INTEGRATION OF SPEECH & VIDEO: APPLICATIONS FOR LIP SYNCH: LIP MOVEMENT SYNTHESIS & TIME WARPING

Jon P. Nedel

Submitted to the Department of Electrical and Computer Engineering
in Partial Fulfillment of the Requirements for the Degree of Master of Science at

Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

May, 1999

ABSTRACT

Throughout the past several decades, much research has been done in the area of signal processing. Two of the most popular areas within this field have been applications for speech recognition and image processing. Due to these extended efforts, today there are systems that can accurately recognize and transcribe the daily television news programs that are broadcast to our homes. There are also systems that can accurately locate and track the faces within those same news programs.

Recently, a new field has emerged which focuses on combining the disciplines of speech recognition and image processing in a practical way. This interest has sparked research in a broad range of new application areas, including:

- Enhancement of speech recognition via visual cues
- Animation of interactive talking agents to facilitate human-computer interaction
- Synchronization of audio and video tracks in the film industry
- Correction of lip movement for films dubbed into a foreign language and in video conferencing

This paper will discuss some of the current efforts in integrating speech and video in a practical way. It will briefly discuss some image processing methods for extraction of lip coordinates and lip features from video.

The focus of this investigation, however, is the use of speech recognition and other signal processing techniques in the development of two practical systems: one for lip movement synthesis and the other for lip synchronization.

First will be the use of a speech recognition technique (Viterbi forced alignment) to segment the lip features extracted from video on a phoneme by phoneme basis. Once the video features are segmented, appropriate models can then be created to link the audio and video features together. Effective models can be built on a very limited amount of training data.

Next will be the development and description of a system for the creation of synthetic lip features based on information contained in the speech recognizer output and the models discussed earlier. These features can then be used to automatically generate accurate and believable synthetic lip movements that correspond to any audio speech waveform.

Also, a separate system to automatically synchronize lip motion and voice is under development. This system is based on dynamic time warping techniques on the output of the speech recognizer and the models that relate the audio and video features.

Finally, there will be some discussion about the methods for performance evaluation of such systems, as well as some ideas for future research in this and other areas of multimodal signal processing.

ACKNOWLEDGEMENTS

I would first and foremost like to thank God for His grace and the many wonderful opportunities and blessings He has granted. His support is truly more precious than gold.

I would like to thank my advisor, Richard Stern, for his support, creativity, and patience as we together worked through many of the details of this work. I would also like to thank Radu Jasinschi and everyone at Tektronix for making this research possible. Special thanks, too, to Matthew Siegler and Sam-Joo Doh for their encouragement and willingness to help a new graduate student get his feet off the ground.

Finally, I would like to thank my family and friends for their unending love and support. I dedicate this work to them.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

In the age of information, multimedia applications abound.

We are able to interact with our computers in unconventional ways. Computers are able to transcribe speech from a variety of sources, such as the microphone of a desktop PC or the broadcast news programs on the television. We can therefore speak, rather than type, our documents. This is one result of the many years of hard work in the area of speech recognition by signal processing researchers.

Other such researchers have looked deeply into images and processing the information within. Does the image contain a face? If so, whose face is it? Can we track the face (or a part of the face such as the eyes or lips) as it moves from frame to frame in a video sequence? Research in the area of image processing tackles such problems.

Now that the areas of image processing and speech recognition are well developed, the research can expand to the area of *multimodal signal processing* [1]. What, if anything, can be gained by combining the information from a variety of sources (video images, audio speech, text, etc.) for processing? What practical applications can come from such research?

This chapter will provide a short overview of the practical things that can be accomplished when speech information is coupled with video information for multimedia signal processing. The chapter then closes with an overview of this report.

1.1 Why Combine Speech & Video? Motivation & Possible Applications

There are many practical applications that can result from multimedia signal processing efforts to combine the information contained in the speech waveform and the corresponding video sequence. Once the lips are accurately tracked via image processing algorithms, it is possible to make creative use of that information. One possibility is to enhance speech recognition results by additional features derived from the lip information. These video features are impervious to any noise that typically corrupts the audio waveform, and thus they can be useful in improving the recognition rates in noisy environments.

Another interesting application stems from the movie industry. When film recordings are done outside a studio, the audio tracks are re-recorded to ensure high-quality sound. Although the actors view themselves in the film while repeating their lines, the resulting audio is always slightly out of synch from the video. These slight synchronization anomalies can prove unacceptable to a viewing audience, and therefore they should be corrected. The WORDFIT system developed by P.J. Bloom makes use of the two audio tracks to correct for the alignment errors [2]. Alternatively, it is possible to utilize additional information from the video to automatically synchronize the two.

The animation industry can also make use of such research. If appropriate models linking the audio and video features are developed, computers can be used to automatically generate fully synchronous lip movement animations that would otherwise need to be painstakingly hand-drawn by professional animators. The same technology for lip movement animation, if applied to foreign language speech, can also be used to drive an application to correct the lip movement in a movie dubbed into a foreign language.

There is much hype now about animated interactive "agents" as a means for human-computer interaction. In essence, the user can "converse" with the animated character on the desktop of the

PC to accomplish certain tasks. The agent responds with both speech and action, and thus lip movment animation technology is useful in this application as well.

Although the technology is still far from solving some of the complex problems presented above, it is clear that the research in the area of audio-visual interaction based on speech recognition and image processing algorithms has many practical applications and thus is well warranted.

1.2 Report Overview

We begin this report with a short review of current research in the field of automatic lip synch and the generation of synthetic lip motion. We then discuss our models that link speech and video features together for practical applications. The focus of the report is on two simple but practical applications of lip movement synthesis and lip time warping that have been developed. Noteworthy is the fact that our applications can function with a minimal amount of training data while still producing good results.

Here is a brief outline of this report:

- Chapter 2 is a survey of other research in the area of lip synch with a focus on lip movement synthesis applications.
- Chapter 3 is a survey of the data collection method developed at Tektronix to extract lip coordinates from video images.
- Chapter 4 discusses the model-building techniques we used to link the speech waveform and the visual features.
- Chapter 5 presents a system we developed for lip movement synthesis based on pre-trained models and cues from the audio features.
- Chapter 6 presents a system we developed for lip time warping to automatically synchronize lip motion to an audio track re-recorded at a later time.
- Chapter 7 summarizes the results and presents ideas for future research.

CHAPTER 2: OTHER RESEARCH IN LIP MOVEMENT SYNTHESIS

2.1 Introduction

In the overview "Audio-Visual Integration in Multimodal Communication", Tsuhan Chen describes many different multimedia signal processing applications that focus on the integration of speech processing and image analysis[1]. Included in this overview is a section on audio-to-visual mapping, the real heart of any system for lip movement synthesis. Chen states that there are two basic approaches for audio-to-visual mapping: a linguistic approach based on speech recognition technologies, and a physical approach based on the inherent relationship between audio and visual features.

2.2 Lip Movement Synthesis: The Linguistic Approach

Generally, the linguistic approach is as follows: the speech is first segmented into a phoneme sequence, and the resulting phoneme sequence is mapped to a sequence of visual features. The mapping is performed by first referencing a lookup table of visual features that correspond to each phoneme. The resulting feature sequence must somehow be smoothed and connected to form a natural looking sequence of lip features.

Michael Cohen and Dominic Massaro have done research in the linguistic approach to audio-to-visual mapping at UC-Santa Cruz [3]. For each phoneme (or cluster of phonemes that correspond to a similar lip shape), a set of "target" features is derived. These targets are placed in time at the core of each phoneme segment. Different "dominance" and "blending" functions are then used to smoothly connect the features, simulating natural coarticulation effects. A typical dominance function is the double-sided decaying exponential; a dominance function of this shape is natural because it assumes that the influence of each target on the resulting feature shape decreases exponentially the further away you are in time from the heart of the segment. Different phonemes in different contexts may exhibit a different level of dominance (specified by the maximum amplitude of the dominance function). In essence, the resulting feature string is a weighted average of the target values.

2.3 Lip Movement Synthesis: The Physical Approach

The physical approach seeks out a relationship between visual parameters and speech parameters (e.g. LPC cepstrum). Such a relationship should exist as an extension of the natural dependence of speech sounds on the shape of the vocal tract producing them. The goal is to learn the best approximation for the unknown visual features based on training examples.

Tsuhan Chen and Fu Jie Huang have done research in the physical approach to audio-to-visual mapping using the Gaussian Mixture Model (GMM) together with the Hidden Markov Model (HMM) [4]. In this work, an overall feature vector is defined as the concatenation of the audio features (e.g. LPC cepstrum) and the video features (e.g. lip height and width). The distribution of the overall audio-visual feature vector is modeled using a mixture of Gaussians and trained using the Expectation-Maximization (EM) algorithm. In the testing phase, the estimate for the visual features is simply the expected value of the visual features given the audio features observed for each frame of input audio. This estimate can be obtained using the GMM. The HMM can be used to further enhance this process with context information. For each word in the training set, a 5 state left-right HMM is trained, and a different GMM is created for each state in the HMM. In the testing phase, the HMM acts to segment a given word into its constituent parts, and the estimate for the visual features is now the expected value of the video features given the audio features observed and the current state of the HMM.

2.4 Lip Movement Synthesis: Visualization & Evaluation

Once the audio-to-visual feature mapping is complete, it is necessary to display these synthetic features so that the resulting lip movement synthesis can be evaluated. A popular avenue of research has been in the development of face models that can be parametrically driven. One such model is the CANDIDE model developed by Mikael Rydfalk at the Linköping Image Coding Group in 1987. The face is composed of approximately 100 polygons that are controlled by "action units" that mimic the way human beings are able to alter the shape of their faces [5]. It is possible to convert the video features generated by an audio-to-visual mapping into action unit specifications that drive such a face model for appropriate visual speech animation.

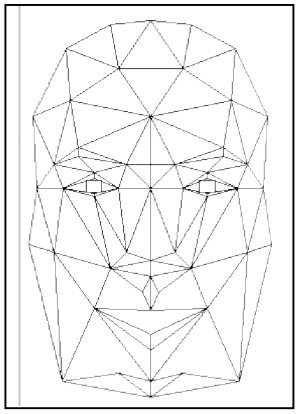

Figure 2.1 The CANDIDE face model

One big challenge to researchers in the area of lip movement synthesis is that there exist no standard metrics by which the resulting animations or synthetic lip features can be evaluated. Metrics based on mean square error of synthetic features and actual features have been proposed [6], but no in-depth studies of error metrics or their correlation to human evaluation of lip movement synthesis have been performed. Eli Yamamoto, Satoshi Nakamura, and Kiyohiro Shikano from the Nara Institute of Science and Technology in Japan have done some experiments with subjective evaluation of synthetic lip animations to test both intelligibility and acceptability [7]. For their intelligibility tests, nonsense Japanese words of the form CVCV were presented to the subjects. They were able to show that the presentation of synthetically generated lips enhanced the intelligibility of the nonsense word. For their acceptability tests, the subjects were presented Japanese sentences 3 words in length. The Mean Opinion Score (MOS) was used to measure the acceptability on a scale of 1 to 5 with category labels bad, poor, fair, good, excellent. The subjects were told that an excellent rating should only be assigned when the lip motion is natural enough to be real human lip movement.

Finally, researchers from Interval Research Corporation have developed an impressive system called "Video Rewrite" [8] which can automatically create video of a person mouthing words he or she did not speak. Speech recognition is first performed to obtain the correct sequence of triphones that compose the speech. Then, the training sequence is searched in the audio domain for the closest triphone sequence that matches the new speech utterance. The process is very much like the modern concatenative speech synthesis systems that tie together small segments of speech from a database to produce synthetic speech. Based on approximately 8 minutes of training data, the Video Rewrite system has produced some very believable video sequences of people "saying" things that they never actually said.

2.5 Lip Movement Synthesis: Our Approach

Our research best fits into the linguistic approach to lip synch applications. We employ Carnegie Mellon's SPHINX speech recognition system to fully analyze the audio speech waveform. We then use this information together with lip features extracted from the video in order to create synthetic lip features which are natural and believable. Similar to Massaro's group, we seek to establish a canonical value for the lip features corresponding to each phoneme. Then, through a series of statistical filtering methods, we attempt to smooth the feature sequence to provide natural transitions from one level to the next. The resulting feature sequences should be as similar to real lip data as possible. In the end, we attempt to do even better. We aggressively scale these features to create lip movement synthesis examples which are in some ways more believable than the data from which the models are derived.

Despite the inherent difficulty in defining an error metric for lip movement synthesis, we also attempt to use a error metric based on mean-square error. We present some sensible modifications to such a metric for better evaluation. However, the search for an accurate and ideal error metric still proves to be one of the most challenging problems to solve.

CHAPTER 3: LIP DATA COLLECTION

This chapter will cover the data collection procedure we used for this research. We include a discussion of the sentences selected for recording and the system developed at Tektronix to extract lip coordinates from the collected video. The chapter closes with an overview of the corpus that was available for research and system development.

3.1 The TIMIT Database: A Phonetically "Rich" Corpus of Sentences

The Spoken Natural Language Processing Group of the National Institute of Standards and Technology (NIST) worked to create a corpus of phonetically "rich" sentences for use in speech recognition research. This database, known as the DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus (or "TIMIT" for short) [9], contains speech and transcription of samples taken from 630 subjects from various parts of the U.S.

