

AUTOMATIC LIPREADING IN THE DUTCH LANGUAGE

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus, prof. dr. ir. J. T. Fokkema,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op
dinsdag 11 november 2003 om 13.00 uur
door

Jacek Cyprian WOJDEŁ
Master of Science in Engineering,
Technical University of Łódź,
geboren te Łódź, Polen.

Dit proefschrift is goedgekeurd door de promotoren:

Prof. Dr. H. Koppelaar

Toegevoegd promotor: Dr. drs. L. J. M. Rothkrantz

Samenstelling promotiecommissie:

Rector Magnificus,	voorzitter
Prof. dr. H. Koppelaar,	Technische Universiteit Delft, promotor
Dr. drs. L. J. M. Rothkrantz,	Technische Universiteit Delft, toegevoegd promotor
Prof. dr. ir. E. Backer,	Technische Universiteit Delft
Prof. dr. H. J. van den Herik,	Universiteit Maastricht
Prof. dr. P. S. Szczepaniak,	Technical University of Łódź
Prof. dr. ir. P. van der Veer,	Technische Universiteit Delft
Prof. dr. P. J. Werkhoven,	Universiteit van Amsterdam

Preface

This thesis is the result of five years of research, hard work, and at the same time having fun. I have managed to survive countless changes in the faculty and different workgroup names over the course of time. I became a father and a climbing instructor, while still working on lipreading by machines. It is impossible to do that much without firm support of other people. Therefore, even though I am the sole author of this book, I feel obliged to mention at least some of the people that helped making it possible.

First of all I would like to thank my supervisor and friend Leon Rothkrantz, who five years ago decided to invest his time in me and has stuck with this decision to this day. I hope he finds this book rewarding enough for the countless hours spent on our talks and on tons of pages of papers.

Other people who contributed to my research were members of two active scientific groups: the Alparon group and the Multimodal Interfaces group. Among them are Niels Anderweg, Patrick Ehlert, Jelle de Haan, Boi Sletterink, Robert van Vark and Pascal Wiggers. A separate acknowledgement goes to Hans de Vreught, who introduced me not only to the world of Unix, but also taught me the difference between Pilsener and Stout.

Obviously, this work would not have started or been completed without my promoter Henk Koppelaar, who was especially helpful in the final stages of my studies.

Even though the above-mentioned people have greatly contributed to the matter of the thesis, their efforts would have been to no avail if it wasn't for my wife. She helped me to get through all difficulties and kept my spirit high. Without her, this work would never have been finished.

Dziękuję Ci Aniu!

Contents

Preface	iii
1 Introduction	1
1.1 Research goals	2
1.2 Presented techniques	3
1.3 Thesis overview	5
2 Lipreading overview	7
2.1 Lipreading by humans	8
2.1.1 The McGurk effect	8
2.1.2 Modeling the human speech perception	9
2.2 Lipreading by machines	10
2.2.1 Automatic lipreading	13
2.2.2 Bimodal speech recognition	14
2.2.3 Visually based speech enhancement	15
2.2.4 Teleconferencing	16
2.3 Primitives of automatic lipreading	16
2.3.1 Phonemes, visemes and viseme syllable sets	17
2.3.2 Signal representation abstraction	18
2.3.3 POLYPHONE word list	19
2.3.4 Separability of utterances	20
2.3.5 Propagation of misclassifications	22
3 Computational models	23
3.1 Feature extraction techniques	23
3.1.1 Raw image processing	25
3.1.2 Tracking specific points	26
3.1.3 Model-based tracking	27
3.1.4 Lip geometry estimation	27
3.2 Principal Component Analysis	28
3.3 Partially recurrent neural networks	34
3.3.1 Basic principles of neural networks	34
3.3.2 Jordan neural network	39
3.3.3 Elman neural network	41

3.4	Hidden Markov models	42
3.4.1	HMM generator	43
3.4.2	Forward pass	44
3.4.3	Viterbi algorithm	45
3.4.4	Baum-Welsh re-estimation	47
3.4.5	Developing a speech recognizer	48
4	Lip geometry estimation	53
4.1	Lip-Selective Image Filtering	53
4.1.1	Hue-based filtering	54
4.1.2	Filtering with ANNs	55
4.1.3	Region of interest extraction	56
4.2	Feature Vectors Extraction	56
4.3	Person-Independent Feature Space	57
4.4	Intensity features extraction	62
4.5	The future of lip-tracking models	65
5	Limited vocabulary lipreading	67
5.1	Connected digits recognition	68
5.2	Data acquisition	70
5.3	Experiments with a single subject	70
5.4	Experiments with multiple subjects	73
6	Towards continuous lipreading	75
6.1	Data acquisition	75
6.2	Speech onset/offset detection	76
6.2.1	Explorative data analysis	76
6.2.2	Classification with artificial neural networks	79
6.3	Vowel/consonant discrimination	80
6.3.1	Explorative data analysis	80
6.3.2	Data labeling	83
6.3.3	Training Procedure	84
6.3.4	Recognition Results	85
7	Continuous lipreading	87
7.1	Multimodal speech data acquisition	88
7.1.1	Recording requirements	88
7.1.2	Prompts	90
7.1.3	Physical setup	92
7.1.4	Storage structure	94
7.1.5	Recorded data	96
7.2	Combining recognizers	97
7.2.1	Early integration	97
7.2.2	Late integration	99
7.2.3	Intermediate integration	100
7.3	Bimodal continuous speech recognizer	103

7.3.1	Choosing integration strategy	103
7.3.2	Baseline system	105
7.3.3	Feature fusion procedure	105
7.3.4	Recognition results	106
8	Conclusions	109
8.1	On experimental results	109
8.2	On lipreading in general	111
8.3	Final words	112
	Notes on the bibliography style	115
	Bibliography	115
	Curriculum Vitae	129
	Samenvatting	131

*“Who ARE you talking to?” said the King, going up to Alice, and looking at the Cat’s head with great curiosity.
“It’s a friend of mine—a Cheshire Cat,” said Alice: “allow me to introduce it.”*

L. Carroll, *Alice’s Adventures in Wonderland*

Chapter 1

Introduction

What actually constitutes lipreading? This non-trivial question should appear in each book that deals with the topic. The utterance “lipreading” contains two separate and meaningful roots and implies that one can *read* some meaningful content from the *lips* of another person. This is as far as one can get by just using the morphology of the word. In order to understand the full meaning, we need to define the context in which the lipreading is considered. For example, we need to understand that the “lips” are not necessarily the only features that one studies when “reading” the visual information from another person. It is understandable that the rest of the face of our interlocutor also provides essential information that concerns both context and content in verbal and non-verbal parts of the communication. Lipreading is simply impossible without additional observations concerning the tongue, teeth or nasal wrinkles, for example.

A very similar ambiguity can be found in the root “reading” that we usually use when dealing with the most strict and well-defined form of communication. If spoken language is governed by rules with their unavoidable exceptions, this is doubly so for written language. On the lowest level, the typographical rules define precisely what a text should look like in order to be readable (even more rules apply if one wants it to achieve the “well written” label). Syntax and semantics, although these also hold for spoken language, are a lot stricter when written text is considered. Do all those restrictions also apply to lipreading or does the root “reading” refer only to the visual aspect of obtaining information transferred through the channel of natural language?

This thesis does not aim to explore all of the possible definitions of the term “lipreading”; we will however try to be conscious of all its ambiguous connotations, and where appropriate we will redefine it over and over again in different contexts and environments. For now, let us define it in a very generic way as follows:

lipreading : the process of obtaining some information related to the acoustic process of speech on the basis of visual observations.

1.1 Research goals

At the time when the research that led to this thesis was started, we had only one clear and distant goal in mind: an automatic lip-reader for the Dutch language. The goal proved to be unreachable within four man years, which can easily be explained when considering the very closely related field of speech recognition research. It took about 25 years of continuous worldwide effort to bring it to its current usability level. Automatic speech recognition research started with simple tasks such as matching a very limited set of words uttered by one specific person and stored in a machine-readable form with the observed utterance produced by the same person. The researchers developed a multitude of signal processing and recognition paradigms, and the tasks that could be performed slowly grew in complexity, both with number of words, and with the number of persons. Moreover, the influence of the audio recording environment can introduce additional level of complexity. Even though the currently available speech recognition engines are person independent, operate on thousands of words and accept continuous and spontaneous speech, they are by no means perfect.

It is therefore understandable that lipreading research must follow a similar route. We had to start from simple problems and had to develop appropriate processing techniques to solve them. The distant goal made it seem worth taking this route, along with the secondary goals that were to be reached along it. When reaching those goals, we sometimes obtained solutions that are applicable in a real world. For example, Section 6.2 describes a system that is able to detect whether a subject is speaking or not at a given point of time. This by itself can be used to augment the speech recognition software in environments where audio based voice activity detection may not be suitable.

In the same manner, Section 5.1 presents a system that recognizes strings of connected digits. This actually represents a class of different problems (such as phone dialing or name spelling) that deal with a limited set of possible words uttered according to some reasonably simple grammar (or no grammar at all).

On the other hand, we reached some goals that are not self-sufficient solutions to a real life problem. Labeling of recorded video frames as being a vowel or a consonant, for example (Section 6.3) can probably be used to some extent as a support for automatic speech recognition, but it is not useful by itself. However, our work was aimed specifically at realizing the following goals:

- developing an efficient processing model for lipreading;
- validating the proposed model and comparing it to other possible approaches;
- developing a system capable of reading connected strings of utterances;
- developing an automatic system that can detect an on- and off-set of speech in video sequences;
- developing a system that recognizes predefined patterns in continuous speech;
- evaluating possible means of combining lipreading with speech recognition;
- developing an audio-visual corpus for evaluating the proposed algorithms.

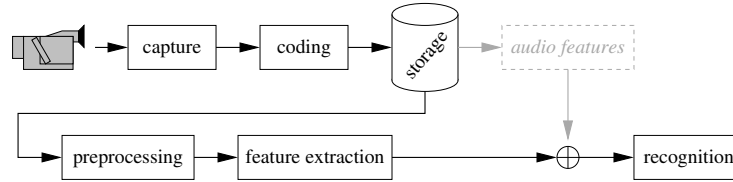


Figure 1.1: Lipreading as a processing pipeline.

All of these goals were reached during our research. At the moment of writing, the results presented in this thesis reflected the state of the art in this research field. The results reported by other researchers are comparable and refer to the same level of complexity. There are therefore still many problems that have not yet been solved; for example:

- obtaining robustness against variations in illumination conditions;
- full person independence, without limitations like skin tone, presence of facial hair, etc.;
- robustness against facial occlusions;
- large-vocabulary lipreading;
- lipreading of spontaneous speech;
- robust integration of audio and video signals depending on the noise level;
- development of a large-scale audio-visual corpus.

All these are the next steps that need to be taken on the path leading to a fully automatic lipreading system.

1.2 Presented techniques

In order to research and develop an automatic lip-reader we had to deal with many different problems, related to both image visual and auditory signal processing at the same time. This thesis describes them all from the very beginning to the end of the processing pipeline. The generic processing pipeline for a lipreading capable system is shown in Figure 1.1. In the following chapters we use the same drawing to indicate which sections of the system are being described.

The first problem that has to be dealt with is the acquisition and storage of video data. In the process of developing the system, we need the ability to retrieve an image sequence from a gathered corpus of recordings. There are many techniques that are available at the current time, but all of them influence further processing stages to some extent. It is, for example, not feasible to store the video in its uncompressed source form. The amount of data that we need to deal with is staggering. For example, the corpus of audio-visual data that was gathered for the purpose of our research

contains about 4 hours of recordings. In uncompressed form it would amount to about 450GB of data. Storing and retrieving this amount of data is technically possible, but not really feasible. Using some compression technique is therefore mandatory in order to keep the data manageable. The compression of a video stream has consequences however. An MPEG coding (and any other coding used to limit the video bit rate down to manageable levels) is a lossy compression method. It introduces some distortions in the video signal in order to make it more predictable. Such a more predictable signal can then be coded with fewer bits. The coding scheme is usually prepared in such a way that the lost (distorted) parts of the signal have the least perceptual influence. This means for example that less space is allocated for coding of the chromacity information than for the illumination part of the signal. The human visual system deals much better with the distortions of the chromacity than with distortions of the illumination and therefore the changes remain almost imperceptible to a human observer. It is not obligatory to build machine-based perception in the same way, however. For example, one can develop video processing techniques that rely on the accuracy of the chromacity representation. Therefore the choice of the storage method must be evaluated with the specific techniques in mind. It also needs to be considered that the choice will affect the on-line processing as well. Any picture deformation resulting from compression must be reproduced in an on-line system or the algorithms tuned on the compressed data will not work as expected.

In the second stage, the visual data must be processed in order to reveal the lip-reading-related information in it. The tasks at hand at this stage are for example: locating the face of the user, finding the mouth region within the face, extracting the geometrical (and non-geometrical) properties of the visible part of the vocal tract. The complexity of each of those tasks depends heavily on the constraints put on the environment in which the system must perform (and therefore on the planned application of the system). For example, locating of the face might range from a trivial (or non-existent) problem in case of a head mounted camera to a complex process in case of a multi speaker environment with a wide-field-of-view camera. A variety of well-known image processing techniques is used at this stage of processing. Especially color-based filtering and morphology operations are of importance here.

Once the interesting image data (such as mouth, teeth or tongue) is recovered from the raw image it usually needs to be converted into a suitable description of the features important for the task of lipreading. One can use many different contour tracking techniques such as active contours or model-based template matching. On the other hand, the data might also be processed on a statistical basis without the need of strict geometrical representation. The choice depends on many factors such as robustness against noise, computational complexity and the significance of the extracted data. This thesis focuses exactly on this area of the lipreading research. The pre- and post-processing stages of the data preparation, together with the mathematical models deployed in the recognition stage, are well-known and broadly used in the fields of image processing, computer vision and speech recognition. The model of data extraction on the other hand is unique, developed with robustness in mind and gradually tuned to the best performance.

At the last stage, the data extracted from the video sequence must be properly recognized by itself or used in combination with the audio speech signal. Depending on

the desired lipreading capabilities of the system, we used two main computational models at this stage. The first one is a data-driven black-box approach as implemented with artificial neural networks. Such an approach can be justified when the information that we want to obtain from the visual speech is related to the short time span characteristics of the signal. Introducing a model of speech with its inherently complex hierarchical structures enforced by grammar, syntax and semantics of the language would be an overhead in such a case. Artificial neural network based classifiers can handle problems such as voice activity detection and coarse phoneme classification.

The neighboring field of speech recognition has taught us however that artificial neural networks are not powerful enough for speech recognition. While it is not inherent to the processing model deployed by them, the staggering amount of training data and the growing complexity of an unstructured classifier that would be needed to implement speech recognition make this approach infeasible at the current time. Even though our brain – the best performing speech recognizer to date – operates on the principles from which the artificial neural networks were derived, the complexity of this specific implementation is far beyond the capabilities of currently available computers. It is therefore much better to use a model-based recognition principle such as Hidden Markov Models. Hidden Markov Models encapsulate our knowledge on the speech production process in a set of small models completed with the set of rules governing their combinations. In this way, the speech signal is seen as a sequence of random processes generated by the underlying grammar. As soon as the production model is well estimated, the recognition can be seen as finding the most probable chain of models that would generate the observed signal. Luckily for us, there are efficient methods for both model estimation and searching for the most probable sequence. All of the lipreading experiments performed during this research employed Hidden Markov Models as the recognition engine.

1.3 Thesis overview

This thesis begins with the chapter that gives an overview of the lipreading by humans and machines. The first part of Chapter 2 describes how people use lipreading techniques to augment their speech recognition in everyday communication tasks. It describes some effects that lipreading has on both normally hearing and hearing-impaired people. It also contains results of our exploratory work in the area of the possible ambiguity of visual perception of speech in Dutch. The second part of Chapter 2 gives an overview of the current research in the area of lipreading by machines. It presents different aspects of lipreading depending on the area of application.

Chapter 3 introduces the main tools that are used in lipreading research; image processing, feature extraction and finally recognition techniques. Their mathematical background is also summarized following the same order.

Chapter 4 describes in full our approach to lipreading from the very first stages of acquiring the data to the resulting preprocessed feature stream.

Chapter 5 shows some experiments with a limited vocabulary class of recognition problems.

Chapter 6 deals with experiments that were aimed at exploring the properties of lipreading for speech data of a continuous nature.

Chapter 7 details our experiments with the continuous lipreading as an augmentation of a speech recognizer. It also deals with some aspects of sensory integration in the context of hidden Markov models.

Chapter 8 summarizes all of the previous chapters, highlighting the achieved goals and other interesting parts of this thesis.

The shop seemed to be full of all manner of curious things— but the oddest part of it all was, that whenever she looked hard at any shelf, to make out exactly what it had on it, that particular shelf was always quite empty: though the others round it were crowded as full as they could hold.

L. Carroll, **Through the Looking Glass**

Chapter 2

Lipreading overview

For a long time, lipreading was predominantly the subject of non-technical research efforts. This research dates as far back as the publication by John Bulwer in 1648 [Bul48]. In those early years, the phenomenon of lipreading by humans was studied in order to understand and improve teaching techniques that allow deaf people to integrate in our speech-oriented society. Lipreading research is, however, no longer confined to that area. As can be found on international conferences such as Auditory–Visual Speech Processing [Mas99, MLG01] or on special sessions of other speech oriented conferences, there is a broad group of researchers interested in both human and machine aspects of lipreading. The topics presented generally include the following:

- acquiring lipreading capabilities by humans
- neurophysiology of lipreading
- sensory integration by humans
- speech production/recognition disorders
- lip-tracking and feature extraction techniques
- computational models for sensory integration
- visually guided speech enhancement
- human perception of computer generated (AV) speech

This chapter presents an overview of some of the above mentioned topics, namely: visual perception of speech by humans, main directions in machine lipreading research, and a feasibility study of using visual modality for speech perception in Dutch. This last part of this chapter is of special significance for the rest of the thesis as it sets upper limits to the performance of any automatic lipreading system.

2.1 Lipreading by humans

Humans can use lipreading to augment their speech perception. This fact is obvious when we refer to people with hearing disabilities; they are forced to use the visual modality to augment – even replace – the auditory input. The studies show that normal-hearing people use lipreading to some extent as well [SP54, BC94]. This process is neither clear nor conscious, yet it influences our perception of speech.

To test the contribution of visual stimuli to the overall speech perception, the aforementioned studies measured the recognition rates of context-free speech with changing levels of noise. It is important to remove the influence of the context from the speech recognition when speech perception – not understanding – is to be measured. To achieve this experimental constraint, the subjects are presented with senseless concatenations of vowels and consonants. The results measured with varying levels of noise and with or without appropriate visual stimuli show clearly that seeing the movements of the lips helps in speech perception. The difference between bimodal and auditory-only recognition rates grows noticeably with the noise level. It is interesting to note that similar experiments with varying amounts of visual clues and a wide range of natural and computer generated speech have been conducted, with results that clearly show a positive influence of a visual aid in perceiving speech [AGMG⁺97, CWM96, Sme95a].

Of course not only lips are observed by humans during discourse. When confronted with a lipreading task we observe a whole person, not only his or her lips. The question as to what extent the lips are crucial in lipreading is indirectly answered when predicting a human response to the visual stimulus [JAAB01]. It appears that if we consider a human response to the consonant-vowel (CV) stimulus, about 65% of it can be explained from the pure lip measurements. By adding measurements from the cheek and chin areas, this ratio can be brought up to almost 80%. These results suggest that human lipreading might be influenced by other changes in appearance, not only by lip movements. It coincides with the results of experiments that show that speech intelligibility changes depending on the visibility of a speaker. Speech is easier to understand when we can see a person's lips, and even more if we can see the entire face.

2.1.1 The McGurk effect

Lipreading by normal-hearing people can also be observed in other circumstances than distracted audio stimulus. In their research on how infants perceive speech, Harry McGurk and John MacDonald discovered what is now commonly called the McGurk effect [MM76]. At some point of their experiments they asked a sound technician to create the videotape with person speaking “ga” and the sound of “ba”. Later on when playing the tape they found out that such a strange combination is perfectly recognizable as “da”. The McGurk effect manifests itself when the auditory and visual information contradict each other even if a sound is perfectly clean. That leads to the conclusion that the visual part of speech is not only used to support the sound when needed but that it is inevitably integrated into the recognition process. Even adding some context to speech does not remove this effect. For example, a sentence:

“My bab pop me poo brive”

$$\begin{array}{ccc}
 & \text{auditory signal} & \\
 \sum a_r \swarrow & & \searrow 1 - \sum a_r \\
 \text{if identified} & & \text{if not identified} \\
 P(r|A, V) = a_r & P(r|A, V) = P(r|V)
 \end{array}$$

Figure 2.1: Auditory Dominance Model.

makes of course no sense if heard. Yet when we dub it on the video with a person speaking:

“My gag kok me koo grive”

it is usually perceived as:

“My dad taught me to drive”

One of the explanations for this phenomenon is that there is no inherent priority hierarchy between auditory and visual modality of speech. They are just complementary parts of the speech signal that are integrated in our brain. For example, Bayesian modeling (choosing the highest conditional probability of spoken utterance given both auditory and visual observations) explains very accurately the McGurk effect.

2.1.2 Modeling the human speech perception

In order to get some insight in the inner workings of human speech perception it is interesting to investigate how, when and where the integration of the modalities takes place. To some extent the answer to this question can be found in real measurements of the working human brain [CCVB01, BPETA01, PB02], but unfortunately, the current state of technology and our understanding of the human brain does not us allow to draw definite conclusions from those observations. Another approach is to define an ad hoc model of modality interaction and test it against a broad spectrum of human responses.

Given the observation that the auditory processing of the speech is much more important than the visual, the Auditory Dominance Model (ADM) has been proposed (see [Sme95a]). In this model the visual processing takes place only in case the auditory processing is not conclusive (see Figure 2.1). The probability of a response (given audiovisual stimuli) predicted by this model is:

$$P(r|A, V) = a_r + (1 - \sum_r a_r) (v_r + (1 - \sum_r v_r) w_r) \quad (2.1)$$

where the a_r and v_r are the probabilities of response r given auditory or visual stimuli respectively. The w_r coefficient represents the bias probability towards a given response.

Another possible approach is the Adaptive Model of Perception (AMP). It is more generic and does not assume any hierarchical dependencies between modalities. In the AMP, the resulting probability of the response is merely a weighted sum of unimodal probabilities:

$$P(r|A, V) = p a_r + (1 - p) v_r \quad 0 \leq p \leq 1 \quad (2.2)$$

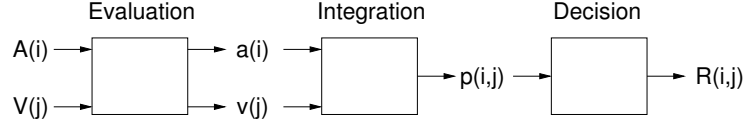


Figure 2.2: Fuzzy Logical Model of Perception (from [Mas89]).

with a weight parameter p that describes a bias towards auditory modality.

The most successful model to date however is the Fuzzy Logical Model of Perception (FLMP). It was developed by D. Massaro in the late 80s [Mas89]. The FLMP is widely known and appreciated because it predicts the results of experiments with great precision. The schematic depiction of the way FLMP works is shown in Figure 2.2. FLMP has three main phases of operation. The input stimuli $A(i)$ and $V(j)$ are first evaluated to yield psychological values $a(i)$ and $v(j)$ respectively. They are then integrated to yield the degree of support $p(i, j)$ for a given alternative. In the last stage the decision operation returns the decision based on the observations. The FLMP assumes late integration of the modalities. Auditory and visual perception processes are assumed to be independent. This model can also be extended to accommodate varying amounts of noise, and still achieve a respectable level of prediction accuracy [ATJL01]. The model itself does not only refer to speech processing, it also applies to other areas of human perception.

There are some problems with FLMP and other late integration models, however. Although the results obtained from such models are correct and much better than the results from any early integration based model, there is at least one problem that does not seem to be resolved if the integration is to be performed after unimodal recognition. There is a known phenomenon of the audiovisual asymmetry (see [CSA01]) that does not fit well in late integration modeling. Even though the FLMP seems to predict the results properly, what it lacks is the explanatory power for this asymmetry phenomenon. It is therefore still somehow undecided whether the human perception is based on early or late integration or even on some hybrid processing model.

2.2 Lipreading by machines

Not so long ago the term *lipreading* was used only in context of hearing-impaired people reading the spoken words from a person's lip movements. However this is not the case anymore. Lipreading is now one of the fast evolving fields in research of human-machine interaction.

Lipreading by machines is, in the same way as for humans, a process in which some (or all) of the spoken information is extracted by the recognition system from the visual changes in the mouth region of the face. Typically, normally hearing people do it subconsciously without knowledge about the dependencies between lip movements and spoken words. The processing of the information acquired in this way also tends to be highly context sensitive. In case of automated lipreading the developers have to translate this subconscious processing into the rigid logic of computers.

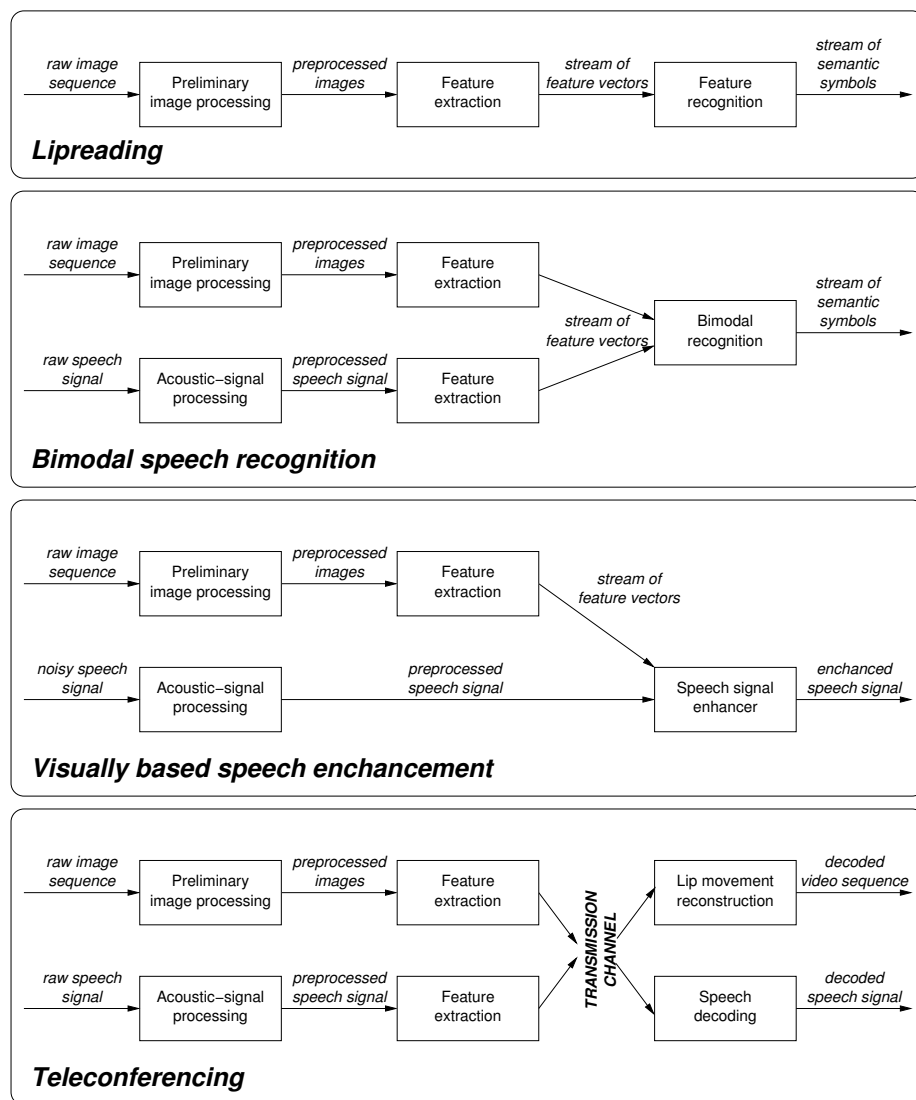


Figure 2.3: Four different applications for a lipreading system.

There are several possible applications for systems capable of lipreading, as depicted in Figure 2.3. They all take as input a sequence of images, and some augment it with the audio input as well. In this thesis we use the terms “image sequence” and “video sequence” equivalently, as our understanding is that video sequence is just a sequence of images put together in some storage format. We will slightly distinguish between them in a sense that “image sequence” does not imply the intricate temporal relationships between the images, as opposed to the “video sequence.” The first and most obvious one is to try to extract spoken information from visual input. This kind of approach may be useful in environments with an extremely low signal-to-noise ratio, where auditory speech recognition does not make much sense. It does not promise high recognition rates, however, because of the inherent lack of information in the visual data stream.

It is much more promising to use the approach based on the way normally hearing people recognize speech; by combining the two modalities that are involved in speech perception. In such bimodal speech recognition, there are two streams of data that have to be processed independently and merged at some point of the recognition process. Depending on the used approach, those two data streams can be integrated either before, after or right in the middle of the recognition engine. This reflects the different ways to understand human models of perception (as described in Section 2.1.2). Early integration of the data streams means that the numerical features extracted from both acoustic and visual data streams are concatenated together and used as an input for the recognizer. In this case the recognizer must provide a mapping directly from the Cartesian product of acoustic and visual observation spaces to semantic symbols. On the other hand, a recognizer that performs a late integration of incoming data may be constructed from two independent sub-recognizers for visual and acoustic signals. The outputs of those recognizers would then be mediated by the additional module. In the last case, the integration may be done on the basis of complicated interactions between smaller modules within the recognizer, each of them processing any (or all) of the data streams. For a further explanation of these issues see Section 2.2.2. The techniques that are used to handle different integration strategies are described in Section 7.2

The third way to use the visual modality of the speech signal is restoring some acoustic information that was lost in noise. In the simplest intuitive example, if we could assess from the visual signal whether a vowel is being spoken in a given time interval, we could conclude that any high frequencies observed in the acoustic signal at the same time are in fact noise. This information could be used later to control the parameters of the low-pass noise-removing acoustic filter.

Finally, the same techniques that are used to extract the semantic information from the video sequence can provide an efficient way of video encoding. Contrary to other approaches that allow generating of facial animation according to a given text, the approach of using the data extracted directly from the observed facial movements can provide us with a person-specific, individualized reconstruction of the speaker. With the appropriate feature extraction models, one can optimally balance the quality of the reconstructed image and the available transmission bandwidth.

Although the above applications of lipreading in human-machine interaction differ a lot, they employ the same basic building blocks. Most notably, all these systems must extract some useful information from the video sequences and advance the results to

further processing modules. As can be seen in Figure 2.3, the first stage in this case is enhancing an image by removing noise and artifacts, histogram equalization etc. None of the tasks performed at this stage is lipreading specific; they are all well-known in the image processing literature. The goal of this part of the system is to produce images of the highest possible quality. It is crucial that the procedures performed at this level are not too computationally expensive as all of the lipreading applications require real-time or close to real-time processing. After this generic image enhancement, the data has to be processed in order to produce some reduced representation of the video sequence. Depending on the used processing model, this representation may contain the geometric model parameters, the principal component weights, some statistical measures of the image data, or any other data structure that seems to provide the crucial information. In contrast to the much more advanced field of speech recognition, in the field of lipreading there are no clear standards yet defining what the data representation should be like. Section 3.1 contains an in-depth description of some of the many possible implementations of this processing stage.

2.2.1 Automatic lipreading

Automatic lipreading, independent from other modalities, forms a basis for any other research in this area. It concentrates on lip-tracking, feature extraction and developing recognition models appropriate for the visual modality of speech. Any system processing visual speech must deal with the same issues as a pure lipreading application. For this reason we concentrate on the different aspects of pure automatic lipreading throughout this thesis (see Chapters 4, 5 and 6).

A typical lipreading system performs its job in several stages:

- image acquisition,
- image preprocessing,
- face and mouth tracking,
- feature extraction,
- recognition.

In the image acquisition stage, the data from the video camera must be transferred to a computer and encoded in some accessible format. This stage is tightly related to a data-storage methods and very much dependent on the development platform. It is not of much interest to researchers nowadays, because image acquisition and storage is now a standard feature of consumer grade PCs. This is something that one can take for granted when researching visual processing of speech. The image preprocessing stage is where the image is enhanced and cleaned from artifacts. It is also a well-researched area, with an available tool set of algorithms from which the lipreading researchers can benefit greatly.

An interesting stage (for a lipreading related research) starts at the face and mouth tracking stage. There are many techniques that can be used for those tasks and new ones are still being developed [Che01, RGBVB97, YW96, SMY97, MGW⁺97, WWR99, Woj97, WRS98]. As soon as the region of interest has been located in the acquired

image, it must be processed in order to extract a set of features from it. The features must be relevant to lipreading and should preferably be person independent. Some of those techniques are described in Chapter 3.

In the last stage, the feature stream that has been extracted from the video signal must be processed in the recognition engine, giving a sequence of recognized entities as a result. In our research we used two different recognition models: data-driven Artificial Neural Networks (ANNs) and production based Hidden Markov Models (HMMs). Both recognition paradigms are briefly explained in Chapter 3 with example implementations in Chapters 5 to 7.

2.2.2 Bimodal speech recognition

Bimodal speech recognition is probably the most intuitive way of using the visual part of the speech signal. Just as humans, the computer is confronted with different observations of the same process and can therefore make a better guess as to the content of the uttered speech. The research in this area is based heavily on the foundations laid by automatic lipreading. The feature tracking and extraction techniques are in principle the same as those described in Section 2.2.1. The main focus of the research is therefore on methods for combining both modalities in order to maximize recognition performance.

There is a multitude of problems to be dealt with when combining modalities. The first type of problems relates to the temporal nature of the incoming data: it is not possible to fully guarantee the synchrony of the incoming signals. Some of the reasons for signal asynchrony lay to some extent within the area that can be controlled in hardware or software. For example, the hardware for audio and video grabbing is based on totally different electronics, with different time constraints, different latencies, and so on. The audio data for example is usually sampled with 44.1kHz frequency while the sampling frequency of the video is either 25Hz or 30Hz. If we take into account the jitter in both frequencies, the video frame cannot be synchronized with the audio more accurately than within a couple of audio observation frames. The latencies in audio and video observations differ also because of the enormous differences in the observation size. While the audio observation is just a single value referring to air pressure, the video frame is constructed from thousands of pixel values. Transferring the video data through any channel (digital or analogue) therefore takes more time than transferring audio data.

All of the hardware related synchrony problems can however be amended by developing specific task-oriented hardware setups that minimize jitter, compensate for latencies etc. There are, however, also other factors that introduce asynchrony in the observations. Muscular actions and sound-producing air flow are complementary parts of speech production. Even though they are closely related to each other, to some extent they operate independently. It may therefore happen that the visual occurrence of some speech phenomenon appears before or after the audible entity. It is not guaranteed that such a temporal distance will be constant for a given utterance. The extent of such disparities is multiple orders bigger than that introduced by hardware imperfections. It is therefore better to develop recognition models that can deal with asynchrony rather than to focus on improving the hardware.

The other issue we are faced with when developing a bimodal recognizer is how to mediate in case of missing or conflicting information. In some implicit or explicit way we have to deal with the confidence measures for different simultaneous observations, we need to allow for conflict resolution and so on. The way it is implemented depends highly on the chosen combining strategy. The modalities can be integrated on the level of the signals, on that of the recognition results, or anywhere in between. In case the signals are combined, there is no real need for conflict resolution; there is just one recognition engine that treats the incoming data as a single (albeit complex) signal. The obvious simplicity of this solution also has disadvantages such as lack of flexibility with respect to the signal quality. On the other hand, integrating the recognition results implies developing two separate recognizers (one for each modality) and developing some inference engine that would combine their outputs. Using a separate combining entity means that we need the recognizers to output additionally some sort of quality or confidence ratings for the recognition they did, which increases the overall complexity of the system. An example of such a system developed for this thesis is described further in Chapter 7.

2.2.3 Visually based speech enhancement

The auditory speech signal may be disturbed in many possible ways and in many places during transfer. There might be noise present in the environment where the speech is being produced. Or if the speech is transferred through some unreliable channel, it may be corrupted by errors introduced in the transfer. Moreover, the speech production itself might be influenced by some additional factors (e.g. “helium speech” or “pressure breathing”). Such a degraded speech signal might be hard to understand for humans and most probably totally incomprehensible for automatic speech recognizers. This creates the need for various speech enhancement techniques.

Speech can be enhanced on the basis of purely auditory information. Many existing noise cancellation techniques can deal with different types and different levels of background noise [PHA01, LMO01]. There are also several techniques aimed at correcting the frequency properties of the signal [PC01]. All those techniques suffer from one inherent problem of the audio-only approach: there is no independent information about the original signal. For example, most noise cancellation techniques assume the noise to cover a wide spread of frequencies, with speech concentrated in a small number of frequency bands. This frequency spectrum spread assumption works well for vowels which can be separated from white noise, but not for fricatives (consonants such as *s* or *f*).

