multidimensional scaling. *Mach. Learn.*, 46(1-3):11–19, 2002.
137. Kai Wolf. 3D measurement of dynamic objects with phase shifting techniques. In *VMV*, pages 537–544, 2003.
138. Tynia Yang, Jinze Liu, Leonard McMillan, and Wei Wang. A fast approximation to multidimensional scaling. In *Proceedings of ECCV 2006*, Graz, Austria, 2006.
139. Wai Chee Yau, Dinesh Kant Kumar, and Sridhar Poosapadi Arjunan. Voiceless speech recognition using dynamic visual speech features. In *VisHCI '06: Proceedings of the HCSNet workshop on Use of vision in human-computer interaction*, pages 93–101. Australian Computer Society, Inc., 2006.
140. Junping Zhang, Stan Z. Li, and Jue Wang. Nearest manifold approach for face recognition. In *The 6th IEEE International Conference on Automatic Face and Gesture Recognition*, Seoul, Korea, May 2004.
141. Li Zhang, Noah Snavely, Brian Curless, and Steven M. Seitz. Spacetime faces: high resolution capture for modeling and animation. *ACM Trans. Graph.*, 23(3):548–558, 2004.
142. Qingshan Zhang, Zicheng Liu, Baining Guo, and Harry Shum. Geometry-driven photorealistic facial expression synthesis. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 177–186. Eurographics Association, 2003.
143. Song Zhang and Peisen Huang. High-resolution, real-time 3D shape acquisition. In *CVPRW '04: Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 3*, page 28, Washington, DC, USA, 2004. IEEE Computer Society.
144. Yu Zhang, Edmond C. Prakash, and Eric Sung. A new physical model with multilayer architecture for facial expression animation using dynamic adaptive mesh. *IEEE Transactions on Visualization and Computer Graphics*, 10(3):339–352, 2004.
145. Yu Zhang, Terence Sim, and Chew Lim Tan. Adaptation-based individualized face modeling for animation using displacement map. In *CGI '04: Proceedings of the Computer Graphics International (CGI'04)*, pages 518–521, Washington, DC, USA, 2004. IEEE Computer Society.

Curriculum Vitea

July 9th 1976	born in Geneva, Switzerland
1983 - 1992	Primary School, Ecublens, Switzerland
1992 - 1996	Scientific High School Degree, CESSRIVE, Lausanne, Switzerland
1996 - 1997	Studies in Electrical Engineering, **Ecole Poytechnique Fédérale de Lausanne** (EPFL)
1998 - 2003	Studies in Communication Systems Engineering, EPFL
2000 - 2001	One year exchange within the ERASMUS program, **Universitat Politecnica de Catalunya** (UPC), Barcelona, Spain
2003	Master's Thesis, Computer Graphics Laboratory **Eidgenössische Technische Hochschule, Zürich** (ETHZ), Switzerland
Fall 2003	M.Sc. Degree in Communication Systems EPFL, Switzerland
2004 - 2008	PhD Thesis, Computer Graphics Laboratory **Max-Planck-Institut für Informatik** (MPII), Saarbrücken, Germany

Lebenslauf

Juli 9. 1976	geboren in Genf, Schweiz
1983 - 1992	Grundschule, Ecublens, Schweiz
1992 - 1996	Gymnasium mit Abiturabschluß, CESSRIVE, Lausanne, Schweiz
1996 - 1997	Studium in Elektrotechnik, **Ecole Poytechnique Fédérale de Lausanne** (EPFL)
1998 - 2003	Studium in Kommunikationssysteme, EPFL
2000 - 2001	ein Jahr Studentenaustausch im ERASMUS Programm, **Universitat Politecnica de Catalunya** (UPC), Barcelona, Spanien
2003	Masterarbeit, Computer Graphics Laboratory **Eidgenössische Technische Hochschule, Zürich** (ETHZ), Schweiz
Herbst 2003	MSc in Kommunikationssysteme EPFL, Schweiz
2004 - 2008	Doktorarbeit in Computergrafik **Max-Planck-Institut für Informatik** (MPII), Saarbrücken, Deutschland