The TIMIT sentences are designed to cover a wide range of phonemes in a wide range of contexts. Since the database is audio only, the recordings themselves are not of interest to research in audio-visual interaction. The phonetically "rich" sentences, however, provide a good basis to ensure coverage of all the English phonemes with a very limited amount of training sentences. For this reason, we decided to use TIMIT sentences as the scripts for our audio-visual recordings. Some example sentences are shown in Figure 3.1.

1. The fish began to leap frantically on the surface of the small lake.
2. Artificial intelligence is for real.
3. Only the most accomplished artists obtain popularity.
4. Don't do Charlie's dirty dishes.
5. Barb burned paper and leaves in a big bonfire.

Figure 3.1 Some examples of phonetically rich sentences from the TIMIT database

With as few as 8 TIMIT sentences recorded for our smallest data set, all but 3 of the phonemes used by the CMU SPHINX speech recognition system were seen at least 1 time.

3.2 Tektronix Lip Extraction Procedure

Tektronix and CMU decided to jointly research the area of audio-video interaction beginning in August 1997. Tektronix's responsibilities included data collection and preparation. Radu Jasinschi, engineer of the Textronix Lip Tracker, has been successful in extracting accurate lip coordinates from the video data collected at Tektronix in Portland, Oregon.

The Textronix Lip Tracker is initialized by hand. A human views the first frame of video and selects the pixel coordinates that are to be tracked throughout the video sequence. Based on intensity information, the Lip Tracker is able to hold onto the points as they move throughout the sequence.

Initially, eight points surrounding the outer perimeter of the lips were chosen for tracking. As we progressed through the project, an additional six inner lip coordinates were also tracked. Figure 3.2 shows an example frame of video and the perimeter pixels selected for tracking.

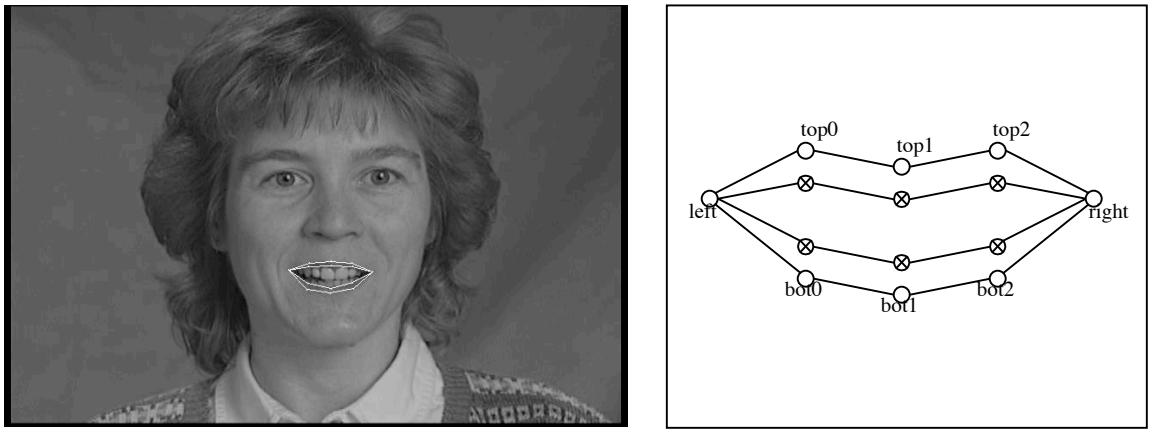

Figure 3.2 Inner & outer lip points selected for tracking by the Tektronix Lip Tracker

The output of the Lip Tracker is a text file containing the x- and y-coordinates for each tracked point on a field by field basis. (The collected video is interlaced; each frame of video consists of two fields. For compression reasons, and in accord with video standards, the first field contains the updated pixel values for the odd-numbered rows of pixels, and the second field contains the updated pixel values for the even-numbered rows.)

Figure 3.3 shows a sequence of video frames with the tracking results overlaid in white.

Figure 3.3 A video sequence tracked with the Tektronix Lip Tracker

The success of the Tektronix Lip Tracker can be seen in this and other example sequences.

3.3 Available Lip Coordinate & Feature Data

All of the data used in this research was collected using the Tektronix Lip Tracker. Table 3.1 details all of the data used for our research.

Speaker	# Sentences	Duration	Data Available	Date Received
Doug	8	40 sec	Outer lip coordinates	August 1997
Jan	9	50 sec	Outer lip coordinates	August 1997
Ted	36	160 sec	Inner & outer lip coordinates	November 1998
Kim	20	100 sec	Inner & outer lip coordinates	November 1998

Table 3.1 Available lip coordinate and feature data

There are two male speakers and two female speakers in the data set. The first two data sets, Doug and Jan, are very limited. Even if all of the data from these sets is used for training, there are some phonemes that are never seen. However, we are still able to develop a believable lip synthesis system trained purely on the data from Doug or from Jan.

CHAPTER 4: LIP DATA ANALYSIS — VISEME FEATURE MODELS

This chapter starts with some basic definitions pertinent to speech recognition and the corresponding standard definitions used in "speech-reading" research. It also discusses the visual lip features (e.g. height and width) we derive from the lip coordinates to serve as an underlying model of the lips. Finally, it details the procedure we use for collecting and building the models that connect the audio features used in speech recognition with the corresponding visual features extracted from the video.

4.1 "Phonemes", "Visemes", & "Viseme Features"

The American Heritage dictionary defines a *phoneme* as "One of the set of the smallest units of speech, as the m of mat and the b of bat in English, that distinguish one utterance or word from another in a given language." Rabiner & Juang remark that "the number of linguistically distinct speech sounds (phonemes) in a language is often a matter of judgment and is not invariant to different linguists [10]."

The Carnegie Mellon University Pronouncing Dictionary (CMU Dict) [11] was developed for use by CMU's SPHINX speech recognition system [12]. The dictionary uses a set of 39 base phonemes to describe over 100,000 words and their North American English pronunciations. These phonemes and example words in which they occur can be seen in Table 3.1. When expanded to include variants for lexical stress and noise phonemes (e.g. +INHALE+, +SMACK+), this set contains a total of 55 distinct phonemes. This augmented set is the set used by the most current version of the SPHINX-III speech recognition system. This phoneme set can also be used as a feature set to appropriately describe and classify the different segments of the speech waveform in time.

Phoneme	Example	Translation	Phoneme	Example	Translation
AA	Odd	AA D	L	lee	L IY
AE	At	AE T	M	me	M IY
AH	Hut	HH AH T	N	knee	N IY
AO	Ought	AO T	NG	ping	P IH NG
AW	Cow	K AW	OW	oat	OW T
AY	Hide	HH AY D	OY	toy	T OY
B	Be	B IY	P	pee	P IY
CH	Cheese	CH IY Z	R	read	R IY D
D	Dee	D IY	S	sea	S IY
DH	Thee	DH IY	SH	she	SH IY
EH	Ed	EH D	T	tea	T IY
ER	Hurt	HH ER T	TH	theta	TH EY T AH
EY	Ate	EY T	UH	hood	HH UH D
F	Fee	F IY	UW	two	T UW
G	Green	G R IY N	V	vee	V IY
HH	He	HH IY	W	we	W IY
IH	It	IH T	Y	yield	Y IY L D
IY	Eat	IY T	Z	zee	Z IY
JH	Gee	JH IY	ZH	seizure	S IY ZH ER
K	Key	K IY			

Table 4.1 Carnegie Mellon's base phoneme set

In the realm of visual speech recognition (or "speech-reading"), the *viseme* is the common visual analog of the phoneme. A viseme is one of the smallest units of lip motion or lip shape that distinguish one utterance or word from another [1]. It is accepted that the number of distinct visemes is far fewer than the number of distinct phonemes. For example, the phonemes B and P are both bilabial consonants produced by a closing and opening of the lips. The only thing that distinguishes a B from a P is whether or not the vocal chords are engaged when the sound is produced. (Linguists define *voiced* consonants as consonants which are produced with vocal chord vibration and *unvoiced*

consonants as those which are produced without.) Visually, the B and P phonemes are identical, and thus the visemes that correspond to B and P should also be the same.

To a greater extent than the phoneme set, the number and type of distinct visemes in English are under dispute by experts in the field. Also, strictly speaking, a viseme can be a sequence of several images, rather than a single still image, which represents the mouth movements [1]. For this study, we avoid the debate by considering only some salient features that can be used to describe the shape or characteristics of the lips at each point in time. These features, which we term *viseme features*, serve as the basis to appropriately describe and classify the various lip shapes extracted from the video information. Some possible members of the set of viseme features include lip height, width, perimeter, and area. Note that there is a different set of viseme features for each stationary image in the video sequence, and we use the information from these stationary images as an approximation for a "strict-sense" viseme.

To summarize, phonemes will serve as the basic descriptor of the speech waveform, and viseme features will serve as the basic descriptor for the visual lip information in this study. The remainder of the chapter will discuss the segmentation and model-building techniques we use to establish a relationship between the two.

4.2 The Hidden Markov Model (HMM) & Acoustic Modeling of Phonemes

The foundation for most modern speech recognition systems is a statistical model known as the *hidden Markov model* (HMM). An illustration of the concept is shown in Figure 4.1. The figure shows the state sequence s(n), a Markov chain, which runs hidden away in a black box to produce the output observation sequence y(n). In theory, the state sequence structure is arbitrary but deterministic. The model is based on the fact that the observation sequence y(n) depends directly on the sequence of states s(n). The HMM is sometimes referred to as a "doubly random" process since the state sequence s(n) is random, and the relationship between y(n) and s(n) is also random.

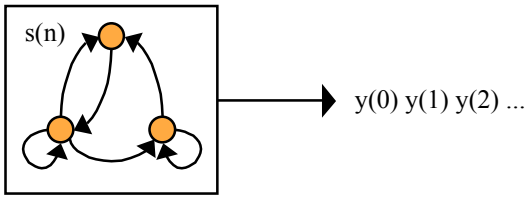

Figure 4.1 A hidden Markov model (HMM)

In speech recognition, the HMM is used as an acoustic model for the phonemes. The observation sequence is a frequency transform of the speech waveform itself. The hidden part of the model can be thought of as the word or phoneme sequence that generated the speech waveform we hear.

Since phonemes are rendered differently in different contexts, it is typical to have several HMMs that describe each phoneme depending on the phonemes that precede and follow it. The basic unit used for phonetic modeling in SPHINX-III and many speech recognition systems is the *triphone*. A triphone is a phoneme in the context of the phoneme that immediately precedes it and the phoneme that immediately follows it. (There are 55^3 possible triphones when using a phoneme set of 55 phonemes. However, a separate HMM is not necessary or practical for each of the 55^3 triphones since many are similar and can be clustered [13].)

In designing HMMs, there is one difficulty: there is no good way to estimate both the transition structure itself and the statistical parameters underlying that structure. In practice, we use our knowledge of the situation and our intuition to design the state sequence structure of the HMM [13]. The structure for SPHNIX-III acoustic models is shown in Figure 4.2. This structure is known as a

left-right HMM or a *Bakis model*. As time increases, the state index of this structure either stays the same or increases monotonically. All of the states of the model are transient; once the model leaves that state, it will never return. The left-right model is highly suited to speech and other signals in which the properties of the signal change successively over time. Each state in the model may be repeated an arbitrary number of times, and it is possible to skip past each of the inner states. The motivation behind the possible repeats and skips is due to variation in speech rates from speaker to speaker and situation to situation.

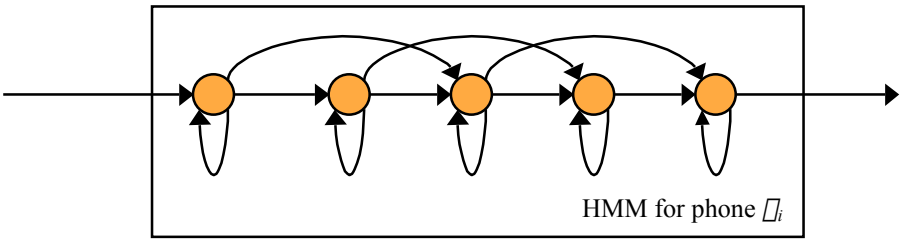

Figure 4.2 HMM structure for acoustic modeling of phonemes in SPHINX-III

Once the HMM structure is determined, the statistics that govern the structure must also be determined. Traditionally, there are two sets of probability distributions needed. First is the probability of transitions between the states p(s´|s). Also needed is an output probability distribution q(y|s, s´) associated with the transition from state s to state s´ [13]. These distributions are trained from many samples of speech to produce a model that is best in the maximum likelihood sense. Finally, the probability of observing the sequence y(0), y(1), … , y(N-1) is

$$P(y(0), y(1), \ldots, y(N-1)) = \sum \prod_{i=1}^{N-1} p(s(i) \mid s(i-1))\, q(y(i) \mid s(i-1), s(i))$$

The sum in the equation above is taken over all possible s(i) when i ranges from 0 to N-1. Also, it is implicitly assumed that the initial state is the leftmost state with probability 1.