We can overcome the shortcomings of auditory based speech enhancement by using a visual modality to provide clues about the speech signal. If we return to noise cancellation: in the video signal, the mouth shape can be tracked to see whether the person is pronouncing a fricative to inform the noise cancellation system that the wide-band audio signal is probably related to speech rather than noise. Different approaches for such audio-visual processing of speech have been presented in e.g. [GFS97] and [DPN02]. In this thesis we will not focus on problems related to speech enhancement. Some of the work presented in Chapter 6 could probably be used to provide visual clues to speech processing, but this does not fall in the scope of our research.

2.2.4 Teleconferencing

In the beginning of this chapter we referred to research results showing that intelligibility of speech in human-to-human communication can be increased by adding visual information. This important fact is probably one of the forces that drive the development of the teleconferencing systems. Such systems aim at bringing the quality of conversations made between physically distant participants to a level comparable with that of the face-to-face meeting.

One of the important issues in teleconferencing systems is extremely efficient video encoding. On the one hand, video encoding in such applications is usually limited by the extremely low bandwidth of available transmission channels. On the other hand, it is possible to restrict the video content to some extent, making it more predictable and therefore more compressible. For example, we may restrict the scene to only a single person's face. In such a case we may pursue so-called model-based video encoding, where the visible face is described in terms of a geometric model. The model must only be sent to the receiver once at the start of the communication. During the course of conversation a lipreading capable system tracks all the occurrences relevant to the speech production and encodes them in terms of model deformations. The parameters for those deformations must then be sent to the receiver, to reconstruct the appearance of the speaker (see [BMP⁺] for an interesting overview of the subject).

To ensure interoperability of software using model based coding, the Moving Picture Experts Group proposed a standard for multimedia content [Ost97, Sik97] (MPEG-4). In this ISO certified standard (ISO/IEC-14496), the shape, texture and expressions of the face are controlled by Facial Definition Parameters (FDPs) and/or Facial Animation Parameters (FAPs). For interoperability purposes it is sufficient that both transmitter and receiver conform to the MPEG-4 standard to facilitate transmission of the facial animation. This allows not only for realistic video transmission but also for platform dependent implementations at both ends. If for example the receiver is low end and cannot handle the computational load necessary to reconstruct a full video image, it can decode the stream of FAPs into a simplified synthetic animation [TO99]. In this way MPEG-4 based teleconferencing may scale down even to the level of hand-held devices.

2.3 Primitives of automatic lipreading

Before regarding further research in the area of visually based speech recognition, it is important to evaluate to what extent it can compete with auditory based methods. There are several issues involved in the implementation of an automatic lipreading system. Some of them are media specific, such as data processing techniques that differ from those used for auditory speech. Others stem from the differences in the primitives of the incoming signal. In this section we deal with the latter issue:

- What are the primitives of auditory and visual speech?
- To what extent does the ceiling on the performance of the speech recognizer depend on the choice of those primitives?

Table 2.1: Viseme sets for the Dutch language reported in [Beu96] (in traditional Dutch phoneme notation).

Consonantal viseme sets		Vocal viseme sets	
1	/p,b,m/	1a	/ie,i,ee,e/
2	/f,v,w/	1b	/ei,aa,a/
3	/s,z,sj/	2	/u,yy,oe,o/
4a	/t,d,n,j,l,r/	3	/eu,oo/
4b	/k,R,x,ng,h/	4	/au,ai/

2.3.1 Phonemes, visemes and viseme syllable sets

A continuous speech signal can be represented using a limited set of primitives called phonemes. Just as the characters of written language, phonemes can be used to write down any spoken utterance. A phoneme is an abstract term describing a class of sounds that share a common articulation pattern. When a word is uttered, the produced sound depends on gender, stress level, position in the sentence and many other factors, yet the representation of that word in terms of phonemes remains roughly the same (with the exception of words that have multiple pronunciations). There are no strict guidelines for choosing the phoneme set. Different phoneme sets describe the articulation patterns with different granularity depending on the area of applicability. For example, the distinction between “soft g” spoken in Limburg and “hard g” from Zuid-Holland is crucial when dealing with different dialects of Dutch, but redundant for speech recognition as both sounds can be interchanged without altering the meaning of the word. For speech recognition purposes the Dutch language comprises about 40 different phonemes.

In the visual modality, the primitives of speech are so-called visemes. As the visual part of speech is but a faint shadow of the auditory channel, the visemes are defined in terms of groups of phonemes. According to different researchers, there are between 10 and 14 different visually indistinguishable phoneme sets in Dutch (see Tables 2.1, 2.2 for examples from the literature). This does not imply that only 14 different lip shapes occur in speech production. There are in fact infinite possible lip configurations. In the same way as a phoneme describes a class of different waveforms, the viseme is an abstract term describing visual occurrences.

Phonemes within the same set correspond to the same viseme, so they cannot be reliably distinguished by a human observer. Yet, one could argue that there exist coarticulation effects that change the way in which these phonemes are perceived in continuous speech.

We may consider for example two phonetically different syllables: *uf* and *oef* (Yf and uF in SAMPA notation). They are very close in the phonetic domain but still distinguishable, and they contain exactly the same visemes (see: [O] [F] in Table 2.2) which makes them potentially indistinguishable in the visual domain. Nevertheless, the experiments with human observers prove that on average those two utterances are perceived as different [Beu96].

The above suggests that using primitives other than visemes may be beneficial for visually based speech recognition. Some utterances – sharing the equivalent viseme

Table 2.2: Viseme sets for the Dutch language in SAMPA notation (from [VPN99a]).

viseme	phoneme class	viseme	phoneme class
0	<i>silence</i>		
1	[F] f v w	8	[I] I e:
2	[S] s z	9	[E] E E:
3	[X] S Z	10	[A] A
4	[P] p b m	11	[@] @
5	[G] g G k x n N r j h	12	[i] i
6	[T] t d	13	[O] O Y y u 2: o: 9 9: O:
7	[L] l	14	[a] a:

Table 2.3: Viseme and phoneme based syllable sets.

syllable type	phoneme sets	viseme sets
V	14	7
VC	308	49
CV	308	49
CVC	6776	343
Total	7406	448

representation – can be visually distinguished if they comprise of different viseme syllables. Therefore, we should use the viseme syllable sets (VSS) as the primitives for the lipreading system. The VSS would have to be constructed in such a way that they would group together the syllables that are indistinguishable for human observers when presented with visual stimulus only.

The biggest problem with such an approach is the large number of potential syllables. Table 2.3 shows how the number of possible syllables grows exponentially with the length of syllable. The number of possible syllables constructed from phonemes is well over 7000; the number of possible syllables constructed from visemes exceeds 400. The number of VSS should be somewhere between those two figures and probably much closer to the lower one. Building a recognition system that can deal with more than 400 classes is definitely more complicated and time consuming than building a system for viseme recognition only (dealing with just 14 classes). Therefore there should be a substantial gain in recognition performance in order to justify that approach.

2.3.2 Signal representation abstraction

The whole speech communication process can be seen as a transfer of a given concept from speaker to listener. The concept is being encoded into an audio-visual signal during speech synthesis. This signal must later be decoded at the receiver side. In order to automate the recognition of a speech signal, one has to design an abstract representation of the audio-visual signal produced by a speaker. For example, in classical

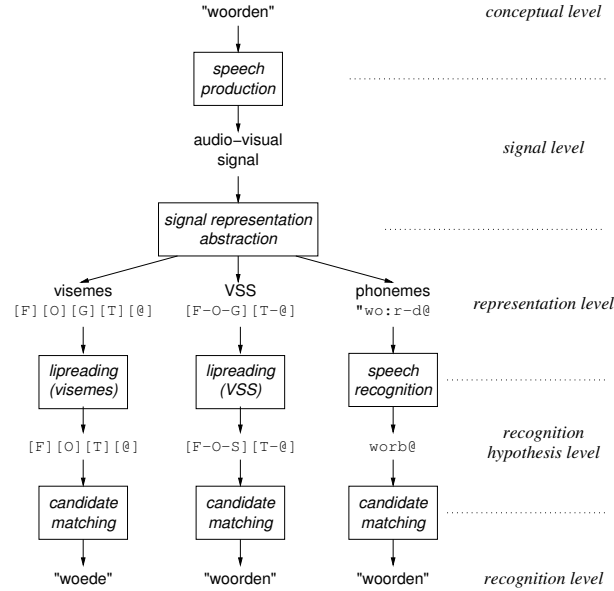


Figure 2.4: From concept to recognition, and how different modalities cope with it.

speech recognition the signal is considered to be a continuous string of phonemes. In this representation, phonemes are the primitives from which higher concepts such as words and sentences are constructed. On the lowest level, the recognition process is about locating the phonemes in the incoming signal; it does not matter whether it is done explicitly or not (e.g. in case of HMM based speech recognition there is no explicit knowledge of phonemes' positioning in the signal). On the other hand, in case of limited vocabulary speech recognition, the speech signal is represented in the form of a string of separate words; the building blocks of the signal are the words from some limited list.

Any signal representation has its limitations with respect to the scope of possible speech recognition. As an obvious example; the limited vocabulary representation limits any recognition system based on it to recognition of the words from the list. That means that its not possible to build a recognition system that would perform better than what is induced by the representation on which it is based.

2.3.3 POLYPHONE word list

In order to compare possible recognition rates we need an exhaustive set of utterances that exist in the considered language. One of the databases used for the development of speech recognition systems for the Dutch language is POLYPHONE [DBI⁺94]. This set of recordings aims at gathering a broad spectrum of spoken Dutch language and incorporates recordings of phone sessions together with their transcriptions and related statistical data.

In this research we only use a POLYPHONE word list, exhausting the separable

utterances in the corpus. Most of those utterances are words, but the list also contains spelled letters of the alphabet, nonverbal utterances (such as 'eh', 'hmm' etc.) and other lexical items (as background noise or background speech). After removing non-word items, the list contains 20539 different words.

Each of the words in the list is represented by its written representation, pronunciation and the count of appearances in the recordings. The pronunciation of the word is written using SAMPA notation, with syllabication and accentuation included. Obviously, the number of times the word appears in the recordings is a very important factor in the investigation of the possible recognition rates: we may assume that in the spoken language, the homonyms appear only rarely, while the words that appear often should have a distinct pronunciation. In total, the number of word utterances is 1355822 (97%) from 1402858 items covered by the POLYPHONE corpus.

2.3.4 Separability of utterances

The report in which VSS were introduced [Beu96] did not specify those sets explicitly. There is only a suggestion that such sets do exist and that there are more of such sets than simple combination of visemes implies. Therefore currently it is not possible to make a proper comparison of word distinguishability between phoneme, viseme and VSS-based representations. The aforementioned work [Beu96] states however that the distinguishability within the syllables composed from the same visemes may be introduced by different durations of the utterances as a result of coarticulation. For example, although *p* and *b* have the same visual appearance, the syllables containing *b* will be usually spoken a bit longer because of the necessity to vocalize the phoneme.

The transcriptions of the words in the POLYPHONE word list contain information about syllable boundaries, duration of the vowels and accentuation of the words. Assuming that all three factors (syllable boundaries, duration and stress) may influence the visual perception of the spoken word, we present the following stages in word distinguishability degradation:

- R-1. Full phonetic representation. This is exactly the representation as presented in the word list. It should allow almost perfect distinguishability with rare exceptions of the words that sound completely the same but differ in textual form.
- R-2. Viseme representation with syllable boundaries, vowel duration and stress point. That is as close as we can get to hypothetical VSS-based representation. The difference in duration and stress placement within the syllables constructed from the same visemes should produce the number of different syllable representations (comparable with number of truly distinguishable viseme syllables).
- R-3. Viseme representation with syllable boundaries and vowel duration but without stress point. The position of stress was not taken into consideration in reported tests, so this representation comes close to the results from the aforementioned report [Beu96].
- R-4. Viseme-only representation.

Table 2.4: Distinguishable words from POLYPHONE word list.

Representation	# words	(%)
R-1	20015	97%
R-2	18440	90%
R-3	18344	89%
R-4	17897	87%

Table 2.5: Distinguishable words from POLYPHONE corpus.

Representation	# words	(%)
R-1	1341346	99%
R-2	1258819	93%
R-3	1252865	92%
R-4	1233189	91%

The decrease in the number of distinguishable words will be clearly seen when moving from representation R-1 to R-2. The lack of difference between vocal and non-vocal utterances together with a relatively large number of consonants that come in a single viseme group should greatly reduce the number of words with different representations. The main question here is, however, to what extent will this number change between representations R-2 and R-4? If the change is significant, the models based on VSS should prove to perform superiorly to those that use visemes separately.

The comparison of the amounts of distinguishable words are presented in Tables 2.4 and 2.5. The first table presents only the number of different words from the POLYPHONE word list that can be distinguished using different word representations. It does not take into account that some of them occur more often than others in real-life use. In Table 2.5, the distinguishability score was calculated using the assumption that if some words share the same representation we will always choose the most common one. In this way the most popular of those words will always be recognized correctly while the others will not be. This calculation method gives us some insight in the way in which a context-less recognizer would treat the incoming data if the recognition was perfect on the specific representation level.

As can be seen, the difference in word distinguishability is not as great as one would expect; the worst-case example lays still within the 10% distance from the phoneme representation. What is even more important is that together with the frequency of the word occurrence (Table 2.5) the differences between all visually based representations are almost negligible. In theory the lipreading system with a perfect viseme recognition would perform with 91% accuracy compared to the 93% accuracy of the VSS-based one.

There seems therefore to be no significant value added by using VSS instead of visemes in the lipreading task. This does not mean that VSS should be completely abandoned in further research. It may turn out that in reality for some reasons, the achievable viseme recognition rate is much lower than the rate at which VSS can be

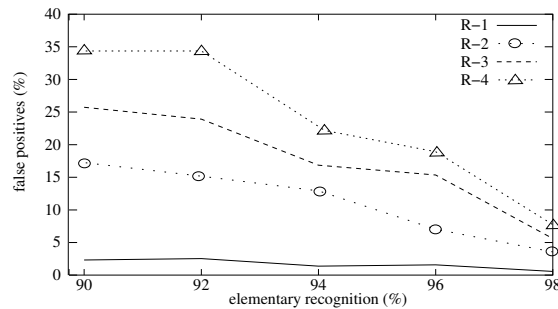


Figure 2.5: False positives as a function of misclassification of basic recognition entities.

recognized. In such a case the VSS based recognizer would have a big advantage over viseme based one.

2.3.5 Propagation of misclassifications

There is another important issue that needs to be considered when comparing different signal representations. It is not possible to achieve a 100% proper classification on the lowest level of recognizer. It is very probable that many of the words will contain some misclassified phonemes (visemes or VSSs). Obviously the recognizer should not discard those words as not recognized straight away. The available word list should be searched for the most probable candidate that matches the recognized phoneme sequence best. In Figure 2.4, the recognition hypothesis level is depicted with misclassified representations of the observed utterance. Only after candidate matching we obtain a resulting word that best matches the observation.

There is, however, one problem with this approach. The candidate matching will succeed in correcting misclassifications only if the hypothesis does not fit one of the other words in the dictionary. Obviously the more redundant the representation of the signal, the less chance that the slightly misrecognized utterance will fit some other valid target and yield a false positive.

The amount of false positives that depend on the level of misclassification on the basic level is depicted in Figure 2.5. As can be seen, even with a recognition level as high as 90% per each observed viseme, already almost 35% of the words will be recognized incorrectly. For a comparison: the amount of false positives for the phoneme representation remains well below 3% at the same level of phoneme misclassifications. It is also worth noting that the typical phoneme/viseme misclassification rates will actually be much higher in a real-life system. In this respect the phoneme representation scores much higher than all viseme representations. If the recognition system uses the visual modality to do the speech recognition, it is much more sensitive to faults in recognizing the basic building blocks of the signal. A small misclassification on the phoneme recognition level will most probably yield a non-word which can later be matched to a proper word. The same misclassification of the viseme produces a valid, but incorrect, word from the dictionary which cannot easily be corrected.

“Hallo!” said Piglet, “what are you doing?”
“Hunting,” said Pooh.
“Hunting what?”
“Tracking something,” said Winnie-the-Pooh very mysteriously.
“Tracking what?” said Piglet, coming closer.
“That’s just what I ask myself. I ask myself, What?”

A. A. Milne, **Winnie-the-Pooh**

Chapter 3

Computational models

In this chapter we describe the generic algorithms and computational models that are used throughout the processing pipeline. We start with the feature extraction techniques that are used in lipreading research, which are described shortly in Section 3.1. The order in which they are presented is by no means random. We start with almost unprocessed images, derive the most simplistic lip geometry information from it and then introduce the extensions to this model.

The next section (3.2) provides a short description of Principal Component Analysis (PCA). PCA is very often used in lipreading research in order to considerably lower the amount of data that needs to be processed by the recognizer. It also usually reduces data redundancy and noisiness. Both of those factors contribute positively to recognizer’s performance. In the last part of Section 3.2 we use the PCA to compare qualitatively the feature extraction techniques described in Section 3.1.

The last two sections of this chapter describe the two recognition models used in the experiments conducted during our research: Artificial Neural Networks (ANNs) and Hidden Markov Models (HMMs). They differ not only on the level of implementation and development; they also represent two separate recognizer paradigms. The ANNs are purely data driven with little to no underlying knowledge about the properties of the signal. The HMMs represent a generative approach to the recognition of the signal, which means that they build upon the predefined model of the process generating the signal. Both approaches have their advantages and disadvantages, which will be discussed in later chapters together with the experimental results.

3.1 Feature extraction techniques

Every lipreading system must somehow process the incoming video data. Obviously, that kind of data is stored and transmitted in a way that is most suitable for decoding it and showing it to the human observer. This data representation does not guarantee that it will be optimal with respect to the automatic processing and recognition. It is therefore essential to preprocess the data in such way that the resulting data stream becomes more suited for further processing. The obtained data stream is not necessarily

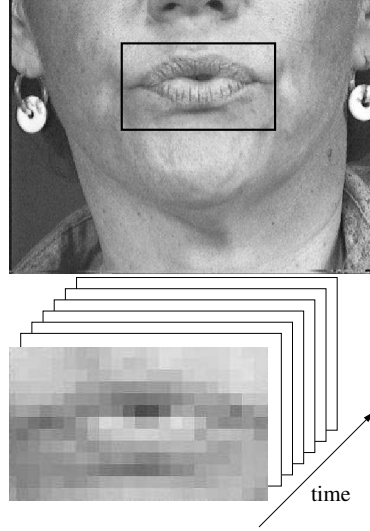


Figure 3.1: Extracting a 512-dimensional vector of raw image data from the video sequence (RI – Section 3.1.1).

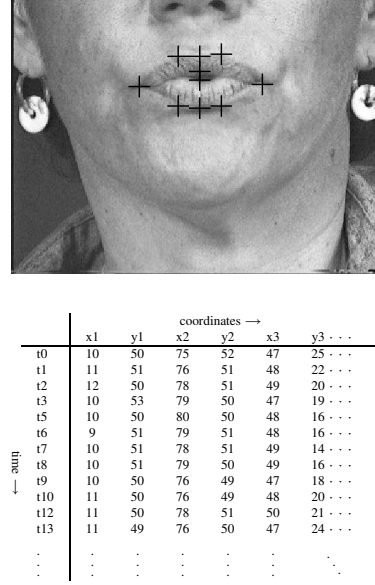


Figure 3.2: Tracking specific points on the mouth contour (PT – Section 3.1.2).



Figure 3.3: Fitting the geometrical model on the mouth image (MBT – Section 3.1.3).

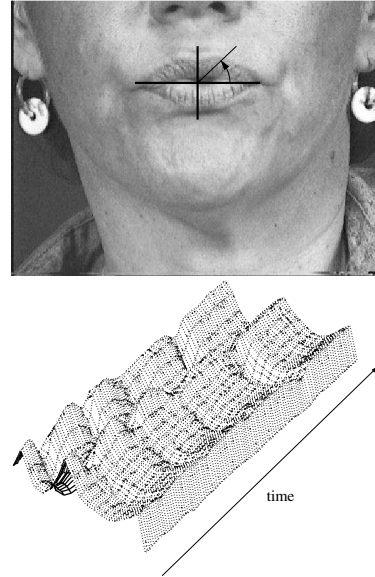


Figure 3.4: Lip Geometry Estimation (LGE – Section 3.1.4).

the final stream of extracted features, but rather the intermediate form that is suited for feature extraction.

Contrary to audio processing, which settled on the use of the frequency domain for representation of the speech signal, the representation forms used in lipreading are varied and usually not directly convertible between each other. The lip movements and other visually distinguishable changes in the vocal tract are represented by different researchers with a multitude of possible geometric and non-geometric models [HSP96]. While one can assume that geometric models are to some extent compatible with each other, there is no obvious method to compare the geometric models with the intensity-based models, for example. Conversion between those two types of representation is certainly possible [LSC00], but nontrivial and current methods do not guarantee accurate conversion. The amount of processing models is not bad per se, but as they are different in nature it is hard to fairly compare all methods for higher level processing of visual speech. Concluding, it is nearly impossible to compare two lipreading system architectures that are based on different processing techniques.

There is however a silent agreement about the fact that usually the data extracted from the visual signal must somehow be reduced in size and complexity. It is essential for methods that deal with raw pixel representation as in this case the amount of data is absolutely staggering. This can be compared to audio processing, where the first step of transforming the input signal into a frequency chart is just the intermediate step before e.g. computing Mel Frequency Cepstral Coefficients (MFCCs). The example data rates in those two steps of signal processing are presented in Table 3.1 (the methods in the table are presented in the same order as in Sections 3.1.1–3.1.4). The table shows that the only processing model that guarantees acceptable data reduction already in the first stage of processing is the model-based lip tracking (with the exception of highly complex models). This does not mean, however, that the obtained parameters of the model are a good representation of the data that can be fed directly to the recognizer [KS96]. It can be seen that all other methods use some sort of linear projection of the obtained data that maximizes some statistical property of the dataset. The optimized properties can either be data related, as in the case of Principal Component Analysis (PCA), which maximizes the variance covered by first components, or recognition related, optimizing features such as maximum likelihood in the case of Maximal Likelihood Linear Transformation (MLLT). In any case, the result is a data stream that is strongly reduced in size and noise filtered.

3.1.1 Raw image processing

One of the most basic approaches to gather data needed for processing visual speech is to use pure and only slightly preprocessed video frames. One then uses a fixed resolution window centered on the mouth region and treats the intensity values of the raw image (RI) from this window as an extracted data vector (see Figure 3.1). This widely used method [LDS95, VPN99b, PLH00, PN01] has some serious drawbacks. Firstly, although it is very straightforward at data extraction level, it produces a vast amount of data that pushes a heavy computational load down the processing pipe. It cannot really be seen as a feature extraction method; it is rather a preprocessing stage for the lip reader. The data of the original image are only reduced with a factor 30,

Table 3.1: Comparison of estimated data-rates in consecutive stages of processing. RI – raw image processing, PT – point tracking, MBT – model-based tracking.

	Input signal	Pre- processing	Intermediate signal	Post- processing	Recognition entity
Audio	256kbps	FFT	80kbps	MFCC	10kbps
Video	38Mbps	RI	819kbps	PCA	8kbps
		PT	13kbps	PCA	4kbps
		MBT	4kbps	–	–
		LGE	28kbps	PCA	4kbps

which makes it hard to store them in uncompressed digital form for off-line processing.

The second disadvantage of this method is that it discards most of the geometric dependencies in the gathered dataset. Two images that are only slightly displaced, for example, will produce highly distant data vectors. The problem here lays in the fact that the two-dimensional function of image intensity must somehow be sampled into the single vector of data.

Another problem with the approach is that it is highly dependent on the illumination condition. Although the image itself may be normalized in order to compensate for this, the resulting changes in contrast, shadows and reflections on the lips are directly transferred into the gathered data. That means that two images of the lips taken from the same person, uttering the same sound, but taken with different light sources will be totally different (in the sense of the distance between extracted data vectors). To remedy this, it is possible, for example, to incorporate a shadow compensation [LK01, KLS02], which utilizes specific properties of the image of the mouth, but those obviously introduce additional computations and therefore remove one of the biggest advantages of the method in question: its simplicity.

This way of representing visual data for lipreading systems also preserves all of the person-dependent features such as skin complexity, skin texturing and person-specific geometric features of the mouth. This could be an advantage for person identification purposes, but it is a substantial overhead for a lip reader.

As a consequence of the above-mentioned drawbacks, the lipreading systems based on such data extraction must be trained on a disproportionally large set of examples. The training set for the system must comprise a wide range of illumination conditions and a wide range of subjects. This could be compared to performing speech recognition based on raw waveforms instead of on well-defined features such as MFCC or LPC.

3.1.2 Tracking specific points

It is possible to define a set of points to be tracked which would represent the mouth shape [GMZRR00, LTB96a] (point tracking – PT). It is important to choose the points such that they are both representative and easy to track automatically in the sequence of images. Using points as a method for describing the shape of the mouth allows for an enormous data reduction and independence of the illumination conditions. The

illumination conditions – even though they are by no means represented in the resulting data – have a big impact on the robustness of the tracking algorithm. There is no straightforward method for obtaining positions of the selected points in the image. One has to use more or less sophisticated edge detection algorithms and/or pattern recognition techniques. It can be argued that in good illumination conditions, tracking points on the inner lip edges is easier than on the outer edges, which is not bad as the inner lip edges are more representative of changes in the vocal tract.

For comparing the different feature extraction techniques presented in Section 3.2, we implemented a simple color-based point tracker that tracks 10 points on the mouth contours (see Figure 3.2). The points are located on different ascending or descending edges of the color filtered image of the region of interest. The outer contour of the mouth is described by the mouth corners and the points laying on the mouth contour at $\frac{3}{8}$, $\frac{1}{2}$ and $\frac{5}{8}$ of the mouth width. On the inner mouth contour, two points laying exactly at $\frac{1}{2}$ of the mouth width are tracked.

3.1.3 Model-based tracking

It is possible to extend point tracking algorithms conceptually and to assume that we actually want to track all of the points on the lip edges (see Figure 3.3). In this case, we may easily put constraints on the geometric features by assuming that those points lay on some predefined type of curves [CTC96, MS98]. By defining the type of the curves we constrain the tracker to some model of the mouth, hence the name: model-based tracking (MBT). This approach allows for more robust tracking as it uses non-local segments of the image. Most of the image distortions are at a local level of detail, so they are discarded by this method.

The model-based trackers may vary greatly depending on the complexity of the model being used. In the simplest case, the mouth can be represented by just two parabolas that approximate the outer edges of the lips. This simple representation may be further expanded by adding models for inner lip contour or allowing for higher degree curves. Moreover, the models can be extended with the internal dependencies that limit their appearance to some predefined shape ranges. This allows for more robust mouth tracking as it eliminates shapes that cannot occur in a real situation. Such models with their constraints can be seen as physical structures with joints, springs etc. [Vog96]. The tracking process of such models then consists of minimizing the model energy given internal and external constraints. At the highest complexity end of the spectrum we find fully-fledged three-dimensional models of the lips.

We have implemented a relatively simple model comprising four parabolas that approximate the inner and outer contours of the lips (see Figure 3.3). The model is constrained with respect to possible configurations, so that the inner contour is limited to the positions within the outer contour and some degree of symmetry is preserved with respect to the vertical axis.

3.1.4 Lip geometry estimation

There is always a subtle balance between the complexity of the model being tracked and the constraints put on it. If the model is highly complicated it contains a lot of degrees

of freedom and therefore possibly produces a lot of invalid shapes. Those unwanted possibilities must be eliminated by using the constraints on the model geometry. The models with more controlling parameters are also more likely to be highly sensitive to small changes in the image. Such a sensitivity is not desirable in a lip tracker as it would amplify the influence of the visual noise and compression artifacts in the image.

In order to compensate for those deficiencies of the highly complex geometric models we propose to get rid of the notion of strict lip geometry tracking. We propose instead lip geometry estimation (LGE), which allows for a geometric model with any chosen degree of complexity. The model is computationally light and noise resistant at the same time. This method was developed entirely by the author during his PhD research and is described in depth in Chapter 4. It is to a large extent independent of the illumination condition. Even though its robustness depends on proper illumination of the subject (the lip-selective filter depends on a proper color range of the lips), the extracted data does not contain any information about the illumination. Moreover, as with all methods that deal only with the geometry of the mouth, it discards a vast amount of person-specific features and preserves the geometrical properties of the mouth. The original image is reduced in size about 2000 times. Although it is not the lowest reduction rate of all geometric models, it is enough to enable efficient off-line storage and processing.

3.2 Principal Component Analysis

Principal Component Analysis (PCA) is a well-known statistical method for linear transformation of data. It is derived from the algebraic notion of eigenvectors, which allow one to express a matrix in the most parsimonious way. The resulting matrix is more suited for further analysis, visualization and/or compression than the original data. PCA can be best described in terms of a statistical representation of a multivariate random variable:

$$X = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \quad (3.1)$$

with with the appropriately defined covariance matrix:

$$\mathbf{C}_X = E\{(X - \bar{X})(X - \bar{X})^T\}. \quad (3.2)$$

Further we will denote the elements of $\mathbf{C}_X$ as c_{ij} . Each element c_{ij} is equal to the covariance between the components x_i and x_j of the random variable X . The PCA relies on finding eigenvectors e_i and corresponding eigenvalues λ_i of matrix $\mathbf{C}_X$. The eigenvalues of the covariance matrix are the n solutions of the following equation:

$$|\mathbf{C}_X - \lambda I| = 0. \quad (3.3)$$

The above equation is of order n and for large dimensions of the X it cannot be easily solved. Fortunately for us, this problem has already been solved many times and with many different techniques. At this time, most of the available math toolkits (e.g. Matlab, Mathematica or Maple) provide a ready-to-use solution for the problem.

The X is immersed in a multidimensional space with an orthogonal basis consisting of eigenvectors e_i . In a case the equation (3.3) has less than n solutions, that is, if there is a linear dependency between some of the rows of covariance matrix $\mathbf{C}_X$, the eigenvectors describe only a subspace of the original n dimensional space. The random variable X is in this case completely contained within that subspace. We can construct the transformation from original space to the space defined by eigenvectors by putting the eigenvectors in a matrix $\mathbf{A}$ with one eigenvector in each row:

$$Y = \mathbf{A}X = \begin{bmatrix} e_1^T \\ e_2^T \\ \vdots \\ e_n^T \end{bmatrix} X. \quad (3.4)$$

The random variable Y represents X in new coordinates.

The eigenvectors e_i can be sorted in the order of descending eigenvalues λ_i . If so, the first eigenvector has the direction of the highest variance of X . The second and following eigenvectors also point in the direction of the highest variance, with the constraint that they are orthogonal to all the previous eigenvectors. This property of the eigenvectors is very useful when we deal with the observations of random variables.

In case of a set of observations, we can estimate the covariance matrix of the underlying random variable. The eigenvectors of this matrix, sorted in order of descending eigenvalues, are further called Principal Components (PC). The properties of PCs tells us a lot about the structure of the dataset. For example, the relative size of eigenvalues tells us how “thick” the dataset is in a given direction. It is very common that the multidimensional observations are actually manifestations of the processes with a much lower dimensionality. The data then concentrates in a very “thin” area around some subspace. In such a case only a few eigenvalues are relatively big. That means that only a few first PCs contribute greatly to the variance of the data, while the rest has almost negligible impact. This can be used for data compression; we only store the values of the PCs that contribute most. It is often assumed that relatively small contributions to the variance in data are not really related to the underlying process, but rather the results of inaccurate measurements. We can therefore reduce the noise by using only a few first PCs instead of the whole dataset.

It is also very common that the PCs relate to some properties of the data that can be intuitively named. For example, in the dataset obtained from tracking the points on the mouth contour, the first PC can be described as the degree of openness of the mouth. The second PC represents the degree to which the mouth is stretched. Surprisingly, this observation holds independently of the feature extraction method and can even be found in three-dimensional tracking [KMT01].

In Table 3.2 we summarized the correlation between the first PC from the different lip-tracking techniques. The table quantitatively shows what is also evident from Figure 3.5: the first component is almost the same independently of the tracking method. The correlation between the data based on the geometric properties of the lips does not fall below 0.9, which means that there is no significant difference between those methods. The raw-data extraction path is totally different from the geometric one and therefore the correlation between its first component and the others is significantly lower; just above 0.65.

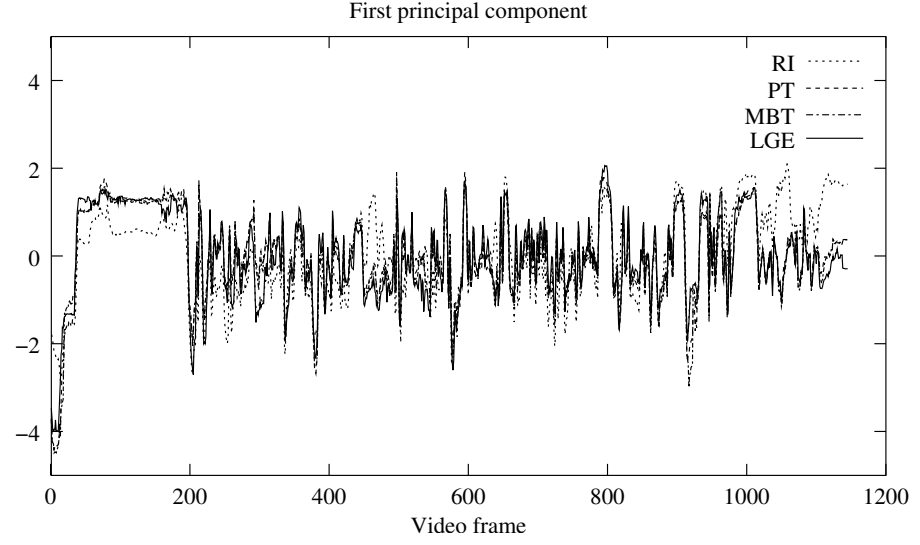


Figure 3.5: First principal component of discussed representations of the lips.

Table 3.2: Correlation between 1st PCs of different tracking techniques and LPC/MFCC coefficients.

	RI	PT	MBT	LGE	LPC	MFCC
RI	1.00000	0.73325	0.69635	0.65130	0.40764	0.26101
PT	0.73325	1.00000	0.98266	0.92246	0.38818	0.19322
MBT	0.69635	0.98266	1.00000	0.93243	0.38349	0.20070
LGE	0.65130	0.92246	0.93243	1.00000	0.42425	0.21802
LPC	0.40764	0.38818	0.38349	0.42425	1.00000	0.51888
MFCC	0.26101	0.19322	0.20070	0.21802	0.51888	1.00000

Table 3.2 also contains the first principal components of the data from the audio channel. We provide here the figures for audio features extracted either as LPC coefficients or as MFCCs. The correlation between those measures of the audio signal and the data extracted from the visual modality is not substantial enough to provide us with the conclusive remarks here. The overall picture, however, is that the raw extraction method and LGE score a bit higher than other methods.

In order to differentiate between the different feature extraction methods, we might ask how well they describe the real visual changes. Obviously, raw-data extraction will have the biggest advantage here as it hardly simplifies the visual information at all. Starting with the first PC, the consecutive components explain the smoothly increasing amount of the original data variation. However, it is interesting to know to what extent the geometric models can be used to reconstruct the original image data. Figure 3.6 summarizes the average reconstruction error of the original image from the extracted features depending on the number of PCs used. We performed a search for the optimal linear transformation from the extracted data to the image data:

$$I(i) = \mathbf{T}F(i) + e(i) \quad (3.5)$$

where:

i is the frame number in the video sequence,

$I(i)$ is the vector representing the original image,

$F(i)$ is the feature vector extracted from the i -th frame,

$e(i)$ is the residual error vector,

$\mathbf{T}$ is the transformation matrix that minimizes the following error measure:

$$E = \sum_i |e(i)|. \quad (3.6)$$

Figure (3.6) contains the error measure (3.6) calculated for different numbers of PCs used in reconstruction.

It can clearly be seen that LGE outperforms the other geometry extraction methods. It is however also evident that we cannot both use the extracted geometric features and expect a simple linear image restoration procedure to work efficiently.

