Videorealistic Facial Animation for Speech-Based Interfaces

by

Stephen J. Pueblo

S.B., Massachusetts Institute of Technology (2008)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2009

Author .....
Department of Electrical Engineering and Computer Science
May 22, 2009

Certified by .....
James R. Glass
Principal Research Scientist
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

Videorealistic Facial Animation for Speech-Based Interfaces

by

Stephen J. Pueblo

Submitted to the Department of Electrical Engineering and Computer Science
on May 22, 2009, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

This thesis explores the use of computer-generated, videorealistic facial animation (avatars) in speech-based interfaces to understand whether the use of such animations enhances the end user's experience. Research in spoken dialog systems is a robust area that has now permeated everyday life; most notably with spoken telephone dialog systems. Over the past decade, research with videorealistic animations, both photorealistic and non-photorealistic, has reached the point where there is little discernible difference between the mouth movements of videorealistic animations and the mouth movements of actual humans. Because of the minute differences between the two, videorealistic speech animations are an ideal candidate to use in dialog systems. This thesis presents two videorealistic facial animation systems: a web-based system and a real-time system.

Thesis Supervisor: James R. Glass
Title: Principal Research Scientist

Acknowledgments

I would like to first thank Jim Glass, my thesis advisor, for giving me the opportunity to work in his group. I am greatly appreciative that I was given such an opportunity, and without his guidance and support, this thesis would not be possible. Secondly, I want to thank Scott Cyphers. Without his technical expertise and assistance on a multitude of aspects dealing with this thesis, I would have been truly lost.

I am also much indebted to Tony Ezzat, whose work this thesis was primarily built upon. The time spent with me with regards to the avatar training and Mary101 is much appreciated. I would like to also thank Paul Hsu for helping me with issues stemming from the original real-time optimized code.

In addition, I would like to thank Ian Mcgraw for his assistance with changing Mary101's video background and with getting the screen capture to work correctly in Linux, and I would like to thank Alex Gruenstein with his assistance on using WAMI. I would also like to express my gratitude to the other researchers at the Spoken Language Group whose research this thesis is built upon.

Last but not least, I would like to thank my family, especially my parents for the support they have given me throughout these five years at MIT. I would not be in the position that I am in today if it was not for them.

This research was supported through the GEM Fellowship.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Traditional dialog systems rely solely on text, speech, or graphical images to interact with the user. There is little to no sensory stimulation for the user other than reading text, listening to audio, or looking at an image, which makes it apparent that the user is still interacting with a machine.

This thesis investigates how the addition of videorealistic animations enhances user's interactions and experiences with computers. Since a realistic video representation of a human is depicted on the screen, instead of just audio, text or an image, it can be hypothesized that the user will feel more comfortable with using such a dialog based system. Ideally, the user will have the sense that he or she is interacting with an actual person, rather than a machine. Two prototype systems have been developed as part of this research. Both systems use the videorealistic animations in several, distinct ways. User satisfaction with the videorealistic avatar is the prime focus of assessment for these systems.

1.1 Motivation

Numerous users would, in all probability, admit that many interactions with computers using dialogue systems still give them the feeling that they are conversing with a machine, and not a human-like entity. A great deal of research has been done with dialogue systems at the Spoken Language Systems Group at MIT and elsewhere over

the years.

Many of these dialogue systems deal with a certain domain and a certain mode of interaction (text, audio, graphics/video, or a combination of the three). For example, Jupiter [28] and Mercury [27] are two conversational systems that converse with the user about weather (Jupiter) and airline flights (Mercury) using only audio over the telephone. Jupiter allows anyone with a telephone to get Intellicast weather forecasts. Someone asking the system what is the forecast in Boston tomorrow is an example of what could be said to Jupiter. Similar to Jupiter, Mercury is a conversational interface that provides flight information and enables users to book itineraries via the telephone.

In addition, CityBrowser, a web-based system developed at MIT, uses both mouth gesture as input and speech to provide information, both graphically and audibly, about user-requested locations in a city [5]. A person using CityBrowser could ask for a Chinese restaurant and then point and circle the location where he or she would like the system to search.