4.3 Viterbi Forced Alignment Algorithm & Segmentation

We use the *Viterbi forced alignment algorithm* to segment the audio waveform along the time axis into its constituent phonemes. The algorithm assumes the words contained in the speech signal are known. Given this information and a phonetic dictionary such as CMU Dict, an HMM of an entire word or sentence is built by concatenating the acoustic HMMs for the phonemes that make up the utterance. Usually, we place an optional HMM for silence between each of the words in the sequence.

Since the phoneme HMMs are each left-right structures, a concatenation of such models forms a connected graph that must be traversed over the course of the acoustic observation sequence. The *Viterbi algorithm* is an efficient way to find the most likely sequence of states through the graph. The algorithm starts at the initial state. Then, it looks at all possible following states and keeps only the one most likely path into each of the possible following states. The remaining paths and scores are pruned away at each step of the algorithm. This process is continued until the final stage is reached. The resulting path is the most likely sequence of states that produced the output seen [13].

The Viterbi forced alignment algorithm makes use of the Viterbi algorithm [13] described above to segment the audio into phonemes. The algorithm is as follows:

- Concatenate the elementary HMMs that correspond to the word sequence in the transcript
- Use the Viterbi algorithm to find the most likely sequence of state transitions s(n) that produced the observed speech signal y(n)

- Temporal boundaries occur when a transition is made from one phoneme's HMM to the next

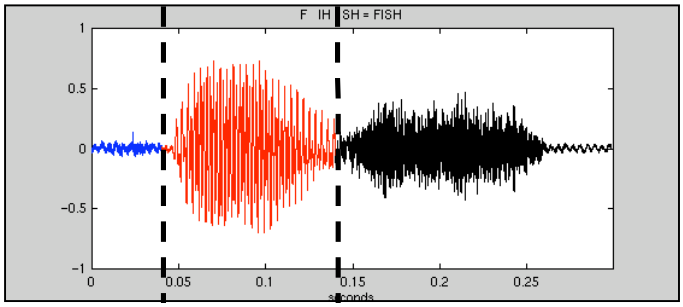

Figure 4.3 Viterbi forced alignment segmentation of "fish" into phonemes F-IH-SH

Figure 4.3 shows the result of running SPHINX-III in forced alignment mode on a very small sample of speech from the Tektronix speaker Jan. In the figure, the word "fish" has been properly segmented into its constituent phonemes F, IH, and SH.

4.4 Combination of Audio & Video: Building Viseme Feature Models

We create our viseme feature model to establish a simple relationship between the audio features (phonemes) and the video features (viseme features). For each phoneme in the phoneme set, our model contains the value of the mean and variance of each of the viseme features. For example, if the chosen viseme features were height and width, our model would contain 4 statistics for each phoneme: the mean height and mean width of all the lip coordinates that correspond to the phoneme, as well as the variance of the observations.

The process we use to calculating a viseme feature model for a particular speaker is as follows:

- We use SPHINX-III in forced alignment mode to segment the audio data for training into its constituent phonemes. This phoneme classification is done for each 10ms window of speech.
- If necessary, we resample the lip coordinate data for training to a sampling rate of 100Hz, i.e. one video sample for every 10ms period.
- We then use the phoneme labels and information from the forced alignment to label each sample of the lip coordinates by phoneme. (Figure 4.4 is an illustration of the result of this step.)
- For each phoneme in the set, we calculate the sample mean and unbiased sample variance of each viseme feature over every lip observation labeled with that phoneme.

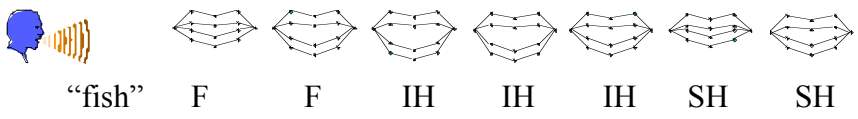

Figure 4.4 An illustration of the labeling of lip coordinate data by phoneme

The results of these calculations are stored in a text file for use by our applications.

4.5 Possible Model Refinements

This closing section discusses some possible refinements to the models for increased flexibility and accuracy. The effectiveness of each model type, however, is closely tied to the application in which it is used. Later chapters will include more discussion on the effectiveness of different model types within various application frameworks.

4.5.1 Context Dependence

In accord with the fact that the sound produced for a phoneme changes depending on the context in which it is found, the viseme features observed for a given phoneme also have different characteristics based on the context in which they are found. As described earlier, speech recognition systems make use of the triphone to account for the contextual effects on the observed phonemes.

We have considered the possibility of extending this idea to the viseme feature model. The new process would be to collect separate viseme feature statistics for each of the possible triphones or clusters of similar triphones. However, since the CMU phoneme set consists of 55 phonemes, there are 55^3 possible triphones. This exponential increase in the number of objects to model requires an exponential increase in the amount of training data. Even with a phonetically rich corpus of training sentences, it is difficult to cover all of the possible phonemes. In our most extensive training set, only 0.74% of the total possible triphones is observed at least once. With data sets this small (on the order of 2 or 3 minutes of speech), it is impossible to construct meaningful context-dependent models.

We are confident that given a large corpus of audio and corresponding lip coordinates for training, context-dependent viseme feature models would produce a highly accurate correspondence between audio and visual features. However, we decided to look for other alternatives due to the limited amount of data available in this experiment.

4.5.2 "Targeting"

We observed one potential problem in the modeling procedure: the basic model-building scheme includes the transition frames from one phoneme to another in the calculations. For example, the phoneme M between two vowels is produced by a mouth shape that starts open, closes down, and then opens again. The height feature of the M will thus start high, proceed to a local minimum when the mouth is fully closed, and then rise again when the mouth reopens. If the M lasts for 7 frames, there may be only 1 frame in the middle where the height feature is at the local minimum, i.e. where the mouth is closed. If we averaging the height feature over all 7 frames, we produce a model whose height feature is well above that local minimum. In lip synthesis applications, the mouth closure is the requisite characteristic of the M, and mouths that do not fully close are perceived as an error.

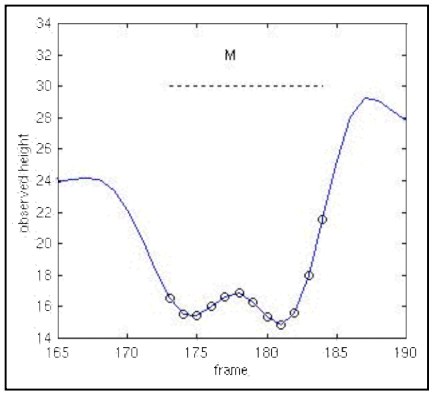

Figure 4.5 An example where averaging over all frames is inappropriate for modeling the height value for M

We can think of lip motion as a process by which the brain establishes a "target" point towards which the lips move when a desired phoneme is formed. For bilabial consonants such as B/P/M,

this target would be a full closure of the lips, and for vowels like AA, this target would be a widened opening of the lips. If we predetermine which direction a particular feature is headed for each of the phonemes, we can create "target" models which select a particular frame or range of frames within the entire duration of a phoneme for modeling purposes.

Practically, it may be difficult to decide for sure where the targets lie for different features and different phonemes. One simplified way that we have built targeting models is to assume that only the core frames of the phonemes are valid. In this case, a certain percentage of the frames at the beginning and ending of a phoneme are disregarded since they are affected by the preceding and following phonemes. The resulting viseme feature models indeed contain feature values that span a wider range of values, corresponding to wider lip openings and tighter lip closings.

Alternatively, we have formed target models by first assuming that all of the consonants are heading towards a local minimum ("closing") and that all of the vowels are heading towards a local maximum ("opening"). Thus every time a phoneme is observed, we select only the features corresponding to one frame and the discard the remnant. For consonants, we keep the frame where the lip features are at the minimum value (over that particular phoneme instance), and for vowels, we keep the frame where the lip features are at the maximum value. This choice naturally applies well for the height viseme features. But we have also seen improvements in synthetic width features that correspond to consonants like W where the width should head towards a local minimum before opening for the following vowel sound.

So continuing the above example, for an M, rather than average all 7 frames of the observed M into the model for M, we could keep only the 3 frames at the "core" of the M and disregard the transition frames. Alternatively, we could take the height value at the local minimum and disregard the rest. The targeting models produced in this fashion are more suited for applications such as lip movement synthesis where openings and closures are crucial.

For both the lip motion synthesis and lip time warping applications, we have chosen to use the targeting feature models built using one feature value (taken at the local maximum/minimum points) per phoneme observance. As we later show, these models produce very clean lip motion animations and the best time alignments of any of the models presented here. Also, there is no problem estimating the targeting models even with very limited training data sets.

4.5.3 Speaker Independence

Although lip shapes and sizes vary from speaker to speaker, there are many trends that are consistent amongst different speakers. For example, everyone closes their mouths to produce the bilabial consonants B/P. We endeavor, therefore, to create a bootstrap model for a new speaker based on the model for another speaker. We accomplish this by a simple normalization of the feature statistics. Our procedure is as follows:

- Calculate the overall mean value of each feature for both the new speaker and the previously modeled speaker. These means should be calculated over the entire training set without regard to phoneme label.
- Calculate the normalization factor by dividing the overall mean for the new speaker by the overall mean of the modeled speaker.
- Multiply the mean feature values of the previously modeled speaker by the normalization factor to get mean feature values for the new speaker.

As an example, assume the overall mean height for the new speaker is given by μ_{new}, and the overall mean height for the previously modeled speaker is given by $\mu_{previous}$. Also assume $H\_AA_{previous}$ is the

mean height feature for the phoneme AA in the original model. The value used to model the new speaker's mean height feature for this phoneme is given by $H\_AA_{new} = (\mu_{new} / \mu_{previous}) H\_AA_{previous}$.

The normalization factor ($\mu_{new} / \mu_{previous}$) gives us a rough way to compensate for the possible differences in lip shape or size. However, it will not compensate for any differences in intensity or degree of motion that exist from one speaker to another. For example, it is not a good idea to use the model of someone who "mumbles" as a starting point to bootstrap a model for someone with a wide range of mouth motion. This method should be very successful if used to model the same speaker when the distance from the camera to the speaker changes.

4.6 Future Research: Clustering Viseme Feature Models

Since it is widely agreed that the number of distinct phonemes is much greater than the number of distinct visemes, having a different viseme feature model for each of the 55 phonemes employed by the SPHINX-III system should not be necessary. There are two possible ways to partition the phonemes into visually equivalent groups: linguistically or automatically. Separate viseme feature models can then be built for each of the groups rather than for each individual phoneme.

The first way to cluster the phonemes is via linguistic knowledge. For example, it makes sense to group together phonemes with identical points of articulation but differences in voicing. Since the only difference between such phonemes is whether or not the vocal chords are engaged, these phonemes should be visually indistinguishable. The phoneme pairs B/P, G/K, and D/T are three such pairs of voiced and unvoiced counterparts.

Alternatively, it is possible to make use of various automatic clustering algorithms on the data themselves to produce the phoneme groups. Agglomerative clustering approaches have been used to group the phonemes based on the height and width viseme features, and it has been shown that the resulting groupings have and an increased between-class variance. The result of automatically grouping the phonemes in this fashion produces classes which are statistically more stable and easily distinguishable.

At this point, we have completed only the preliminary stage of research with clustering viseme feature models. We have created the both the linguistically and automatically generated clusters of phonemes. Future research will include actual clustered viseme feature model generation and testing in both our lip motion synthesis and our lip time warping systems.

CHAPTER 5: LIP MOVEMENT SYNTHESIS

This chapter discusses the first of our applications that makes use of the viseme feature models discussed in the previous chapter for a practical purpose. The problem is to develop synthetic lip movements for a new sentence recorded by the modeled speaker or by another speaker. These lip movements should be synchronized to the audio and move in a believable way. This chapter will discuss our system for producing synthetic lip movements using speech recognition together with viseme feature models. It will discuss the appropriate choice of viseme features and model type, as well as a simple method for evaluating the effectiveness of our system. It will close with other possibilities for further system development and enhancement.

5.1 Goal: Generate Synthetic Features to Match Observed Features

The problem of lip movement synthesis using viseme feature models can be broken into two parts. The first step is to generate synthetic features that are natural and similar to features that would be observed had the speaker's lips been recorded when the audio recording was made. Once appropriate synthetic features have been generated, there then must be a way to map these features into an animated lip shape. The features can be used to drive a face model such as CANDIDE [5] (Figure 2.1), or they can be systematically mapped from features to lip coordinates which exhibit those features.

The first part of the system is the core to the solution of the problem. Our goal is to develop a system which, when trained and implemented properly, will produce synthetic features that match natural features extracted from real lip coordinate data. This formulation of the problem is flexible in that it is independent of the number or type of viseme features used to model the lips. Also, it is extensible for the addition of more features in the future.

5.1.1 Performance Evaluation Metric

One difficult challenge faced when tackling the problem of lip movement synthesis is that there are no established guidelines for evaluating the performance success or failure of the system. One possibility is to develop a set of perception-based tests in which synthetic lip movements are viewed and scored by a trained human audience. It is difficult to ensure that perception-based evaluations are fair and unbiased, and it is difficult to ensure that different evaluators are judging consistently. Also, such evaluations are costly and time-consuming.