The next step in evaluating the compared feature extraction methods is to produce the similar results for reconstruction of the audio data instead of the original image. Substituting the $I(i)$ in the equation (3.5) with the vector of the audio data $A(t_i)$ extracted from the appropriate time interval t_i we can test how well the extracted visual data can be used to predict the audio data. Figure 3.7 shows the results with $A(t_i)$ as a vector of the LPC coefficients extracted synchronously with the image acquisition. We use here overlapping Hamming windows corresponding to each video frame. The windows have 80ms span and overlap by 20ms at both sides. We obtain therefore 25 feature vectors $A(t_i)$ per second. In this case, the LGE outperforms all other extraction methods. Note however that the graph's scale has been stretched substantially as the

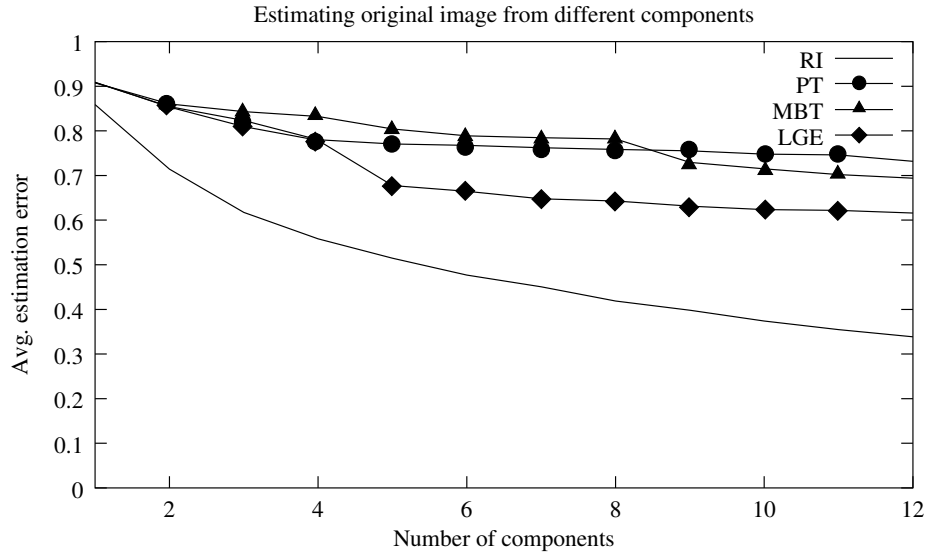


Figure 3.6: Estimating the original image from different representations of the lips.

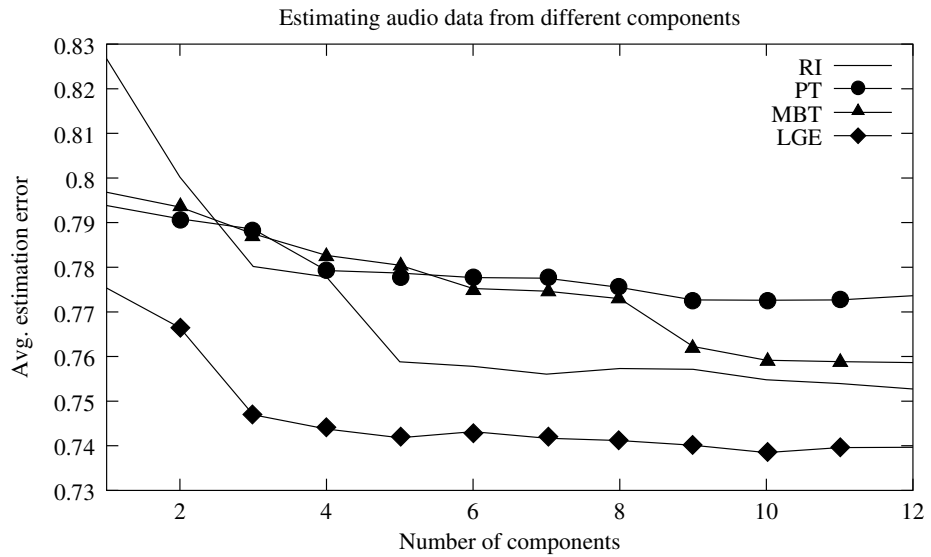


Figure 3.7: Estimating the LPC coefficients of the audio signal image from different representations of the lip movements.

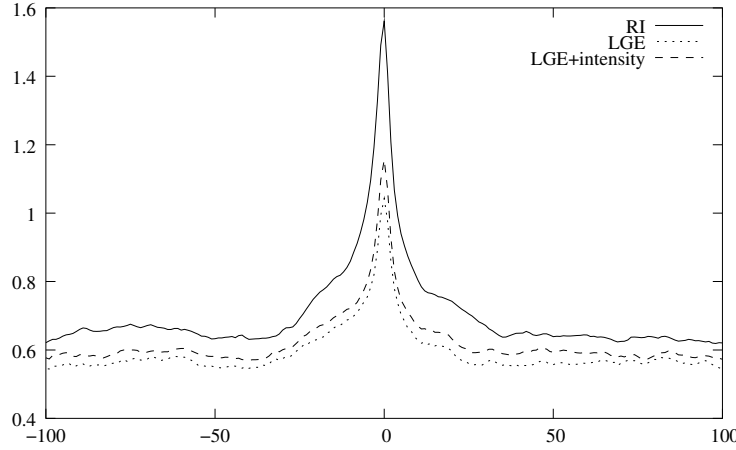


Figure 3.8: The mutual information measure for the audio and video data streams shifted between -100 frames (audio 4s before video) and +100 frames (audio 4s after video).

results do not differ enough. The linear prediction of the audio signal from the visual data remains very poor independently of the feature extraction method chosen.

The poor results of audio data prediction from the video signal may indicate some underlying problems. One of the possibilities is that the underlying relation between audio and video is highly non-linear, which seems a very plausible hypothesis. We could also hypothesize that the extracted features do not relate to the audio data in any way (linear or not). That would of course be very bad and it would make it impossible to develop a lipreading system on the basis of such measurements. One of the possible ways of testing this phenomenon is to look for the mutual information measure [MH99] of the audio and video signals. If those signals are somehow related, we may expect that the mutual information measure will peak when they are in sync and be negligible when they are far out of sync.

In order to simplify the calculations we assumed that both signals have a Gaussian nature, in which case the measure of mutual information can be calculated as:

$$H = \frac{1}{2} \log \frac{|\Sigma_A| |\Sigma_V|}{|\Sigma_{AV}|} \quad (3.7)$$

where:

Σ_A is the covariance matrix of the audio vectors distribution,

Σ_V is the covariance matrix of the video vectors distribution,

Σ_{AV} is the covariance matrix of the joint AV distribution.

The results of such a calculation where the audio and video streams are shifted out of sync are presented in Figure 3.8. It can clearly be seen that there is a relevant amount

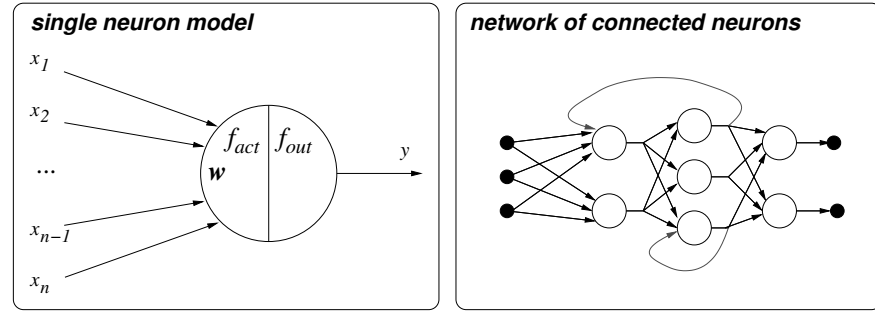


Figure 3.9: A single neuron and an example of a neural network build of such neurons.

of information that is shared between the two modalities of the speech and that this information is at least to some extent preserved in the stream of features extracted from the visual data.

3.3 Partially recurrent neural networks

The computational techniques described so far in this chapter can all be regarded as “data preparation” tools. Their aim is to provide a data stream that represents the visual part of the speech as closely as possible and at the same time is most suitable for recognition engines. These beginning stages of visual speech processing are of course necessary but it is the recognizer that does the most spectacular job. It is the recognizer that assigns the meaning to the incomprehensible data stream. This task can be performed in many different ways, based on different recognition principles. In this section we present examples of a data driven, black-box approach with the use of Artificial Neural Networks (ANNs). Although it is possible to use a wide range of ANN architectures (as can also be seen in Section 6.2), we have found that for speech related experiments, the most promising ones are so-called Partially Recurrent Neural Networks (PRNNs). Two examples of this kind of recognizers will be described following a short introduction to the principles of neurocomputing.

3.3.1 Basic principles of neural networks

An artificial neural network is a complex information processing device that is modeled after our own brain structure [HN89a]. Obviously, it is not yet possible to construct an ANN with the complexity coming even close to the complexity of the human brain, yet ANNs provide us with a flexible tool for a variety of applications. The ANN is a collection of simple processing units (called neurons) that are connected to each other in some way and which process the information in a defined order. Further an ANN is also defined by the way it is trained to perform a specific task.

A single neuron is defined as a processing unit with multiple inputs and a single output (see Figure 3.9). It is therefore a simplified model of a real neuron with its

synapses, multiple dendrites and a single axon. The network of such neurons transmits signals from the neuron output to the (possibly multiple) inputs of other neurons. There are many possible functions that can be implemented using such a broad definition, but in most of the ANN architectures, the inner workings of the neuron are implemented as follows:

Calculating activation — At this stage, the activation function (f_{act}) is used to calculate a single value of activation. The calculations use incoming input values ($\mathbf{x}$) and internal weights ($\mathbf{w}$) of a neuron. Usually a dot product is used for the activation function; there are, however, also other possibilities.

Calculating output — As soon as the activation of the neuron is known, the output function (f_{out}) is used to calculate the output of the neuron.

We can therefore formally define a single neuron as the unit that performs the following calculation:

$$y = f_{out}(f_{act}(\mathbf{x}, \mathbf{w})). \quad (3.8)$$

In the simplest case, where both activation and output functions are linear, the neuron is also called linear. Early research in the area of ANNs concentrated on linear neurons because of the simplicity of their implementation and analysis. Unfortunately, it is relatively easy to prove that the networks constructed of such neurons cannot perform complicated tasks, and can always be reduced to a simplified version containing only a single layer of independent neurons. It is this revelation that stopped the progress in this research area for a long time until the beginning of the 80s. Only after the rediscovery of the proposal of McCulloch and Pitts to use a nonlinear output function [MP43], ANNs gained scientific interest again. The activation function remained a linear dot product of input and weight vectors:

$$f_{act}(\mathbf{x}, \mathbf{w}) = \mathbf{w} \cdot \mathbf{x} = \sum_{i=1 \dots N} w_i x_i. \quad (3.9)$$

Out of many possible output functions, researchers decided to limit the range to the functions that would provide some resemblance to the real biological neuron. It had already been observed that the neurons in living organisms stay inactive as long as the net excitation of the inputs does not exceed some threshold limit. Further, if the neuron reacts to the input, its activity does not depend on the activation anymore, but remains more or less constant. If we want to represent the same behavior in the mathematical terms of the equation (3.8), the output function must satisfy the following terms:

1. $\lim_{x \rightarrow -\infty} f_{out}(x) = f_{min}$
2. $\lim_{x \rightarrow \infty} f_{out}(x) = f_{max}$
3. $\forall x, y \ x \leq y \Rightarrow f_{out}(x) \leq f_{out}(y)$

There are many functions that fulfill the above criteria. Some of them are:

Binary threshold function:

$$f_{out}(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}. \quad (3.10)$$

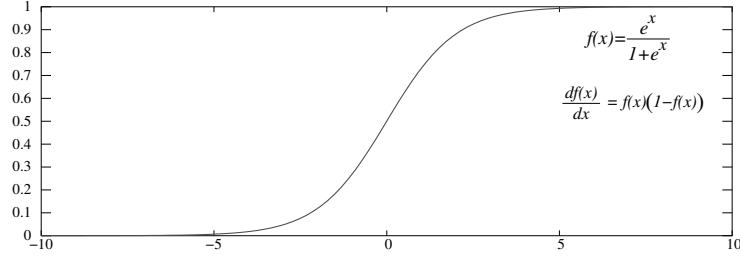


Figure 3.10: Sigmoid function.

Arc tangent function:

$$f_{out}(x) = \tan^{-1} x. \quad (3.11)$$

Sigmoid:

$$f_{out}(x) = \frac{e^x}{e^x + 1}. \quad (3.12)$$

Bipolar sigmoid:

$$f_{out}(x) = \frac{e^x - 1}{e^x + 1}. \quad (3.13)$$

Although the binary threshold function is computationally the least complex of the above functions, it is non-continuous and therefore not differentiable in the switching point, which severely limits the choice of training algorithms for such neurons. It is therefore not commonly used. From the remaining functions, the sigmoid (see Figure 3.10) is usually chosen because its derivative is easily represented in terms of the function itself:

$$\frac{df}{dx}(x) = f(x)(1 - f(x)). \quad (3.14)$$

Given the above considerations, the generic model of a neuron is usually implemented in the form of the famous perceptron:

$$y = \frac{e^{\mathbf{w} \cdot \mathbf{x}}}{e^{\mathbf{w} \cdot \mathbf{x}} + 1}. \quad (3.15)$$

Before we can describe the architecture and training of the PRNNs, we have to say something about the least complex form of ANNs – the feed-forward networks (FF-ANN). In the FF-ANN there are no loops in the connections between neurons. That is, the output of each neuron does not (recursively) depend on itself. The FF-ANN can be dissected into several layers of neurons that share common input but do not influence each other's processing. In the FF-ANN the signal can be propagated step by step from the input to the output of the network. If we consider only the straight links (disregarding the curved loops), Figure 3.9 presents indeed a FF-ANN example.

It is a proven theorem that for any continuous function that maps from a finite interval of $\mathbb{R}^N$ to $\mathbb{R}^M$, there exists a FF-ANN with just two layers of neurons that will approximate it with an a-priori given precision [HN89b]. The theorem is existential,

i.e., it does not provide any way of constructing or estimating such a network. It does not even support specification of the exact number of neurons for that network.

Functioning and useful FF-ANNs are obtained in a laborious process of supervised training. In that process, the network is presented with a set of examples that contain both inputs and expected outputs. During the training, the weights of all neurons in the network are adjusted so that the network output matches the desired output as closely as possible, minimizing the squared error of the neuron's output. For a single neuron, where at any time all inputs and the single output are directly accessible, this can be done with a simple gradient descending algorithm. Each time the neuron is presented with the input/output pair it adjusts its weights as follows:

$$\begin{aligned}
 \Delta w_i &= -\eta \frac{\partial}{\partial w_i} (T - y)^2 \\
 &= -2\eta(T - y) \frac{\partial}{\partial w_i} (T - y) \\
 &= 2\eta(T - y) \frac{\partial}{\partial w_i} y \\
 &= 2\eta(T - y) \frac{\partial}{\partial w_i} f_{out}(f_{act}(\mathbf{x}, \mathbf{w})) \\
 &= 2\eta(T - y) f'_{out}(f_{act}(\mathbf{x}, \mathbf{w})) \frac{\partial}{\partial w_i} f_{act}(\mathbf{x}, \mathbf{w}), \quad (3.16)
 \end{aligned}$$

where:

- η is a training coefficient
- y is the output of the neuron
- T is the target value for the output

Although the weight adjustment rule from the equation (3.16) seems pretty complicated, it gets a bit simpler when it is implemented for the perceptron. If we take a neuron function from (3.15), the weight w_i (acting on an x_i input) can be adjusted as follows:

$$\Delta w_i = 2\eta(T - y)y(1 - y)x_i. \quad (3.17)$$

Although the above allows for training a single neuron (or after a straightforward extension: a single layer of neurons), it does not directly provide us with the means to train larger ANNs. The problem here is that at any time, inputs are known only for the first layer while target outputs are known only for the last layer of the network. The neurons in the inner layers of the network lack appropriate T and x_i values for the equation (3.17) to be used.

One of the widely used optimization algorithms that overcomes this problem is called Error Backpropagation (BP). In this algorithm, during training the network performs a forward and backward pass for each presented input/output pair. In the forward pass, the signal values are propagated from the inputs through the following layers until all the outputs are calculated. This is a normal feed-forward pass, with the small difference that outputs of all neurons need to be stored for retrieval in the backward pass.

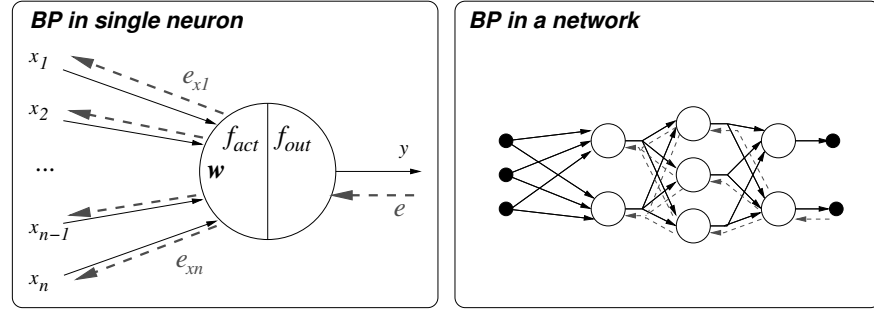


Figure 3.11: Error backpropagation in single neuron and in a network.

In the backward pass the modified version of the equation (3.17) can be applied to each neuron, starting from the output layer and ending with the input layer. The modifications are as follows:

- For a network consisting of N neurons with M inputs, we denote the output of n -th neuron as y^n and its weight vector as $\mathbf{w}^n = [w_1^n, \dots, w_{N+M}^n]$. Note that this notation suggests a fully connected neural network. Each neuron has $N + M$ inputs, each connected to one of the other neurons in the network ($w_{1 \dots N}^n$), or the external input ($w_{N+1 \dots N+M}^n$). To obtain the desired layered structure, appropriate weights must be set to 0 (disconnecting the neurons) and they must be kept constant during the whole training process.
- In the output layer, T^n is just the target output for an n -th neuron. An additional term e_n^i is calculated for each input as follows:

$$e_n^i = 2(T^n - y^n)y^n(1 - y^n)w_i^n. \quad (3.18)$$

This term represents an error propagated from neuron n to neuron i . In case the neurons i and n are not connected, this quantity is assumed to be 0.

- In the other layers, the $T - y$ term from equation 3.16 is replaced by the summed error propagated from the layers closer to the output. The expression becomes:

$$\Delta w_i^n = 2\eta \left(\sum_{k=1}^N e_k^n \right) y^n (1 - y^n) y^i. \quad (3.19)$$

The errors from these neurons are propagated further back the network in exactly the same manner as in the output layer (with the exception of the first layer, which has no neurons to propagate the error back to).

The schematic working of the BP algorithm is depicted in Figure 3.11. The process of performing forward and backward passes must be repeated multiple times for each available example. The training stops when the network outputs reach a specific stop criterion such as for example the sum-squared error threshold. The method itself can

also be improved in many ways, e.g. by using a momentum term that accumulates the weight changes or by varying the η coefficient according to the dynamics of the weight changes.

The obvious constraint of BP is that it does not allow for any loops in the network. Implications of this fact are for example that BP may be used only to train memory-less networks: networks in which at any given time the output does not depend on the network state of the previous run. This is obviously a big shortcoming for a speech-related area. Recognizing a time dependent process using a network that has no notion of the past is a challenging idea. Of course, any time varying observation can just be sampled and presented as a static entity, but the growing amount of input data that need to be passed to the network in this case makes this approach not feasible. One of the first methods suggested to solve this problem was to provide some way of fixing the network parameters (weights) in an architecture that would reflect the repeatability of processing for each time frame. The architecture obtained in this way is called the Time Delayed Neural Network (TDNN).

Another approach was to allow for some limited recurrence in the network and to modify the BP algorithm appropriately. This limited use of recurrence explains the word “Partially” in the name PRNN (the PRNNs are sometimes also called Simple Recurrent Neural Networks). In each forward pass, the neurons that have recursive links as their inputs take the appropriate output values from a previous forward pass instead of recursively looping the real output. In each backward pass, the error is propagated back only along the non-recursive links, even though all of the weights are adjusted. In this way, the network can be trained with the ease of a BP even though it contains some recursive links.

This simple approach can be extended in order to consider not only the last forward pass, but also the error of several passes prior to the current one. This extension is called Backpropagation Through Time (BPTT). We may actually treat BPTT as a normal BP algorithm but performed on the network constructed of multiple copies of the original network “unfolded” through several time steps. This way of training the PRNN outperforms the simple modification of BP discussed previously because it takes into consideration more global trends in the network dynamics. The details of BPTT implementations can be found in [RHW86].

3.3.2 Jordan neural network

Probably one of the first implementations of PRNN was presented in a widely cited technical report by Michael I. Jordan in 1986 [Jor86]. In this network there are two basic types of recursive connections. There are connections going from the network output to a special type of neurons in a so-called “plan layer”. The second type of recurrent connections are self-recursive loops within that layer (see Figure 3.12). Both connections are actually delayed, so that they transmit the values from the previous forward pass of the network. All of the recurrent connections are fixed and cannot be trained. The connections from output to plan layer usually have a value 1.0, while the weight of self-recurrent connections within this layer is the network parameter λ . In Chapter 6 we present an application of this architecture for a voice activity detection task. The Jordan recurrent neural networks (JNNs) are relatively simple to implement

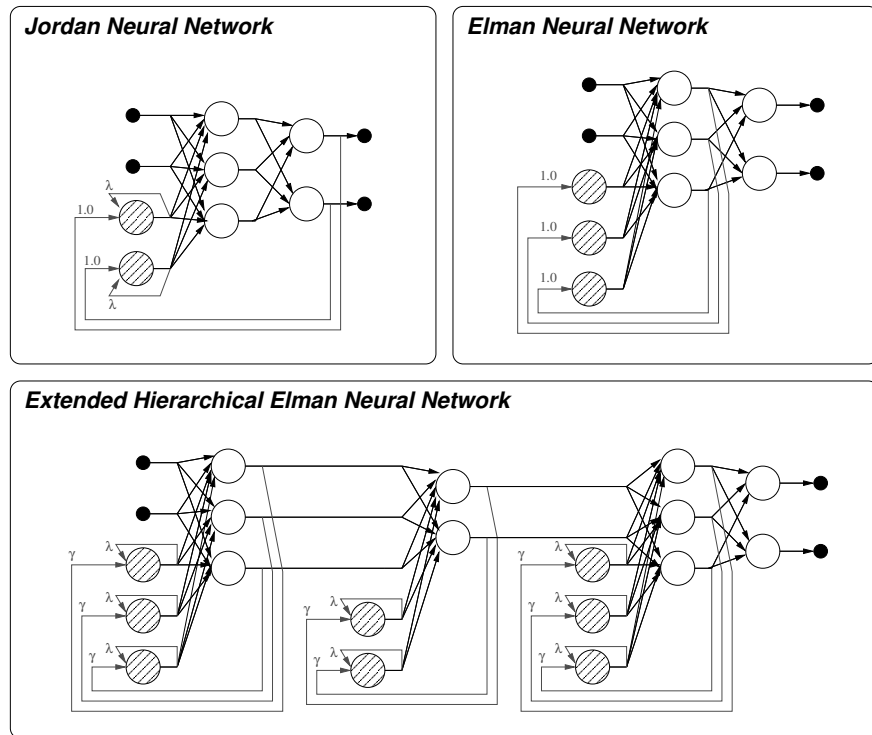


Figure 3.12: Schematic architectures of partially recurrent neural networks.

in software, the recursive nature of the connections is limited and does not incur much computational penalty. There are however some problems when we apply this approach to real-life data.

Firstly, the training of the JNN is highly sensitive to the training parameters. Errors from the output units propagate back to the network and stay in the “plan layer” for an unspecified number of forward passes. If the number of self recurrent connections in this layer is high, the influence of previous inputs will last longer. If it is low, the network will forget its previous state quickly. In order to counter the prolonged influence of output errors, it is often advised to introduce “teacher forcing” to the training procedure. This computational trick is performed by using a weighted average between the real network output and target network output in all training passes. This way, the inputs of the neurons in the “plan layer” are biased towards desired values. During the training process, the weight of this bias can be gradually changed so that the influence of the target output on the network decreases. It is advisable to also decrease the training rate in this process so that the adjustments to the weights become smaller. In this way, the network becomes self-sufficient only at a later stage of fine tuning, when the chaotic behavior is limited.

Another shortcoming of the JNN is that it allows only a very limited internal representation of time to be learned. The plan layer is limited in its size to the number of actual outputs and therefore cannot be extended independently of the network’s task. For example, in our experiments with voice activity detection (Section 6.2), there was only one output and subsequently only one neuron to constitute network’s memory. It was sufficient for the presented task, but it proved to be insufficient for, for example, vowel/consonant discrimination.

3.3.3 Elman neural network

The disadvantages of JNNs were addressed in a paper by J. Elman in 1990 [Elm90]. His design changes the JNN in two ways. Firstly, there are no self-recurrent connections to be found in the Elman recursive neural network (ERNN). The neurons that receive recursive connections (this time called “context neurons”) do not loop their output back to themselves. This change aims at limiting the chaotic behavior of the network and at simplifying the training procedure. Teacher forcing is still a desirable feature during training, but it is much easier to choose the appropriate values of teacher and training coefficients.

The second change in the network’s design, in our view the most important one, is that the context neurons are connected to the outputs of the hidden layer instead of to the outputs of the network. This change allows for a greater flexibility in designing the network. The amount of context neurons (and therefore the amount of the network’s memory) can be adjusted by changing the amount of hidden neurons. As the network is allowed to allocate the hidden neurons freely during the training (their meaning is not application bound), it is possible for the network to evolve even multi-step memory, with the length of history limited only by the number of available neurons.

There are tasks in which a neural network with only one hidden layer is not a good choice. Introducing multiple hidden layers in ERNN yields an Extended Hierarchical Elman Neural Network. This kind of network was used in our experiments with

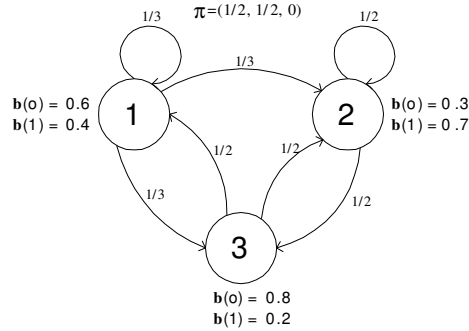


Figure 3.13: Example HMM.

vowel/consonant discrimination (see Section 6.3). The implementations of this kind of network very often re-introduce the parameter λ for self-recurrent connections together with the complementary parameter γ for the connections from hidden layers ($\lambda + \gamma = 1$).

3.4 Hidden Markov models

The Hidden Markov models (HMMs) are a powerful tool for modeling and recognizing different time-varying signals. They have been successfully applied in automatic speech recognition and lipreading, but their use is not restricted to speech related processing. One can find them for example in papers on sign language recognition [SP95] or even on modeling of human interactions [ORP00]. This flexibility of HMMs stems from their “time-warping” capabilities and the availability of efficient training algorithms.

HMMs are based on the well-known Markov chains that are often used in probability theory. The Markov chain is a probabilistic signal generator with a single state memory. It is constructed of a number of states that can be observed and probabilistic transitions. The probability of the transition from the current state to the next one depends solely on the state the model is in the current time step. Markov chains have one disadvantage that actually excludes them from direct application to most real life problems. Each state of the Markov chain is rigidly tied to one possible observation. That means that the number of states must be at least as large as the number of possible observations. This restriction renders Markov chains unsuitable for speech processing, which has an infinite number of possible observations, as the speech signal is continuous and cannot easily be quantized in small number of (useful) classes.

HMMs allow for loosening of the strict equivalency between state and observation. In each state, the HMM produces the observation chosen in a probabilistic manner. In this way, the models become “hidden”, because their current state cannot be assessed based on the observation. We can only calculate the probability that it is in a given state. The observation probability distributions at each state can either be discrete or continuous, allowing for the flexibility that contributed to the HMMs popularity. In

the following sections we will describe the inner workings of HMMs on the example of models with discrete probability distributions; the extension to continuous ones is rather straightforward though.

3.4.1 HMM generator

The HMM is defined in the following manner:

- number of states N
- number of observation symbols M
- output alphabet $\mathcal{O} = \{O_1, O_2, \dots, O_M\}$
- state space $\mathcal{Q} = \{q_1, q_2, \dots, q_T\}$ where T is the number of observations at discrete time slots $1 \leq t \leq T$
- transition probabilities $\mathbf{A} = \{a_{ij}\}$ where:

$$a_{ij} = P(q_{t+1} = j | q_t = i), \\ 1 \leq i, j \leq N, 1 \leq t < T$$

- observation probabilities $\mathbf{B} = \{b_j(k)\}$:

$$b_j(k) = P(o_t = O_k | q_t = j), \\ 1 \leq k \leq M, 1 \leq j \leq N$$

- initial state distribution $\pi = \{p_i\}$ where:

$$p_i = P(q_1 = i)$$

- full parameter set of a given model:

$$\lambda = \{\mathbf{A}, \mathbf{B}, \pi\}$$

example in Figure 3.13

3
2
 $\{0, 1\}$

$$\begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix}$$

$$\begin{array}{ll} b_1(0) = 0.6 & b_1(1) = 0.4 \\ b_2(0) = 0.3 & b_2(1) = 0.7 \\ b_3(0) = 0.8 & b_3(1) = 0.2 \end{array}$$

$$\left\{ \frac{1}{2}, \frac{1}{2}, 0 \right\}$$

Given the above definitions, the HMM generator of observations can be implemented in the following manner:

1. choose the initial state $q_1 = i$ according to distribution π and set the time counter $t = 1$
2. generate the observation $o_t = O_k$ according to probabilities $b_i(k)$
3. choose the transition to state $q_{t+1} = j$ according to probabilities a_{ij}
4. increment the time counter t and change to the new state
5. return to step 2 if $t < T$

It is therefore relatively easy to generate any desired number of sequences generated by a predefined HMM with its parameters λ . What is much more challenging (and much more useful) is the reverse process. Given the observation $\mathbf{O} = o_1 \dots o_T$, what is the probability that it was generated by the HMM with the parameters λ ? After that we may also ask the question how we can construct a model such that the class of observation strings will have the highest probability of being generated by it. The next sections deal with those questions and describe the basic mathematics behind the solutions.

3.4.2 Forward pass

Directly from the definition of HMM, if we consider a sequence of states $\mathbf{q} = q_1 \dots q_T$ and a sequence of observations $\mathbf{O}$ of equal length, the probability of producing $\mathbf{O}$ through $\mathbf{q}$ is:

$$P(\mathbf{O}|\mathbf{q}, \lambda) = \prod_{t=1}^T P(o_t|q_t, \lambda) = \prod_{t=1}^T b_{q_t}(o_t). \quad (3.20)$$

The state sequence itself can be generated with the probability:

$$P(\mathbf{q}|\lambda) = p_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}}. \quad (3.21)$$

In order to calculate the probability that $\mathbf{O}$ is generated by a model with parameters λ , we take product of the equations (3.20) and (3.21) and sum it over all possible state sequences:

$$\begin{aligned} P(\mathbf{O}|\lambda) &= \sum_{\mathbf{q} \in \mathcal{Q}} P(\mathbf{q}|\lambda) P(\mathbf{O}|\mathbf{q}, \lambda) = \sum_{\mathbf{q} \in \mathcal{Q}} p_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}} \prod_{t=1}^T b_{q_t}(o_t) = \\ &= \sum_{\mathbf{q} \in \mathcal{Q}} \left(p_{q_1} b_{q_1}(o_1) \prod_{t=2}^T a_{q_{t-1} q_t} b_{q_t}(o_t) \right). \end{aligned} \quad (3.22)$$

Calculating this probability directly in the manner presented in the equation (3.22) would not be feasible because the number of operations required grows exponentially with the length of the observed sequence. It would require $2TN^T$ multiplications and N^T additions to perform.

In order to compute this probability in an efficient way we consider all the partial sequences $o_1 \dots o_\tau$ for each time step $1 \leq \tau \leq T$. We define $\alpha_\tau(i)$ as the probability of ending in the state i after the observation $o_1 \dots o_\tau$ has been made:

$$\alpha_\tau(i) = P(o_1 \dots o_\tau, q_\tau = i|\lambda). \quad (3.23)$$

It is easy to compute $\alpha_1(i)$ from its definition:

$$\alpha_1(i) = P(o_1, q_1 = i|\lambda) = p_i b_i(o_1) \quad (3.24)$$

for each $1 < \tau \leq T$ we have from equation (3.22):

$$\begin{aligned}
 \alpha_\tau(i) &= P(o_1 \dots o_\tau, q_\tau = i | \lambda) = \sum_{q_1 \dots q_\tau, q_\tau = i} \left(p_{q_1} b_{q_1}(o_1) \prod_{t=2}^{\tau} a_{q_{t-1}q_t} b_{q_t}(o_t) \right) = \\
 &= \sum_{q_1 \dots q_\tau, q_\tau = i} p_{q_1} b_{q_1}(o_1) a_{q_1q_2} b_{q_2}(o_2) \dots a_{q_{\tau-2}q_{\tau-1}} b_{q_{\tau-1}}(o_{\tau-1}) a_{q_{\tau-1}i} b_i(o_i) = \\
 &= \sum_{j=1}^N \left(\sum_{q_1 \dots q_{\tau-1}, q_{\tau-1}=j} p_{q_1} b_{q_1}(o_1) \dots a_{q_{\tau-2}j} b_j(o_{\tau-1}) \right) a_{ji} b_i(o_i) = \\
 &= \left(\sum_{j=1}^N \alpha_{\tau-1}(j) a_{ji} \right) b_i(o_i). \tag{3.25}
 \end{aligned}$$

The above equation shows that it is possible to compute $\alpha_\tau(i)$ iteratively if we know $\alpha_{\tau-1}(j)$ for all possible states ($1 \leq j \leq N$). The computations here involve only $N + 1$ multiplications and $N - 1$ summations. Moreover, if we know $\alpha_T(j)$ for all possible ending states, we can calculate the total probability that $\mathbf{O}$ is produced:

$$P(\mathbf{O} | \lambda) = \sum_{j=1}^N \alpha_T(j). \tag{3.26}$$

Taking all the above into consideration, in order to calculate $P(\mathbf{O} | \lambda)$, we start from equation (3.24), compute it for each starting state, then iterate $T - 1$ times using equation (3.25), and finally obtain the result according to equation (3.26). This iterative algorithm for computing the probability is known as “forward pass” and can be performed in the order of TN^2 calculations. This compares quite favorably to the direct method of calculation, which requires the order of $2TN^T$ operations.

Even though the forward pass requires some additional memory for storing intermediate results, its size is only linearly related to the size of model and does not depend on the length of observation sequence. At each iterative step of calculations, only N results from previous step are needed. This can be best shown on a diagram such as that in Figure 3.14. In typical implementations, the algorithm can be optimized even further by removing the calculations that involve $a_{ij} = 0$ in equation (3.25). Figure 3.14 depicts this: in it the appropriate connections between states in consecutive time steps have been removed.

3.4.3 Viterbi algorithm

A forward algorithm enables implementing a working HMM recognizer of incoming signals. We can construct a number of models and classify the observation sequences according to which the model most probably produced them. There is one fundamental problem though: we have absolutely no means of guessing the appropriate set of parameters λ . In some very simple cases with only a small number of parameters (a limited output alphabet $\mathcal{O}$, short observation sequences etc.) it might be possible to

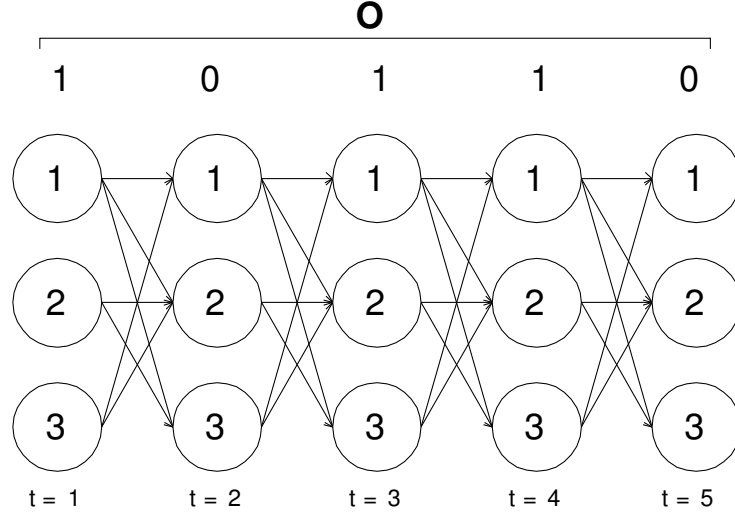


Figure 3.14: Forward pass in HMMs.

obtain those parameters analytically, but this is not the case for any speech-related application. In order to obtain a useful recognizer framework, we also need algorithms for estimating the model parameters based on set of training data.