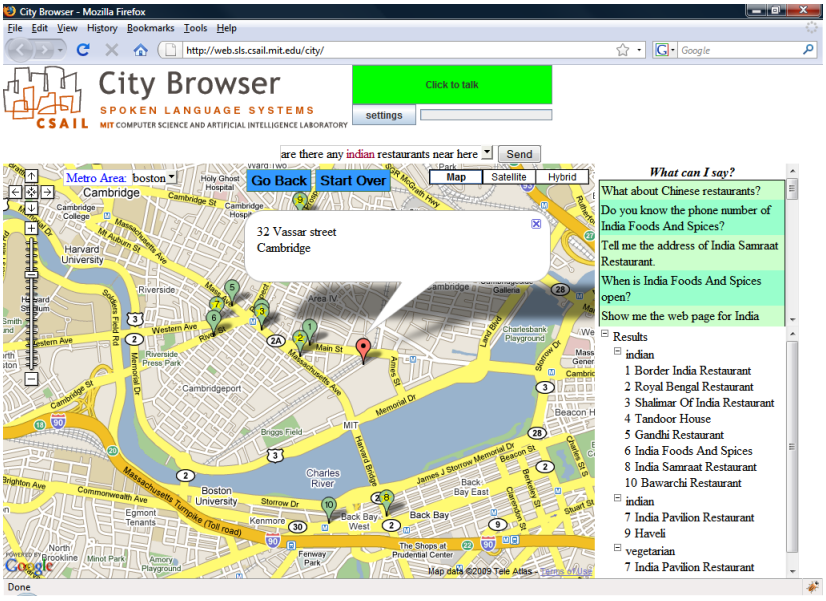

Figure 1-1: Screenshot of City Browser

All of the systems described above are advanced systems that perform their specified tasks well, but these systems lack the feeling that one is talking to an actual

person.

In 1987 a future concept video was created by Apple Computer. The video introduced a visionary device called the Knowledge Navigator [12]. In this video a professor comes home and interacts with the Knowledge Navigator, which is a book-like gesture based electronic device. A computer generated avatar appears on screen, and the professor converses with the avatar asking the avatar to compile data about deforestation in the Amazon rainforest. The avatar onscreen immediately agrees to obtain the data, and in a short period of time, the data is shown. The avatar inside the Knowledge Navigator acts and behaves like an actual real-life agent with all the human qualities one would expect. The avatar presented in this video inside the Knowledge Navigator is the ultimate motivation and eventual goal for the technology presented in this thesis.

Figure 1-2: Screenshot of the Knowledge Navigator [14]

According to Shechtman and Horowitz, people behave differently when they interact with what they perceive to be either a computer or a human being [23]. For their study, participants had a text-based conversation with a computer that gave scripted responses. However, some participants were told that the computer was a person and others were told that the computer was actually a computer. In Shechtman and Horowitz's findings, participants used more words and spent more time conversing with the other partner when they believed the other partner was a person.

The participants' behavior was entirely different when they believed that they were conversing with a person. There was much greater engagement on the relationship track for the participants that believed they were talking to a person than those who believed they were talking to a computer.

With Shechtman and Horowitz's study in mind, it is hypothesized that these prototype systems will give users the feeling that they are no longer conversing with a machine, but instead, they will have the feeling of video-conferencing with a living, breathing human being. This illusion that users are conversing with a person will hopefully enhance their interactions with the computer. As technology and computational power have increased over the past decade, it is now possible to create systems where the videorealistic video is not only pre-generated, but also generated in real-time.

1.2 Chapter Summary: Prototype systems

The real-time generation technology and the two prototype systems discussed in this thesis are based on the video generation system developed by Ezzat et al. at MIT [9]. Having the choice between both prototype systems allows for flexibility, in the sense that each system gives the developer a different option on how he or she wants to integrate such an avatar system with his or her own system.

The first system that is discussed is Mary101.NET. Built solely to run in a web browser, Mary101.NET allows the user to easily converse with a simple dialogue system through speech recognition and pre-generated videos. The system is also flexible in the sense that the developer can add as many pre-generated videos as he or she wishes, in order to create as complex a dialogue system as deemed necessary.

The second system discussed is CGI.Mary101, and it is similar to Mary101.NET. The main difference is that CGI.Mary101 runs locally on a machine, instead of on a server. Computer facial animation and a speech recognizer are still used, as with Mary101.NET; however, a custom video display is created to properly display the different video states of the dialogue system, and the video is generated in real time.

1.3 Outline

The remainder of the thesis is organized in the following manner:

An overview of the related works that deal with facial animation is discussed in detail in Chapter 2. Here, both research in photorealistic human-like animations and research in cartoon animations are discussed. The research presented also shows different approaches to creating such animations. The approaches include methods where training is involved and methods that solely rely on computer algorithms that are not based on any previous training.