In the interest of expediency, we employ a mathematical measurement to evaluate the performance of the lip movement synthesis engine. Ideally, the measure must be accurate in assigning higher scores to synthesis examples which humans would score high and lower scores to those which humans would score low. Since the goal is to create synthetic features that match real, observable features, it makes sense to choose a mathematical measure that is based on the mean-square error between the actual lip features and the synthetic lip features for the same utterance.

However, the measure must also be independent of possible variations in speaker, lip size, range of motion, distance from speaker to camera, etc. We propose a normalization of the standard mean-square error by the variance of the original signal compensates for such differences. (The variance is used as the normalization rather than the energy of the signal because the features in general are not zero-mean. The variance, therefore, is a good measure of the relative amount of motion or "energy" in the original signal.) Assuming $f(n)$ is the observed feature sequence and $\hat{f}(n)$ is the synthetic feature sequence for the same utterance, the normalized mean-square error (NMSE) between the two is given by the following equation:

$$NMSE = \frac{\sum \left(f(n) - \hat{f}(n)\right)^2}{\sum \left(f(n) - \bar{f}\right)^2}$$

We will use the NMSE to evaluate our lip movement synthesis results.

Based on observation and experience with many lip movement synthesis examples, we attempted to make a refinement to the performance metric by adding a step before the NMSE is calculated. At times, there is a slight temporal offset between a local maximum/minimum of the observed features and of the synthetic features. Although this offset can be clearly seen on a plot of the features themselves (as shown in Figure 5.1), this slight delay does not have a noticeable effect on the synthesized lip movements that are produced. However, the delay acts to inflate the NMSE score assigned to the resulting lip animation. Thus, we decided to warp the synthetic features in time before calculating the NMSE. (This is done via the Dynamic Time Warping algorithm to be discussed in detail in Section 6.1.) The warping algorithm makes slight shifts to better align the "peaks and valleys" of the feature curves. The resulting score should theoretically better correlate with a human being's rating of the synthesis.

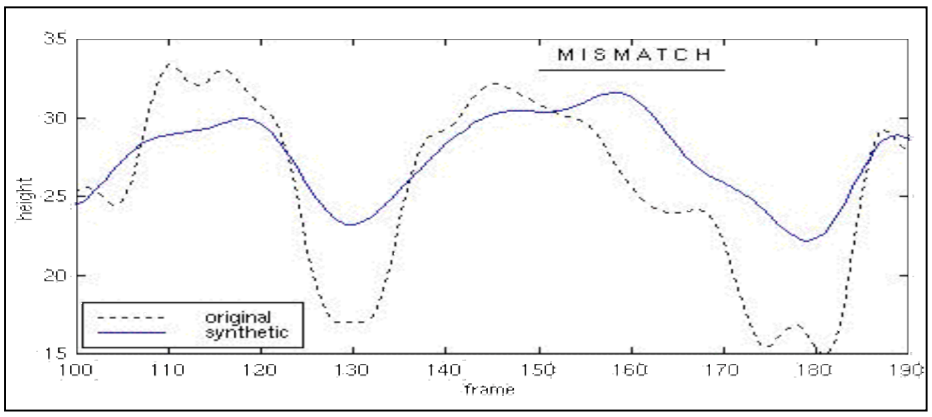

Figure 5.1 Slight timing mismatch when synthetic height falls 10 frames (~0.1s) after the observed height. The synthesis still looks reasonable, but the NMSE is increased by such a mismatch.

Interesting to note is that the DTW algorithm allows us to limit the maximum amount that a peak or valley can be moved. If this parameter is adjusted properly, we can ensure that the DTW-NMSE scores are not artificially inflated beyond what they should be.

5.1.2 Appropriate Viseme Feature Set

The choice of viseme features depends on the specifics of the lip movement synthesis. The simplest set required for synthesis consists of the height and width of the outer lips. With these two features, it is possible to derive synthetic lip movements that are believable. These features are sufficient enough to drive a face model or a morphing system used for lip dubbing. This is also the minimum set sufficient to derive coordinates for an arbitrary, predefined lip shape. For development and evaluation purposed, we focus on the accuracy of these two features.

Given the inner and outer lip coordinates provided by later data collection efforts of Tektronix, we decided to expand the feature set to include multiple lip heights and widths to capture information from both the inner and outer edges of the lips. Figure5.2 illustrates both the basic and expanded viseme feature sets used for lip movement synthesis. These additional features allow for synthetic

examples that exhibit more intricate lip motion. The resulting animations are more precise and believable. The inner lip height features provide some additional information about lip closure and allow for more aggressive post-processing which ensures that the synthetic lip animations close fully when appropriate. Section 5.5 contains a more detailed look at the lip synthesis results. Further experimentation is still needed to determine how much the additional features help the realism of the resulting lip motion animations.

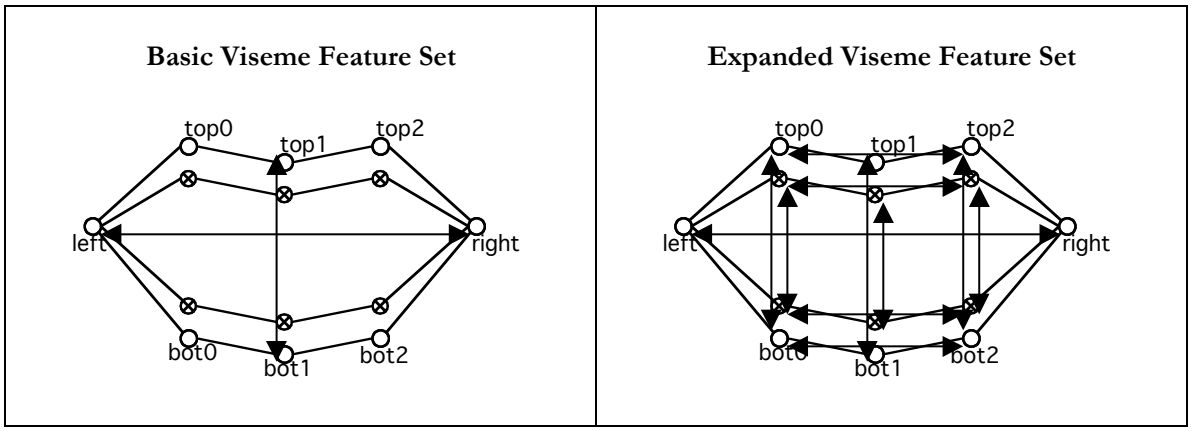

Figure 5.2 Heights and widths used as viseme feature sets for lip synthesis

Again, the basic viseme feature set contains 2 features, height and width. The expanded set contains a total of 11 features, 6 heights and 5 widths.

5.2 System Description

An overview of our approach to lip movement synthesis is shown in Figure 5.3.

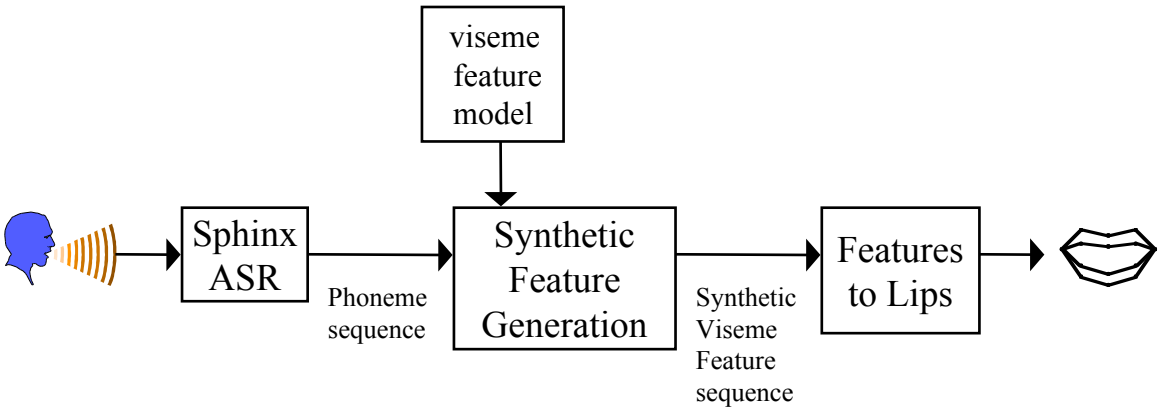

Figure 5.3 Block diagram overview for the lip movement synthesis system

In the first stage of the lip movement synthesis process, we recognize the recorded speech. For this recognition either SPHINX-II or SPHINX-III is used in allphone recognition mode to produce the most likely sequence of phonemes that compose the speech. This recognition can be done either in forced alignment mode if a transcript of the speech is available or in "blind" mode if a transcript is not available. Using either SPHINX-II or SPHINX-III, the forced alignment recognition is more accurate than "blind" recognition on the same speech. For "blind" recognition, the SPHINX-II system is faster but less accurate than the SPHINX-III system. This is due to the fact that the

SPHINX-II system performs a search over the space of possible isolated phonemes while SPHINX-III performs a search over the larger space of possible triphones. [Reference? Ravi]

In the second stage, the phoneme sequence from the audio is used along with a viseme feature model to create a sequence of synthetic viseme features that correspond to the speech. We accomplished this by initializing the features with the values contained in the model and then filtering and scaling the feature sequences to produce a natural-looking motion. Section 5.3 discusses this stage of the synthesis process in detail.

The final stage for lip animation is a mapping from synthetic features to lip coordinates. In most cases, we employ a mapping to an arbitrary lip shape that exhibits the features observed. This stage is discussed in Section 5.4. Also, this stage may be replaced by a face model or another application which uses features to drive the lip motion.

5.3 Synthetic Feature Generation

A block diagram of the synthetic feature generation stage of our lip movement synthesis is shown in Figure 5.4. The input to the process is the phoneme sequence output from the recognizer (1 phoneme per 10ms frame of speech) and the viseme feature model to be used for synthesis. The first step is to initialize a synthetic feature sequence that is the same length as the phoneme sequence. We set the initial feature values for the synthetic sequence to the corresponding sample mean values stored in the viseme feature model. The result is a rough, square-wave approximation of the desired synthetic feature sequence.

It is then necessary to perform some filtering and post-processing to ensure that the synthetic feature sequence is natural and smooth with dynamics that closely resemble the dynamics of the actual feature data. We accomplish this by passing the features first through a Wiener-Hopf filter which has been trained on the same training set used to train the viseme feature model. Then, we smooth the result with a lowpass filter and scale the features to ensure closure and opening of the synthetic lip animations. The relevant theory and processing steps are described in more detail in Sections 5.4 and 5.5.

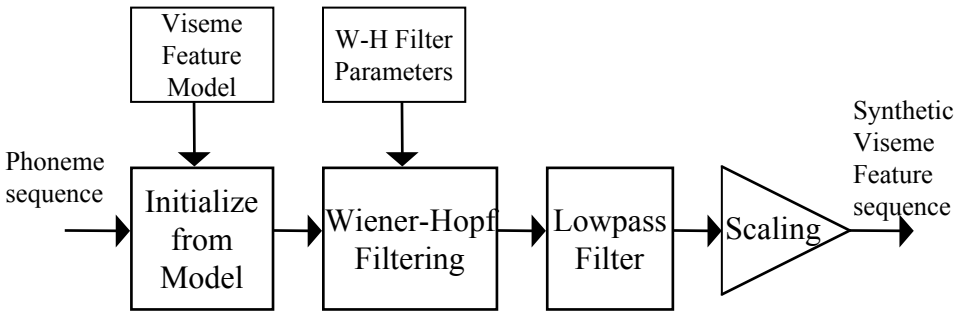

Figure 5.4 Synthetic feature generation block diagram

Our motivation for this choice of processing stages is as follows: The Wiener-Hopf filter should act to expand the dynamic range of features and create more natural looking synthetic lip movements. The lowpass filter is then used to remove any high frequency artifacts incurred as the order of the Wiener-Hopf filter increases. In the end, the Wiener-Hopf filter and the lowpass filter together work to produce a smooth sequence of natural-looking lip features. Nevertheless, the Wiener-Hopf filter

does not correct the dynamic range of motion sufficiently enough to ensure complete lip closures and openings. Thus, the final stage of feature scaling is designed to tackle this problem specifically.

5.3.1 Wiener-Hopf Filtering

Wiener-Hopf filtering is a statistically driven technique for creating an estimate that is as close as possible (in the mean-square sense) to a desired signal based on a fixed number of samples taken from an observed sequence. In the situation of synthetic lip features, we seek to estimate the natural lip features from the observed "square wave" sequence generated by model lookup in the first step of our lip synthesis process.