The forward algorithm does not provide any useful information about the inner workings of the HMM. It provides a single value $P(\mathbf{O}|\lambda)$, without any clues to which parameters contributed in which way to the result. If we want to estimate the parameters of the model, we need to know more. One piece of useful information is, for example, which state sequence contributed the most to the total probability, that is, the path $\mathbf{q}$ that maximizes $P(\mathbf{q}|\mathbf{O}, \lambda)$.

An efficient way of finding the most probable path through the model is with the Viterbi algorithm. This algorithm is actually very similar to the forward pass, except that it requires the use of the maximum operator instead of summation in equations (3.25) and (3.26). Using the following set of equations:

$$\delta_1(i) = p_i b_i(o_i) \quad (3.27)$$

$$\delta_\tau(i) = \left(\max_{j=1 \dots N} \delta_{\tau-1}(j) a_{ji} \right) b_i(o_i) \quad (3.28)$$

$$\tilde{P} = \max_{j=1 \dots N} \delta_T(j) \quad (3.29)$$

we can efficiently compute the probability of the most probable path. Moreover if at each time step and at each state we also store the information about the state that was the “winning” contributor in equation (3.28):

$$\psi_\tau(i) = \arg \max_{j=1 \dots N} \delta_{\tau-1}(j) a_{ji} \quad (3.30)$$

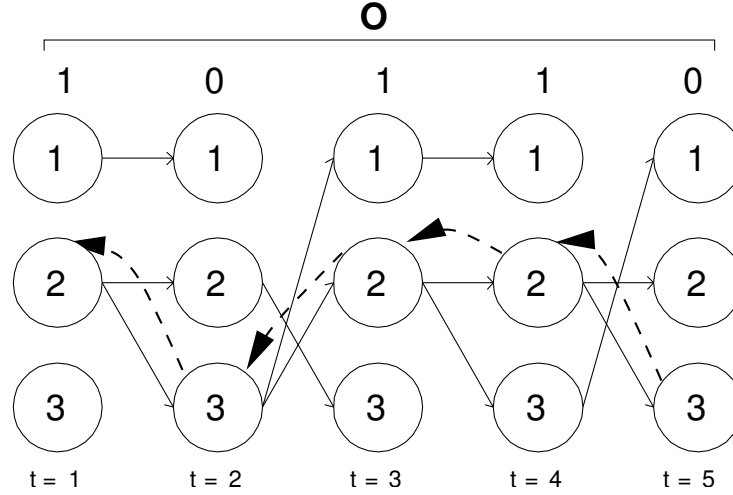


Figure 3.15: Viterbi algorithm.

we can use those values to “back trace” from q_T to q_1 . This process is schematically visualized in Figure 3.15.

The Viterbi algorithm by itself can be used to train the models to some extent. When we perform the Viterbi algorithm on some model and on a set of observations, each best path through the model constructs in fact the correspondence between each state and a set of observations. If we want the model to maximize the probability of generating the whole set of observation sequences, we should maximize the probability of generating the observations matched to each of the states. This procedure called “Viterbi alignment” makes it possible to adapt the parameters within the state ($b_i(k)$), but it does not provide a means for adjusting the transition probabilities (a_{ij}) and therefore it is usually only used as an initialization step for further training.

3.4.4 Baum-Welsh re-estimation

The procedure that allows training of all parameters of the model is called Baum-Welsh re-estimation. It is a procedure that – given the set of observation sequences – yields the parameters λ that generate all of the sequences with the highest probability.

In order to perform Baum-Welsh re-estimation we need to define a few more variables. Firstly, a backward probability $\beta_\tau(i)$ that is analogous to the forward probability $\alpha_\tau(i)$, but describes the probability of generating the part of the observed sequence remaining after time step τ :

$$\beta_\tau(i) = P(o_{\tau+1} \dots o_T | q_\tau = i, \lambda). \quad (3.31)$$

This probability can be obtained in an iterative way very similar to the one described for $\alpha_\tau(i)$.

Using both forward and backward probabilities we can calculate the probability

that HMM is in state i at time τ given the observation:

$$\gamma_\tau(i) = \frac{\alpha_\tau(i)\beta_\tau(i)}{P(\mathbf{O}|\lambda)}. \quad (3.32)$$

Another thing that can be expressed in terms of backward and forward probabilities is the probability that HMM is in state i at time τ and in state j at time $\tau + 1$:

$$\xi_\tau(i, j) = \frac{\alpha_\tau(i)a_{ij}b_j(o_{\tau+1})\beta_{\tau+1}(j)}{P(\mathbf{O}|\lambda)}. \quad (3.33)$$

The probabilities from equations (3.32) and (3.33) are enough to formulate the estimates of the model's parameters:

$$\tilde{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \quad (3.34)$$

$$\tilde{b}_i(k) = \frac{\sum_{t=1, o_t=k}^T \gamma_t(i)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \quad (3.35)$$

$$\tilde{p}_i = \gamma_1(i). \quad (3.36)$$

The above equations are the core of Baum-Welsh re-estimation procedure. Unfortunately they are recursive; at each point of the training procedure we can only operate on the estimated values ($\tilde{\mathbf{A}}$, $\tilde{\mathbf{B}}$ and $\tilde{\pi}$), not the real ones ($\mathbf{A}$, $\mathbf{B}$, π). They appear therefore on both sides of the equations. The procedure must therefore be repeated multiple times and slowly converge to the appropriate values.

3.4.5 Developing a speech recognizer

In previous sections we focused on the mathematical foundations of HMMs. After this introduction it is now time to describe briefly how this can be put to use in speech signal recognition. Obviously, the speech signal can be observed and somehow parametrized. An auditory signal is usually processed in the frequency domain in a way that mimics the human sound perception system. Typical feature extraction techniques for the visual modality of speech were already discussed in Section 3.1. The obtained observation vectors (auditory, visual or both) must be stored and properly labeled, so that it forms suitable training material.

After gathering training data and a test set we have to decide what kind of models would best represent speech production mechanisms. We have to choose the granularity of representation as well as the actual form of the models. The most common choice is to model each phoneme with a single model. The models are also constrained to some extent so that they form chains without loops between different states (loops within a state are allowed). The motivation for such a choice is the observation that, because of the way the phones are produced, each phone proceeds from its onset through the middle part to its offset always in the same order. Even though different parts of the phone have different frequency characteristics, may vary in duration and can

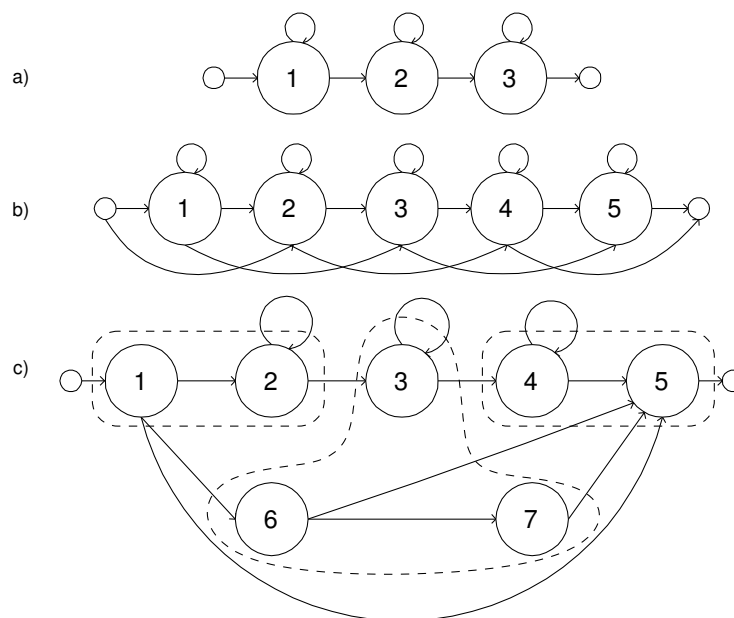


Figure 3.16: HMM architectures used for phoneme level recognition.

sometimes even be omitted, they are never reordered. The variations in the phone realizations can be seen as nonlinear stretching or compression of time with respect to the “ideal phone”. The HMM architectures that allow for such time deformations are depicted in Figure 3.16. Figure 3.16a shows the simplest time-warping model which allows for different durations of the three phases of the phone, but it does not allow the omission of any of them. The model from Figure 3.16b is not only extended with two more states, but it also contains transitions that skip some of them on the way from the beginning to the end. The model in Figure 3.16c allows for even more variation in time distribution between the states. It is also the most computationally heavy of the depicted models, with its seven states and twelve transitions. Yet its complexity can be reduced by tying the parameters of some of the states (see dashed surroundings), so that it actually contains only three different output probability distribution functions.

Developing any kind of speech recognizer is laborious work. It involves a lot of time invested in data acquisition and editing in order to obtain the speech data in a suitable form. The acquired data must be labeled and validated so that it is both clean of errors and irrelevant signals and at the same time representative of a real-life application. Then, it is usually beneficial to perform feature extraction beforehand in an off-line manner so that this need not be done each time the training process is reiterated. Only after the data has been preprocessed, the actual training and validation can take place. The amount of work involved in all those steps would actually be already sufficient for a doctoral dissertation (see [vA92]) if they were all to be implemented from scratch.

Fortunately, there is a freely available tool called the Hidden Markov Toolkit (HTK)

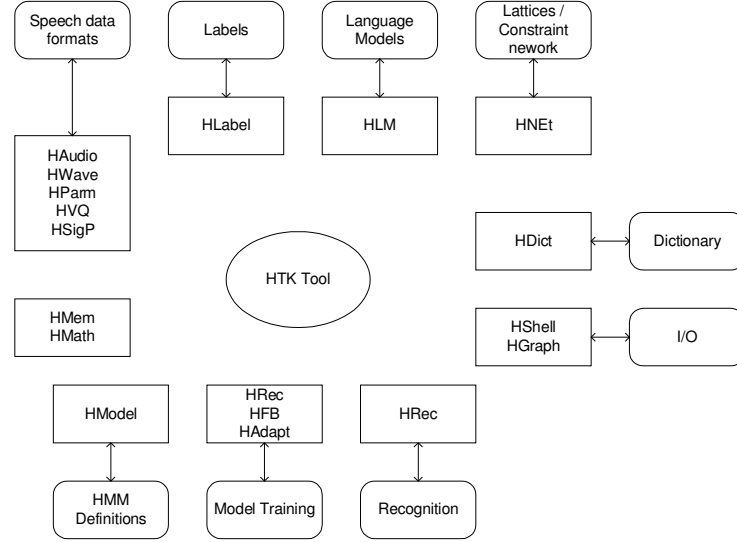


Figure 3.17: HTK toolkit overview.

that supports all those development stages [You93, WOY94, WLO⁺95]. The HTK is a highly modular software package which covers most of the tasks that need to be performed in order to obtain a functional recognizer (see Figure 3.17). The modules can be used independently of each other without any penalty on further processing. For example, even though the HTK does not contain any tools for working with visual modality of the speech, we executed all of the experiments presented in this thesis using HTK. We only had to replace the audio processing modules from HTK with our own processing software. The data files that are used within HTK are well documented so that writing HTK compatible software modules is a straightforward task.

The HTK provides command-line tools for data preparation such as `HSLab` that supports labeling of the sound files, or `HList` for inspecting contents of any data file. A lot of work with data labels can be automated and scripted using the `HLEd` tool. Moreover, the `HLStats` and `HQuant` support building vector quantization codebooks for discrete probability HMMs. The training procedure can be performed either using the generic Baum-Welsh re-estimation procedure (`HRest`) or with the embedded variant (`HERest`) in which case the strict frame-per-frame labeling of data is not required. Finally, the recognition is performed using a single program `HVite`. The results of recognition on large data sets can be summarized with `HResults`. In this way, the whole recognizer development process can be performed within the HTK framework, as depicted in Figure 3.18.

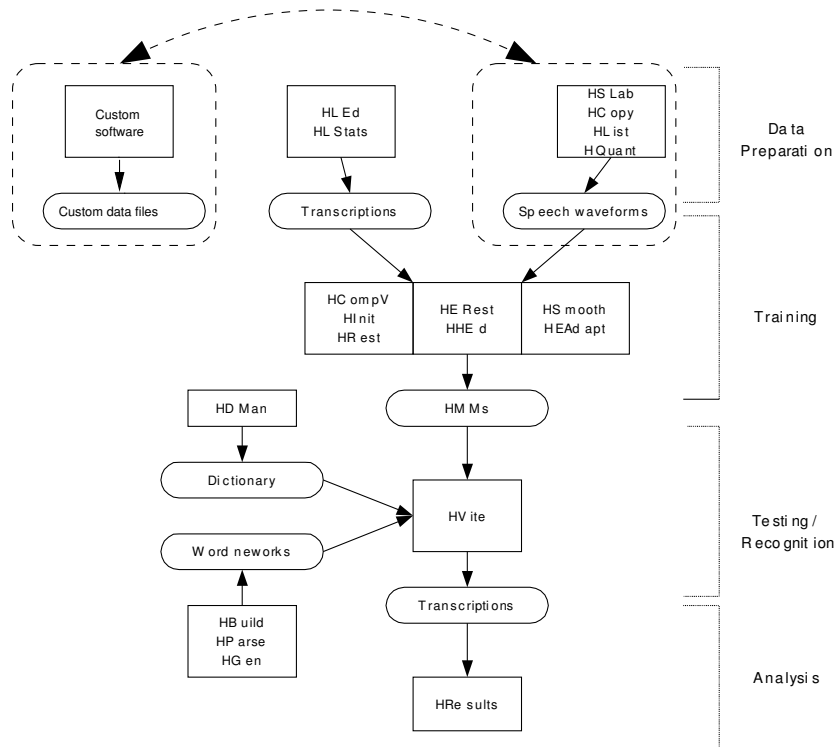


Figure 3.18: Data and processing flow in HTK during recognizer development.

Peter jumped at the drawers, scattering their contents to the floor with both hands, as kings toss ha'pence to the crowd. In a moment he had recovered his shadow, and in his delight he forgot that he had shut Tinker Bell up in the drawer.

J. M. Barrie, **Peter Pan**

Chapter 4

Lip geometry estimation

In this chapter we present a robust model for video sequence processing that extracts the data describing the mouth shape in the image [WR01a]. The method allows computationally light data extraction from the video sequence. The extracted data can then be used in lipreading applications. This chapter deals with the two stages of the lipreading processing pipeline that are highlighted in Figure 4.1: preprocessing and feature extraction.

Because of the huge amount of data contained in a visual signal it is usually computationally not feasible to apply signal processing techniques directly. Commonly, specific features or parameters are derived from each video frame and further processed by fitting a geometrical model on the image and using the obtained model parameters as input to the recognition system [CTC96, LTB96a]. This way of video processing was already discussed in Chapter 3. In the following sections we will describe step by step a feature extraction method developed during this doctoral research. The presented technique is unique because it does not rely on any a-priori geometrical lip model.

4.1 Lip-Selective Image Filtering

As the first step in video processing, we have to locate the mouth in the image somehow. Fortunately for us, in a video containing only the face, the lips have a distinct

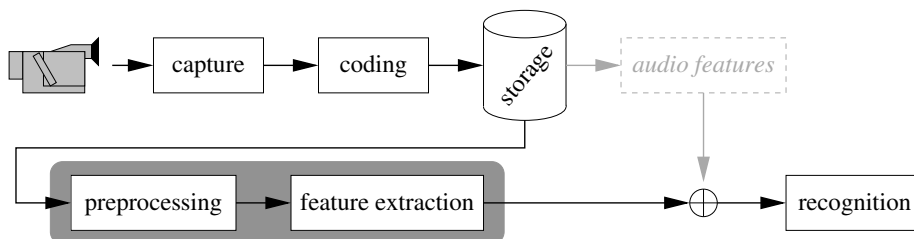


Figure 4.1: Lipreading as a processing pipeline.



Figure 4.2: Parabolic shape hue-based lip-selective filtering example, (a) original image, (b) hue-filtered image with the mean value of resulting distribution depicted ($h_0 = 0$, $w = 0.25$).

coloring that allows us to find them without the need for complicated object recognition techniques. In order to utilize this fact, we need to apply some sort of *lip-selective* filter to the image. In our current research we use several different filters depending on the illumination conditions and the quality of the recorded video sequences.

4.1.1 Hue-based filtering

This is probably the simplest possible way of performing *lip-selective* filtering. It was first proposed in [CTC96] and incorporates a parabolic shaped filter that cuts off the colors that fall outside a given interval:

$$F_{hue}(h) = \begin{cases} 1 - (h - h_0)^2/w^2, & |h - h_0| \leq w \\ 0, & |h - h_0| > w \end{cases} \quad (4.1)$$

The filter is defined by the center of the interval h_0 and its half width w . Filtering the image starts with calculating the *hue* color component of a given pixel and then passing it through the F_{hue} function. In order to select lips in the image, the value of h_0 must be around 0 (red color) and w about 0.25 (see Figure 4.2). Both values have to be adjusted for a specific video sequence in order to retain the filtering quality.

We also may use the *hue*-based filter in combination with the similarly shaped parabolic *value* filter if the difference between the *hue* components of the face and lips is not sufficient for robust lip detection. Using a product of those two filters, one can for example remove the noise of the dark or bright background, where the *hue* values behave rather randomly. Both dark and bright ends of the HSV (*hue*, *saturation*, *value*) color space allow for large changes in both *hue* and *saturation* components with relatively small changes in the overall color perception. Therefore the filter only based on *hue* does not perform well in those color space regions.

Parameters of both *hue* and *value* based filters can be set automatically by optimizing the filter's separation of color histograms prepared for the lip region and the rest of the face. We use here a simple interaction with the user in which the lip region in the initial video frame must be selected. From the selected region we extract the color histogram and perform the optimizations. Resulting filters are then presented to the user

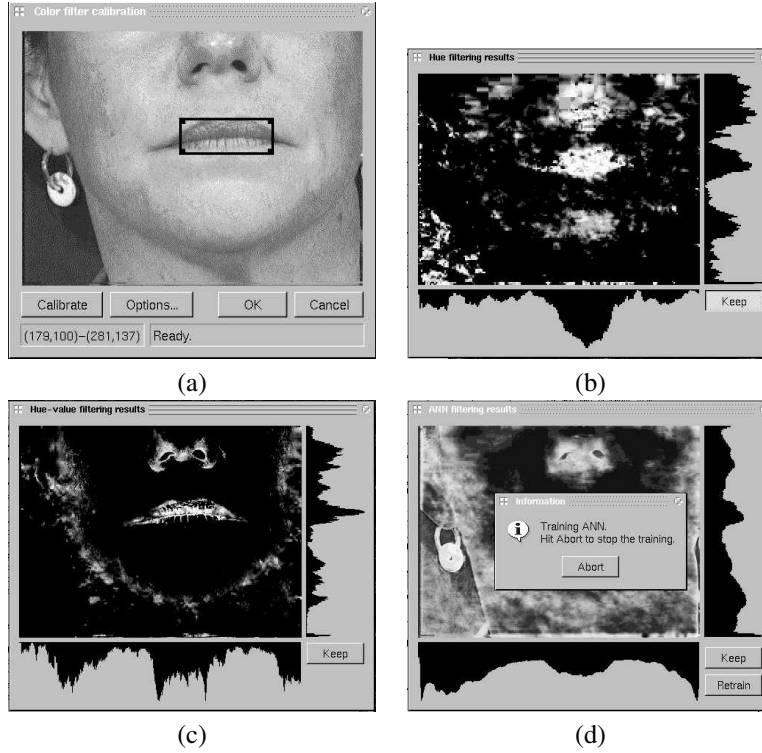


Figure 4.3: User-supported calibration of lip-selective filters. (a) Region selection, (b) *hue* based filter, (c) combination of *hue* and *value* based filters, (d) ANN during training.

for confirmation (see Figure 4.3). The total calibration time for both filters does not exceed a couple of seconds, even on a mid-range computer, and has to be performed only once for each recording setup.

4.1.2 Filtering with ANNs

A *hue*-based filter is very simple and computationally very effective. Unfortunately, during our experiments we found that in many cases, if the illumination of the face is not perfect, the *hue* component itself is not sufficient for proper lip selection (see Figure 4.4b). Instead of putting additional constraints on the filtered color (such as a threshold value of *saturation* or *value* component), we decided to use a black-box approach. We trained a simple feed-forward neural network and used it as a filter. The network that was used has only 5 neurons in a hidden layer and one output neuron. It was fed with the *RGB* values from a pixel in the image. The small size of the network allows for on-line training, so that we could use different networks for different video sequences as long as the first frame in the video sequence was prepared with the tar-

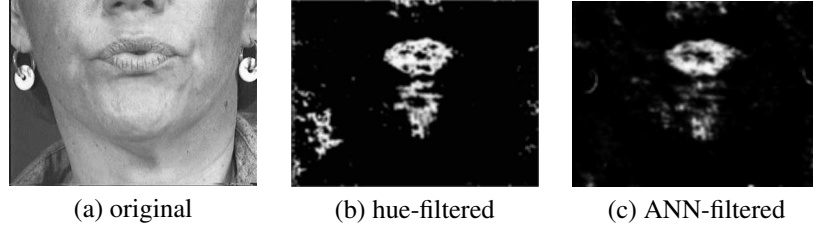


Figure 4.4: Using different lip-selective filters, (a) original image, (b) hue-filtered image with optimal filter parameters, (c) ANN filtered image.

get outputs for the network. Using the same user interface as that for calibrating the parabolic filters described earlier, we used the user-selected region to mark pixels as belonging to lips. The training of the ANN based filter took significantly more time than optimizing parabolic filters, but it could be performed well within a minute on a typical system.

4.1.3 Region of interest extraction

Even with optimal filtering of the mouth, in some cases the filtered image contains unwanted artifacts. A typical example can be seen in Figure 4.4b-c, where the shadow under the chin cannot be distinguished from the lips with sufficient certainty. In order to reduce the impact of such occurrences, we must extract a region of interest (ROI) from the whole image. We do this by using vertical and horizontal signatures of the image. Each signature is a vector containing sums of the pixel values for a specific row or column. The boundaries of the ROI can be found by looking for raising and falling edges in each of the vectors. Even if the artifacts are big enough to appear on the image signatures, their influence can be removed by assuming temporal consistency of the ROI boundaries between consecutive frames. Only the part of the filtered image that falls within the ROI was further processed.

4.2 Feature Vectors Extraction

From this point on, the filtered image is treated as a bivariate distribution $I(X, Y)$ (after normalization). The mean of this distribution: $[EX, EY]$ approximates accurately and in a stable way the center of the mouth. Using this value, we transform the image into polar coordinates:

$$J(a, r) = I(EX + r \cos(a), EY + r \sin(a)) \quad (4.2)$$

and we concentrate on the conditional distributions $J(a = \alpha, r)$. We therefore define the following functions of mean and variance values for any angle α :

$$M(\alpha) = \frac{\int_r J(\alpha, r) \cdot r \cdot dr}{\int_r J(\alpha, r) dr} \quad (4.3)$$

$$\sigma^2(\alpha) = \frac{\int_r J(\alpha, r) \cdot (r - M(\alpha))^2 dr}{\int_r J(\alpha, r) dr}. \quad (4.4)$$

As the image is discrete rather than continuous, all of the values are obtained from summation rather than integration, so we only operate on estimations of those values, namely $\widehat{M}(\alpha)$ and $\widehat{\sigma}^2(\alpha)$. The vectors resulting from sampling of those functions for one of the video sequences can be seen in Figure 4.5.

The value of $\widehat{M}(\alpha)$ for a specific angle relates directly to the distance of the lips from the center of the mouth in a given direction. Therefore a vector constructed from those values for a number of angles describes the shape of the mouth on a given image. Such shape profiles were gathered from a video sequence of spoken Dutch and grouped based on the visemes. Figure 4.6 shows three profiles for visemes [A], [o u O y Y 2: 9:] and [p b m]. In order to obtain scale independence, the vectors were scaled so that all their values fit in the $\langle 0, 1 \rangle$ interval. It can be seen that the [ou] viseme can be distinguished from the others by its relatively high $\widehat{M}$ value in the middle. There is however no feasible way of distinguishing between [A] and [p b m] visemes. The lip shapes that correspond to those visemes are completely different (mouth closed versus mouth opened), but scaling of feature vectors removes all of those differences.

As the values of $\widehat{M}$ alone are not sufficient for extracting useful data from the video sequence, we use additional information about the variance of conditional distributions ($\widehat{\sigma}^2$). Those values are also scaled in order to obtain size independence, but the scaling factor is directly determined by the values of $\widehat{M}$ rather than $\widehat{\sigma}^2$. Therefore the resulting feature vectors provide additional information about the lips. As an example, the [A] viseme clearly shows much lower $\widehat{\sigma}^2$ values than the other visemes (see Figure 4.7). Obviously, the lips of a wide-stretched mouth, appear thinner than those of a closed mouth when related to the overall size of the mouth.

The last stage of feature vector extraction is sub-sampling of the $\widehat{M}$ and $\widehat{\sigma}^2$ functions into vectors that can be further processed. Obviously, the chosen sampling rate (vector dimensions) is a compromise between accuracy and processing efficiency. The longer the vectors, the more information on the original distribution they contain but the longer it takes to extract and process them. After some experiments with the data we chose to use the 18-dimensional vectors for both $\widehat{M}$ and $\widehat{\sigma}^2$ (see [WR00]).

4.3 Person-Independent Feature Space

One of the biggest challenges in developing a signal processing model for speech recognition is to separate the production system dependent part from the speech dependent part of the signal. In other words, it is important to transform the signal in such a way that it will no longer be person dependent. Using PCA as one of the processing steps guarantees the dimensionality reduction, but it does not guarantee that the resulting space will be person independent. Even worse, we may anticipate that for some techniques (such as raw-data extraction), the main varying factor will be person dependent (or even dependent on the recording setup and environmental factors).

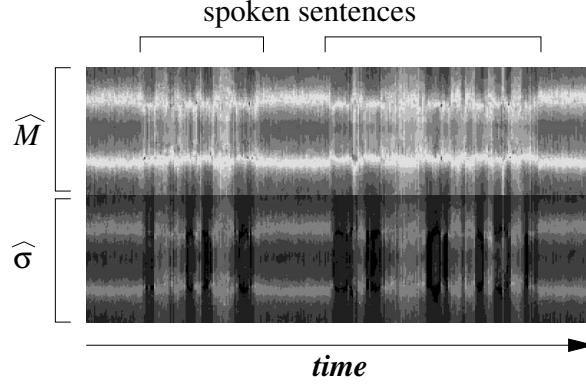


Figure 4.5: Pairs of $\widehat{M}(\alpha)$ and $\widehat{\sigma^2}(\alpha)$ vectors extracted from a video sequence. The periods of silence around two spoken sequences can be seen clearly.

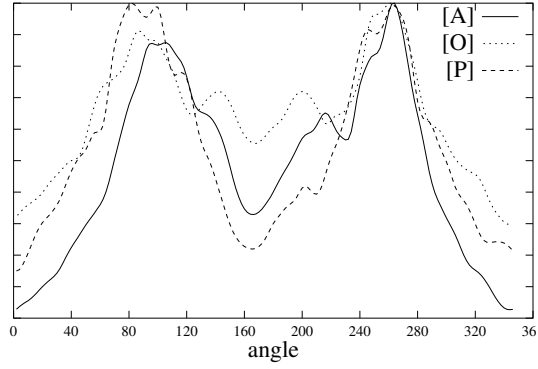


Figure 4.6: Average $\widehat{M}(\alpha)$ values for specific visemes in a video sequence.

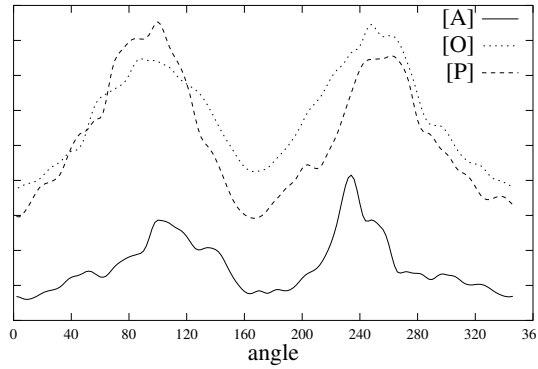


Figure 4.7: Average $\widehat{\sigma^2}(\alpha)$ values for specific visemes in a video sequence.

Table 4.1: Raw data distribution overlap between 4 persons.

	P1	P2	P3	P4
P1	1.000000	0.038908	0.572069	0.475456
P2	0.038908	1.000000	0.065154	0.204072
P3	0.572069	0.065154	1.000000	0.684076
P4	0.475456	0.204072	0.684076	1.000000
average overlap: 0.33996				

Table 4.2: LGE distribution overlap between 4 persons.

	P1	P2	P3	P4
P1	1.00000	0.84940	0.55526	0.86293
P2	0.84940	1.00000	0.22421	0.59294
P3	0.55526	0.22421	1.00000	0.71962
P4	0.86293	0.59294	0.71962	1.00000
average overlap: 0.63406				

In order to test whether the data representation is person independent we may first check whether or not measurement distributions for different persons overlap. If they do not overlap, the measurement space is much more suited for person identification than for the speech recognition. To investigate this overlap we here use a multi-modal overlap measure (MOM) [KS96] with the *min* operator as the objective function:

$$MOM_{p_i, p_j} = \int_{v_p} \min(p_i(v), p_j(v)) dv_p \quad (4.5)$$

where p_i and p_j are the measurement distributions for person i and j respectively. The integration is done over the total space v_p in which any of the distributions has a non-zero value. Tables 4.1 and 4.2 present the results of such calculations. The plots of the corresponding distributions can be seen in Figures 4.8 and 4.9. It can be seen that the average overlap between features extracted using raw-data extraction is almost twice lower than the overlap from LGE features. That indicates that the unprocessed image data contains a lot of person-specific components that are of no interest to us. Even though the LGE scores are much higher, they are still not entirely satisfactory.

The problem here is that even though the data extraction techniques capture the proper and relevant information from the incoming signal, they cannot guarantee that for different subjects the same dynamic ranges will be preserved. We may visualize the problem the way it is depicted in the first row of Figure 4.10. Experiments show that independent of the subject, the first few components represent the same underlying processes. Independently of a person's lip shape, the first PC reflects the mouth opening and the second mouth stretching. The consistency between the data extracted from the measurements and our descriptive labels referring to the physical changes of the mouth shape suggests that using a Person Adaptive PCA (PAPCA) we may actually acquire a more reliable and more useful projection of the data set. The underlying principle is depicted in the second row of Figure 4.10.

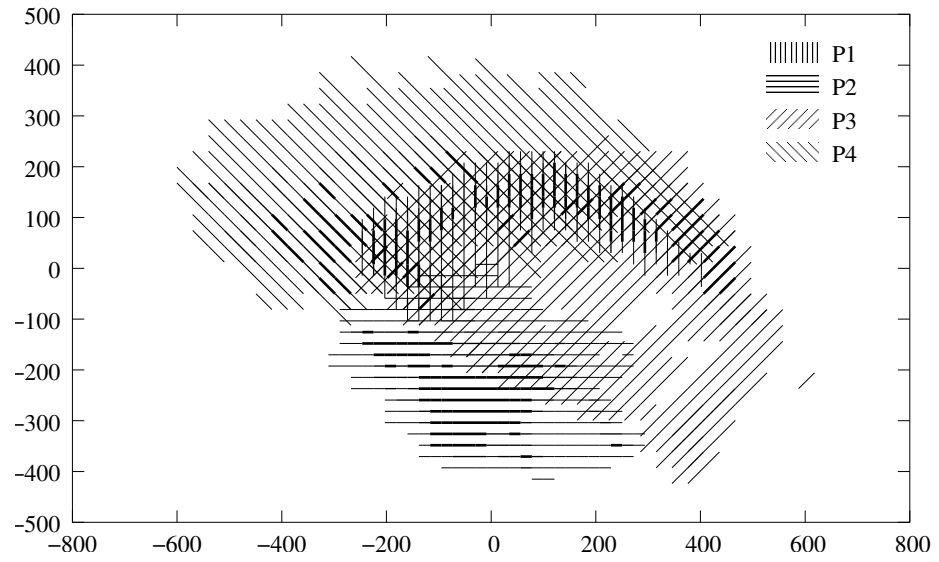


Figure 4.8: Four persons, raw image data extracted.

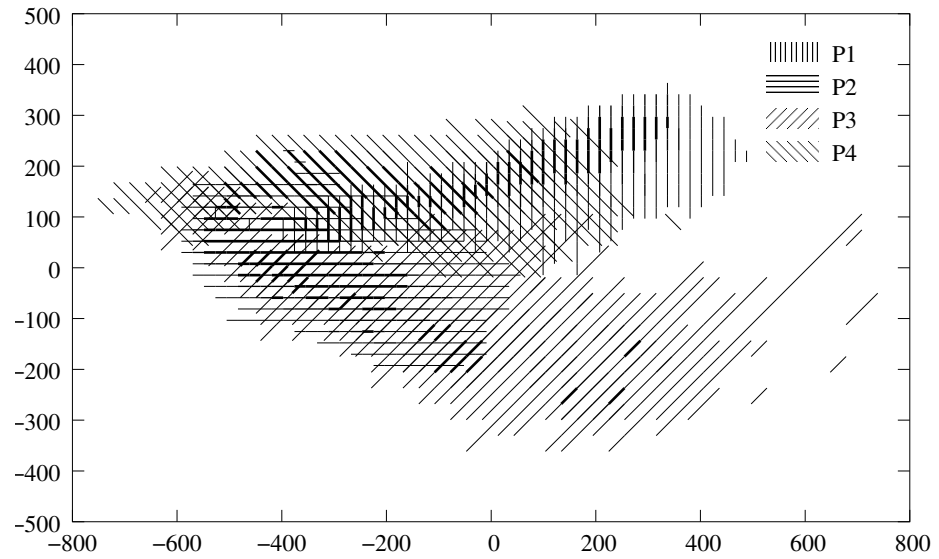


Figure 4.9: Four persons, LGE data extracted.

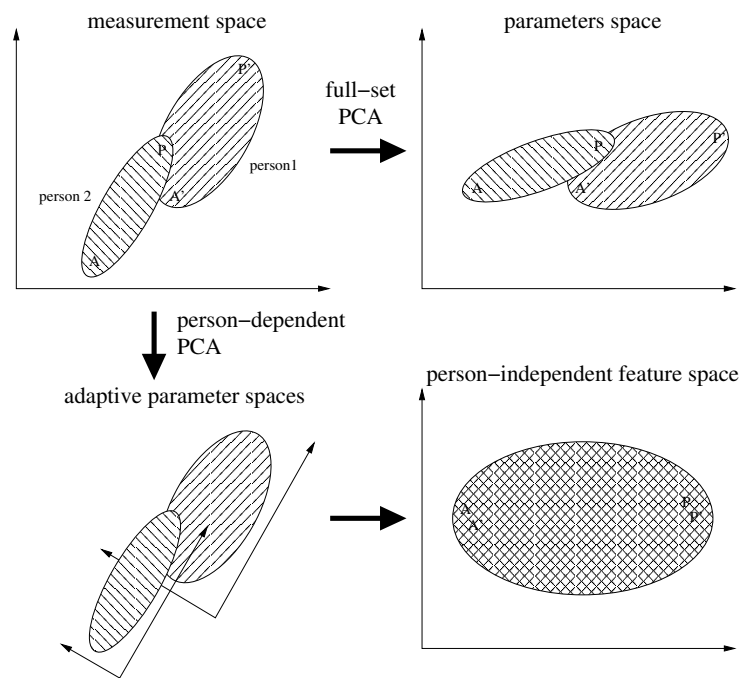


Figure 4.10: Full-set and person-adaptive PCA illustration.

There are some potential issues that have to be considered when performing a PAPCA. Firstly a suitable projection matrix should be constructed within a short period of time; this matrix should not require a large set of recordings from each subject. Our experiments show that in case of the LGE, the projection stabilizes within about 30 seconds of the speech data. That is, 30 seconds of speech data usually covers a whole range of possible mouth shapes. What is true for the geometric models that provide a limited number of extracted features does not hold for the raw-image processing technique. In that specific case the amount of data extracted from the image is about two orders larger. That makes this method in combination with PAPCA unsuitable for use in on-line systems.

The other caveat in performing PAPCA is that although the directions of the components reflect the information in the data set, their orientation is purely coincidental. There is no guarantee that the PCs will not *flip* to opposite orientation. This can be remedied in different ways. It is possible for example to build a recognizer in such a way that it is invariant of the sign of the feature vector components. Although this solution is very simple, it influences negatively the number of classes in the input that can be distinguished by the recognizer. Another solution would be to put specific constraints on the projection matrix based on our knowledge about the possible mouth movements; this is the one we chose to implement.

The third relatively simple problem that remains is that the dynamic ranges of the shape changes may be different for different people. Fortunately this can be fixed relatively easy by using the variance in each of the PCs to scale the data appropriately.