Mathematically, we develop the Wiener-Hopf filter as follows: Define x[n], the "observation sequence", to be the square wave synthetic features generated by model-lookup. Define d[n], the "desired sequence", to be the real feature string extracted from the data. The goal is to construct an estimate of the desired sequence d[n] based on N+1 observations (x[n], x[n-1], … , x[n-N]) of the square wave synthetic sequence, where N is the order of the Wiener-Hopf filter. The estimate is constructed as a filter with impulse response h[n]. The result is given by the following equation:

$$\text{Estimate } \hat{d}[n] = h[0]x[n] + h[1]x[n-1] + \cdots + h[N]x[n-N] = x[n] * h[n]$$

The goal is to choose h[n] so that the error $d[n] - \hat{d}[n]$ is minimized. The orthogonality principle implies that the solution must satisfy the following equation:

$$R_{xx}h = R_{xd} \quad \text{where } R_{xx} \text{ is the autocorrelation matrix of } x[n]$$
$$R_{xd} \text{ is the cross-correlation of } x[n] \text{ and } d[n]$$

Our filter is trained from a sequence of features from real data and the corresponding sequence of synthetic features derived from the viseme feature model. The training sentences from real data are concatenated to form d[n]. The corresponding square wave synthetic features are concatenated to form x[n]. We estimate R_{xx} and R_{xd} are directly from the concatenated sequences, and we derive the filter coefficients h[n] by solving the equation above.

For practical purposes, we need to determine an appropriate choice of filter order N to complete our design of this stage. The experimentation leading to this choice will be discussed in section 5.5.

5.3.2 Post-processing: Lowpass Filter and Feature Scaling

Following the Wiener-Hopf filtering stage are two additional steps to ensure the resulting viseme feature sequences are smooth and that the dynamic range of motion covered by the features is sufficient.

We first use a lowpass filter to smooth out any high frequency artifacts that occur. These artifacts occur because the Wiener-Hopf filter is driven by a square wave input with high frequency transitions from one phoneme to the next. We observed that this anomaly becomes more prominent as the order of the Wiener-Hopf filter increases.

Our smoothing filter is an equiripple FIR lowpass filter with an order of 108, a stopband frequency of 0.2π rad/sec, and a stopband attenuation of 40dB. We generated the fixed filter coefficients using

MATLAB's implementation of the Parks-McClellan algorithm. The filter exhibits linear phase, and the corresponding delay is half the order of the filter, i.e. 54 samples.

After the features have been filtered, the final step is to scale the features and thus enhance the resulting lip motion. Figure 5.5 (a) shows a histogram of the inner height feature observed in the training data for Ted. Figure 5.5 (b) shows a histogram of the corresponding synthetic feature after the filtering stage of the synthesis process. We observe that the resulting synthetic features span a much narrower range than the same features in the training data. This implies that the resulting synthetic lip movements consistently fall short of the extreme opening and closing points. As explained in Section 4.5.2, this is due to the fact that the models are built by an averaging over all available frames. Even when targeting models are used in attempt to compensate, the extreme points are rarely achieved, and therefore, a scaling is necessary to produce high-quality synthesis with clear openings and closings.

We devised two possible methods for scaling the lip features to enhance their dynamic range:

- Linear transformation based on constraints
- Percentile scaling based on the feature distributions

The linear transformation technique is the one we make use of in our system. The percentile scaling is presented later as a possible approach for future experiments.

We use a linear transformation of the following to perform the feature scaling:

$$f' = af + b$$

In this formula, f' is the newly scaled feature value, and f is the synthetic feature value coming out of the LPF stage of the synthesis process. The constraints used to determine the scaling factor a and the linear shift b are chosen from the following list:

1. $\text{mean}(f') = \text{mean}(f_{obs})$
2. $\text{var}(f') = \text{var}(f_{obs})$
3. $\max(f') = \max(f_{obs})$
4. $\min(f') = \min(f_{obs})$

In these equations, f_{obs} is the observed feature sequence from the training data. Constraint 1 ensures that the mean of the scaled feature is the same as that of the observed feature, constraint 2 ensures that the variance of the scaled feature is the same as that of the observed feature, etc.

Together, constraints 1 and 2 ensure that the scaled synthetic lip features have the same average value and excursion as the observed features in the data. We calculate the means and the variances over all observations. The resulting parameters are given by the following equations:

$$a = \sqrt{\frac{\text{var}(f_{obs})}{\text{var}(f)}}$$

$$b = \text{mean}(f_{obs}) - a \cdot \text{mean}(f)$$

When we use constraints 3 and 4 in conjunction, we ensure that the desired maximum and minimum values are achieved. Since the scaling process is designed to produce synthetic lip movements that close down and open wide, it makes sense to constrain the parameters in this fashion. The values $\max(f_{obs})$ and $\min(f_{obs})$ do not refer to the global maximum or minimum over the entire training set. Rather, for each utterance in the training set, we note the maximum and minimum values of each

feature. We then average these maximum and minimum observations to produce $\max(f_{obs})$ and $\min(f_{obs})$. The maximum and minimum constraints give rise to the following linear transformation parameters:

$$a = \frac{\max(f_{obs}) - \min(f_{obs})}{\max(f) - \min(f)}$$

$$b = \max(f_{obs}) - a \cdot \max(f) = \min(f_{obs}) - a \cdot \min(f)$$

Figure 5.5 (c) shows the resulting inner lip height feature after scaling with the minimum/maximum constraints. Note that the scaled synthetic features span the same range as the observed features with approximately the same frequency. Thus, the scaling is successful.

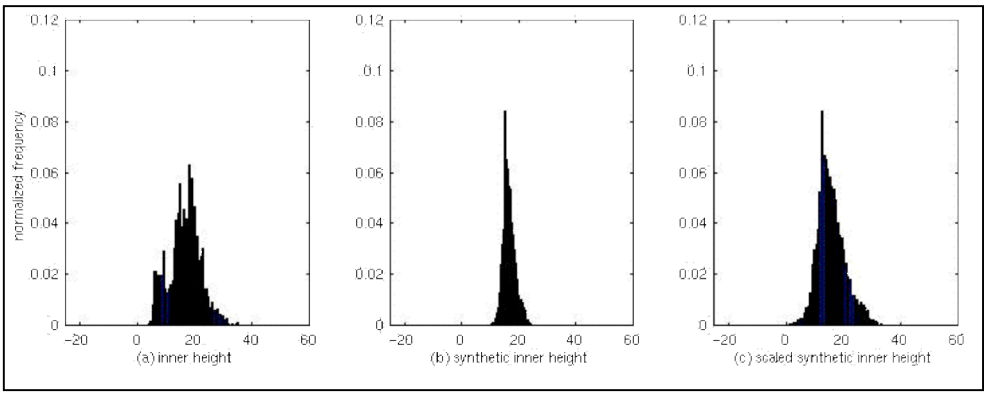

Figure 5.5 Observed inner height features (a) and the corresponding synthetic features before (b) and after (c) scaling

The linear transformation scaling is good when the distribution to be scaled has approximately the same shape as the reference from the training data. This is often the case for synthetic lip features. However, if the distribution shapes are quite different, we propose a scaling based on the distributions themselves. This "percentile scaling" is presented here for reference. It has yet to be implemented, but the resulting scaled distributions achieved using this method should more exactly match the original distributions seen in the data.

The percentile scaling approach is as follows: For a given synthetic feature value f from the filtering stage of the synthesis process, numerically calculate the cumulative distribution up to f on the histogram for the synthetic feature. Choose a corresponding scaled feature value $\boxed{\times}$ so that the cumulative distribution up to $\boxed{\times}$ on the histogram for the observed feature is the same as the cumulative distribution up to f on the histogram for the synthetic feature. In this fashion, the entire synthetic feature distribution (e.g. Figure 5.5 (b)) is warped to match the actual distribution in the data (e.g. Figure 5.5 (a)). The mapping from f to $\boxed{\times}$ can be computed in advance and a lookup table can be employed for rapid synthesis.

We have produced many synthesis examples using the linear transformation approach with very good results. This approach has proven to compensate well for synthesis examples that otherwise do not move adequately enough at all. We have yet to experiment with the percentile scaling approach. We present it as an idea for future experiments and expect that it produces lip motions which very accurately cover the same range of motions as those in the original training data.

5.4 Mapping Synthetic Features to Lip Coordinates

The main part of the synthesis problem, the derivation of synthetic lip features, is now complete. We can now use the synthetic features to drive a variety of systems to view and test the results. If the application is cartoon synthesis, the features can be used to drive a head or face model. If the application is lip dubbing, the synthetic features can be used to drive a video morphing system to correct the lip movements contained in an actual video sequence.

For our research, we use a simple viewer application to animate cartoon lips for viewing. The lip viewer takes the same set of coordinates extracted by Tektronix (Figure 3.2) as input. Thus, if the synthetic lip features are mapped back to a set of lip coordinates, then the same viewer can be used to view either the original data sent from Tektronix or the resulting synthetic data obtained from the synthesis process. Figure 5.6 shows a screen shot of the lip viewer interface developed at CMU.

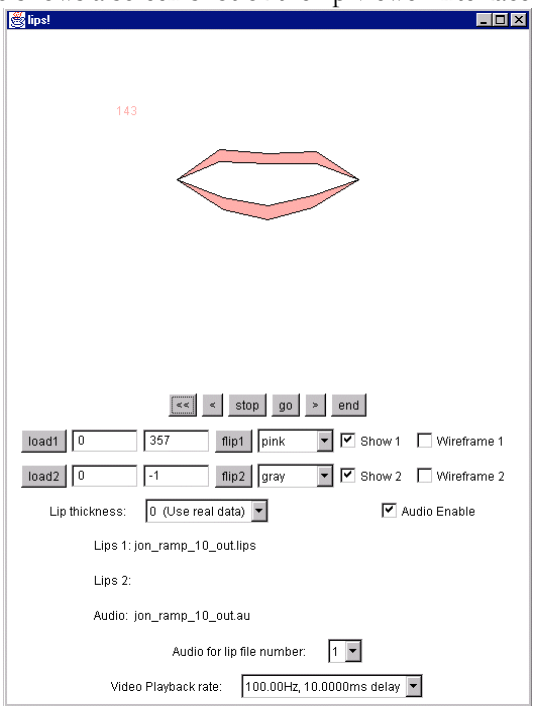

Figure 5.6 CMU Lip viewer application

So, we need a mapping from synthetic features to lip coordinates. Figure 5.7 illustrates the arbitrary mapping that we use to derive the outer lip coordinates from the following feature values: height center (h), width center (w), height left (hl), height right (hr), inner width top (iwt), inner width bottom (iwb). In most cases, the height feature values are evenly distributed about the x-axis, half above and half below. The width features are distributed evenly about the y-axis. The only variation is that the upper and lower coordinates at the center of the lips are distributed 40% above the x-axis and 60% below. This arbitrary mapping produces lips which are symmetric in some respects, but the independence of the left and right height features allows the rendering of asymmetrical lip shapes.

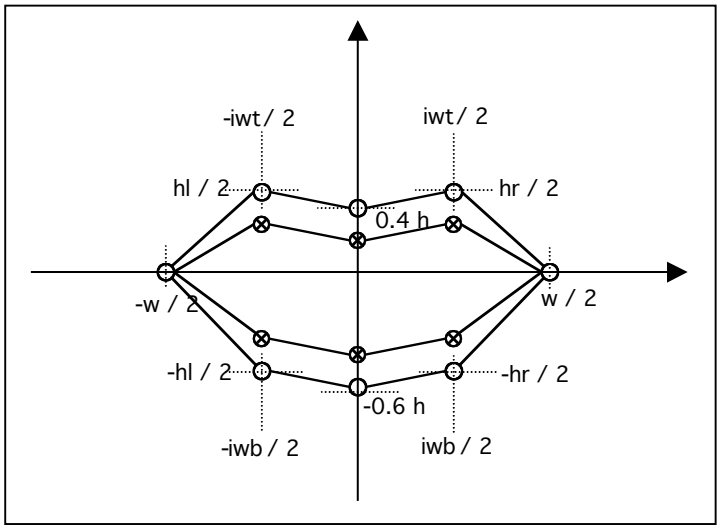

Figure 5.7 Arbitrary mapping from synthetic features to lip coordinates

We use a similar mapping is used to derive inner lip coordinates from the corresponding inner lip features.

Based on observations from the data, it is possible to determine which percentage of each feature should fall above/below the axis. This would create a lip shape that is more tuned to the basic lip shape of the speaker from which the models are derived. But, for simple synthesis and qualitative evaluation of the lip movement synthesis results, an arbitrary shape is sufficient.

5.5 System Performance

Figure 5.8 shows an example synthesis of the main height feature from beginning to end.