The results of projecting data from three different subjects using the adapted transformations is depicted in Figure 4.11. All of the gathered data sets have the same *U*-shape structure, which relates to the fact that in normal speech the mouth is a bit narrowed when it is closed or fully open, and when it is maximally stretched it is usually also half opened. Moreover the distribution overlap for all subjects remains above 0.8.

What remains to be checked is whether the seemingly similar data structure also results in similar placement of the shapes relating to the same visemes. For this purpose we investigated the degree of overlap between different phonemes and between different subjects. The results for phonemes *a* and *p* for four different persons are summarized in Table 4.3. The results in this table show that the overlap between the same phoneme is always above 0.5 independently of the subjects, while the overlap between different phonemes is always below 0.5. That means that in the end the projection based on the PAPCA in the geometry data provides us with the Person-Independent Feature Space (PIFS) where phonemes/visemes can be distinctively located. The example of improvements gained by using PIFS in lipreading tasks can be found further on in this thesis in Section 5.4. The same technique is also used in our bimodal speech recognizer described in Chapter 7.

4.4 Intensity features extraction

The shape of the lips is not the only determinant of a spoken utterance. There are some other important factors such as the position of the tongue, teeth etc. Some of them

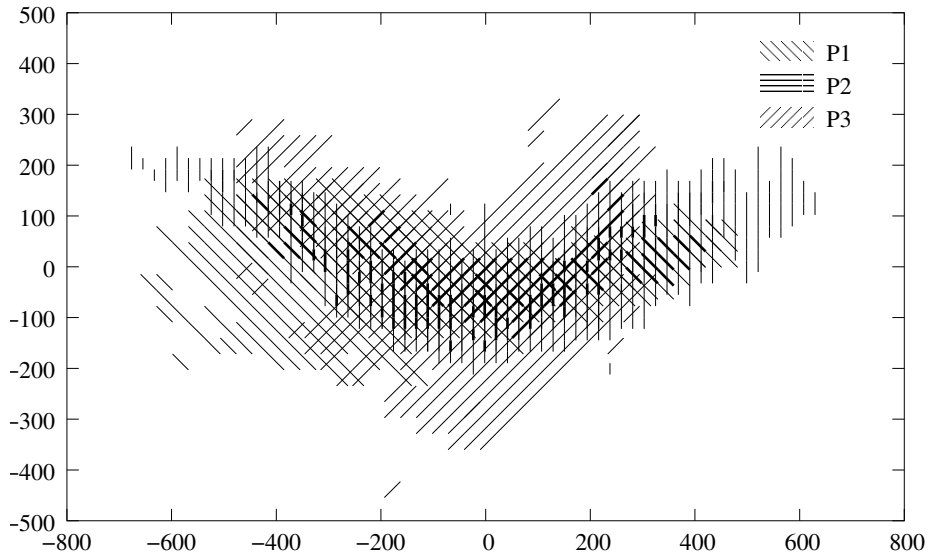


Figure 4.11: Estimated geometry data visualized in PIFS.

Table 4.3: Overlap between phoneme distributions in person-specific LGE data.

	phoneme a				phoneme p			
	1	2	3	4	1	2	3	4
a	1.000	0.790	0.760	0.668	0.544	0.531	0.498	0.416
	0.790	1.000	0.556	0.625	0.400	0.477	0.459	0.394
	0.760	0.556	1.000	0.660	0.252	0.303	0.326	0.251
	0.668	0.625	0.660	1.000	0.378	0.387	0.450	0.415
p	0.544	0.400	0.252	0.378	1.000	0.713	0.683	0.546
	0.531	0.477	0.303	0.387	0.713	1.000	0.678	0.609
	0.498	0.459	0.326	0.450	0.683	0.678	1.000	0.715
	0.416	0.394	0.251	0.415	0.546	0.609	0.715	1.000

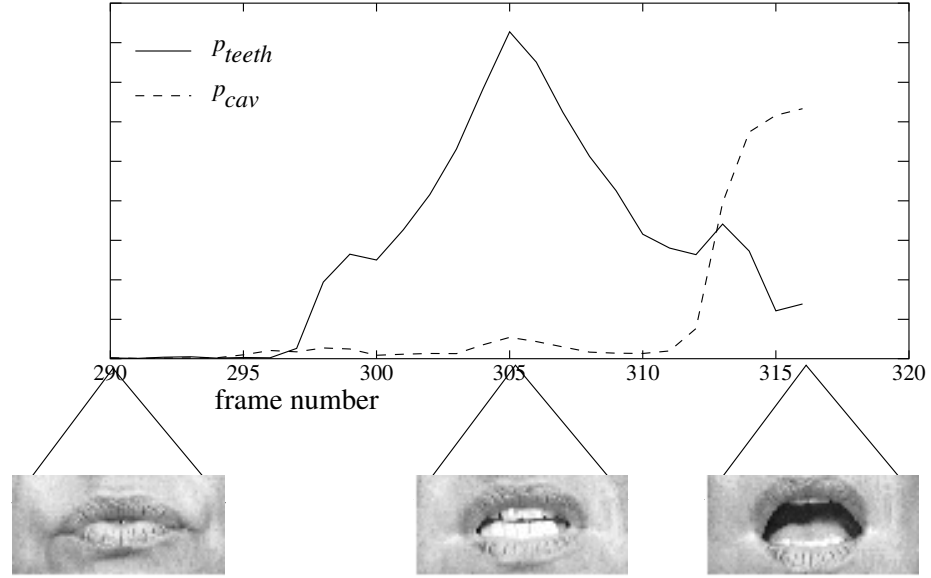


Figure 4.12: Changes in two of the intensity features in the sequence containing word *zes* (["zEs], six) in frames 290–310 just before the word *acht* (["Axt], eight) beginning in frame 315.

can be observed in the video sequence, the others not. It is essential in the case of lipreading to extract from the visual channel as much information as possible about the utterance being spoken. We propose therefore to use several additional features that complement the geometric information about the shape of the lips.

It would probably be possible to track the relative positions of the teeth and tongue with respect to the lips. The tracking accuracy would be limited by the fact that the visibility of lips and tongue is normally very poor. Such a task would also be too complex and therefore infeasible for a lipreading application. There are however some easily traceable features that can be measured in the image which relate to the positions and movements of the crucial parts of the mouth. The teeth for example are much brighter than the rest of the face and can therefore be located using a simple filtering of the image intensity:

$$f_{teeth}(v) = \begin{cases} 0 & \text{for } 0 \leq v \leq t_{teeth} \\ \eta(v - t_{teeth}) & \text{for } t_{teeth} < v \leq 1 \end{cases} \quad (4.6)$$

The above filter has a step-wise linear shape and in fact only one parameter: the threshold value t_{teeth} . The slope steepness factor $\eta = (1 - t_{teeth})^{-1}$ is calculated so that the resulting filter produces values in the $[0, 1]$ interval.

The visibility and the position of the tongue cannot be determined as easily as in case of the teeth, because the color of the tongue is pretty much indistinguishable from the color of the lips. We can however easily determine the amount of mouth cavity that is not obscured by the tongue. While teeth are distinctly bright, the whole area of the

mouth behind the tongue is usually darker than the rest of the face. Therefore we use the following filter to detect it:

$$f_{cav}(v) = \begin{cases} 0 & \text{for } t_{cav} \leq v \leq 1 \\ \gamma(t_{cav} - v) & \text{for } 0 \leq v < t_{cav} \end{cases} . \quad (4.7)$$

Both f_{teeth} and f_{cav} filters generate images with distributions I_{teeth} and I_{cav} respectively. In order to use the information presented in those images, we need to extract from them some quantitative values. We chose to use the total area of the highlighted region and the position of its center of gravity relative to the center of the mouth:

$$p_\phi = \sum_{x,y} I_\phi(x,y) \quad (4.8)$$

$$X_\phi = \frac{1}{p_\phi} \sum_{x,y} x I_\phi(x,y) - X_{center} \quad (4.9)$$

$$Y_\phi = \frac{1}{p_\phi} \sum_{x,y} y I_\phi(x,y) - Y_{center} \quad (4.10)$$

$$\phi \in \{teeth, cav\} \quad (4.11)$$

It is arguable that the above calculations are oversimplified, especially with respect to the visibility of the teeth. Obviously, the distribution obtained from the p_{teeth} filter is not unimodal if both the upper and lower teeth are visible. Fortunately this does not happen a lot in case of normal speech. Also our further experiments proved that the above features are descriptive enough (see Chapter 5). The example changes of the p_{teeth} and p_{cav} are depicted in Figure 4.12. In this figure it can be well seen how the visibility of the teeth dominates during the pronunciation of the word *zes* (["zEs]) and how the increase of the p_{cav} at the end of the sequence relates to the phoneme [A] being spoken later.

4.5 The future of lip-tracking models

The quest for the most accurate and robust processing model for going lip tracking is still on. Many researchers strive to achieve the perfect processing pipeline with hopes to overcome the inherent problems in lipreading. In Chapter 3 we showed that this noble quest may not be that relevant to lipreading recognition results as generally assumed. The entirely different methods can be transformed in each other with accuracy that is sufficient to preserve the obtained recognition rates. This is a very important result for the lipreading community. The developers of lipreading systems may now start to concentrate on the recognition models themselves and assume that they will be able to plug in any mouth tracking method that appears to be the most robust one at a later point in time.

The lip geometry estimation method developed by us and used in several experiments up to now appears to be very competitive with the others with some strong points on its favor. The previously mentioned equivalence of different geometry-based methods suggests that the results from our current and previous experiments also apply to other processing models.

Finally, the PAPCA based on the variability of the normal speech data provides a good base for a feature space that is person independent. It satisfies several conditions that need to be met before it can be used in a lipreading application. First of all, the projection can be established on the basis of a relatively small amount of calibration data. Secondly, it allows us to remove the inter-person differences from the extracted features. And last but certainly not least it provides us with a relatively good separation of the phoneme/viseme distributions.

There is still a long way ahead before the presented concepts can be applied in a real appliance. For example, even though the raw-image extraction model is too data intensive to be used, it still shows some superiority to the models only based on the geometry. Attempts at getting the best of both worlds have already been made [WR01b, LTB96b] and they probably indicate which direction future research will take.

“Well! I’ve often seen a cat without a grin,” thought Alice; “but a grin without a cat! It’s the most curious thing I ever saw in my life!”

L. Carroll, *Alice’s Adventures in Wonderland*

Chapter 5

Limited vocabulary lipreading

Automatic speech recognizers are very often classified by the number of words that they can be recognize. Each of the classes has specific applications in real life. Most commonly, those classes are defined as follows:

Limited vocabulary recognizers are able to recognize properly a small amount of words. The number of words depends on the application, but it does not exceed 50. If built with the use of HMMs, those recognizers usually implement one model per word. They can be applied for tasks such as speech controlling of a simple device (e.g. car audio, telephone dialing, etc.).

Medium vocabulary recognizers operate on up to 1000 words. Because of the number of recognition units, it is not feasible to model each word separately. This kind of recognizers is therefore implemented on the phone level, where words are modeled by strings of phone models. The search space for this kind of recognizers is usually limited to a predefined dictionary of words. Such recognizers are commonly used in dictation systems, information booths etc.

Large vocabulary speech recognition is the most sophisticated form of automatic speech processing. The intended set of recognized words is usually far above 1000, with a lot of focus on dealing with out-of-vocabulary utterances.

Another classification of speech recognizers is based on whether they recognize the continuous speech signal or only specific, time-limited chunks of it:

Single word recognizers deal only with a signal that starts and ends on the word boundary. This kind of recognizers is not very common, not even among the limited vocabulary recognizers, and it does not make sense in case of bigger dictionaries. They are to be found only in systems that have severely limited computing power (such as cellular phones or other embedded systems).

Single sentence based recognizers are much more powerful, yet their implementations are not significantly more complex. This kind of recognizers is usually built on the basis of task-specific grammars that define a rigid set of possible sentences.

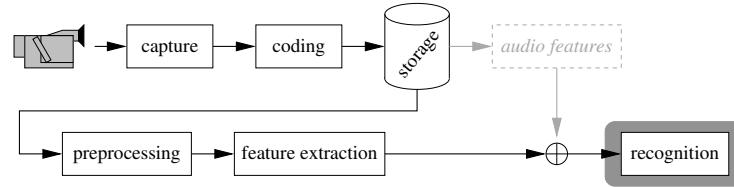


Figure 5.1: Lipreading pipeline.

Continuous speech recognizers deal with the signal that is not bound in time. The segmentation of the signal into sentences and words must happen in the recognizer itself. This task is next to impossible with the current technologies. Therefore continuous speech recognizers often use certain approximate techniques (such as word spotting) to extract the meaning from the signal even if it cannot recognize its exact contents.

In this chapter we describe several experiments with a limited vocabulary lipreading system based on a strictly defined grammar.

5.1 Connected digits recognition

Recognizing a connected string of digits is a typical example of limited vocabulary speech recognition [GPN02, ITF01, YSFC99]. The task for the automatic speech recognizer is to infer the sequence of the digits under the constraint that no other utterances are present in the speech signal. Despite this strong constraint on the type of signal, systems capable of such a task can be used in a real-life applications. There are several possible scenarios in which the situation naturally limits the user to speak only a sequence of digits. For example, dialing a phone number or providing a bank account number to the system are such situations.

The connected digits recognition can be based on a simple grammar with only ten terminal symbols and a small set of rules. Depending on the application, the grammar may allow for any number of digits being spoken (an unconstrained string of digits) or limit the utterances to a fixed number of digits (a constrained string of digits). Such grammars can easily be represented in the form of word nets such as those presented in Figure 5.2. Even though there are only ten meaningful terminal symbols in the grammar, it is often beneficial to introduce additional symbols for silent periods before and after the sequence (*sil*) as well as a short pause (*sp*) symbol that may or may not be used between digits. Such approach allows for flexible modeling of gaps between digits. If neither *sp* nor *sil* models were used, the burden of modeling the signal in between the meaningful utterances would fall on first and last emitting states of each digit's HMM. In our lipreading experiments we stuck to the convention of naming the models as “digit models” and “silence models”, even though no audio signal was involved in the processing (and so one could argue that they are all “silence models”). The silence models are therefore used to model the movements of the lips that are not speech related and appear between the uttered digits.

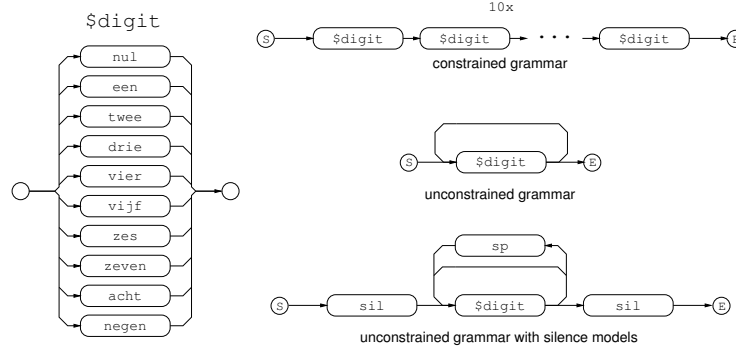


Figure 5.2: Graphical representation of word nets used in connected digits recognition.

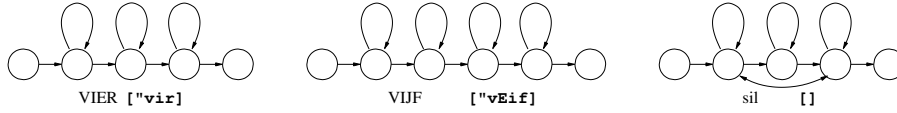


Figure 5.3: Examples of different length HMMs for different digits.

For all the following experiments we used the Hidden Markov Toolkit (HTK) to train 12 HMMs; one for each digit, one for the silence periods at both ends of the digit sequences and one for short pauses between digits. As the digits differ in pronunciation complexity, the models we used differed in number of states (see Figure 5.3). We chose to use one state per phoneme in each of the models; there was however no enforced correlation between the same phonemes in different digit models. The silence and short pause models had three and one emitting states respectively. As can be seen in Figure 5.3, the *sil* model has a structure that differs somewhat from that of the digit models. It contains a backward-going transition that allows a loopback from the last emitting state to the first one. In this way, the silence model can better fit in the varying time span of the parts of the signal that do not carry meaning.

In all of the models, each state was modeled by a single Gaussian distribution. The dimensionality of this distribution varied depending on the feature extraction techniques that were used for each experiment.

Further on in this chapter we will compare the results of the experiments, based on two measurements: the percentage of correct answers and the accuracy percentage. The correct answer and accuracy percentages given there adhere to the definitions provided in HTK manual:

$$\%correct = 100\% \times (N - D - S)/N \quad (5.1)$$

$$\%accuracy = 100\% \times (N - D - S - I)/N, \quad (5.2)$$

where N , D , S and I are respectively: the total number of words, the number of word deletions, the number of word substitutions, and the number of word insertions.

Both (5.1) and (5.2) are to some extent related to each other. If one of them increases during the training process, the other one is also likely to increase. The difference between those measurements indicates the amount of insertions that occurred in the recognition process.

5.2 Data acquisition

For the initial set of data, we recorded a native Dutch speaker (female), speaking multiple sets containing 10 random digits each. We recorded in total 30 such sets (300 digits). The subject was asked to pause between the sets and to speak the digits in a varying pace. There are some recordings with clearly visible pauses between words and some with no pauses at all. The recorded video sequences contain only the lower half of the face from nostrils to chin (see Figure 5.4). The movement of the subject's head was constrained only by the fact that she had to read the numbers from the screen in front of her. We recorded the video using a consumer-quality CCD PAL camera and stored it digitally in MPEG1 format. In order to reduce the color quality degradation we used a high bit rate coding.

From the obtained data the training and test sets were chosen randomly. The training set contains 21 digit sets (210 words) while test set contains 9 digit sets (90 words). Both sets were then processed in order to extract geometric and intensity related features from the video sequences following the model described in Chapter 4. As the recordings were not done in a single continuous shot, some illumination-induced color variations occurred between different video sequences. This had to be compensated with the recalibration of the used *lip-selective* filter. Fortunately, this happened only a couple of times in the whole recorded set. The intensity related filters (f_{teeth} and f_{cav}) proved to be much less sensitive to the illumination changes and remained the same for all of the data sets. The resulting data were later converted into a format that is compatible with HTK data files, so that it could be seamlessly used in the system.

At a later stage, the data set was extended to comprise more subjects. The newly recorded data set was aimed at developing a medium vocabulary bimodal speech recognizer, so the digit sequences were only a small part of it. A more in-depth description of the data set and the way it was recorded can be found in Chapter 7. For the limited-vocabulary experiments described here, we used recordings of 5 subjects with over 650 uttered digits.

5.3 Experiments with a single subject

Based on the data from a single female speaker we, trained four different lipreading systems. They differ in the type of grammar being used (constrained/free) and in the way the feature vectors were extracted.

In the first attempt, we trained the system on LGE feature vectors (see Chapter 3). We chose for 36 dimensional LGE vectors. As is common in speech recognition, we also used the difference values between consecutive vectors: so-called “deltas”. This way of preprocessing the data provides additional information about the dynamics of



Figure 5.4: Example images from the recorded material.

the signal with little to no computational overhead. The resulting observation vector was therefore computed as follows:

$$\mathbf{o}(t) = \begin{bmatrix} L_1(t) \\ \vdots \\ L_{36}(t) \\ L_1(t) - L_1(t-1) \\ \vdots \\ L_{36}(t) - L_{36}(t-1) \end{bmatrix}, \quad (5.3)$$

where $\mathbf{L}(t) = [L_1(t) \dots L_{36}(t)]$ is the LGE vector extracted from the video sequence at frame t .

As a result, each of the states in the HMM contains a 72-dimensional Gaussian probability density function (PDF) for calculating its observation probability. As each PDF is characterized by its mean vector and covariance matrix, the resulting amount of free parameters in a system containing 12 HMMs is more than 250 thousand. Even though this amount might seem a bit large, the system trained without problems and on the free grammar task scored:

$$\%correct = 80.0, \%accuracy = 36.7.$$

There is a large discrepancy between the percentage of correctly recognized digits and the achieved accuracy. That means of course that a large amount of insertions occurred in different places. We can guess that the above figures can be improved by using a grammar with a constrained number of digits; this would greatly reduce the amount of spurious insertions. It indeed prove to be true, as the results obtained for a constrained grammar are as follows:

$$\%correct = 75.6, \%accuracy = 64.4.$$

The improvement in accuracy of the recognition is so large that we decided to investigate the nature of the word insertions somewhat closer. Figure 5.5 depicts the beginning of one of the digit sequences together with the results of recognition based on constrained and free grammars. It can clearly be seen that a vast amount of insertions occurs in the part of the signal that comes before the beginning of the digit sequence.

target						0	5	5	6	8	8
unconstrained	3	8	8	8	7	0	5	5	6	8	8
constrained					7	0	5	5	6	8	8

Figure 5.5: The first few digits in the recognized sequence of combined sets of features. The *target* row represents the spoken sequence and the two lower rows show the output from the Viterbi algorithm.

This part of the signal should be modeled by the `sil` model and yet it appears as if any random sequence of digits matches this part of the signal better than that model. A similar pattern of insertions being placed before or after the real signal can be found in the whole data set. This led us to the conclusion that the silence model is heavily undertrained.

The problem with the silence models in lipreading is that in contrast to an auditory speech recognition, the silent periods do not imply reduced dynamics of the incoming signal. Even when not speaking, people usually move their lips, inhale and exhale, smile etc. The mouth region is in constant movement. In order to be able to train the silence models for lipreading properly, we would need the same amount of training material for them as for any other model. Moreover, there is a large number of possible facial expressions and gestures that are not speech related and yet they influence the lip movements in distinct ways. If we want to model them properly, we may need more than one silence model to cover this broad range of movements.

In Chapter 4 we proposed extending the LGE features, which are purely geometric, with a set of intensity related features. Our motivation was that not only lips are involved in speech production but also the teeth and tongue and that additional information about these might be beneficial for a lipreading system. In order to verify this hypothesis, we trained another system, this time with 84-dimensional vectors containing 36 LGE features, 6 intensity features and their respective deltas. The resulting system scored:

$$\%correct = 91.1, \%accuracy = 60.0$$

on a free grammar task and:

$$\%correct = 86.7, \%accuracy = 81.1$$

on a constrained grammar task. The results are therefore significantly better than when only LGE features are used. Most important here might be the fact that these improvements in recognition rates could be achieved with an amount of free parameters in the system that was only 20% larger.

The above result shows that even a small number of relevant features can make a significant difference in the system's performance. We can therefore ask the question whether the opposite is also true. That is, could it be possible to retain the same performance level by using only a small number of carefully chosen features? As we already described in Chapter 4, it is possible to choose such relevant features by performing principal component analysis (PCA) of the gathered data.

In our case, the first 5 PCs of LGE data cover about 98% of the data variation. This is therefore the number of features that we chose to use to train the recognizer.

This 5-dimensional geometry vector is extended with the intensity features and deltas of both vectors. As can easily be calculated, this system has only about 25 thousand free parameters. This is about ten times less parameters than those of the system that we started with. The performance figures of this system with both types of grammar are within 2% of the performance of the system based on raw LGE features augmented with intensity information. The exact figures can be found together with all other presented figures in Table 5.1 at the end of this chapter.

5.4 Experiments with multiple subjects

When training the lipreader for multiple persons, we started right away with the system based on 5 PCs of the LGE vector. We thus chose not to repeat the step with using raw LGE data. This decision was of course based on the good results obtained by such feature preprocessing. There was also another important reason for choosing PCA; the data reduction. The material recorded with 5 persons was more than twice as long as the material from our first recordings. The time that the training process would take if the raw version of LGE vectors were used could have posed a serious problem. The training time grows with the square of the complexity of the models. Using PCA to reduce the length of the observation vectors (and therefore the amount of parameters within each state) reduces the computational complexity around 100 times in our case.

Unfortunately, the results obtained in these experiments were far from encouraging. As can be seen in Table 5.1, a recognizer based on free grammar performs pretty much on a chance level (10%). The situation is a bit better when the grammar constrains the number of digits, but the 27% correct answers and 17% accuracy are results that are far from satisfactory.

In Chapter 4 we showed that performing PCA on a full dataset containing data recorded from different people might not be such a good idea. The feature vectors extracted from the recordings of different subjects have different dynamic ranges, and the resulting projection might end up reflecting the inter-personal differences rather than features related to speech production. As a solution we may use a PAPCA projection and let the recognizer operate in the Person-Independent Feature Space (see Section 4.3). Applying this technique to our data set and utilizing both LGE and intensity components we got a system that scored:

$$\%correct = 35.8, \%accuracy = 18.3$$

on a free grammar task and:

$$\%correct = 32.5, \%accuracy = 25.8$$

on a constrained grammar task. Those figures are significantly better than those obtained in our first attempt.

Table 5.1: Summary of the recognition results for different setups presented in this chapter.

dataset	constrained grammar		free grammar	
	LGE	LGE+I	LGE	LGE+I
1 person	75.6/64.4	86.7/81.1	80.0/36.7	91.1/60.0
N= 90	D= 10	D= 5	D= 7	D= 4
	S= 12	S= 7	S= 11	S= 4
	I= 10	I= 5	I= 39	I= 28
1 person (PCA)	75.6/65.6	86.7/81.1	71.1/37.8	91.1/58.9
N= 90	D= 11	D= 5	D= 7	D= 4
	S= 11	S= 7	S= 10	S= 4
	I= 9	I= 5	I= 39	I= 29
5 persons (PCA)	20.8/10.0	26.7/16.7	23.3/ 0.8	27.5/ 6.7
N= 120	D= 35	D= 32	D= 33	D= 31
	S= 60	S= 56	S= 59	S= 56
	I= 13	I= 12	I= 27	I= 25
5 persons (PIFS)	25.8/16.7	32.5/25.8	25.0/ 4.2	35.8/18.3
N= 120	D= 32	D= 30	D= 33	D= 28
	S= 57	S= 51	S= 57	S= 49
	I= 11	I= 8	I= 25	I= 21

"Would you tell me, please, which way I ought to go from here?"
"That depends a good deal on where you want to get to," said the Cat.
"I don't much care where—" said Alice.
"Then it doesn't matter which way you go," said the Cat.
"—so long as I get SOMEWHERE," Alice added as an explanation.
"Oh, you're sure to do that," said the Cat, "if you only walk long enough."

Chapter 6

L. Carroll, *Alice's Adventures in Wonderland*

Towards continuous lipreading

This chapter describes a set of experiments conducted at Delft University of Technology to examine the feasibility of applying the lipreading approach proposed in Chapter 4 on a continuous speech input. Continuous speech recognition differs from limited vocabulary recognition in many ways. Firstly, the input stream is not constrained to a small set of occurrences, but rather it comprises the whole spectrum of possible inputs. At a higher level of representation, we cannot put too many restrictions on the grammar of the language being used. It is not feasible to use formal grammars such as the ones presented in Chapter 5. The complexity of natural language is so high that it is better to approach it in a probabilistic rather than in a deterministic way. Those are the issues that make continuous lipreading a much more complex task. The following sections do not deal with language complexities or language understanding issues. In order to investigate the feasibility of using LGE for continuous recognition tasks, we focus our attention on low level speech processing.

6.1 Data acquisition

We did experiments with multiple speakers (only one of them was not a native Dutch speaker). The subjects were seated in front of the camera and asked to speak a set of five sentences. The tripod holding the camera was then adjusted so that only the lower part of subject's face was recorded. The sentences were chosen from the POLY-

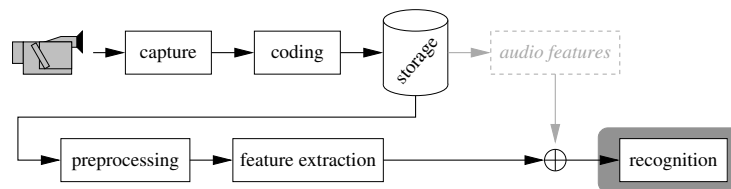


Figure 6.1: Lipreading pipeline.

PHONE [DBI⁺94] corpus. The sentences in this corpus are sentences used in everyday life and grouped in sets of five so that all of the phonemes used in the Dutch language appear at least once in the set.

6.2 Speech onset/offset detection

The processing interval of an automatic speech processing system is defined by the presence of speech. It is reasonable that such a system should be active and process the incoming audio data only when there is somebody actually speaking to it. Detecting this simple occurrence is not always that easy. For example, using the volume level to determine the beginning and the end of the utterance may be very misleading in a noisy environment.

In order to detect this activity time interval, another modality can be used. For example, street information kiosks are often equipped with a weight sensor in front of the information panel. In this way the processing is activated only when somebody is standing in front of the display.

Another modality to be used is of course the visual stimuli such as lip movements or changes in the visibility of the teeth and tongue. In this way the activation of the speech recognition module would be triggered only when the registered movement of the lips appears to relate to the speech production process. We will further present the experiments that were conducted with this problem in mind.

After acquisition of the video sentences, we manually labeled the speech onset and offset points on the basis of only the auditory signal. These boundary points were used to label all the frames in the video sequences as either 1 for silence or 0 for speech. These labels were later used as target output values for the silence recognition system.

6.2.1 Explorative data analysis

In order to get insight in the high-dimensional data that is extracted from the video sequence using LGE, we had to use some dimension reduction techniques. We decided to use the Sammon mapping [Rip96]. It yields a more accurate representation of the small distances in the data set than other methods (see [Rip96]). Sammon mapping is a technique in which each (highly dimensional) vector from the data set is explicitly mapped to some vector of a lower dimensionality space. The mapping is done in a way that minimizes the stress measure, which is defined as follows:

$$E = \sum_{i \neq j} \frac{(d_{ij} - d'_{ij})^2}{d_{ij}} \quad (6.1)$$

where d_{ij} is the distance between i -th and j -th vector in the data set and d'_{ij} is the distance between mappings of the i -th and j -th point. By minimizing the formula (6.1), the obtained mapping preserves the distances and therefore also the clustering properties of the initial data set as accurately as possible.

There is however one issue that needed to be addressed first, before applying this technique to the data coming from a video sequence. As described in Chapter 4, the

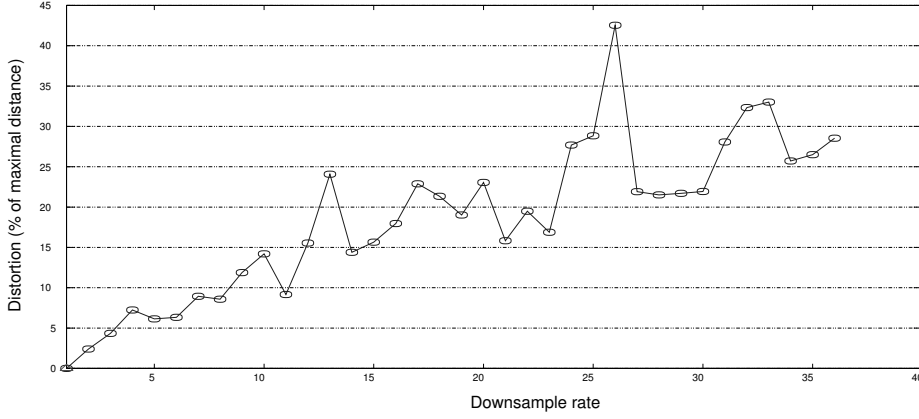


Figure 6.2: Distance matrix distortions.

last step of the geometry extraction process provides us with estimates of the mean and variance of a distribution for a given angle ($\widehat{M}(\alpha)$, $\widehat{\sigma}^2(\alpha)$). Those functions must be sampled to represent the data vector that will be used in further processing. Therefore we needed to choose an appropriate number of sampling points in order to achieve a robust, yet sufficiently accurate representation of the data. If we choose to sample the data densely, we gain the accuracy of the data representation, but the resulting feature vectors might be too long to be efficiently processed.

In order to find the appropriate sampling rate, we first extracted the feature vectors with 180 uniformly distributed sampling points for each of both functions. This dataset with its distance matrix was used as a reference set. The reference set was then downsampled with different rates and the obtained distance matrices were compared with the reference matrix using formula ((6.1)) (see Figure 6.2). As can be seen, the distortions in the dataset grow smoothly up to a factor 10 (resulting in 36 dimensional feature vectors). After this point, the distortions become rather chaotic, which means that the distances in the dataset are not preserved properly. These results motivate our choice of 36 dimensional vectors (18 uniformly distributed sample points for mean and variance) as the data vectors for further processing.

After that, we applied Sammon mapping on the data set (see Figure 6.3). Even with straightforward mapping of vectors coming from the video sequence the frames that were labeled as silent can be clustered (see Figure 6.3a). Most of the *silent* frames contain the image of a closed mouth. There are however also such frames for the phonemes like *p*, *b* and *m* at some point of their pronunciation. Therefore there is no possible way of differentiating between a *[pbm]* viseme and a silence on the basis of a single frame. Figure 6.3a therefore shows *[pbm]* frames scattered around in the silence cluster.

In order to introduce the time dependencies in the feature vector, we can combine the data from some number of different frames. Figure 6.3b shows the result of a concatenation of 3 feature vectors from consecutive frames from the video sequence (a very nicely clustered *[pbm]* viseme; just on the edge of silence cluster).

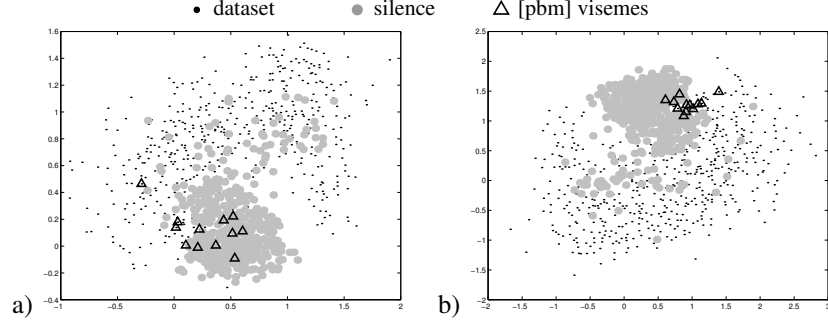


Figure 6.3: Sammon mapped dataset: (a) vectors extracted from single frames, (b) concatenated data vectors from three consecutive frames.

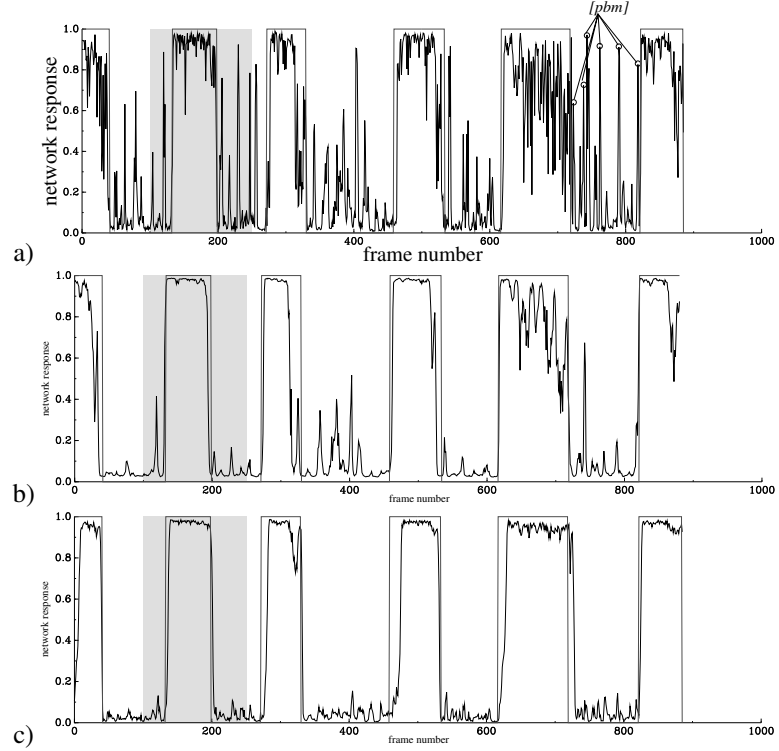


Figure 6.4: The output graphs from different neural network architectures. (a) FF-ANN, (b) TDNN, (d) JNN. Straight lines depict target outputs; the grey area represents the part of the sequence that was used as a test set.

6.2.2 Classification with artificial neural networks

Preliminary analysis of the data showed that the silence segments can be separated (to a limited extent) from the rest of the sequence. This suggests that it would be possible to use clustering techniques in order to develop a classifier. It proved however that relying only on distances between feature vectors is not sufficient for classifying the silence segments. The underlying process' evolution over time is much more important than the current observed state. The information about time changes in the observed sequence could be incorporated in the feature vectors themselves. This could be done by concatenating several consecutive vectors, using their deltas or acceleration values. However, all those approaches would increase the dimensionality and complexity of the problem. Therefore we decided to conduct several experiments with Neural Network-based classifiers which can cope internally with the time evolution of the signal. All of the experiments were done using the Stuttgart Neural Network Simulator (SNNS) version 4.1 running on a Sun Ultra1 workstation.

For training and validating the neural networks we used different video sequences with varying speech rates. All the extracted feature vectors were labeled 0 or 1 (*non-silent*, *silent*) according to audibly found boundaries of spoken sentences. From the full dataset we chose about 10% of the vectors to form the test set and used the rest for training. The training set was chosen such that it consisted of several intervals containing both a beginning and ending of silence together with some non-silent frames. We measured the fitness of all the networks with mean square error of their response for a single epoch.

Figure 6.4 shows responses of the networks for a single video sequence containing 5 sentences. The video sequence was chosen such that it contained one of the sentences that were put in test set.

In the experiments we used three different NN models: the Feed Forward Neural Network (FF-ANN), the Time Delayed Neural Network (TDNN), and the Jordan Recurrent Neural Network (JNN). An extension to the JNN, the Elman Hierarchical Neural Network, is used in the vowel/consonant discrimination experiments described further on in this chapter. A fully connected feed-forward back-propagation trained neural network was fed with a single frame feature vector, therefore it had 36 input units and the single output unit. As can be seen in Figure 6.4a, the performance of the network is rather poor. Most notably, there are a lot of frames within sentences that are recognized as silence. There is, however, evidence that the proposed data processing technique is suitable for lipreading: the network tends to highlight the *[pbm]* viseme in the sentences even though it cannot classify the silent segments properly.

A time-delayed neural network allows exploitation of time dependencies in the data set by using a so-called *perception window*. A perception window spans several consecutive input vectors and slides along the input sequence. The length of the window is called the *delay*. Moreover, the hidden layers can also be time delayed; this allows the activation patterns from earlier time steps to pass through the network. This network gives reasonable results with very little variations of the output for silent periods and reasonable results for spoken ones (see Figure 6.4b). Those improvements come directly from the time-dependency-finding capabilities of TDNNs.

In a Jordan recurrent neural network the time dependency is represented by the

recurrent nature of the network itself. The input to the JNN is only based on a single frame; the time related information is preserved within the network through *context neurons*. Context neurons are neurons that take as input the output of the whole network and their own output from a previous time step. The single hidden layer is fed with activation of both input and context neurons. In the presented case there is only one output neuron, and therefore also only one context neuron. This network architecture gave the best results by far (Figure 6.4c). Not only did it train much faster than the rest of the networks, but it also gave the smoothest and most accurate output. There are only two possible disadvantages of using JNN for this task. Firstly, according to our experiences, the JNN can very easily be overtrained, in which case the training procedure becomes unstable. Secondly, the network is always a little bit late when changing from speech to silence (as seen in Figure 6.4c), which presumably comes from the fact that the output of the network heavily depends on the loop between the context neuron and the output; the change in the activation of this neuron cannot be propagated to the output in a single data pass.

6.3 Vowel/consonant discrimination

As a second example of continuous speech signal processing we chose to extend the previously described system with the capability to discriminate between vowels and consonants [cWR00]. This kind of system does not have a significant application value, but it can support other speech processing applications. For the following experiments we used the same set of recordings as in previous case.

6.3.1 Explorative data analysis

In the previous set of experiments we investigated whether the dynamics of the lip movements can be used to distinguish between silent and non-silent segments. We have shown that the LGE features form a good data representation for this task. However, the previous success does not guarantee that the LGE is appropriate for any other type of continuous speech processing. We do not know whether it can be used to accurately describe the multitude of lip shapes that is used in speech production.

To get some insight in this issue, we used a Kohonen's self organizing map (SOM). SOM is implemented in the form of an n -dimensional grid of neurons and can be trained to perform a multidimensional scaling [Koh95]. After training, each of the neurons in the grid represents a single vector in the input space. Neurons that are close to each other in the grid represent vectors that are close to each other in the input space. The grid as a whole covers as much of the training data as possible.

The results of training 2-dimensional SOM on our data set are presented in Figure 6.5. It can be seen that the LGE vectors are indeed a good representation of the lip geometry. In trained SOM the neurons that are close to each other also represent lip shapes that are visually similar. Moreover, we can also distinguish clusters of similar shapes and the smooth transitions between them. The architecture of SOM provides an objective way of defining such clusters together with the borders between them. In order to do that, we need to investigate the Euclidean distances between weights of

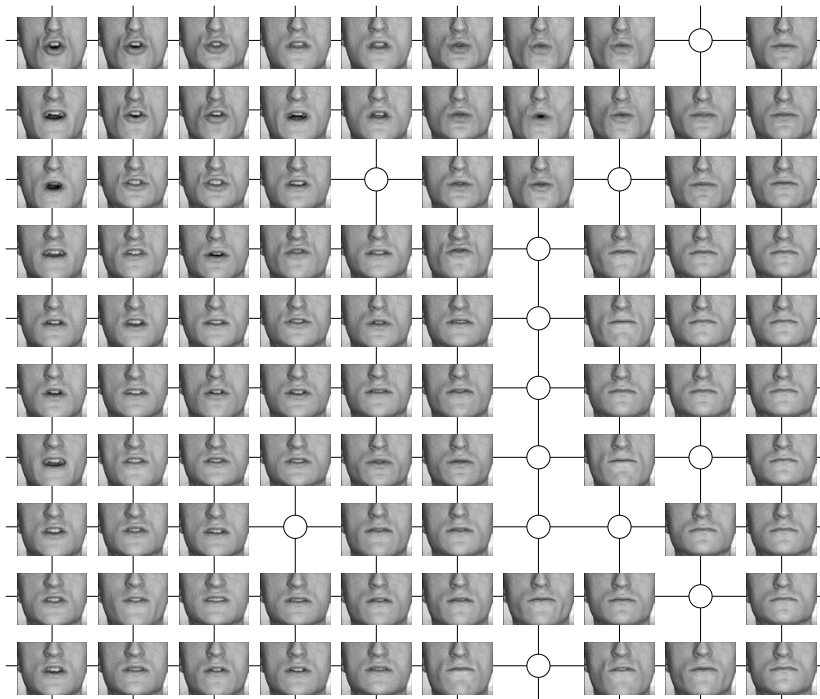


Figure 6.5: A self-organizing Kohonen map after training on the recorded data set. Each cell shows an image that best matches the single respective neuron's weights.

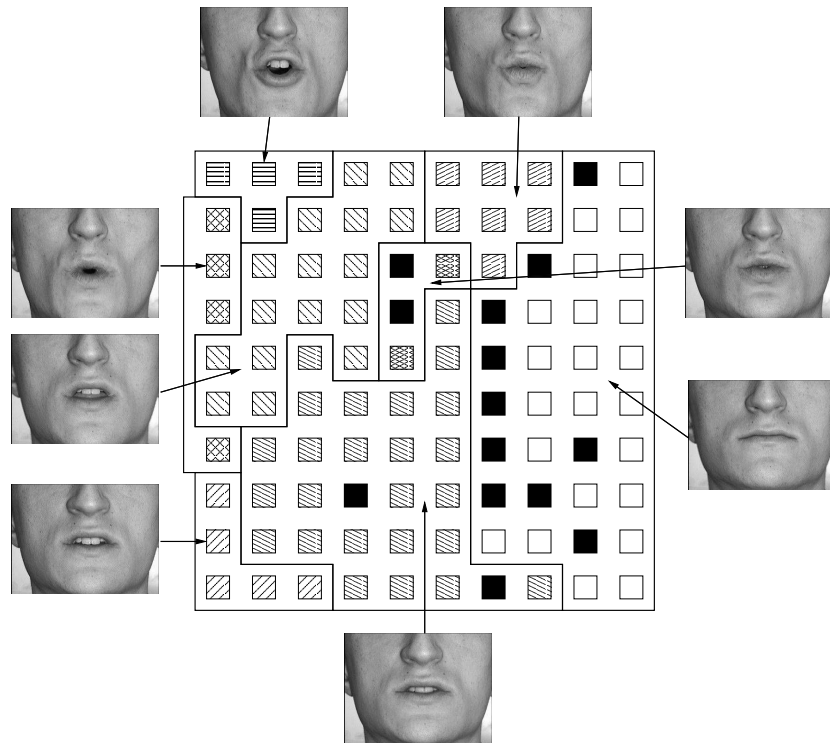


Figure 6.6: The SOM from Figure 6.5, with the boundaries between clusters and the mean representatives of each cluster.

neighboring neurons. The neurons in the same cluster have weights that are close to each other; the weights of neurons from different clusters differ greatly. Therefore, we can obtain either fine-grained or coarse clusters just by choosing the appropriate threshold distance value. An example of such clustering is shown in Figure 6.6.

6.3.2 Data labeling

The labeling of the data was simplified by the fact that we already had the transcriptions of the sentences in our data set. All we had to do was to assign the consecutive labels from the transcription to the appropriate video frames. We used both visual and auditory information in order to place the labels appropriately, and followed a set of rules related to the speech production process. All the frames labeled as vowels captured the mouth as being maximally stretched during phoneme pronunciation. The frames labeled as consonants that have a clear visual point of pronunciation, such as *p* or *l*, showed the appropriate visual occurrence. Frames labeled as other consonants were located in the middle of the audibly recognizable pronunciation. Example rules used in data labeling had the following form:

- [*p*] - the last video frame before opening the mouth
- [*l*] - the video frame in which the bottom side of the tongue is visible between the teeth
- [*A*] - the video frame in which the mouth is maximally stretched when pronouncing the phoneme
- [*t*] - the video frame in the middle of an audibly recognizable phoneme *t*

In cases where the occurrence of the phoneme could not be objectively located from either an auditory or visual channel (often for consonants such as *h* or *k*), the middle frame between the neighboring labels was labeled as showing this phoneme. This way of labeling the frames works pretty well because on average a phoneme lasts about 3 frames. Therefore we can treat those frames as representatives of three stages of phoneme pronunciation: *fade-in*, *pure* and *fade-out*. In cases when the same rule could have been applied to more than one frame, only the one in the middle was labeled. This is often true for vowels whose *pure* form tends to last longer than one frame.

The rules used for labeling were designed such that the process of manually labeling of the frames was as simple as possible. There was, however, some concern that they would produce a pattern set that was not suitable for training a neural network. Some occurrences such as opening of the mouth occupy several frames and if labeled too early, these would in fact force the network to predict the given viseme instead of detecting it. To prevent this problem we forced the resulting labels to appear with a predefined delay. This was done by shifting the patterns by a number of frames. In this way phonemes labeled too early in the sequence were shifted to the appropriate places. This shifted the phonemes that were in appropriate places before the shifting procedure to the wrong frames, but this was not a big problem as the input of the network contains some information on the previous frames.

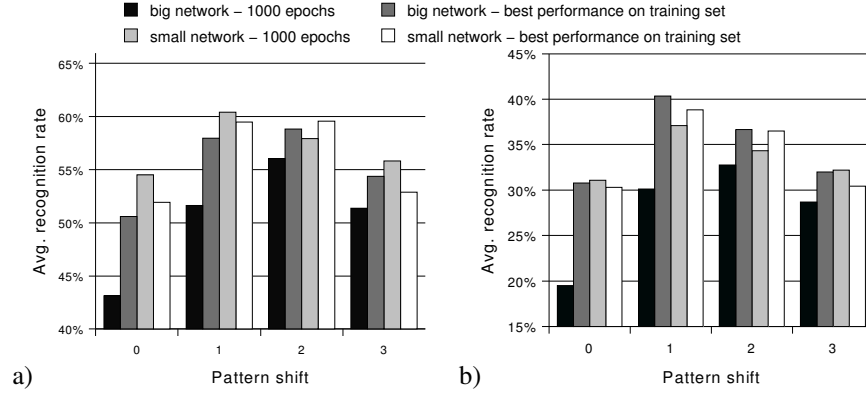


Figure 6.7: Changing recognition rate when shifting output patterns. a) recognition rate for all outputs, b) rate excluding silence detection output.

From earlier research we expected that recursive networks would perform better than other neural architectures, so to investigate the influence of pattern shifting we used four Elman neural networks with varying sizes and trained them with patterns shifted 0 to 3 frames. The resulting recognition rates can be seen in Figure 6.7. The results clearly indicate that shifting the patterns by one frame gives an optimal network performance.

6.3.3 Training Procedure

From each video frame a data vector was extracted using the LGE technique described in Chapter 4. The resulting feature vector $L(t) = [L_1(t) \dots L_N(t)]^T$ was obtained for each frame t . In order to achieve mouth size independence the obtained vector was scaled so that all of the values in it fit the $\langle 0, 1 \rangle$ interval:

$$L_i^*(t) = \frac{L_i(t)}{\max_{j=1 \dots N} L_j(t)} \quad i = 1 \dots N. \quad (6.2)$$

The stream of those vectors formed the input for the neural network. We used 18-point sampling of the $\widehat{M}(\alpha)$ and $\widehat{\sigma}^2(\alpha)$ functions, which proved to be adequate in our earlier experiments [WR00]. Therefore the input pattern contained in total 36 values for each video frame ($N = 36$ in the above formula.)

The appropriate output pattern was formed from three values, each representing one of the classes:

0 - silence – the frames that contain no utterances should be classified in this way,

v - vowel – the frames with labels I e: E E: A @ i O Y y u 2: o: 9 9:
O: a:

c - consonant – the frames with labels f v w s z S Z p b m g k x n N r
j t d l

Table 6.1: Example recognition rates for different NN architectures.

Architecture	Recognition rate	V/C only
FF-ANN	48.9%	44.4%
TDNN	49.8%	47.7%
Elman NN	63.4%	70.6%

Table 6.2: Recognition results for 36-20-15-10-3 Elman network.

Target	Network response			
	0	c	v	-
0	70.5%	14.3%	3.8%	11.4%
c	2.6%	72.2%	19.5%	5.2%
v	0.0%	26.5%	67.3%	6.1%
-	1.7%	45.2%	12.2%	40.9%

As typically done, we constructed the outputs in such a way that the value 1 is put in the video frame labeled as the given class, with neighboring frames taking lower values of 0.5 and 0.25 depending on their distance to the labeled one. Example output data can be seen in Figure 6.8 (note that for clarity, one output: *silence*, is not depicted there). The output patterns prepared in this way were then shifted one frame forwards in order to avoid the problem of too early labeling discussed in the previous section.

From the whole recording set, we chose 10% of the data to form our test set and used the remaining 90% for training the networks. The test sentences were chosen such that we had one sentence from each of the subjects, with all *normal*, *slow* and *whispered* sentences present in this set.

We trained several different neural network architectures. The architectures used here varied from a feed-forward back-propagation trained neural network, through a Time Delayed neural network to partially recursive neural networks such as the Jordan Neural Network and the Elman Neural Network.

6.3.4 Recognition Results

The results of the comparison between different NN architectures were consistent with our previous experiments in this field [WR00]. The typical feed-forward neural networks are not really suitable for this task and so they score poorly for the overall recognition rate. The TDNN performs a bit better, but its recognition rate is still significantly lower than that of the Elman recurrent NN (see Table 6.1).

The Elman Hierarchical Neural Network constructed from three hidden layers with respectively 20, 40 and 20 neurons in each of them proved to be the most efficient in vowel/consonant discrimination. The overall results of recognition are summarized in Table 6.2. Together with the three classes that can be recognized by the network (0, v, c) this table also contains a fourth column labeled (-), which corresponds to the output patterns that should not be classified as any of the three classes. Such patterns occur in our data set in, for example, slow speech between some phonemes that last a long time.

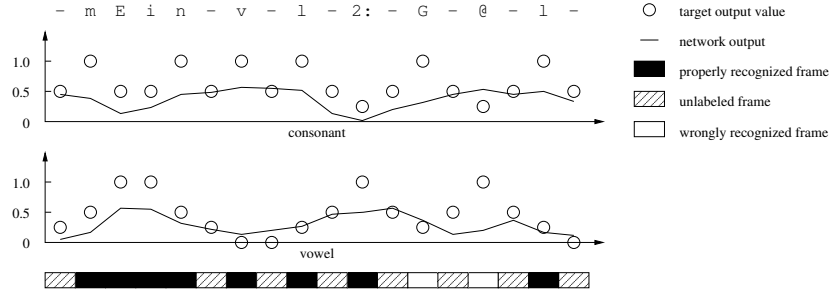


Figure 6.8: Recognition results for utterance “mijn vleugel” (“my piano”) in one of the sentences from test set.

According to our labeling method, if a phoneme lasts longer than 5 frames (~ 200 ms), it will have a single output peak associated with it in the middle of the utterance and some unspecified frames around it. In the experiments we assumed that the network response was unspecified if none of the network’s outputs had a higher activation than a threshold value of 0.1.

In case of *silence* detection, the achieved recognition rate of about 70% conceals how well speech (and silence) is being detected. Almost all of the incorrectly recognized frames appear at the speech onset/offset points, which means that the actual silence detection is either too early or too late with respect to the labeled data. The amount of 30% incorrectly recognized frames is equivalent to an average mismatch of around 0.5s (12 video frames) in detecting the onset or offset of speech.

Figure 6.8 presents an example flow of the network outputs for the words “mijn vleugel” from one of the test sentences. The first two rows of numbers in this figure represent the target outputs with boxes around the dominating values. The second two rows show the output of the network for the same frames. The lowest row of symbols is the transcription of the words in SAMPA notation. As can be seen, the network output is not ideal and for example the phonemes [G] and [ə] are not recognized properly. The rest of the phonemes seem to be recognized fairly well, however.

“I shall sew it on for you, my little man,” she said, though he was tall as herself, and she got out her housewife, and sewed the shadow on to Peter’s foot.

J. M. Barrie, **Peter Pan**

Chapter 7

Continuous lipreading

In this chapter we present the development of a continuous speech processing system with lipreading capabilities. The system is not based purely on lipreading but rather on interaction between auditory and visual modalities of the speech signal. At this moment it does not seem possible within the foreseeable future to develop a lipreading system that is capable of processing a continuous speech signal independently. The developed techniques, together with the processing capabilities of the current hardware and the available data sets, are not sufficient for developing a synthetic lipreader of reasonable quality. Therefore the only possibility for testing the developed methodology in a continuous context is to apply it together with auditory-based recognizers.

Bimodal speech recognizers are a very interesting research topic. It is not only because of their improved robustness in comparison to purely auditory recognizers. The more we know about the technical challenges that come with multiple modalities, the more precise can be our questions about our own human capabilities for integrating modalities. Our interaction with the world around us is highly multimodal. We use all five senses to gather information from our surroundings, while our brain integrates them into a single coherent experience. Bimodal speech recognition is a very valuable playground at which theories and models of modality interactions can be tested and quantitatively verified. In the ongoing quest for artificial intelligence, multimodal interaction and sensory integration are of the uttermost importance. Our guess is that bimodal speech recognition will be the first of the goals that will be achieved on the way to the intelligent, sensing machine.

In the following sections we present how we developed a bimodal speech recognizer for the Dutch language. We start with describing the process of acquiring a bimodal speech database. Further we explain the tools and techniques that are necessary for combining modalities. Finally we present the results of the experiments with the developed recognizer.

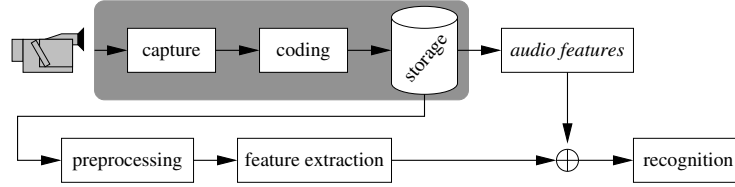


Figure 7.1: Lipreading pipeline.

Table 7.1: Comparison of data sets available for speech processing applications.

	POLYPHONE	M2VTS	XM2VTSDB	TULIPS1	DUTAVSC
respondents	5,050	37	295	12	8
words	>1,000,000	2,000	64,000	96	14,000
continuous?	yes	no	no	no	yes
AUDIO					
sampling	11kHz	48kHz	32kHz	11kHz	48kHz
bits/sample	8	16	16	8	16
VIDEO					
resolution		288 × 360	720 × 576	100 × 75	384 × 288
frame rate		25fps	25fps	30fps	25fps
lips only?		no	no	yes	yes

7.1 Multimodal speech data acquisition

This section describes the Dutch audio-visual speech corpus (DUTAVSC) developed for this thesis research. The corpus was prepared with multi-modal speech recognition in mind and has been used in our research on lipreading and speech recognition.

The availability of training and testing data is crucial when developing speech processing systems. There are already many commercially available speech corpora that contain audio data only. Certainly the TIMIT [GLF⁺93] data base is one of the most popular in developing English-language-based ASRs. For the Dutch language, the POLYPHONE [DBI⁺94] data set is comparably comprehensive. There is however a lack of such data sets containing both audio and visual information. One of the few available resources is the M2VTS database together with its successor, XM2VTSDB [MMK⁺99]. However, these comprehensive audio-visual data sets were designed and recorded with person identification applications in mind, and therefore they are not well suited for the development of speech processing systems (see Table 7.1 for comparison). It was therefore crucial for our research to gather our own data set that would be appropriate for speech-related research.

7.1.1 Recording requirements

From our earlier experiments with lipreading (reported in Chapters 5 and 6) we learned which requirements for the recorded data should be specified beforehand. These are requirements that need to be satisfied in order to develop a multimodal ASR and/or

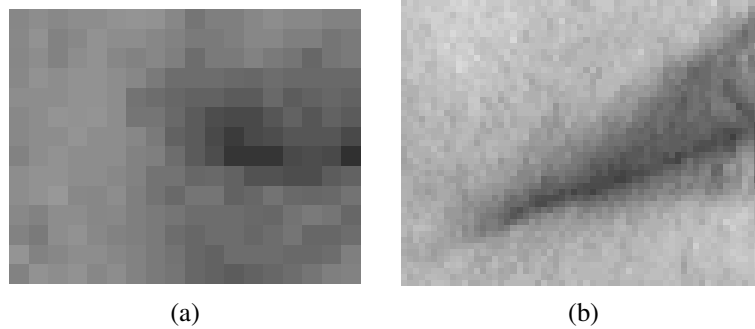


Figure 7.2: Example of a detail in mouth image in M2VTS (a) and in DUTAVSC (b).

lipreading system. Those requirements in turn influenced both the content of the recordings and the physical setup during the sessions.

Audio requirements

Only audio data of a reasonably high quality are useful for speech recognition. We agreed that the audio recordings sampled at 44kHz with 16-bit resolution would be sufficient for developing a speech recognizer. The present speech recognition methods are sophisticated enough to allow for limited noise in the recordings and the use of middle-class recording equipment. As the audio signal is substantially less storage expensive than the video signal, we decided to keep all of the audio data in uncompressed form, so that no signal degradation happens would occur during storage.

Video requirements

It is not feasible to store the video in uncompressed form (unlike the audio data). We therefore decided to use the MPEG1 compression with a high bit rate in order to make an optimal choice between the image quality and file size. In order to speed up the development of the system we also decided that the camera will be focused only on the lower part of the face. This leads to simplification of the lip-tracking process and allows the use of a lower video resolution. At a resolution comparable to the one used in M2VTS, we obtain much finer detail in the mouth region images (see Figure 7.2). Such a restricted field of view is of course not easily achievable in most real-life situations. It is, however, justified in the development stage.

An additional concern when recording the video was the color reproduction quality of the used equipment. In earlier experiments we noticed that commercially available camcorders are very sensitive to changes in the illumination conditions. It is inherent to all video coding standards (both analog and digital) to put more emphasis on image intensity than on its chromacity information. Therefore we had to ensure that the recorded scene was well lit, preferably with a natural light source.

7.1.2 Prompts

The set of textual prompts that were used in DUTAVSC was derived from the prompts recorded for the POLYPHONE [DBI⁺94] data set. The POLYPHONE data set consists of an extended set of telephone-quality recordings of Dutch utterances. Although the recordings contain only an audio signal and therefore are not suitable for developing a bimodal speech recognizer, we used some of the prompts from this collection. In POLYPHONE, among utterances as answers to specific questions or separate digit sequences, there are also the phonetically rich sentences. Those sentences were gathered from Dutch newspapers and grouped in sets of five sentences in such a way that in each set each of the Dutch phonemes occurs at least once. We used those sets together with separate words, spelling examples and application-specific utterances when preparing prompts for our recordings.

Our prompts collection is divided in 24 sections, each of which has the same structure described later. Recording all of the 24 sections with each of the participants would not have been really feasible as it would have required sessions of almost 2 hours. All of the subjects agreed that one hour of recordings is already a burden. We constrained ourselves to the one-hour sessions, which resulted in a recording of between 10 and 14 sections of the prompt set. Because of organizational issues such as introducing the subject, resetting the equipment etc. during a full recording session hour we gathered between 25 and 45 minutes of actual material.

Each section of the prompt set contains a fixed number of different utterances. The example section can be seen in Figure 7.3. There are always 10 separate words, 10 phonetically rich sentences (2 sets from POLYPHONE), 3 ten-digit sequences, 4 spelled words and 5 application-oriented utterances.

Words

The 10 words that open each section are meant for single-phoneme experiments. As it is hard for a non-trained subject to pronounce properly an isolated phoneme, we decided to choose the smallest possible words that contain each of the phonemes. The letters corresponding to the selected phoneme are highlighted in each word on the subject's display. The subjects were asked to pronounce this phoneme as well and as clear as possible (possibly by prolonging or stressing it.) The words that have a vowel highlighted are in the form of CVC (consonant–vowel–consonant), CCVC (consonant–consonant–vowel–consonant) or CVCC. In case of consonants we used words containing no more than two syllables. The consonant in question was in the middle of the word. The set of such words is pretty limited in Dutch and therefore most of them occurred more than once in the whole prompt set.

Phonetically rich sentences

The 10 sentences in each section are in fact a pair of randomly chosen phonetically rich sets from POLYPHONE. Thanks to the size of POLYPHONE we had enough sentences to fill all of our prompt set and none of them appeared more than once.

Although the phonetically rich sentences guarantee occurrence of each of the phonemes, they do not guarantee the natural distribution of the phonemes. One might be

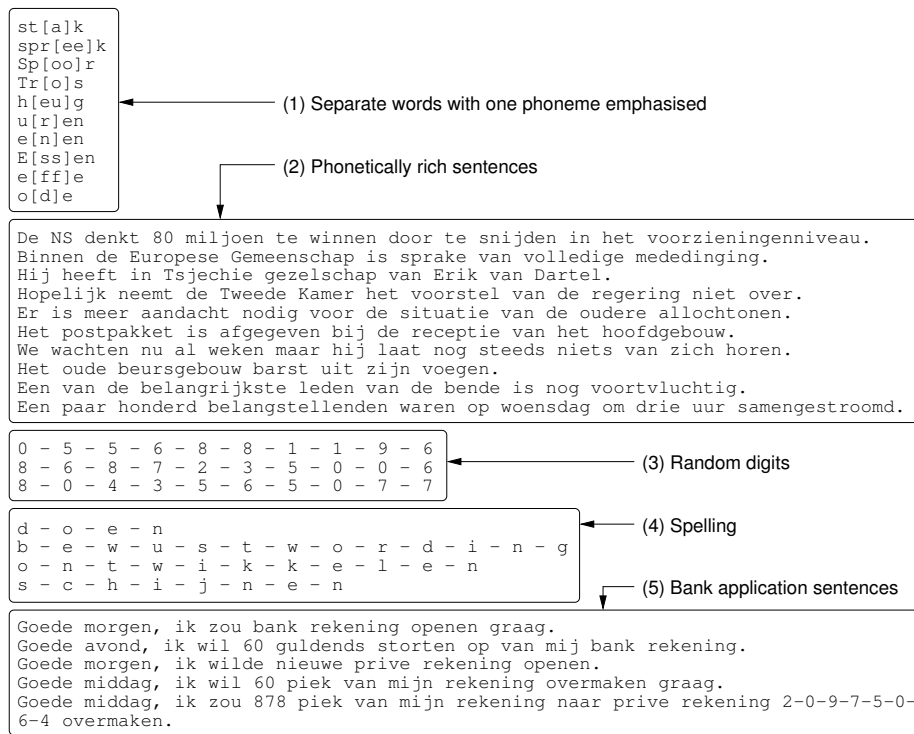


Figure 7.3: Example set of prompts.

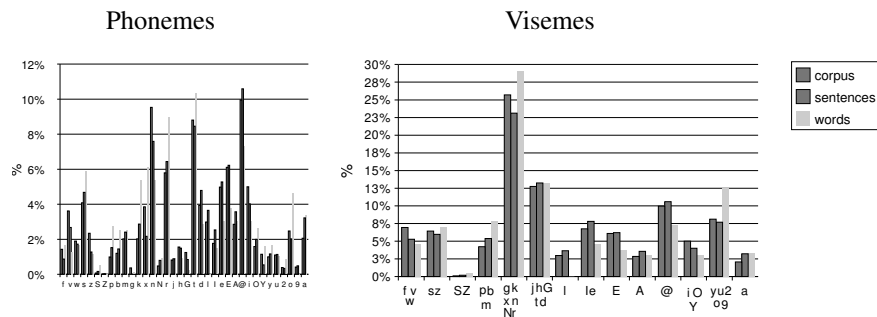


Figure 7.4: Phoneme and viseme histograms for the POLYPHONE corpus, phonetically rich sentences in our prompt set and the phonemes selected in isolated words.

afraid that the phoneme distributions in those sentences will be skewed in the direction of the least common phonemes. In order to check this we compared the phoneme histogram of the selected sentences with the histogram of the whole POLYPHONE data set (see Figure 7.4.) The histogram of the whole POLYPHONE may be assumed to be a natural distribution of the phonemes in Dutch as the forced utterances are only a small part of the whole data; most of the POLYPHONE corpus consists of spontaneous answers to the questionnaire.

Digits

Digits part of the prompt section is made up of 30 digits in total. They were presented to the subject in 3 sequences, 10 digits each. The digits were randomly generated and have a uniform distribution in the whole prompt set. There was however no uniformity forced on a per-section basis. The digit recordings can be used in experiments with the limited-vocabulary recognition.

Spelling

Spelling-based recognition is in fact a specific case of a limited-vocabulary recognition. It is, however, sometimes necessary to use this approach. An example could be information retrieval from a phone book. Using the phonetic transcriptions of all of the names is not feasible in this case, especially if the speech recognizer and the database are separate entities connected through some common protocol for data retrieval (see [vVdHR00]). Especially in this case spelling the name is a rather intuitive approach, even in human-to-human communication. For this reason the spelling of 4 randomly chosen words was included in each section.

Application-oriented utterances

Testing the real-world performance of the recognizer requires also some utterances with a constrained grammar and vocabulary. For this purpose we have chosen a telebanking application and prepared a simple grammar for the opening user utterance. The Hidden Markov Toolkit (HTK) was then used to create a corresponding word net and later to generate a set of random utterances from it. The grammar was prepared with recognition in mind, so some of the generated utterances are not grammatically correct. This is not a big issue however as we do not intend to deploy such a system, but rather just want to provide the possibility to test the capabilities of bimodal speech recognition in a constrained-grammar situation. Each section contains 5 sentences generated by HTK and was later manually supplied with punctuation marks.

7.1.3 Physical setup

The recordings were conducted in the setup shown in Figure 7.5. The subject was seated in front of a laptop computer on which the prompts were displayed to her/him. The prompts were sequentially read from the prompt set and displayed using a simple `PromptShow` application (see Figure 7.6 for the actual view of it). We developed this

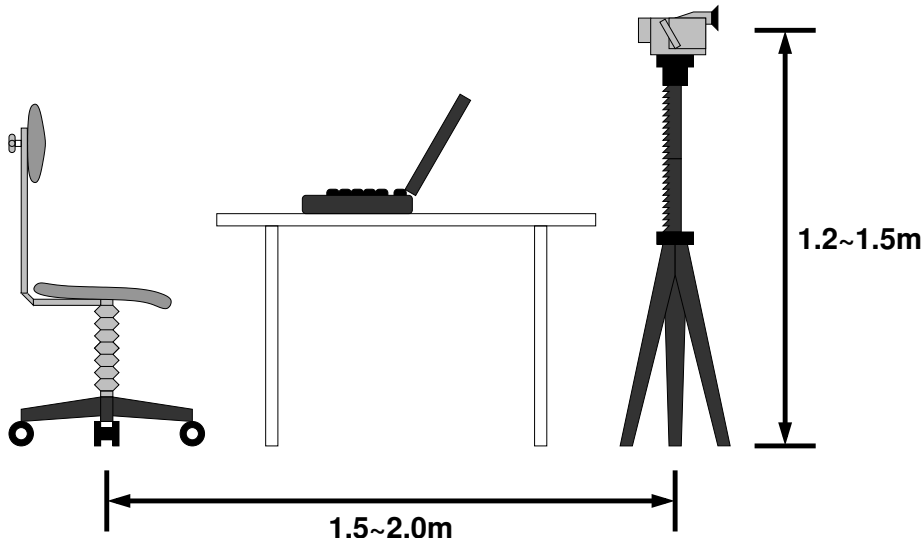


Figure 7.5: The physical setup of recording environment. The subject sits in front of the laptop on which the prompts appear. The camera is placed on a tripod behind the laptop.

application in a portable manner using the Qt library so that it can run in both Windows and Unix environments. `PromptShow` allows for choosing the appropriate prompt set file, selecting the section from which it will start etc. It also gives the possibility to highlight any part of the displayed text by using **bold**, *italics*, underlined or a user-specified color for the text (or any combination of the aforementioned). The progress of the prompts was controlled by the operator, so that the only task of the subject was to read aloud the contents of the prompts.

We recorded the material with a SONY TRV20E digital camcorder on standard DV tapes equipped with Cassette Memory chips. In this way we could digitally store the code of the recorded session for further reference. In order to record an audio signal with a satisfactory quality, we had to use an external microphone hung around the subject's neck. We used a standard low-cost computer microphone because of its availability, light construction and satisfactory quality.

The camera was placed on a tripod behind the laptop. During each recording, we used the camera's LCD screen to monitor the position of the subject's mouth in the field of view. It proved that the setup was comfortable enough so that most of the participants did not move substantially during the recordings. The camera's direction was adjusted usually only in the beginning of each new section (the participants were allowed short breaks between sections.)



Figure 7.6: Example of prompt presented to the subject by PromptShow software.

7.1.4 Storage structure

After recording on mini-DV tapes, we had to convert all data to a more convenient digital form. The storage structure is depicted in Figure 7.7. Each of the recorded sessions was edited using video editing software and cut into smaller sequences. The video sequences were then converted from a standard DV format into an MPEG1 stream. Moreover, from all of the scenes the audio data was extracted and saved separately. Further, proper transcriptions of the utterances were saved in textual form alongside the media files.

The video sequences are in a half-PAL resolution (384×288); they were sampled at 25 frames per second and then saved with 600kbps bit rate. This is a relatively high bit rate and together with low dynamics of changes in the video it provides us with a fairly undistorted picture.

The resulting files were stored on CD-ROMs with the following directory structure:

Top-level directory – This directory contains only subdirectories with the names corresponding to the video tape on which the session was recorded. They are called S001, S002 etc.

Session directory – Each session directory contains two text files that describe the recording session and several section directories. The `info.txt` file is a strictly structured file with the basic information concerning the number of recorded

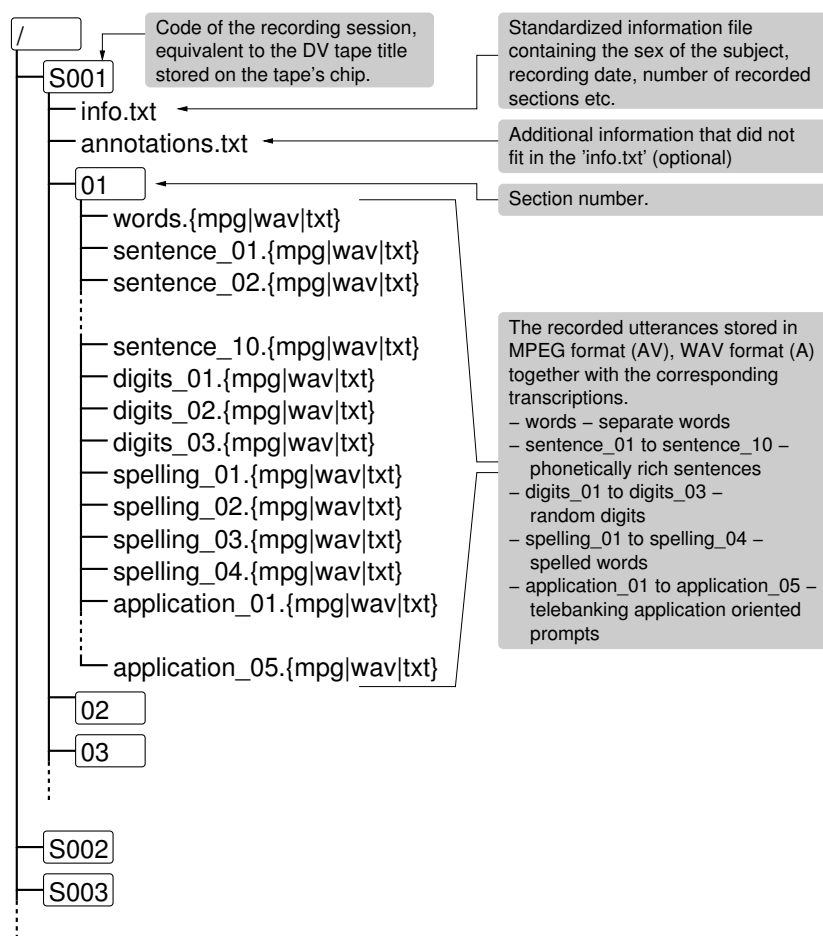


Figure 7.7: The way the recorded data is stored on CD-ROMs.