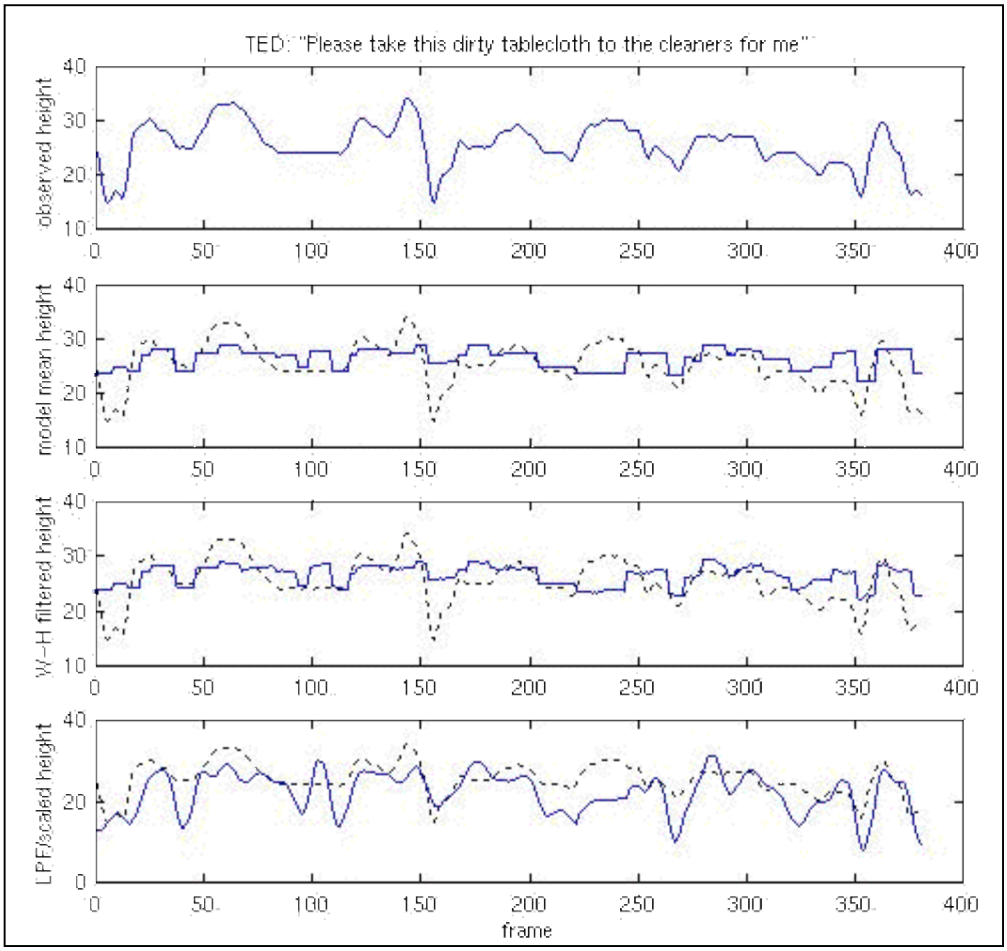

Figure 5.8 Synthesis example for Ted. Plots from top to bottom are: original height from training data, mean heights filled in from the viseme feature model, output of the Wiener-Hopf filter, and output of the low pass filter after scaling.

The resulting height feature NMSEs for each step shown above are given in the following table. Note, the NMSE was calculated both with and without the DTW time re-alignment of the reference and the test signals.

	NMSE	DTW-NMSE
Mean height	0.872	0.566
W-H filtered height	0.871	0.569
LPF/scaled height	1.706	0.832

Despite of the fact that the NMSE increases significantly after the lowpass filtering and scaling of the features, the believability of the resulting synthesis example is slightly better than that of the original data when replayed by the CMU lip viewer. This example and others point to the fact that the NMSE metric is not the most important deciding factor when evaluating the performance of the lip movement synthesis system.

5.5.1 Experimental Results

This section details a series of experiments we performed to fine tune the system and decide which viseme feature model type is most appropriate for the lip movement synthesis system. We ran

experiments to determine which is the optimal order of the Wiener-Hopf filter to be used for lip motion synthesis. Then we did a comparison to evaluate the performance increase obtained by using targeting viseme feature models. We also did an experiment to determine whether the system needs to be trained on the examples which we are synthesizing. We continued with an experiment to see if the lip synthesis is still adequate when the speech recognizer does not have a transcript of the speech for phoneme recognition ("unforced" or "blind" recognition mode). Finally, we performed an experiment to evaluate the decrease in performance when speaker independent viseme feature models are used.

We present the results in terms of NMSE and DTW-NMSE scores for objective evaluation. We also make some comments on our own subjective evaluation of the resulting synthetic lip animations. Both sets of scores are presented below for reference. Note that a crucial part of our future research is to perform a comparative analysis between human-evaluated opinion scores and both the NMSE and DTW-NMSE scores. Such experiments would allow us to see how well each of the objective metrics correlates with actual human ratings, and in this way get a handle on the appropriateness and effectiveness of the metrics. As presented below, there are many cases where subjective evaluation does not correlate well with the NMSE numbers.

Optimal Wiener-Hopf Filter Order Experiment
The first series of experiments is designed to determine the optimal filter order for the Wiener-Hopf filter. We completely trained the system using Ted's 36 sentences with Wiener-Hopf filter orders of 8, 16, 32, 64, 128, and 256. We then evaluated on the same 36 sentences to determine the average NMSE and average DTW-NMSE of the height feature for the system. We carried out the same procedure using Kim's 20 sentences. The results are as follows:

TED	Average NMSE		Average DTW-NMSE	
	After W-H filter	After LPF/scaling	After W-H filter	After LPF/scaling
before filtering	0.8813		0.5722	
8 taps	0.8815	2.1743	0.5691	1.0052
16 taps	0.8819	2.1699	0.5711	1.0070
32 taps	0.8808	2.2023	0.5717	1.0297
64 taps	0.8787	2.1710	0.5668	1.0161
128 taps	0.8760	2.1930	0.5628	1.0250
256 taps	0.8702	2.1443	0.5562	1.0043

KIM	Average NMSE		Average DTW-NMSE	
	After W-H filter	After LPF/scaling	After W-H filter	After LPF/scaling
before filtering	1.0061		0.7676	
8 taps	0.9557	2.6033	0.7720	1.2737
16 taps	0.9357	2.9585	0.7176	1.4272
32 taps	0.9264	2.8266	0.7150	1.3563
64 taps	0.9259	2.9255	0.7143	1.4426
128 taps	0.9240	2.8772	0.7111	1.3959
256 taps	0.9199	2.8884	0.7092	1.3946

As the number of taps is increased, there is very little change in NMSE after the Wiener-Hopf filtering stage of the synthesis process. The DTW-NMSE is also exhibits essentially no change as the filter order increases. In all cases, the nominal best scores are seen for 256 taps, and thus we fixed the Wiener-Hopf filter order at 256 taps.

Subjectively, there is little noticeable difference between the lip movements synthesized using 8 taps and those synthesized using 256 taps. This is in accord with the NMSE metrics.

Targeting v. Ordinary Viseme Feature Model Experiment

This experiment is designed to evaluate the lip movement synthesis performance when targeting viseme models are incorporated into the system. We trained targeting viseme feature models are trained on the entire data sets for Ted and Kim. We then tested the system driven by the targeting feature models on the same data sets. The performance is compared with the system performance achieved when using ordinary viseme feature models. The results are as follows:

TED	Ordinary Models		Targeting Models	
	Avg NMSE	Avg DTW-NMSE	Avg NMSE	Avg DTW-NMSE
before filtering	0.8813	0.5722	1.0010	0.5048
after W-H filter	0.8702	0.5562	0.9342	0.5790
after LPF/scale	2.1443	1.0043	2.2354	1.0229

KIM	Ordinary Models		Targeting Models	
	Avg NMSE	Avg DTW-NMSE	Avg NMSE	Avg DTW-NMSE
before filtering	1.0061	0.7676	1.1883	0.3801
after W-H filter	0.9199	0.7092	0.7264	0.3690
after LPF/scale	2.8884	1.3946	1.8556	0.8893

Subjectively, the synthesis examples built with targeting feature models are better than those built with original models due to a more extensive range of lip motion. However, the objective NMSE numbers prove inconclusive as to which is better. For Ted, it appears that both models are essentially equivalent, but for Kim, it appears that the targeting models are much better.

Testing on Data that is not a Part of the Training Set

The following experiment is designed to determine if the system needs to be trained on the data that is being synthesized. For Ted, we broke the data into two sets, each containing 15 sentences. We ran the complete training process on each of the sets to produce two models. We applied both models to set 1 for testing, and then both models to set 2 for testing. We repeated the experiment for Kim with a set size of 10 sentences each. The resulting NMSE scores for the main height feature are as follows:

TED	Train Set 1 / Test Set 1		Train Set 1 / Test Set 2	
	Avg NMSE	Avg DTW-NMSE	Avg NMSE	Avg DTW-NMSE
before filtering	0.6089	0.3574	1.0869	0.6726
after W-H filter	0.5951	0.3232	1.1083	0.6539
after LPF/scale	0.8044	0.3598	3.0917	1.4552

TED	Train Set 2 / Test Set 1		Train Set 2 / Test Set 2	
	Avg NMSE	Avg DTW-NMSE	Avg NMSE	Avg DTW-NMSE
before filtering	1.1535	0.5792	0.8698	0.5660
after W-H filter	1.1820	0.5435	0.8245	0.5062
after LPF/scale	1.7824	0.6463	2.1997	1.1554

KIM	Train Set 1 / Test Set 1		Train Set 1 / Test Set 2	
	Avg NMSE	Avg DTW-NMSE	Avg NMSE	Avg DTW-NMSE
before filtering	1.1438	0.8985	1.2480	0.5874
after W-H filter	1.0750	0.8709	1.2266	0.5545
after LPF/scale	5.1263	2.7497	3.2018	1.2414

KIM	Train Set 2 / Test Set 1		Train Set 2 / Test Set 2	
	Avg NMSE	Avg DTW-NMSE	Avg NMSE	Avg DTW-NMSE
before filtering	1.0929	0.9115	0.7004	0.4782
after W-H filter	0.9978	0.8244	0.6339	0.4031
after LPF/scale	3.8762	2.1315	1.5134	0.6852

Focusing on the average NMSE scores, it is clear that the system performs better on test examples that are part of the training set. As expected, there is a considerable increase in average NMSE when the test set is not a part of the training set.

Subjectively, for a given example, the synthesis resulting from a model trained on the utterance and the synthesis resulting from a model not trained on the utterance are both acceptable and convincing. However, the synthetic features better match the original features in the case where the utterance is in the training set, and the NMSE measurement detects this fact.

Supervised v. Unsupervised Recognition Experiment

We designed this experiment to determine if we need to know beforehand what words are being said in order to do effective lip movement synthesis. Supervised recognition assumes that we have prior knowledge of the words being spoken, and thus the correct phoneme sequence. The task of the recognizer is simply to align the phoneme sequence in time with the speech waveform. Unsupervised phoneme recognition assumes that the transcript of the speech is unknown, and thus the recognizer must determine both the correct phoneme sequence and the proper start frame and duration of each phoneme. Unsupervised recognition is thus more prone to errors.

We performed speech recognition on all of the utterances for Ted and Kim using SPHINX-III in supervised mode and unsupervised mode. We also performed unsupervised recognition using the older SPHINX-II architecture. The SPHINX-III unsupervised phoneme recognition should be more accurate than the corresponding SPHINX-II unsupervised recognition because SPHINX-III performs its search over the more accurately modeled triphones to determine the phoneme sequence while SPHINX-II searches over connected models of individual context-independent phonemes.

The results are as follows:

TED	Supervised Recognition SPHINX-III		Unsupervised Recognition SPHINX-II (monophone search)		Unsupervised Recognition SPHINX-III (triphone search)	
	Avg NMSE	Avg DTW-NMSE	Avg NMSE	Avg DTW-NMSE	Avg NMSE	Avg DTW-NMSE
before filtering	0.8813	0.5722	1.1522	0.7373	1.1502	0.7616
after W-H filter	0.8702	0.5562	1.1578	0.7257	1.1484	0.7514
after LPF/scale	2.1443	1.0043	3.0602	1.5059	2.8730	1.4758

KIM	Supervised Recognition SPHINX-III		Unsupervised Recognition SPHINX-II (monophone search)		Unsupervised Recognition SPHINX-III (triphone search)	
	Avg NMSE	Avg DTW-NMSE	Avg NMSE	Avg DTW-NMSE	Avg NMSE	Avg DTW-NMSE
before filtering	1.0061	0.7676	1.1700	0.8752	1.1667	0.8473
after W-H filter	0.9199	0.7092	1.1029	0.8248	1.1156	0.8068
after LPF/scale	2.8884	1.3946	3.2813	1.5749	3.9999	2.0203

Both the NMSE and the DTW-NMSE scores indicate a decrease in performance when the recognizer is run in unsupervised mode. It is interesting to note that there is little difference in performance when SPHINX-III is used for more accurate phoneme recognition.

Speaker Independent v. Speaker Dependent Model Experiment
The final experiment sequence presented here is designed to determine the efficacy of speaker independent models. We trained models on Kim's speech and then normalized the means via the procedure presented in Section 4.5.4 to produce a bootstrap viseme feature model for Ted. We then tested the two models on all 36 of Ted's sentences. The results are as follows:

TED	Ted's standard models		Ted's models bootstraped from Kim	
	Avg NMSE	Avg DTW-NMSE	Avg NMSE	Avg DTW-NMSE
before filtering	0.8813	0.5722	1.0720	0.8746
after W-H filter	0.8702	0.5562	1.0729	0.8694
after LPF/scale	2.1443	1.0043	1.5616	0.8555

In general, there is a degradation in performance when the bootstrapped models are used. However, an interesting anomaly in this case is that the final results for the bootstrapped models more closely match the original features, and thus have a lower NMSE score.