Table 7.2: Utterances recorded in the corpus.

	Normal	Slow	Whisp.	Total
Sections	58	22	7	87
Sentences	865	330	105	1300
Words	9380	3616	1153	14149
Words (sep.)	571	219	70	860
Digits	1683	627	218	2528

sessions, speaker characteristics and other similar information. This file is mostly intended for automated use. The `annotations.txt` file on the other hand contains the verbal description of the recordings together with the description of any anomalies in the data and other things not captured in the `info.txt` file.

Section directory – The directories were numbered subsequently depending on the section number in the prompt set. There are three types of files:

- .mpg** – MPEG1 encoded video sequences
- .wav** – uncompressed audio in a standard wave format
- .txt** – the transcription of the utterances from the above files

The different parts of the prompt set, as described in Section 7.1.2, were stored in the files with the following names (without the type-dependent suffix):

- words** – a set of 10 phoneme specific words
- sentence_number** – each of 10 phonetically rich sentences
- digits_number** – ten digits spoken with short pauses
- spelling_number** – one of the 4 spelled words
- application_number** – one of the 5 telebanking-related sentences

7.1.5 Recorded data

We have recorded in total 87 sessions with 8 different respondents. This gives in total over 4 hours of constant recordings. The recorded respondents were all native Dutch speakers; 7 males and only one female. This gender skew could not be avoided with the volunteering students of Delft University of Technology (the actual male-to-female ratio on the computer science faculty is even higher).

We asked the respondents to vary the speech rate during the recordings. Some of the recorded sections were marked as being “slowly spoken”, which means that the respondents were asked to slow down the speech rate. A small amount of sessions were also recorded with respondents whispering the prompts in order to allow investigation of this type of articulation as well. The total numbers of recorded utterances are summarized in Table 7.2.

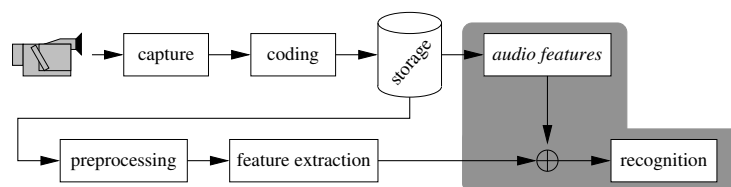


Figure 7.8: Lipreading pipeline.

In the following experiments we used the data from 7 CDs with 31 sections recorded with a group of 5 speakers (the single female speaker included). The available data set was broad enough to allow us to develop a person-specific lipreading system that recognizes strings of digits [WR01b] and another system for person-independent continuous audio-visual speech recognition [WWR02b].

7.2 Combining recognizers

In order to develop a bimodal speech recognizer, we had to face a challenging problem: how to combine the modalities together in the recognition pipeline. In Chapter 2 we presented already some models of human perception combining different modalities. Obviously, the way humans perform sensory integration may be used as an inspiration for integration strategy used in the developed system. But at the same time, there is no guarantee that the human perception model is by any means optimal for machines. Concluding, we have to use this knowledge with some caution.

There are three different integration strategies to be considered (see Figure 7.9):

- early integration (or feature fusion),
- intermediate integration (or model fusion),
- late integration (or decision fusion).

The above-mentioned modality integration strategies are named either based on the place at which the interaction between modalities occurs (early, intermediate or late) or based on what exactly is combined in the process (features, models or decisions). In the further sections we will tend to use the first division. We will discuss first the simplest case of early integration strategy, follow it up with late integration and describe the most complex intermediate integration at the end.

7.2.1 Early integration

The early integration scheme is conceptually the simplest approach to bimodal speech recognition. In this case, the interaction between the signals happens in the earliest possible stage. The feature extraction must be done on each of the signals separately. They differ so much that it is not possible to use any conceivable processing technique that would fit them both. Yet as soon as the features are extracted from the incoming

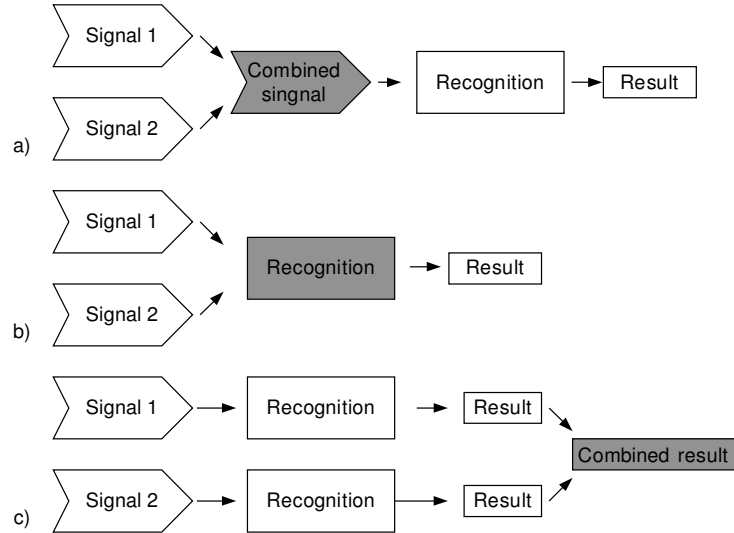


Figure 7.9: Different integration strategies. Early integration (a), intermediate integration (b) and late integration (c).

data, the integration can take place. The features from both an audio and a video stream are concatenated together to form a new audio-visual feature vector.

There are some issues that need to be resolved when applying this scheme. First of all, the feature streams are not necessarily sampled at the same rates, so there is no direct one-to-one relationship between the visual and auditory observations. It is very common that the audio features are sampled in 20ms intervals, while the video frames arrive every 40ms (or 33ms depending on the video standard common to the country).

This discrepancy in the timing of the observations must be handled in some way. It could be done either by copying the missing observations in the video stream from the available ones. After all, we may assume that the observed feature values are valid for the whole 40ms extent of the observation window. This would introduce some high-frequency noise in the video stream, however. It is therefore much more common to interpolate the missing observations from the available ones (see Figure 7.10). The process of interpolation (either linear or of higher order) is bound to introduce some artifacts, but for the sake of simplicity they are usually not considered to be important.

Another issue that relates to the timing of the observations in both channels is the fact that it is not guaranteed that the changes in both signals occur simultaneously. For example, the visual occurrence of [p] being spoken (that is, closing the lips tightly together) happens before the auditory observation (a sudden energy burst in a wide range of frequencies). Some of the visemes are visible before the corresponding phoneme can be heard, and some after. There is no way to compensate for this lack of synchronization between the two channels when doing integration early. Experiments with human observers show that it is beneficial for the intelligibility of the speech to include the vi-

sual modality even if it is in perceivable asynchrony with the auditory signal [Sme95b]. It is hard to believe that the same will hold for machine recognition.

After obtaining a stream of concatenated feature vectors, some post-processing of this data can be done. For example, the combined features can be transformed using MLLT so that the resulting vectors are best suited for discrimination between different phonemes. Such a combined postprocessing stage can also be used to limit the redundancy in the combined data and to decrease the size of the feature vectors (see [PN01, DPN02]).

7.2.2 Late integration

The concept of late integration of different speech modalities is very clean and intuitive. It is the same one that is often used to model human perception patterns. Models for human sensory integration (as described earlier in Section 2.1.2) differ in their mathematical formulation and obtained predictions. Yet they all assume that the perception of each modality happens independently. That is, the visual and auditory signals are processed by different parts of the brain and they both yield some hypothesis about the spoken utterances. The signals are integrated by comparing the two hypotheses and concluding on a result. The stimuli that activate the perception of speech are therefore to some extent separated from the integration process. The validity of models such as FLMP goes well beyond speech perception. That in turn suggests that the integration process is indeed disconnected from the nature of the signal processing and operates only on abstract hypotheses.

In context of machine lipreading, late integration means the existence of two independent recognizers and an arbitrating module which consolidates their outputs. A simplest implementation of such a module could be using a Bayesian rule for calculating conditional probability:

$$P(\alpha|AV) = \frac{P(\alpha|A)P(\alpha|V)}{P(A|V)}. \quad (7.1)$$

This fundamental equation tells us that it is possible to calculate the probability of the utterance α being spoken given a set of audio-visual observations from other better tractable probabilities. The probabilities $P(\alpha|A)$ and $P(\alpha|V)$ are easily derived from the output of the separate recognizers if they both are properly trained. This is obviously true for HMMs that directly provide the a-posteriori probabilities as their output. This is also true for some ANN based classifiers. When properly trained, their outputs

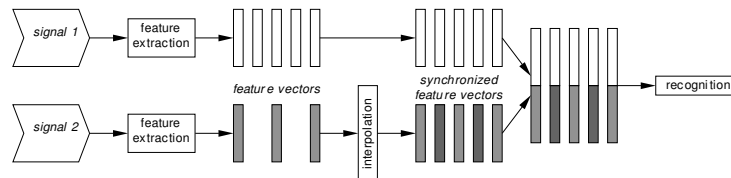


Figure 7.10: Feature fusion.

will also converge to the a-posteriori probability values. In this way, it does not matter what kind of recognizer paradigm is used. The process of modality integration is abstracted from the underlying recognition. The determinant $P(A|V)$ of Bayes' equation (7.1) performs only a normalizing role and for single audio-visual observations it can be omitted. After all, we are not interested in the real probability values, but rather in their relative distribution across the set of possible utterances α .

The Bayesian approach to the audio-visual integration is very simple in its form, but also severely limited. First of all, it is very hard to guarantee that the results from the two recognizers really reflect the necessary probabilities. The experiments show that by varying the weights with which the modalities contribute to the result, we can improve the performance of the combined recognition [HWBK01, LSC01]. Such a weighing mechanism is often implemented in the following form:

$$P(\alpha|AV) \sim P(\alpha|A)^{w_A} P(\alpha|V)^{w_V}. \quad (7.2)$$

While at first sight this weighing mechanism seems plausible as a method of introducing the concept of reliability of recognition, it completely defies the logic behind the Bayesian rule. Because, if the output of one of the recognizers were to become less reliable for any reason, it would directly cause a decrease in entropy of the output produced across the whole set of possible results. In other words, the distribution of the outputs should become more uniform, making this modality less influential. By putting additional weighing coefficients in the sensory-integration process we actually account for the reliability of the recognizer twice (or counteract it if our weighing guess is wrong). The fact that this technique gives good results shows that there are some serious problems in our assumption that the values obtained from the unimodal recognizer represent a-posteriori probabilities (for further investigation of this subject see [LSC02]).

Another problem with Bayesian integration of modalities is that it is only tractable for a small, discrete set of possible recognition results α . In the context of speech processing, this approach can only be used for limited vocabulary recognition with a single word as the recognition entity. Even connected digits recognition, such as that described in Chapter 5, does not fall into this category. Therefore, if we intend to perform late integration in a continuous speech processing system, we need to implement other, much more sophisticated integration models.

7.2.3 Intermediate integration

It is a common belief that even though the late integration principle seems to be the one that is utilized for sensory integration by the human brain, it is not the one that would be the most advantageous for machine speech processing. The intermediate integration approach is often viewed as the most promising one. It comes however at the cost of a highly increased complexity of the recognizer architecture.

The intermediate modality integration means that the signals are combined place inside the recognizer architecture. The recognizer itself must handle two different data streams, combine them together, model their interdependencies and provide a unified result. We will discuss some of the issues involved in this process using the example of HMM based recognizers.

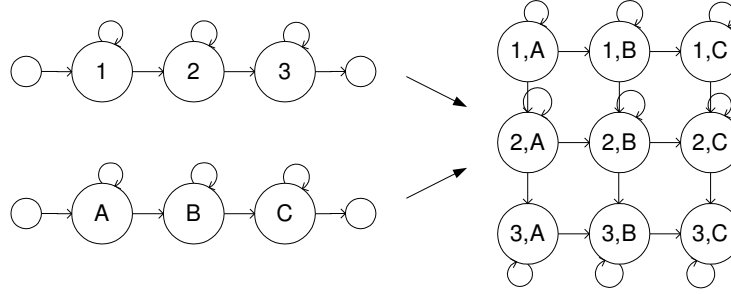


Figure 7.11: Cartesian product of HMMs.

At first sight, it seems that there is a simple solution for combining two modalities in the HMM framework. We can consider a set of models, one for each modality and construct a multimodal one in form of their Cartesian product. The process of obtaining such a Cartesian product HMM from two independent models is depicted in Figure 7.11. Each state of a newly created HMM represents a unique combination of states from the original models. Each possible combination of paths through both unimodal models can be represented as a single path through the product HMM.

The Cartesian product HMMs have one fatal disadvantage though: they are practically intractable. The amount of free parameters is enormous when compared to those of the original models. This simple fact results in much longer training, slower recognition, problems with over-fitting etc. While the product HMMs are not interesting for developing a bimodal speech processing system, they do form a theoretical foundation for some other models such as Linked HMMs or product multistream HMM [Wig01]. By putting constraints on different parameters of the product HMM, we can significantly limit the number of its free parameters.

Another possible way to implement intermediate integration is to use so-called multi-stream HMMs. Using this approach, each of the modalities is modeled separately. The models can have different topologies, they operate in an asynchronous way and model signals with potentially different sampling rates. The modality integration is done by forcing the models' synchrony at some key points in time. The combined model M is therefore constructed from a number of sub-models M_j connected by synchronization states. Between those states, each modality of the signal is modeled by a separate model M_j^k . The synchronization states are non-emitting and therefore do not relate to any specific modality. They are only points in which the separate Viterbi paths through the complex model must arrive at the same time. At those points, the probabilities obtained from different models can be used to calculate the joint probability of generating a multimodal observation. An example multi-stream HMM topology is shown in Figure 7.12.

The multi-stream approach has many advantages. Firstly, different streams of data are modeled independently of each other. This allows the streams to have different sampling rates and different dynamic ranges. It also allows for variations in HMM topologies used for different modalities. While the same can be said about the product HMMs, the complexity of multi-stream HMMs is much lower than that of the Cartesian

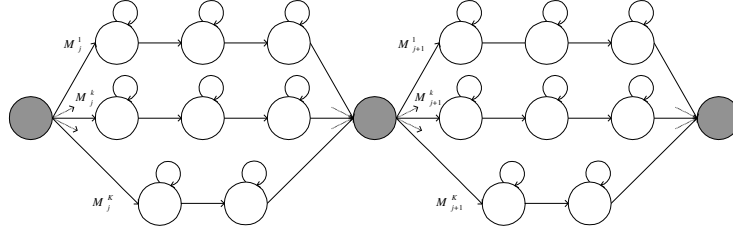


Figure 7.12: Multi-stream HMM. The filled circles represent non-emitting synchronization states.

product of separate models. The number of free parameters in multi-stream HMM grows proportionally to the number of modalities, in contrast to the exponential growth in case of product HMMs. Moreover, there exists the possibility of extending the multi-stream approach by adding other types of recognition paradigms. It would therefore be possible to replace a sub-model M_j^k with an ANN, as long as we can guarantee that its output at the synchronization stage can be treated as a probability measure.

Multi-stream HMMs essentially do not belong to the HMM class. In contrast to a generic HMM type (as in Section 3.4), multi-stream HMMs have more than one emitting state at each point of time. Moreover, if the observation streams do not have the same sampling rate (or sampling times), the execution of a multi-stream HMM generator will proceed at different rates in different sub-models. This fact complicates the algorithms to be used for training and recognition using multi-stream HMMs.

If observed signals are sampled in synchrony, at the same rate, the multi-stream HMMs can be converted into an equivalent Cartesian product HMM. In this case it is possible to use the generic recognition algorithms. In order to achieve this equivalence during the training process as well, the number of parameters in the product multi-stream HMM must be limited by parameter tying (see Figure 7.13). By tying the parameters of different states together it is possible to guarantee that the product multi-stream HMM remains equivalent to a multi-stream HMM during estimation of its parameters. This technique is especially useful if the independent models M_j^k from the original multi-stream topology are relatively small and synchronized at short time intervals.

The complexity of multi-stream HMM can be decreased even further by forcing a synchrony between auditory and visual states. This approach, often called state synchronous multi-stream HMM, is actually almost equivalent to feature fusion and therefore it is rather an early integration technique. The only difference with pure feature fusion lays in the fact that by keeping the unimodal models separated it is possible to weigh the streams. That is, the observation probability for a state j , given the observation o_t is:

$$b_j(o_t) = b_{Aj}(o_t)^{w_A} b_{Vj}(o_t)^{w_V}. \quad (7.3)$$

By choosing different combinations of weights w_A and w_V , we can vary the importance of each of the streams in the recognition. Just as with weighing in the late integration procedure (Equation (7.2)), this approach actually defies the notion that the

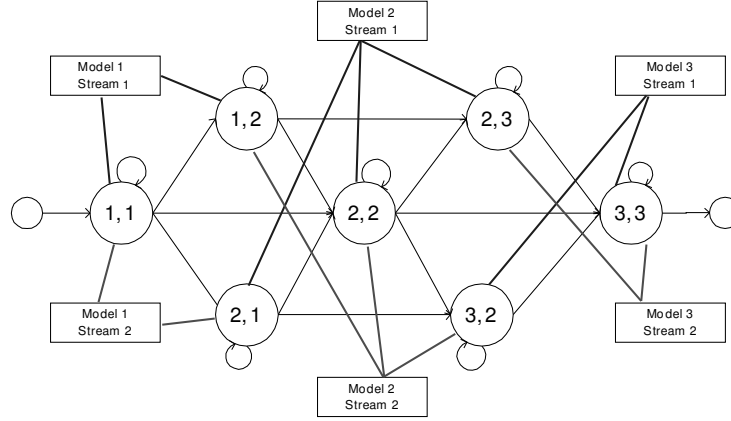


Figure 7.13: Tying parameters of product multi-stream HMM.

values of $b_{\{A,V\}j}(o_t)$ represent the probabilities, but it proves to be very useful in real-life experiments.

7.3 Bimodal continuous speech recognizer

In this section we will describe our continuous speech processing system augmented with lipreading capabilities. The system was developed for the purpose of testing the LGE technique in this context. This work has been done in cooperation with Pascal Wiggers, who wrote his MSc. thesis on this subject [Wig01].

7.3.1 Choosing integration strategy

The first thing to do when developing a bimodal speech recognizer is to select a possible integration strategy. All three of them have their advantages and disadvantages and need to be evaluated in the context of the available data, computing power, and time limits. At the time the DUTAVSC corpus was completed, we had in our lab working a prototype of a speech recognizer built on HTK toolkit [WWR02a]. All of the experiments with processing continuous visual input described in Chapter 6 had also been finished, which gave us the confidence that a working continuous lipreading system was close at hand. It therefore seemed plausible to start by investigating the late integration strategy.

To our dismay, all attempts at developing the lipreading part of the system failed. The HMM based lipreader built using the HTK toolkit would not train on continuous speech data. We experimented with different model architectures, different PDFs for the states, and different parameter tying approaches. Nothing could bring the recognizer above the “pure chance” performance level. This bitter failure on the lipreading front forced us to re-examine the input data in search for clues as to why things went wrong. This led to the discovery of the Person-Independent Feature Space (see Sec-

tion 4.3) and further refinements in LGE algorithms. Yet even those improvements in signal processing did not change the outcome of the training process.

It became apparent that there are two main reasons for this lipreading failure. Firstly, the recorded data set was just too small for developing a continuous lipreader. Because of the nature of the lipreading process, the HMMs used to model must be rather complex. The complexity of HMMs for lipreading exceeds the complexity that is needed for speech recognition. The coarticulation effects are much more evident in a visual than in an auditory channel. Therefore a tri-phone approach is a must for automated lipreading. That means that instead of having a single model for each viseme we need a model for each viseme in a context of any other viseme that precedes or follows it. Even with highly aggressive parameter tying it still multiplies the number of free parameters by at least a factor 10. Of course, the tri-phone approach is rather common in audio-based speech recognition, but it is only applied as a refinement to the monophone-based recognizers and only if there is a sufficient amount of available data to justify it. Further, a typical string of viseme observations is much less static than a string of observations for a given phoneme. That means that we need to increase the number of states within a HMM if we want to model the changes during viseme observation correctly. All in all, the DUTAVSC was sufficient for training a small speech recognizer, but far from sufficient for training a lipreader with at least 20 times more free parameters.

The other reason our attempts failed lays on a much more fundamental level. In order to evaluate the data set from a human perspective, we asked a hearing-impaired student with some lipreading experience to have a look at the data. After showing him different video recordings, we found out that even he did not have a clue about what was being said. While it is obvious that the performance of human lipreading is highly dependent on context, the extent to which the lack of context can impair this process was a surprise. Obviously, the sentences we recorded were not placed in any context. They were in fact picked *out* of context and selected purely based on the distribution of their phonemes. Also, the lipreading architecture is not context aware. The only context-related knowledge that is available to the recognizer consists of probabilities on the order of word occurrence. If a lack of context can confuse a human, it is obvious that its influence on the machine will be devastating. It is our opinion that whoever plans developing an independent continuous lipreading machine needs to take context awareness very seriously.

After deciding against late integration, the intermediate integration strategy was not considered any more either. This decision was based on the assumption that the complexity of models dealing with multiple modalities would once again challenge the limitations of the DUTAVSC. This left early integration as the only available option. At the time the decision was made we found some encouraging results with early integration schemes when applied to a limited vocabulary recognition [PN01]. We decided therefore to start with the earlier developed speech recognizer and to extend it with lipreading capabilities by using a state synchronous multi-stream HMM architecture.

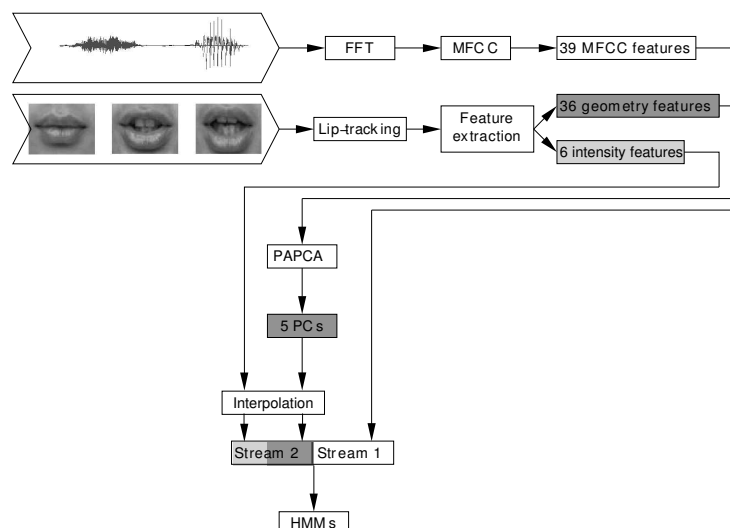


Figure 7.14: The feature fusion for multimodal speech recognizer.

7.3.2 Baseline system

The prototype speech recognizer was developed using the HTK toolkit and was trained on the POLYPHONE data set. The POLYPHONE provides a large training data set for a variety of Dutch dialects and a broad spectrum of speakers. Yet it is limited by the quality of the recordings. All of the recordings were made through a telephone line and therefore they are rather noisy and severely limited in frequency range. The data in DUTAVSC is of a much higher quality and clearly differs from the POLYPHONE recordings. Therefore, the prototype system was only used as a starting point for further refinements based on data from the audio-visual recordings. The audio data in DUTAVSC was divided into the training and evaluation sets. After retraining, the performance of a speech recognizer was at the level of about 84% correctly recognized words in clean speech conditions.

7.3.3 Feature fusion procedure

The feature fusion procedure we used is depicted in Figure 7.14. We used 13 MFCC features together with their deltas and acceleration values (a total of 39 features for an audio stream). For the lipreading part of the system we used the LGE extraction procedure augmented with intensity features (see Chapter 4). The geometry data has been processed with PAPCA, so that the resulting geometry features were transformed into the Person-Independent Feature Space (see Section 4.3). The projection parameters for PAPCA were estimated off-line beforehand for each of the persons for both the training and the validation sets. The lipreading-related data stream was sampled with a rate twice lower than that of the audio stream, so we had to interpolate every second observation in order to bring both streams in synchrony.

Table 7.3: Recognition performance for different weighing of bimodal input streams.

weights	word recognition rate (%)	accuracy (%)
$w_V = 1.0, w_A = 1.0$	83.69	78.61
$w_V = 0.8, w_A = 1.2$	84.43	79.41
$w_V = 0.6, w_A = 1.4$	84.22	78.88

Table 7.4: Recognition performance with viseme-tied parameters for different weighing of bimodal input streams.

weights	word recognition rate (%)	accuracy (%)
$w_V = 1.1, w_A = 0.9$	84.76	80.48
$w_V = 1.0, w_A = 1.0$	85.56	80.28
$w_V = 0.9, w_A = 1.1$	85.92	82.09
$w_V = 0.8, w_A = 1.2$	85.03	80.75

The models in the baseline system were extended with the parameters describing the lipreading observation PDFs. With the HTK toolkit this proved to be a relatively straightforward task. We extended each state in each of the models with the single Gaussian observation PDF (b_{Vj}) with its parameters set to the mean and variance of the whole lipreading data training set. In this way we could gain from the fact that the already trained auditory part of the models would allow for better initial segmentation of the data and improve the speed of training the visual part. Such a fused system was retrained on the bimodal input.

7.3.4 Recognition results

By using a multi-stream HMMs for our recognizer, we could modify the extent to which the modalities influenced its performance. The results in Table 7.3 show that it is beneficial to increase an audio weight by a small amount, but that it should not be increased too much.

Another thing that can be considered is that some parameters in the lipreading part of the system are redundant. We may recall that the basic building block of the visual speech is the viseme. As visemes represent a group of phonemes that are indistinguishable from a visual point of view, it is rather logical to reflect this in the recognizer architecture somehow. This can best be done by tying the observation probability parameters for the lipreading part of the system across the phoneme groups. The results obtained for the viseme-tied bimodal speech recognizer are presented in Table 7.4. The viseme tying resulted in better results, but without outstanding improvement.

Because of the fact that the recognition rates are already pretty high for a continuous speech recognizer, we cannot really expect much further improvement from fine-tuning of the parameters of the recognizer architecture. Even in case of humans, the influence of lipreading in clean-speech conditions is relatively small.

What is much more interesting is whether the lipreading-capable system would

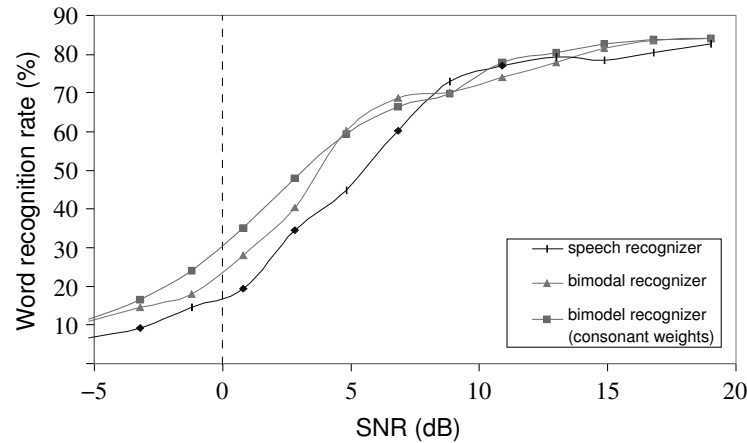


Figure 7.15: Recognition performance of three different speech recognizers for different SNR levels.

show an improvement over our baseline speech recognizer in noisy conditions. As can be seen in Figure 7.15, our bimodal speech recognizer did not perform significantly different from the baseline system if the signal to noise ratio (SNR) was above 8dB. However at lower SNR values, the lipreading appeared to help the speech recognition significantly. The improvement can be seen as a perceptual noise reduction by on average 1.4dB for the SNRs below 8dB.

The results depicted in Figure 7.15 show that the benefits of lipreading for speech recognition depend on the level of acoustic noise in the incoming data. We decided to investigate which part of the recognizer is most affected by the noise, and where lipreading might help even more. It turned out that the recognition of consonants suffers most from the presence of noise. At the same time, the lipreading part of the system seemed to perform especially well on consonants. Concluding, we can improve the performance of our system by varying the stream weights for consonants depending on the SNR level. The results of such experiments are also depicted in Figure 7.15. The effective noise reduction is on average 2.5dB, more than 1dB better than the noise reduction of the fixed weight system. The disadvantage of this approach is that the noise level must be known beforehand in order to adjust the weights accordingly. In typical situations such information is not available during speech processing. There are however methods for estimating the weighing coefficients based on different measures of the incoming signal [LSC02].

When we asked Pooh what the opposite of an Introduction was, he said “The what of a what?” which didn’t help us as much as we had hoped, but luckily Owl kept his head and told us that the Opposite of an Introduction, my dear Pooh, was a Contradiction; and, as he is very good at long words, I am sure that that’s what it is.

Chapter 8

A. A. Milne, **The House at Pooh Corner**

Conclusions

In this last chapter of the thesis we will summarize the results of the experiments and put them in broader perspective. It seems inevitable to write about more than just pure facts and quantitative descriptions of the results. We will also try to extrapolate to the near future of this research field. As a result, the chapter is divided into two sections: the first one concluding on our research experiments, the second one focusing more on the future of lipreading as a research field.

8.1 On experimental results

The first experiment presented in this thesis deals with the separability of different spoken utterances given a specific type of representation (Section 2.3). The results from this experiment confirm that the acoustic representation of speech is the one that is most reliable and most error proof. This simple fact is more qualitatively presented in Figure 2.5. It is important to realize that the lines in this figure represent hard boundaries that limit the recognition rates. For example, if we have a recognizer that achieves a 90% recognition rate on a per-viseme basis, its WRR will not exceed 65% unless some context knowledge is introduced. Also, the architecture based on viseme syllable sets instead of visemes would make twice less mistakes. Figure 2.5 can be seen as a map depicting the dependencies between the complexity of representation (and therefore the complexity of the recognizer), the reliability of the recognition engine on the basic level, and the limits of achievable recognition rates. The flat line at the bottom of the drawing, representing the phonetic representation of speech, is a dark reminder of the fact that a visual signal really is no more than a shadow of the accompanying acoustic speech.

The possible recognition rate of a developed lipreading system depends highly on the robustness of the lip-tracking algorithms used, of course. The choice here does not seem to be obvious. On the one hand, simple algorithms such as point trackers can be efficiently implemented and are not overly complex. The results from such trackers, even if not entirely reliable, can later be processed with different noise-removal and correction techniques (such as temporal consistency checks), if there is enough

computing power left, that is. On the other hand, complex approaches yield much better results right away. There is a fine balance to be found between “track coarsely – correct later” and “track as accurately as possible”. Yet the results presented in Section 3.2 show that even the differences between the most distant lip-tracking models are not overwhelming. To a great extent, different geometrical models of the lips are compatible with each other. This is a very important result that suggests that future researchers may concentrate on developing new recognition models independently of the lip-tracker architecture. The choice of lip-tracking algorithm remains application bound. Depending on the features and quality of available video input, one can choose the most appropriate tracker. As an example, if the video is constrained to black-and-white only, the LGE is excluded, while point and model-based trackers fit the situation perfectly. The high level of equivalency between feature extraction techniques also exists between geometric models in general and image intensity based approaches. For example, if the video signal is highly compressed with some visible artifacts, the raw-image (RI) or Discrete Cosine Transformation based methods can be used instead of the lip tracker. In all examples, the recognizer architecture may remain fixed. It only needs to be re-trained for the specific data.

For this thesis research we developed a novel approach to feature extraction from a video stream. The Lip Geometry Estimation (LGE) is presented in detail in Chapter 4. In our opinion its complexity versus accuracy has exactly the right balance for lipreading task. It is based on simple and computationally effective lip-selective color filtering techniques. LGE is also rather insensitive to the presence of noise in the image and effectively filters a large amount of person-specific information, such as texture, skin color, exact shape of the mouth etc. One additional advantage of LGE is that while it has been conceived as a purely geometric model, it can be easily extended with intensity information obtained from color filters which are conceptually similar to the lip-selective filter. The advantages of such a combination were further discussed in Chapter 5.

Even if some of the person-specific features are filtered out by using LGE, the resulting features are still highly person dependent. In this respect LGE does not differ that much from other geometry-based models. Tables 4.1 and 4.2 show a much better overlap of LGE-extracted effective feature spaces between people than for the RI-extracted features (qualitatively this can be observed in Figures 4.8 and 4.9). The following problem remains though, and it is common to all geometry-based methods: they do not provide sufficient separation between system and speech parameters. In Section 4.3 we present a novel method for dealing with this lack of separation. The schematic representation of the problem and the proposed solution can be found in Figure 4.10. By using person-adaptive PCA (PAPCA) we can project the features into a Person-Independent Feature Space (PIFS). The validity of the approach is to some extent verified by comparing viseme distribution overlaps of different visemes across several persons (see Table 4.3). The advantages of using PIFS for representing visual speech signals are mentioned at the end of Chapter 5.

In that chapter we also present the results of our experiments in developing the limited vocabulary lipreading system. Obtained recognition rates are sufficiently high to conclude that the proposed data processing techniques are suitable for lipreading applications. Yet the obtained recognition rates are not the most interesting part of

that chapter. A comparison between a pure geometric model (using LGE) and a model augmented with intensity features gives the most important information learned from the experiments. Table 5.1 shows clearly that there is more to lipreading than just the lip geometry. It shows that the introduction of features related to tongue movements and visibility of the mouth cavity improves the accuracy of the recognition from around 60% to around 80% for unconstrained grammar cases.

In all of the presented experiments we used primarily two recognition engines: Artificial Neural Networks (ANNs) and Hidden Markov Models (HMMs). While the Chapters 5 and 7 concentrate on using HMMs, Chapter 6 deals almost exclusively with ANNs. This specific chapter deals with the short-time-span characteristics of the speech signal. In the presented experiments we looked at the visualization and recognition aspects of sub-word units. For such tasks the ANNs performed much better than the HMMs given the same amount of training data.

During our research we needed audio-visual data. As a solution to this problem we gathered a set of recordings called the Dutch audio-visual speech corpus (DUTAVSC). In Section 7.1, the whole data acquisition procedure is described in detail. To our best knowledge, this data set is the most comprehensible speech-oriented audio-visual corpus for the Dutch language to date (it compares favorably with audio-visual corpora for other languages as well). The data has been made available to the scientific community and can be obtained at request as a set of CDs.

In the final experiments of our research we developed a bimodal continuous speech recognizer (a speech recognizer augmented with lipreading capabilities). Its development together with necessary theoretical background is described in Chapter 7. The performance of this recognizer is summarized in Figure 7.15, but it can best be described in terms of perceptual noise reduction. The addition of the lipreading component to the speech recognizer can be seen as a 2.5dB reduction in the level of noise.

8.2 On lipreading in general

The sheer number of papers dealing with lipreading and presented on international conferences shows that there is a broad interest in this aspect of speech processing. Yet there are almost no available applications or development tools that would bring lipreading to a broader audience. This perplexing situation shows that audio-visual speech processing is still in its infancy. In order to grow further it is in desperate need for appropriate audio-visual speech corpora. As has been discussed in Section 7.1 the availability of such data sets is crucial for progress in lipreading research. Unfortunately, the amount of resources needed to prepare a good speech corpus in general is staggering. The result is a chicken-and-egg problem: there are no lipreading applications because of the lack of data, while there are no commercially available audio-visual corpora because there is no application market that would pay for their development. This situation is slowly changing as big research labs recognize the potential of integrating the lipreading capabilities in frameworks of the existing human-computer interfaces.

In the near future, when standard audio-visual corpora emerge, another unification is likely to occur. Just as FFT became a de facto standard audio processing technique,

one of the image processing models will probably win over the world of lipreading. Based on our experiments we can make an educated guess that the technique will be a hybrid of geometry and intensity related features. The geometrical description of the mouth and other visible parts of the vocal tract is crucial for performing person-independent lipreading. At the same time, parts such as the tongue and teeth are extremely hard to track accurately, while at the same time they leave an easily distinguishable trace in the intensity of the image. Combining both types of image processing (as described in Chapter 4) makes it possible to combine their respective advantages into a robust processing architecture.

The recognizer architectures that are typically used for audio-based speech recognition (HMMs, ANNs) can easily be used in lipreading research, as this thesis shows, but it is not necessarily the best approach. Concluding our research: we are firmly convinced that lipreading needs a new recognition paradigm. There are several reasons for this. Firstly, it appears that the phoneme-level representation of the visual speech is too error prone (see Figure 2.5). It would be much better if we could model it on the syllable-level using Viseme Syllable Sets (VSS). The problem with using VSS, together with an HMM or ANN based recognizer, is that the number of free parameters grows explosively. The more free parameters, the larger the data set needed to train them properly. It is therefore not conceivable to develop a generic VSS-based lipreader in the near future. A syllable based approach to auditory speech has already been presented [Aha00]. It makes heavy use of language-specific features, parameter tying and preliminary training on the phoneme level.

Secondly, visual speech does not seem to be well described by the state-to-state paradigm employed by HMMs. The speech signal on the audio level does indeed contain short, phoneme-related intervals that have some stable features. Those can be effectively modeled by separate states in HMM. The visual speech signal is in this respect totally different. It is always changing without stationary points. The phoneme cannot be well described by the mouth shape, but rather by the trajectory embedded in the space of possible shapes. Fortunately, there are recognition methods that use such trajectory-based speech models [JR02]; those can be tested for lipreading as well.

Thirdly, the lipreading efficiency, even that of humans, depends heavily on the knowledge of the context in which the speech is perceived. If humans cannot do without extensive knowledge of the situation, how could a machine? The task of integrating the context awareness and problem-specific knowledge with the recognition itself is one that needs to be pursued. Again, in the field of auditory speech recognition, such approaches are emerging [WR03].

8.3 Final words

Lipreading itself is not a feature that users who interact with computers request too often. In fact, quite possibly machines will never become good at lipreading; it will not be necessary. Audio-visual speech processing is really just the beginning, not the ultimate goal of lipreading research. It is often more useful to get just a rough idea about the speech signal from the visual modality than to perform a full recognition. Existing speech recognizers can be augmented with the visual capabilities that may help

in segmenting the incoming signal. They can also give spatial clues for the acoustic beamformers in order to reduce the background noise ¹.

The quest is therefore not for lipreading but for a complete multimodal environment awareness by machines. Even right now computers can listen (through microphones), watch (through digital cameras) and feel touch (through pressure-sensitive gloves and other VR manipulation tools). All those senses utilized together may bring the interaction between human and computer to the next level. Lipreading research fulfills its role in teaching us multimodal awareness, sensory integration and other related topics. Eventually, lipreading will probably slowly fade away. The barriers between acoustic, visual and gesticulative signals are purely artificial. They are erected by researchers in order to reduce the complexity of the interaction. The machines must learn to process those signals together in order to become credible partners in interaction. While other communication modalities (auditory speech, gestures etc.) are often considered separately and appear self sufficient, visual speech cannot stand on its own. It is a mere shadow of the speech signal and therefore must be considered together with other modalities. This forces researchers to think more about the interactions between this modalities, to develop models for their cooperation. A full multimodal environment awareness of the machine is crucial if humans are to treat it as a partner, not as a simple tool. In this respect, lipreading is a cornerstone of Artificial Intelligence research, even if it does not look like one.

¹See for example <http://www.tm.tue.nl/uce/crimi/>

Notes on the bibliography style

In this thesis we use a standard `alpha` style from `BIBTEX` package for generating the list of references used throughout the book (see [Lam94] for more information on `BIBTEX` and `LATEX` in general). According to this style, different types of citations are referenced in the following manner: a book is cited as [BooAu] or [BooEd] depending on whether it was written by a single author or collected by an editor(s). An article in a scientific journal is cited as [Journ]. A citation of an article in conference proceedings may contain the full information on the proceedings itself [ProcA1] or it may refer to the proceedings indirectly in case of multiple articles cited from the same conference [ProcA2,ProcA3]. A specific chapter or other unit of a book is cited like this [Chapt], although it may also refer to the book indirectly in a manner similar to the conference example above. Finally, the PhD and MSc theses are cited as [ThePhD] and [TheMSc].

The bibliography has been collected and presented with the greatest care, but some mistakes in it are inevitable. Specifically, in some cases, the lack of availability of full bibliographic information on the item led to the omission of some of the information. The author of this thesis apologizes for any inconvenience arising from those omissions.

- [BooAu] Author's Name. *Title*. Publisher, year.
- [BooEd] Editor's Name, editor. *Title*. Publisher, year.
- [Chapt] Author's Name. *Title*, chapter Chapter title, page pages. Publisher, year.
- [Journ] Author Name. *Title*. *Journal Name*, volume(number):pages, year.
- [ProcA1] Author's Name. *Title*. In Editor Name, editor, *Proceedings Title*, page pages, Address, month year. Publisher.
- [ProcA2] Author's Name. *Title*. In Proceedings [ProcEd], page pages.
- [ProcA3] Author's Name. *Title*. In Proceedings [ProcEd], page pages.
- [ProcEd] Editor's Name, editor. *Title*, Address, month year. Organization, Publisher.
- [ThePhD] Author's Name. *Title*. PhD thesis, University, year.
- [TheMSc] Author's Name. *Title*. Master's thesis, University, year.

Bibliography

- [AGMG⁺97] A. Adjoudani, T. Guiard-Marigny, B. L. Goff, L. Reveret, and C. Benoît. A multimedia platform for audio-visual speech processing. In Kokkinakis et al. [KFD97].
- [Aha00] S. M. Ahadi. Reduced context sensitivity in Persian speech recognition via syllable modeling. In Barlow [Bar00], pages 492–497.
- [ATJL01] T. S. Andersen, K. Tiippana, and M. J. Lampien. Modeling of audio-visual speech perception in noise. In Massaro et al. [MLG01], pages 172–176.
- [Bar00] M. Barlow, editor. Canberra, December 2000. Australian Speech Science and Technology Association.
- [BC94] K. S. Benoît C., Mohamadi T. Audio-visual intelligibility of french speech in noise. *Journal of Speech and Hearing Research*, 37:1195–1203, 1994.
- [Beu96] D. H. Beun. Viseme syllable sets. Technical Report 130, Institute of Phonetic Sciences, University of Amsterdam, Amsterdam, The Netherlands, 1996.
- [BMP⁺] C. Benoît, J. C. Martin, C. Pelachaud, L. Schomaker, and B. Suhm. Audio-visual and multimodal speech systems. In D. Gibbon (Ed.) *Handbook of Standards and Resources for Spoken Language Systems* - Supplement Volume, 1998.
- [BPETA01] L. E. Bernstein, C. W. Ponton, and J. Edward T. Auer. Elektrophysiology of unimodal and audiovisual speech perception. In Massaro et al. [MLG01], pages 50–53.
- [Bul48] J. Bulwer. *Philocopus, or the Deaf and Dumbe Mans Friend*. Humphrey and Moseley, London, 1648.
- [CCVB01] D. Callan, A. Callan, and E. Vatikiotis-Bateson. Neural areas underlying the processing of visual speech information under conditions of degraded auditory information. In Massaro et al. [MLG01], pages 45–49.

- [Che01] T. Chen. Audiovisual speech processing. *IEEE Signal Processing Magazine*, pages 9–21, January 2001.
- [CSA01] M.-A. Cathiard, J.-L. Schwartz, and C. Abry. Asking a naive question about the McGurk effect: Why does audio [b] give more [d] percepts with visual [g] than with visual [d]? In Massaro et al. [MLG01], pages 138–142.
- [CTC96] T. Coianiz, L. Torresani, and B. Caprile. 2D deformable models for visual speech analysis. In Stork and Hennecke [SH96].
- [CWM96] M. Cohen, R. Walker, and D. Massaro. Perception of synthetic visual speech. In Stork and Hennecke [SH96], pages 153–168.
- [cWR00] J. c. Wojdeł and L. J. M. Rothkrantz. Silence detection and vowel/consonant discrimination in video sequences. In Barlow [Bar00], pages 104–109.
- [DBI⁺94] M. Damhuis, T. Boogaart, C. In 't Veld, M. Versteijlen, W. Schelvis, L. Bos, and L. Boves. Creation and analysis of the Dutch polyphone corpus. In *Proceedings of the International Conference on Spoken Language Processing, ICSLP'94*, pages 1803–1806, Yokohama, Japan, 1994.
- [DLB01] P. Dalsgaard, B. Lindberg, and H. Benner, editors. *Proceedings of Eurospeech 2001 – Scandinavia*, Aalborg, Denmark, September 2001. Kommunik Grafiske Løsninger A/S, Aalborg.
- [DPN02] S. Deligne, G. Potamianos, and C. Neti. Audio-visual speech enhancement with avcdcn (audio-visual codebook dependent cepstral normalization). In Hansen and Pellom [HP02], pages 1449–1452.
- [Elm90] J. L. Elman. Finding structure in time. *Cognitive Science*, 14:179–211, 1990.
- [GFS97] L. Girin, G. Feng, and J. Schwartz. Noisy speech enhancement by fusion of auditory and visual information: a study of vowel transitions. In Kokkinakis et al. [KFD97].
- [GLF⁺93] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, N. L. Dahlgren, and V. Zue. Timit acoustic-phonetic continuous speech corpus. Published on CD-ROM by NIST, 1993. ISBN: 1-58563-019-5.
- [GMZRR00] R. Goecke, J. Millar, A. Zelinsky, and J. Robert-Ribes. Automatic extraction of lip feature points. In *Proceedings of the Australian Conference on Robotics and Automation ACRA 2000*, pages 31–36, Melbourne, Australia, September 2000.

- [GPN02] R. Goecke, G. Potamianos, and C. Neti. Noisy audio feature enhancement using audio-visual speech data. In *Proceedings of International Conference on Acoustics Speech and Signal Processing*, Orlando, Florida, 2002. IEEE.
- [HN89a] R. Hecht-Nielsen. *Neurocomputing*. Adison-Wesley, 1989.
- [HN89b] R. Hecht-Nielsen. *Neurocomputing*, chapter 5, pages 110–163. In Robert Hecht-Nielsen [HN89a], 1989.
- [HP02] J. H. L. Hansen and B. Pellom, editors. *Proceedings of ICSLP 2002*, Denver CO, USA, September 2002. ISCA.
- [HSP96] M. E. Hennecke, D. G. Stork, and K. V. Prasad. Speechreading by humans and machines. In Stork and Hennecke [SH96], chapter Visionary Speech: Looking Ahead to Practical Speechreading Systems, pages 331–349.
- [HWBK01] M. Heckmann, T. Wild, F. Berthommier, and K. Kroschel. Comparing audio- and a-posteriori-probability-based stream confidence measures for audio-visual speech recognition. In Dalsgaard et al. [DLB01], pages 1023–1026.
- [ITF01] K. Iwano, S. Tamura, and S. Furui. Bimodal speech recognition using lip movement measured by optical-flow analysis. In *International Workshop on Hands-Free Speech Communication*, pages 187–190, Kyoto, Japan, April 2001. ISCA.
- [JAAB01] J. Jiang, A. Alwan, E. Auer, and L. Bernstein. Predicting visual consonant perception from physical measures. In Dalsgaard et al. [DLB01], pages 179–182.
- [Jor86] M. I. Jordan. Serial order: A parallel distributed processing approach. Technical Report ICS Report 8604, Institute for Cognitive Science, University of California, 1986.
- [JR02] J. Jackson and J. Russell. Models of speech dynamics in a segmental-hmm recognizer using intermediate linear representations. In Hansen and Pellom [HP02], pages 1253–1256.
- [KFD97] G. Kokkinakis, N. Fakotakis, and E. Dermatas, editors. *Proceedings of ESCA, Eurospeech97*, Rhodes, Greece, 1997. ESCA.
- [KLS02] J. Kim, J. Lee, and K. Shirai. An efficient lip-reading method robust to illumination variations. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E85–A(9):2164–2168, September 2002.
- [KMT01] S. Kshirsagar and N. Magnenat-Thalman. Viseme space for realistic speech animation. In Massaro et al. [MLG01], pages 30–35.

- [Koh95] T. Kohonen. *Self-Organizing Maps*. Springer-Verlag, Berlin, 1995.
- [KS96] D. H. Kil and F. B. Shin. *Pattern Recognition and Prediction with Applications to Signal Characterization*, chapter Feature Extraction and Optimization, pages 73–109. AIP Press, March 1996.
- [Lam94] L. Lamport. *TEX: A Document Preparation System*. Addison-Wesley, 2nd edition, 1994.
- [LDS95] N. Li, S. Dettmer, and M. Shah. Lipreading using eigensequences, 1995.
- [LK01] J. Lee and J. Kim. An efficient lipreading method using the symmetry of lip. In Dalsgaard et al. [DLB01], pages 1019–1022.
- [LMO01] E. Lleida, E. Masgrau, and A. Ortega. Acoustic echo control and noise reduction for cabin car communication. In Dalsgaard et al. [DLB01], pages 1585–1588.
- [LSC00] S. Lucey, S. Sridharan, and V. Chandran. An improvement of automatic speech reading using an intensity to contour stochastic transformation. In Barlow [Bar00], pages 98–103.
- [LSC01] S. Lucey, S. Sridharan, and V. Chandran. An investigation of hmm classifier combination strategies for improved audio-visual speech recognition. In Dalsgaard et al. [DLB01], pages 1185–1188.
- [LSC02] S. Lucey, S. Sridharan, and V. Chandran. A link between cepstral shrinking and the weighted product rule in audio-visual speech recognition. In Hansen and Pellom [HP02], pages 1961–1964.
- [LTB96a] J. Luettin, N. A. Thacker, and S. W. Beet. Active shape models for visual speech feature extraction. In Stork and Hennecke [SH96].
- [LTB96b] J. Luettin, N. A. Thacker, and S. W. Beet. Speechreading using shape and intensity information. In *Proceedings of ICSLP'96*, pages 44–47, 1996.
- [Mas89] D. W. Massaro. A fuzzy logical model of speech perception. In D. Vickers and P. Smith, editors, *Proceedings of XXIV International Congress of Psychology. Human Information Processing: Measures, Mechanisms and Models*, pages 367–379, Amsterdam: North Holland, 1989.
- [Mas99] D. W. Massaro, editor. *Proceedings of AVSP'99: International Conference on Auditory-Visual Speech Processing*, Santa Cruz, 1999. Perceptual Science Laboratory, University of California, Santa Cruz 95064.
- [MGW⁺97] S. McKenna, S. Gong, R. P. Würtz, J. Tanner, and D. Banin. Tracking facial feature points with gabor wavelets and shape models. In J. Bigün, G. Cholle, and G. Borgefors, editors, *Proceedings of the First International Conference on Audio- and Video-based Biometric Person Authentication*, volume 1206, pages 35–42, Crans-Montana, Switzerland, March 1997. Springer LNCS.

- [MH99] J. Movellan and J. Hershey. Using audio-visual synchrony to locate sounds. In *Advances in Neural Information Processing Systems*, volume 12, pages 813–819. Massachusetts Institute of Technology Press, 1999.
- [MLG01] D. W. Massaro, J. Light, and K. Geraci, editors. *Proceedings of AVSP 2001*, Aalborg, Denmark, September 2001. Perceptual Science Laboratory, University of California, Santa Cruz 95064.
- [MM76] H. McGurk and J. MacDonald. Hearing lips and seeing voices. *Nature*, 264:746–748, 1976.
- [MMK⁺99] K. Messer, J. Matas, J. Kittler, J. Luetlin, and G. Maitre. XM2VTSDB: The extended M2VTS database. In *Second International Conference on Audio and Video-based Biometric Person Authentication*, March 1999.
- [MP43] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.
- [MS98] D. W. Massaro and D. G. Stork. Speech recognition and sensory integration. *American Scientist*, 86:236–244, 1998.
- [ORP00] N. M. Oliver, B. Rosario, and A. P. Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843, 2000.
- [Ost97] J. Ostermann. MPEG–4 overview. In Y.-F. Huang and C.-H. Wei, editors, *Circuits and Systems in the Information Age*, pages 119–135. IEEE, 1997.
- [PB02] C. W. Ponton and L. E. Bernstein. Neurocognitive basis for audiovisual speech perception: Evidence from event-related potentials. In Hansen and Pellom [HP02], pages 1697–1700.
- [PC01] A. Podhorski and M. Czepulonis. Helium speech normalisation by codebook mapping. In Dalsgaard et al. [DLB01], pages 1519–1522.
- [PHA01] J. P. Plucienkowski, J. H. L. Hansen, and P. Angkititrakul. Combined front-end signal processing for in-vehicle speech systems. In Dalsgaard et al. [DLB01], pages 1573–1576.
- [PLH00] H. Pan, A.-P. Liang, and T. S. Huang. A new approach to integrate audio and visual features of speech. In *IEEE International Conference on Multimedia and Expo (II)*, pages 1093–1096, 2000.
- [PN01] G. Potamianos and C. Neti. Automatic speechreading of impaired speech. In Massaro et al. [MLG01], pages 177–182.

- [RGBVB97] L. Reveret, F. Garcia, C. Benoît, and E. Vatikiotis-Bateson. An hybrid image processing approach to liptracking independent of head orientation. In Kokkinakis et al. [KFD97], pages 1663–1666.
- [RHW86] D. Rumelhart, G. Hinton, and R. Williams. *Parallel Distributed Processing*, chapter 8, Learning internal representations by error propagation. Massachusetts Institute of Technology Press, Cambridge, MA, 1986.
- [Rip96] B. D. Ripley. *Pattern Recognition and Neural Networks*, chapter Multidimensional scaling, pages 305–311. Cambridge University Press, 1996.
- [SH96] D. G. Stork and M. E. Hennecke, editors. *Speechreading by Humans and Machines*. NATO ASI Series, Series F: Computer and Systems Sciences. Springer Verlag, Berlin, 1996.
- [Sik97] T. Sikora. The MPEG–4 video standard and its potential for future multimedia applications. In *Proc. IEEE ISCAS Conference*, Hongkong, June 1997.
- [Sme95a] P. M. Smeele. *Perceiving speech: Integrating auditory and visual speech*, chapter Cross-linguistic Comparisons in the Integration of Visual and Auditory Speech, pages 15–44. In [Sme95c], 1995.
- [Sme95b] P. M. Smeele. *Perceiving speech: Integrating auditory and visual speech*, chapter Perception of Asynchronous and Conflicting Visual and Auditory Speech, pages 55–76. In [Sme95c], 1995.
- [Sme95c] P. M. Smeele. *Perceiving speech: Integrating auditory and visual speech*. PhD thesis, Delft University of Technology, 1995.
- [SMY97] R. Stiefelhagen, U. Meier, and J. Yang. Real-time lip-tracking for lipreading. In Kokkinakis et al. [KFD97], pages 2007–2010.
- [SP54] W. H. Sumby and I. Pollack. Visual contribution to speech intelligibility in noise. *Journal of the Acoustical Society of America*, 26:212–215, 1954.
- [SP95] T. Starner and A. Pentland. Real-time american sign language recognition from video using hidden markov models. In *SCV95*, page 5B Systems and Applications, 1995.
- [TO99] M. Tekalp and J. Ostermann. Face and 2–D mesh animation in MPEG–4. In *Signal Processing: Image Communication*, 1999.
- [vA92] P. van Alphen. *HMM-based continuous-speech recognition*. PhD thesis, University of Amsterdam, 1992.
- [Vog96] M. Vogt. Fast matching of a dynamic lip model to color video sequences under regular illumination conditions. In Stork and Hennecke [SH96], pages 399–408.

- [VPN99a] M. Visser, M. Poel, and A. Nijholt. Classifying visemes for automatic lipreading. In *Proceedings of TSD'99*, 1999.
- [VPN99b] M. Visser, M. Poel, and A. Nijholt. Classifying visemes for automatic lipreading. In T. Jelinek and E. Nöth, editors, *Proceedings of TSD'99*, pages 349–352, Berlin Heidelberg, 1999. Springer Verlag.
- [vVdHR00] R. J. van Vark, J. K. de Haan, and L. J. M. Rothkrantz. A domain-independent model to improve spelling in a web environment. In *Proceedings of ICSLP 2000*, pages 1081–1084, Beijing, China, 2000.
- [Wig01] P. Wiggers. Hidden markov models for automatic speech recognition (and their multimodal applications). Master's thesis, Delft University of Technology, Delft, The Netherlands, August 2001.
- [WLO⁺95] P. Woodland, C. Leggetter, J. Odell, V. Valtchev, and S. J. Young. The development of the 1994 htk large vocabulary speech recognition system. In *Proceedings of the ARPA Spoken Language Systems Technology Workshop*. ARPA, January 1995.
- [Woj97] J. C. Wojdeł. Recognition of mouth expression in video-sequence. Master's thesis, Technical University of Łódź, Institute of Computer Science, Łódź, Poland, April 1997.
- [WOY94] P. C. Woodland, J. Odell, and S. J. Young. Large vocabulary continuous speech recognition using HTK. In *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing*, pages 125–128, 1994.
- [WR00] J. C. Wojdeł and L. J. M. Rothkrantz. Visually based speech onset/offset detection. In *Proceedings of Euromedia2000*, Antwerp, Belgium, May 2000.
- [WR01a] J. C. Wojdeł and L. J. M. Rothkrantz. Robust video processing for lipreading applications. In *EUROMEDIA'2001*, pages 195–199, Valencia, Spain, 2001.
- [WR01b] J. C. Wojdeł and L. J. M. Rothkrantz. Using aerial and geometric features in automatic lip-reading. In Dalsgaard et al. [DLB01], pages 2463–2466.
- [WR03] P. Wiggers and L. J. M. Rothkrantz. Using station-to-station travel frequencies to improve recognition in a train table dialogue system. In *Text, Speech and Dialogs (TSD) 2003*, Ceske Budejovice, Czech Republic, September 2003.
- [WRS98] J. C. .Wojdeł, L. J. M. Rothkrantz, and P. S. Szczepaniak. Mixed fuzzy-system and artificial neural network approach to the automated recognition of mouth expressions,. In *Proceedings of the 8th International Conference on Artificial Neural Networks*, pages 833–838, Skovde, Sweden, September 1998.

- [WWR99] J. C. Wojdeł, A. Wojdeł, and L. J. M. Rothkrantz. Analysis of facial expressions based on silhouettes. In *ASCI'99 annual conference of Advanced School for Computing and Imaging*, 1999.
- [WWR02a] P. Wiggers, J. C. Wojdeł, and L. J. M. Rothkrantz. Development of a speech recognizer for the dutch language. In *Proceedings of 7th annual scientific conference EUROMEDIA 2002*, pages 133–138, Modena, Italy, April 2002.
- [WWR02b] P. Wiggers, J. C. Wojdeł, and L. J. M. Rothkrantz. Medium vocabulary continuous audio-visual speech recognition. In Hansen and Pellom [HP02], pages 1921–1924.
- [You93] S. J. Young. The HTK hidden Markov model toolkit: Design and philosophy. Technical Report TR.153, Department of Engineering, Cambridge University, UK, 1993.
- [YSFC99] K. Yao, B. E. Shi, P. Fung, and Z. Cao. Liftered forward masking procedure for robust digits recognition. In *Proceedings of European Conference on Speech Communication and Technology*, volume 6, pages 2873–2876, Budapest, Hungary, September 1999.
- [YW96] J. Yang and A. Waibel. A real-time face tracker. In *Proceedings of WACV'96*, pages 142–147, Sarasota, Florida, USA, 1996.

Index

- accentuation, 20
- ADM, *see* perception model, auditory dominance
- AMP, *see* perception model, additive
- ANN, *see* neural network, *see* neural network
- artifacts, 13, 28, 56, 98, 110
- ASR, *see* speech recognition, automatic
- asymmetry, 10
- asynchrony, 14, 99
- backpropagation, 37–39
 - through time, 39
- Baum-Welsh re-estimation, 47–48, 50
- Bayesian rule, 9, 99, 100
- black-box, 34, 55
- BPTT, *see* backpropagation through time
- CCD, 70
- chromacity, 89
- classifier, 79
- coarticulation, 17, 20, 104
- codebooks, 50
- compression, 28, 29, 89
- consonant, 8, 41, 42, 79, 80, 85, 90
- context neurons, 41, 80
- data labeling, 50, 83, 85, 86
- decoding, 23
- dictation, 67
- DUTAVSC, 88, 90, 87–97, 103–105, 111
- DV, 93, 94
- eigenvector, 29
- ERNN, *see* neural network, Elman
- facial animation parameter, 16
- facial definition parameter, 16
- FAP, *see* facial animation parameter
- FDP, *see* facial definition parameter
- feature extraction, 7, 12, 13, 23, 25, 27, 29, 31, 33, 48, 49, 53, 69, 97, 110
 - LGE, *see* lip geometry estimation
 - model-based, 27
 - point tracking, 26–27
 - raw images, 25–26
 - techniques, 23–28
- feature fusion, 97, 102, 105, 105–106
- FFT, *see* Fourier transform
- filtering, 53–56, 64, 110
- FLMP, *see* perception model, fuzzy logical
- Fourier transform, 111
- grammar, 68, 70–73, 75, 92, 111
- Hamming window, 31
- hidden Markov models, 14, 19, 23, 42, 43, 44, 45, 46, 49, 42–50, 68, 71, 100–104, 111, 112
 - Cartesian combined, 101, 102
 - forward pass, 45, 44–45, 46
 - generator, 43, 102
 - multi-stream, 101–104, 106
- Hidden Markov Toolkit, 49, 50, 69, 70, 92, 103, 105, 106
- HMM, *see* hidden Markov models, *see* hidden Markov models
- HSV (color space), 54
- HTK, *see* Hidden Markov Toolkit
- hue filtering, 54, 55
- intensity features, 62, 64, 72, 73, 105, 111

ISO, 16

LGE, *see* lip geometry estimation

linear prediction coding, 26, 31

lip geometry estimation, 27–28, 31, 59,
62, 52–66, 70–76, 80, 84, 103–
105, 110, 111

lip-selective filter, 28, 53, 54, 110

lip-tracking, 13, 29, 65, 89, 109, 110

lipreading, 7, 23, 25, 26, 33, 42, 53,
64–66

by humans, 7–10

by machines, 10–16

continuous, 75, 103, 104, 87–108

limited vocabulary, 68, 70, 67–75,
92, 104, 110

limits, 16–22

processing pipeline, 23, 53, 65

LPC, *see* linear prediction coding

M2VTS, 88, 89

Markov chain, 42

maximal likelihood linear transforma-
tion, 25, 99

McGurk effect, 8–9

Mel frequency cepstral coefficient, 25,
26, 105

MFCC, *see* mel frequency cepstral co-
efficient

misclassification, 22

MLLT, *see* maximal likelihood linear
transformation

MOM, *see* multi-modal overlap mea-
sure

monophone, 104

MPEG, 16

multi-modal overlap measure, 59

neural network, 34, 35, 55, 56, 99, 102,
112

Elman, 41, 41, 41–42, 79, 84, 85

feed forward, 36, 79

Jordan, 39–41, 79, 80, 85

Kohonen, 80

partially recurrent, 34, 36, 39, 39,
34–42

principles, 34–39

time delayed, 39, 79, 85

NN, *see* neural network

noise-removal, 109

nonverbal, 20

overtraining, 80

PAL, 70, 94

parameter tying, 49, 102–104, 106, 112

PCA, *see* principal component analysis

PDF, *see* probability density function

perception model

additive, 9

auditory dominance, 9

fuzzy logical, 10, 99

Perceptron, 36, 37

Person-Independent Feature Space, 62,
110

phoneme, 17, 20–22, 48, 62, 65, 66,
69, 83, 86, 90, 92, 96, 98,
104, 106, 112

phonetically rich sentence, 90, 91, 96

PIFS, *see* Person-Independent Feature
Space

POLYPHONE, 19–21, 75, 88, 90, 92,
105

principal component analysis, 23, 25,
28, 28, 28–34, 57, 72, 73
person adaptive, 59, 62, 66, 73,
105, 110

PRNN, *see* neural network, partially re-
current

probability density function, 71, 106

Gaussian, 33, 69, 71, 106

prompts (recorded set), 90–93, 96

PromptShow, 92–94

quantization, 50

recognition

data driven, 23, 34

generative approach, 23

region of interest, 56

RGB (color space), 55

ROI, *see* region of interest

- Sammon mapping, 76, 77
- SAMPA (notation), 17, 20, 86
- sensory integration, 7, 9, 10, 12, 15, 87, 97, 103, 113
 - early, 10, 12, 97–99, 104
 - intermediate, 12, 97, 100–103, 104
 - late, 10, 12, 97, 99–100, 103, 104
- sigmoid, 36
- signal to noise ratio, 107
- silence, 58, 69, 72, 76, 77, 79, 80, 86
- silence model, 68, 69, 72
- SNNS, *see* Stuttgart Neural Network Simulator
- SNR, *see* signal to noise ratio
- SOM, *see* neural network, Kohonen
- speech corpus, 19, 20, 76, 88, 92, 111
 - audio-visual, 88, 103, 111
- speech enhancement, 12
- speech recognition, 16, 17, 19, 22
 - application-oriented, 90, 92
 - automatic, 88
 - bimodal, 8, 12, 14, 15, 62, 70, 87, 90, 92, 97, 101, 106, 107, 103–107, 111
 - continuous, 67, 68, 75, 80, 87, 100, 103, 106, 111
 - limited vocabulary, 19, 67, 68, 75, 100
- spelling, 2, 90, 92
- Stuttgart Neural Network Simulator, 79
- syllabication, 20
- synchronization, 14, 98, 101, 102

- teacher forcing, 41
- teleconferencing, 12, 16
- texturing, 26
- TIMIT database, 88
- transcriptions, 19, 20, 83, 92, 94

- viseme, 17, 18, 20–22, 57, 66, 77, 79, 83, 104, 106, 109, 110
 - syllable set, 18, 20–22, 112
- visualization, 28, 111
- Viterbi algorithm, 45–47, 101
- vowel, 80
- VSS, *see* viseme syllable set

- word-net, 92
- XM2VTSDB, 88

Curriculum Vitae

Jacek Cyprian Wojdeł was born in Łódź, Poland, on October 14th, 1973. While in high-school, in July 1992, he was a member of Polish team for the International Physics Olympiad in Helsinki, Finland. In the same year he enlisted as a student of the Faculty of Technical Physics, Informatics and Applied Mathematics on Technical University of Łódź, Poland. Since 1993, he studied on basis of individual program under a supervision of Dr. P. S. Szczepaniak from Institute of Computer Science, Technical University of Łódź. His studies in this period were focused on: Artificial Neural Networks, Fuzzy-Systems and Artificial Intelligence. In the academic year 1996/1997, he has done his Master's Thesis at Delft University of Technology under supervision of Dr.Drs. L. J. M. Rothkrantz – Delft University of Technology – and Dr. P. S. Szczepaniak – Technical University of Łódź. He graduated with honors from Technical University of Łódź in April 1997.

From January 1998 to January 2003, he worked as a Ph.D. student at the Knowledge Based Systems Group of the Department of Information Technology and Systems at the Delft University of Technology. This work has been financed by OVR – the Dutch public transport information service company. From January 2002 to January 2003, he took active part in joint research project of Delft University of Technology, University of Nijmegen, and Technische Universiteit Eindhoven: Creating Robustness in Multimodal Interaction.

Samenvatting

Automatic Lipreading in the Dutch Language

door Jacek Cyprian Wojdeł

De belangrijkste aspecten van bimodale spraakverwerking worden in perspectief gezet in Hoofdstuk 2, door middel van een raamwerk ten bevoege van visueel verbeterde spraakverwerkingssystemen.

Omdat er in principe vele methoden bestaan ten bevoege van visuele spraakverwerking wordt de menselijke bimodale spraakwaarneming besproken aan de hand van gepostuleerde voorbeeldmodellen, die de twee modaliteiten (beeld en spraak) samen voegen). Verder biedt het proefschrift kwantitatieve analyse van het inherente verschil in verstaanbaarheid tussen geluid- en beeld-informatie van spraak.

Na deze inleiding in het kennis domein van spraakverwerking worden in Hoofdstuk 3 alle wiskundige bouwstenen van een audio-visuele spraakverwerker beschreven; van elementaire beeldverwerking voor machinaal liplezen tot invoering van toonaangevende herkenningssystemen: data-gestuurd en model-gestuurd. Kunstmatige neurale netwerken worden gebruikt voor uitleg van de basisbeginselen van data-gestuurde klassificatie. Terwijl Hidden Markov modellen de model-gestuurde aanpak illustreren.

Hoofdstuk 4 verwoordt een der belangrijkste bijdragen uit dit promotiewerk. Het beschrijft gedetailleerd de nieuwe aanpak van visuele kenmerk extractie die geschikt is voor liplezen door een spraakverwerkend systeem. Deze nieuwe Lip Geometrie Schattingsmethode (LGS, in het Engels: LGE) stelt ons in staat de gebruikelijke lip-traceringsmethoden te omzeilen.

De LGS wordt toegepast in latere hoofdstukken, maar eerst worden (in dit Hoofdstuk 4) verdere verbeteringen aan de LGS besproken. Bijvoorbeeld: meenemen van intensiteitskenmerken naast geometrische informatie. Nog een nieuw en krachtig concept is de Persoon Onafhankelijke Kenmerkenruimte (POK, in het Engels: PIFS), deze wordt kwalitatief geanalyseerd op basis van opgenomen data. Kwantitatieve verbeteringen zoals verkregen door gebruik van de POK voor liplees toepassingen worden in het volgende hoofdstuk gepresenteerd.

Het experimentele deel van het proefschrift start in Hoofdstuk 5 met het eenvoudigste lip lees probleem: lip lezen van aaneengeregen woorden. Dit hoofdstuk bevat beschrijving van de data zoals verzameld door middel van experimenten met een enkele proefpersoon te zamen met de experimentele resultaten. De experimenten met meerdere personen zijn uitgevoerd op data uit een veel groter audio-visueel corpus, dat ook in dit proefschrift gedocumenteerd wordt. De voordelen van de POK worden belicht.

Verder volgt, in Hoofdstuk 6, een onvermijdelijk pad naar complexere lip lees systemen. De experimenten zoals beschreven in dit deel geven inzicht in de haalbaarheid van LGS voor het verwerken van continue visuele spraaksignalen. De LGS wordt toegepast op verschil in waarneming van spraak versus zwijgen en op onderscheid tussen klinker versus medeklinker. Kunstmatige neurale netwerken worden ingezet als herkenningssarchitectuur voor beide systemen.

Hoofdstuk 7 mankeert een radicale verschuiving van lip lezen naar bimodale spraakherkenning en tegelijkertijd van kunstmatige neurale netwerken naar Hidden Markov modellen. Daarom bevat het niet slechts experimenten, maar begint met beschrijven van het in dit onderzoek gebouwde Delftse Audio-Visuele Spraakcorpus (DUTAVSC). Dit corpus is, naar de mening van de auteur, de op een na belangrijkste bijdrage van dit werk. Het is het eerste vrij verkrijgbare audio-visuele spraak corpus voor de Nederlandse taal en het is in vergelijking met beschikbaar materiaal in andere talen, op zeer hoog niveau.

Verder gaat Hoofdstuk 7 over pure bimodale spraakherkenning. Hedendaagse methodieken voor combineren van gehoor en visuele spraakherkenning wordt besproken, gevolgd door gedetailleerde beschrijving van de ontwikkeling van het herkenningssysteem in dit proefschrift. ten tijde van de formele publicatie (September 2002) was dit systeem de "state of the art in audio-visual speech recognition".

Het is een persoons-onafhankelijke continue spraak herkenning die werkte op middelgrote woordenlijsten en als zodanig is het erg geschikt voor afhandelen van zowel de complexiteit van het spraaksignaal als de toepassing ervan in praktische problemen. De verkregen herkenningresultaten zijn zeer bemoedigend en de visuele wijze draagt aanzienlijk bij aan de betrouwbaarheid van het systeem onder ruisrijke spreekomstandigheden.

De resultaten van de experimenten in dit proefschrift tonen duidelijk de voordelen van bi- en mogelijk multi-modale spraakverwerking. De LGS uitgebreid met intensiteitskenmerken is aanbevelenswaardig als kenmerk-extractiemethode. Dit proefschrift eindigt met aanwijzingen (gebaseerd op gepresenteerde resultaten) ten behoeve van een interessante richting waarin spraakverwerking zou moeten voortborduren, op zoek naar robuustere systemen: meenemen van externe kennis (bijv. fonetisch, contextueel of anderszins) waardoor robuustere implementaties van audio-visuele spraakverwerkingstoepassingen mogelijk wordt.