5.6 System Refinement

Up until this point, we have focused on the development of a system to create synthetic lip features with a minimum mean square error from the observed features. While statistically this is the correct thing to do, subjectively it is not. A closer look at the results shown in the previous section reveals that the error numbers consistently increase from the output of the Wiener-Hopf filter to the final output of the system. However, a simple glance at the resulting synthetic lip animations makes it clear that the subjective quality improves greatly after the final stages of processing. We therefore conclude that minimizing the mean square error is not the correct goal. From this point forward, we will act as a cartoonist of sorts, sacrificing "error" points to create synthetic lip motions with a more convincing subjective reality.

We implemented two main system refinements to improve the overall quality and believability of the lip movement synthesis results. One consistent complaint of synthesis viewers is that the lips still do not close down completely or open as widely as they should. The other complaint is that there is a slight but distracting trembling of the lips during silence periods within an utterance. Ad-hock solutions have been derived to combat these observable artifacts, and the resulting synthesis is more believable.

We designed our first refinement, termed "overshoot scaling", is designed to ensure the lips open and close to a sufficient extent. One possible cause for this persistent problem lies in the training data itself. When Tektronix performs lip tracking, there is a "guard band" of approximately 5 pixels that ensures the upper lip coordinates can never crash into the opposing lower lip coordinates. Thus, the training data itself never fully comes to a close. (This can be seen by taking a closer look at Figure 5.5 (a). There are no inner height values below 5 on the histogram.)

A solution to this problem is to use the linear transformation of the features to perform a more aggressive scaling. Consider the following new constraints:

5. $\max(f_{\square}) = \max(f_{obs}) + \square_1$
6. $\min(f_{\square}) = \min(f_{obs}) - \square_2$

The "overshoot parameters" $\square_1$ and $\square_2$ allow us to generate a transformation of features that exceeds the original data by a specified amount. Typically, we choose $\square_2$ so that the inner lip height features go to zero, i.e. so that the lips will close completely. The corresponding outer lip height features are then scaled using the same $\square_2$ value. Adding an extra overshoot of 10 pixels to the overshoot parameters produces synthesis results that often times exceed the quality of the original data when both are presented by the lip viewer application. Note that the resulting normalized mean square error of the features derived with overshoot scaling will be larger than those derived by the standard procedure. Nevertheless, the quality of the synthesis improves, in spite of the increase in "error".

We created the second refinement to compensate for feature trembling during silence periods within the utterance. This trembling is an artifact of the filtering process used to derive the synthetic features; it occurs as the features head towards the stable silence level specified by the viseme feature model. An example of this can be seen in Figure 5.9 (a). We find it necessary to treat the silences in a special fashion and correct this phenomenon.

Our correction is simple: we fit a decaying exponential (with a relatively fast time constant, less than or equal to 3 frames) to each of the features in the region of silence. The resulting smoothed silence region can be observed in Figure 5.9 (b).

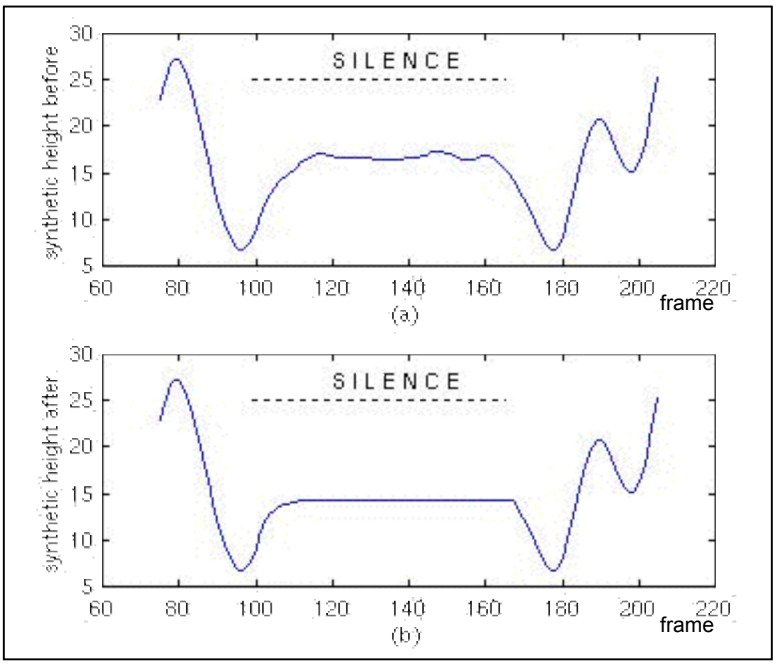

Figure 5.9 Synthetic height within silence region, shown before (a) and after (b) exponential correction

The resulting synthesis looks much smoother and more natural because the distracting wavering of the lips has been removed. Also, the added pause and subsequent restart in synch with the audio enhances the believability of the lip movement synthesis.

5.7 Conclusions

In conclusion, we have shown that is possible to build a lip movement synthesis system stemming from the viseme feature models developed to link the audio and video speech in a natural way. The resulting lip animations are believable and in synch with the corresponding audio, as if the cartoon lips are speaking what was said. Our filtering methodology proves successful in creating lip features that move in the right direction at the right times.

However, our filtering methodology comes up short because it consistently produces features that have a smaller range of dynamic motion than that of observed features. Fortunately, since the underestimation of the features is consistent and regular, we have shown that it is possible to correct for the phenomenon via a linear transformation of the features. After scaling, the features lead to synthesis examples that move in accord with real lip features, both in direction and excursion.

Also, we have employed a sensible error metric that is flexible enough to allow for slight timing misalignments of the features which produce imperceptible errors in the resulting lip movement synthesis examples. Despite this correction, the DTW-NMSE is still limited, and perceptual testing needs to be done to verify whether or not it correlates well with human evaluation of the synthetic lips.

We decided that generation of features to minimize the mean square error is not good enough to create synthetic lip motions with a natural and believable appearance. Further system refinements such as overshoot scaling lead to better looking synthesis examples despite an increase in mean square error. We have observed good improvements in subjective quality of lip animations generated via such techniques.

Finally, there are many applications that can be built around our lip movement synthesis process. The resulting synthetic features have been used to successfully drive a cartoon lip viewer application. In the future, these same features can also be used to drive any of the popular face models, or even to drive a system to morph the lip data in real video sequences to correct the lip movements of a dubbed foreign film.

CHAPTER 6: LIP TIME WARPING

In the film or television industry, a new dialog recording is often used to replace the original dialog recorded in the field. In practice, the actors watch themselves in the film and try to speak at the same rate, but the timing of the dialog rarely matches the original well enough to synchronize with the lip movements in the film. In 1984, P.J. Bloom developed the WORDFIT system [2] that analyzes both the original and re-recorded speech segments of a movie soundtrack, produces a time-alignment path, and then automatically edits the replacement speech so that the timing matches the images on the screen.

In this chapter, we present a system to meet a similar goal, but with a different focus. Our lip time warping system is designed to time align the *lip shapes* to a given re-recording of the audio segment. The re-recording can be performed by the same speaker or by a different speaker. Our system time warps the original lip sequence so that it is in synch with the re-recorded speech.

We start the chapter with a description of the Dynamic Time Warping algorithm before presenting our preliminary system for time warping of lips. The chapter closes with a small evaluation of our system's performance and some conclusions.

6.1 Dynamic Time Warping Algorithm

The Dynamic Time Warping (DTW) algorithm [10] is designed to time-align two instances of the same pattern that are produced at different rates. DTW was used in early speech recognition systems to compensate for the fact that the test speech is almost always spoken at a different rate than the reference templates used for recognition. The problem is visualized as a two-dimensional plot with the reference pattern or signal on the horizontal axis and the test signal on the vertical axis. The DTW algorithm produces a time-alignment path that shows the temporal correlation between the two signals. This is illustrated in Figure 6.1.

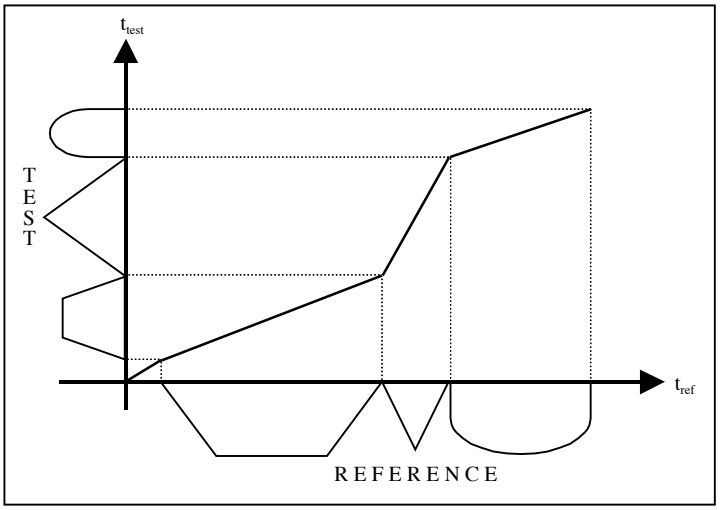

Figure 6.1 DTW example warping path for time alignment of two patterns

To find out which point in the reference corresponds to a given point in the test sequence, trace from the test sequence horizontally to the right until the warping path is reached, then trace down to the corresponding point in the reference.

Given a reference pattern, a test pattern, and some measure for "distance" or "distortion" between the two, the DTW algorithm efficiently derives the time-warping path with the minimal cumulative distortion or "cost".

In the basic implementation, DTW assumes that the first and last points of each pattern are aligned. The algorithm must find the best path from beginning to end. This is accomplished as follows: First define $D(i_r, i_t)$ to be the smallest cumulative distortion possible up to the point (i_r, i_t) in the warping grid. Define $d(i_r, i_t)$ to be the incremental distortion between the reference signal at time index i_r and the test signal at time index i_t. Then specify local path constraints, i.e. all the possible ways to arrive at a given point on the graph. Note that a weight can be associated to each path. An example local path constraint with weighting is shown in Figure 5.2.

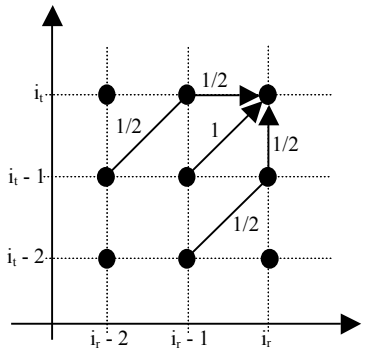

Figure 6.2 An example set of local path constraints that can be used by the DTW algorithm

Using the example constraints, there are three possible paths to reach the point (i_r, i_t). The cumulative distortion to (i_r, i_t) along a given path is given by the sum of the cumulative distortion at the tail of the path and the incremental distortions of all other points along the path. This gives rise to the following three formulas for possible cumulative distortion at (i_r, i_t) (1 is the upper path, 2 is the center path, and 3 is the lower path):

1. $D(i_r, i_t) = D(i_r - 2, i_t - 1) + \frac{1}{2} d(i_r - 1, i_t) + \frac{1}{2} d(i_r, i_t)$
2. $D(i_r, i_t) = D(i_r - 1, i_t - 1) + d(i_r, i_t)$
3. $D(i_r, i_t) = D(i_r - 1, i_t - 2) + \frac{1}{2} d(i_r, i_t - 1) + \frac{1}{2} d(i_r, i_t)$

Given this framework, it is possible to derive the correct warping path. The algorithm examines all points (i_r, i_t) in the warping grid from left to right. The cumulative distortions for each of the possible paths to (i_r, i_t) are calculated, and the minimum result is recorded. The path that resulted in the minimal cost is also recorded for reference. The cumulative distortion calculation continues until all possible points in the warping grid have been examined and the final point has been reached. The final step is to make use of the noted path choices to trace back the resulting time warping path from the end to the beginning.

6.2 Automatic Time Warping of Lip Video for Synchronization with Audio

We use the DTW framework in conjunction with the viseme feature models derived for a speaker to solve the problem of aligning a set of lips to a re-recorded audio sequence. In this case, the reference pattern is the phoneme sequence that corresponds to the re-recorded audio, and the test pattern is

the lip feature sequence from the video. We perform speech recognition to obtain the phoneme sequence, and we derive the sequence of lip features from the points obtained by lip tracking. We assume that the distributions of the lip features for a given phoneme are Gaussian; therefore, the distributions are fully specified using the means and variances stored in the viseme feature model.

For this application, we define $d(i_r, i_t)$ to be a *likelihood* metric, i.e. how likely it is to see the lip features at time i_t given that the phoneme recognized at time i_r is P_r. Recall that the viseme feature model associates each phoneme P_r with a mean $\mu_{f,r}$ and variance $\sigma^2_{f,r}$ for each feature f. The metric $d(i_r, i_t)$ used to perform the time warping is based on the distributions of only two features: the outer height (h) and outer width (w). Let h_t and w_t be the features observed in the lip sequence at time i_t. Using this notation, we define the incremental likelihood metric as follows:

$$d(i_r, i_t) = N(\mu_{h,r}, \sigma^2_{h,r})\Big|_{h=h_t} \cdot N(\mu_{w,r}, \sigma^2_{w,r})\Big|_{w=w_t}$$

Evaluation of the Gaussian pdfs gives an indication as to the likelihood of observing the lip features given the reference phoneme. For implementation purposes, it is possible to use the natural log of the right side of the equation to define the likelihood metric as a sum rather than a product.

We perform the DTW search as usual, but the path chosen is the one that yields the *highest* score, i.e. $D(i_r, i_t)$ is defined to be the *maximum* cumulative likelihood of all possible paths to the point (i_r, i_t). Once the time warping path is derived, we use it to generate a new sequence of lips that are in synch with the re-recorded audio reference.

6.3 System Performance

We need to define an error metric to evaluate the performance of the lip time warping system. Given the reference and re-recorded audio segments, we perform phoneme recognition on both and use the resulting phoneme strings to derive the correct warping path. We choose to use a simple error metric: the standard mean square error between the correct warping path and the warping path created by the system.

However, since we are now working with tracked lip points rather than synthetically generated lip points, subjective evaluation by humans is also quite feasible. The lips "said" the same thing earlier; therefore, the shapes should be correct. The question is whether or not the warped lips move in synch with the re-recorded audio being played.

Figure 6.3 presents a typical result we produced by using our system to align Ted's lips to Jan's speech. For this example, both Ted and Jan read the sentence "The fish began to leap frantically on the surface of the small lake." Except for some error at the beginning, the resulting path is very close to the true path obtained from phonetic recognition of both utterances.

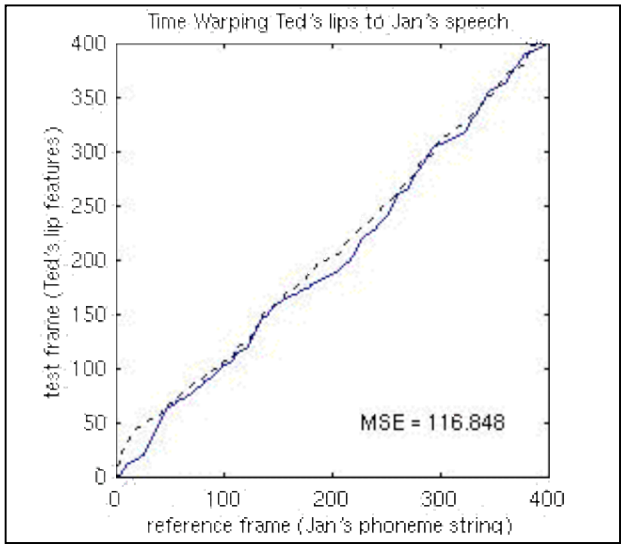

Figure 6.3 An example lip time warping result. True path is shown as a dashed line.

After viewing some preliminary lip time warping examples, we see that initial and final periods of silence require special treatment. This is due to the following facts:

- silences are not well modeled by a context-independent viseme feature model
- the DTW algorithm fixes the initial and final points and allows free warping in between

We decided to exclude any initial and final silence periods before performing DTW on the sequences. It is trivial to exclude initial and final silence frames from the reference phoneme string, but we need to perform phoneme recognition on the test audio in order to remove features corresponding to the initial and final silences from the search. As illustrated in the Figure 6.4, the added benefit of the extra computation far outweighs the cost.

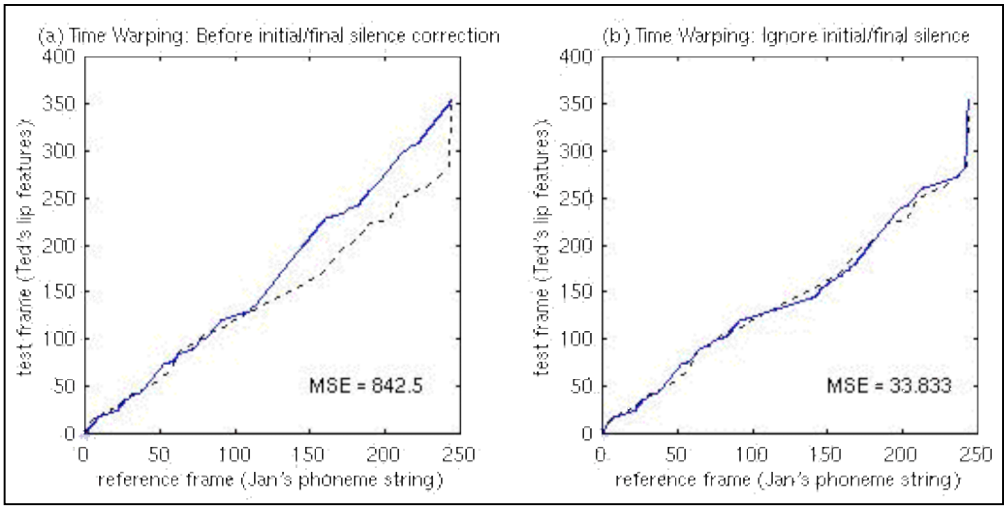

Figure 6.4 Lip time warping results (a) before and (b) after initial and final silences are removed from the search. True path is shown as a dashed line.

In our data set, Jan and Ted speak the exact same utterance 5 times. We performed warping experiments using viseme feature models trained on Ted's lip data to align Ted's lips to Jan's speech for each of those 5 sentences. The results are as follows:

Warping Ted's lips to Jan's speech (standard model)	Average MSE
No initial and final silence compensation	324.5
Ignore initial and final silence regions	118.5

As illustrated in Figure 6.3, an MSE score around 100 corresponds to a derived warping path that matches the correct path very well. When the initial and final silence regions are ignored, the resulting warping paths for these examples are quite good.

Interestingly, when we repeat the same experiment using a targeting viseme feature model, the end results are clearly better. This is most likely due to the fact that the means of the target model are spread further apart, providing for more discrimination between the lip shapes that correspond to each phone.

Warping Ted's lips to Jan's speech (target model)	Average MSE
No initial and final silence compensation	458.2
Ignore initial and final silence regions	53.5

As a final test, I recorded myself speaking the sentence "The fish began to leap frantically on the surface of the small lake." I recorded this same sentence 8 times. We then performed warping experiments to align Ted's lips to my speech for each of the 8 instances of the same utterance. The results are as follows:

Warping Ted's lips to Jon's speech (target model)	Average MSE
No initial and final silence compensation	245.8
Ignore initial and final silence regions	56.9

6.4 Conclusions

We developed and presented a preliminary system for time warping of lip shapes to automatically synchronize them with a re-recorded audio track. Although our test set is not extensive, we do get some good preliminary results on the examples we tried. Clearly more investigation of the effectiveness of the procedure is needed.

A noteworthy observation is that the silence regions, especially at the beginning and ending of utterances, must be eliminated in order for the DTW-based warping technique to function properly. Also, target based viseme feature models produce better results than the standard models. This leads us to believe that context dependent models would do even better.

CHAPTER 7: CONCLUSIONS & FUTURE RESEARCH

This chapter summarizes our results and presents some ideas for future research in the areas of lip movement synthesis and time synchronization.

7.1 Making Good Use of Additional Data

Presently, our training data sets are still very limited. Lip tracking is a computationally expensive task, and recording new speakers for data collection takes much time and energy. How much data is enough? What could be done if more training data were available?

7.1.1 Practical Applications with Limited Training Data

Our current experience indicates that we can develop an effective lip movement synthesis system using 11 lip features (6 heights and 5 widths) with as little as 20 sentences of lip coordinate training data, assuming the sentences read come from a phonetically rich corpus such as TIMIT. Our earlier experiments show that our 2-feature (height and width) system is effective with only 8 sentences of training data. The resulting animations are synchronized to the video and believable. More experimentation is still needed to determine the optimal number of features to be used for synthesis, and to determine exactly what impact on perception the addition of features makes.

Our lip time warping system works with only the 2 basic lip features: height and width. It has proven effective in accurately time warping Ted's lip motion to two different speakers, Jan and myself. The targeting viseme feature models used for time warping are trained on 36 sentences of lip coordinate data from Ted.

7.1.2 Context-Dependent Modeling

It is clear that context-dependent modeling of the lip features would provide more precise models and demonstrate some of the natural coarticulation effects seen in speech and the corresponding video. Currently, our system is trained to believe that the lip features corresponding to each phoneme do not depend on the context in which the underlying phoneme is found. We know that this indeed is not the case. However, since only 0.74% of the possible triphones is seen in our most extensive training set, it is currently not possible to investigate the effectiveness of such models. We would require a massive training effort to collect and process the speech and lip coordinate data to create well-trained viseme feature models based on triphone context. Based on our experience with speech recognition, we are sure that both our lip movement synthesis and lip time warping systems would benefit tremendously from context-dependent modeling. The resulting lip animations would be more natural and believable than the current examples. Also, the time warping system would produce more accurate and precise warping paths.

7.2 Perception-Based Evaluation

As stated earlier, we are convinced that generating minimum mean square error features is not necessarily the best way to create synthetic lip features for lip motion synthesis. In fact, our later work shows that the most convincing animations are ones which sacrifice mean square error points for the sake of subjective realism. Although we have derived an error metric for measuring the performance of our synthetic lip features, we have not yet performed any testing to see how well this numerical metric correlates with human perception of the animations. It is our impression that the metric we presented, while numerically convenient, is not a good measure for perceptual quality of

the lip motion synthesis. Future research should include more focus in this area, including detailed perceptual tests in which our animations are judged by a group of human subjects.

7.3 Applications for Foreign Language Film Dubbing

Most recently, we have begun to investigate the possibility of applying our lip movement synthesis algorithm to foreign language film dubbing. In essence, the problem we are trying to solve is that of correcting the lip motions to match the re-recording of a film in a language other than the original. In theory, there are three main problems to solve:

- Time alignment of foreign speech to original speech
- Generation of synthetic lip features to match the foreign speech
- Lip morphing system to correct the lip motions on the video to match the synthetic features

In modern movies, it is clear that the actors do a very good job of matching their speech timing to that of the original actor in the original language. We have developed a very simple system based on correlation of the smoothed energy contours of the reference and foreign speech to better align the foreign speech to the original speech. However, since the actors do such a good job of time alignment on their own, this step may not be necessary.

For foreign language lip movement synthesis, we have tested our system (trained and designed solely for the English language) on foreign speech in Spanish, French, and Portuguese. Although the results are imperfect, they are more or less time aligned with the foreign speech and perceptually convincing. Clearly, if a speech recognizer were trained in the foreign language and lip coordinates were tracked, the same methods used to create our English language system could be used to create more precise systems for lip movement synthesis in any language. None of our work depended on the language being considered.

Lip morphing of actual movie footage is probably the most difficult problem to solve, given all possible camera angles and lighting conditions. Preliminary demonstrations of the effectiveness of systems such as Video Rewrite lead us to believe that such a system is possible. For future research in this aspect of foreign language film dubbing, it may be useful to first limit the problem to a domain such as broadcast news where the faces and lips are usually in the same position and facing the camera. Since Tektronix has provided the video processing backbone for this research, we leave this problem in their competent hands.

REFERENCES

[1] T. Chen and R. R. Rao, "Audio-Visual Integration in Multimodal Communication," *Proceedings of the IEEE*, Vol. 86, No. 5, pp. 837–852, May 1998.

[2] P. J. Bloom, "Use of Dynamic Programming for Automatic Synchronization of Two Similar Speech Signals," *Proceedings ICASSP 84*, Vol. 1, March 1994.

[3] M. M. Cohen and D. W. Massaro, "Modeling Coarticulation in Synthetic Visual Speech," *Models and Techniques in Computer Animation*, p. 139–156, June 1993.

[4] F. J. Huang and T. Chen, "Real-Time Lip-Synch Face Animation Driven by Human Voice," *1998 IEEE Second Workshop on Multimedia Signal Processing*, Dec 1998.

[5] M. Rydfalk, "CANDIDE: A Parameterized Face", Linkoping Univ., Sweeden, Rep. LiTH-ISY-I-0866, Oct. 1987.

[6] E. Yamamoto, S. Nakamura, and K. Shikano, "Speech to Lip Movement Synthesis by HMM," *Proceedings of the ESCA/ESCOP workshop on Audio-Visual Speech Processing*, pp. 137–140, September 1997.

[7] E. Yamamoto, S. Nakamura, and K.Shikano, "Subjective Evaluation for HMM-Based Speech-to-Lip Movement Synthesis," *Proceedings of the ESCA/ESCOP workshop on Audio-Visual Speech Processing 1998*, 1998.

[8] C. Bregler, M. Covell, and M. Slaney, "Video Rewrite: Driving Visual Speech with Audio," in *Proc. ACM SIGGRAPH '97*, pp. 353–360, 1997.

[9] W. M. Fisher, V. Zue, J. Bernstein, and D. Pallett, "An Acoustic-Phonetic Data Base," in *113th Meeting of the Acoustical Society of America*, May 1987.

[10] L. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, 1993.

[11] A. Rudnicky, L. Baumeister, K. DeGraaf, and E. Lehmann, "The Lexical Access Component of the CMU Continuous Speech Recognition System," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, April 1987.

[12] K. F. Lee, *Automatic Speech Recognition: The Development of the SPHINX System*, Kluwer Academic Publishers, 1989.

[13] F. Jelinek, *Statistical Methods for Speech Recognition*, The MIT Press, 1997.