Chapter 3 dives into the various speech technology that is used in the real-time technology and prototype systems presented in this system. The speech technology includes the WAMI infrastructure, which Mary101.NET is built upon, and other pieces of technology, like speech recognition and speech synthesis.

Chapter 4 discusses the main technology behind the video generation of the facial animation presented in this thesis: the morphable model approach. The morphable model approach allows the video generation system to create novel mouth movements from a pre-recorded training corpus.

Chapter 5 goes into the real-time technology that can generate the facial animation video in real-time, given any text. This technology is the basis for the second prototype system presented in this thesis, CGI.Mary101. This technology is built upon the work done by Ezzat et al. [9].

The two prototype systems are presented in Chapter 6 of this thesis. These prototype systems are summarized in Section 1.2.

In Chapter 7 of this thesis, the user study conducted for this research is discussed in great detail. In this chapter, the results of the study are discussed, in which forty-one people consented to participate. These subjects were asked to assess three scenarios that used three different forms of media: audio, image, and video.

The final chapter, Chapter 8, discusses how the technology and the prototype systems solve the variety of issues presented at the beginning of this thesis. This chapter also discusses the future work that could be implemented to further advance

the technology.

Chapter 2

Related Work

This chapter provides an overview of the related work done in the field of speech animation. Two forms of speech/facial animation are discussed in detail: Photorealistic Speech Animation and 3D-Cartoon Speech Animation. All of the research that is presented in this chapter approach the synthesis of speech animation in distinctive ways.

2.1 Photorealistic Speech Animation

2.1.1 Video Rewrite

Bregler, Covell, and Slaney took an unique approach in synthesizing new speech animations. The system was named Video Rewrite [7]. In order to create novel utterances from a pre-recorded corpus, Video Rewrite stitches together previously recorded mouth movements; however, no new mouth movements could be produced.

In order to train Video Rewrite, video must be recorded from a subject. From the recorded video, Video Rewrite uses HMMs (Hidden Markov Models) to label the recorded video into segments of triphones. The middle phoneme of each triphone is emphasized, and the outer phonemes of the triphone are overlapped and cross-faded in the video stitching to make the transitions smooth and to ensure that the transition points are not critical. Since several phonemes have the same visual lip structure,

the phonemes are classified into visemes. In the same manner how phonemes are the fundamental units of speech, visemes are the fundamental units of the visual lip structure that will be meshed together in Video Rewrite. A mapping from phonemes to their corresponding viseme classifications are used to determine how to properly align and overlap the different triphone video segments.

When newly synthesized video is created from Video Rewrite, the triphone videos calculated must be aligned properly to the speech transcription. The starting time of the center phone in each triphone video is aligned with the time of the corresponding phoneme label in the transcript. In the final step, a mask is used in the video to determine where the transformed mouth movements should be stitched together. The system automatically corrects the different head positions of the background face to the different positions of the mouth movements that will be meshed together. Video Rewrite works relatively well, but since the system can not produce novel, new mouth movements, whenever it can not find a certain phoneme in its training, the system instead finds the closest approximation to it. Finding the closest phoneme approximation in creating a video could potentially create video defects that are visibly noticeable by the viewer.

2.1.2 Facial Animation System for Interactive Services

Using a similar concatenative approach as Video Rewrite, Liu and Ostermann set out to create a photorealistic speech animation system that, like Mary101.NET, works with web-based applications [19]. Given a question from a user through his or her browser, Liu and Ostermann's system creates a photorealistic avatar through the user's browser that responds to the user's question in a realistic manner. However, the approach Liu and Ostermann took to create the avatar is vastly different than the video generation approach used in Mary101.

A corpus of pre-recorded video was used to train the system, but the video frames were aligned beforehand. A frame is aligned to the corresponding phoneme being said and the phoneme context. The phoneme context is an issue because certain mouth movements depend on the preceding and succeeding phonemes. Once the video frames

from the corpus are aligned beforehand, their mouth segment images are stored into a database on a web server. When the dialogue system interprets what to say from the user's audio input, a unit selection algorithm is used to determine what is the best mouth segment image to pull from the database to output onscreen that meshes well with what the system wants to say.

2.2 3D-Cartoon Speech Animation

2.2.1 Baldi

Baldi is similar to the other research presented in this section on speech animation, but Baldi is a 3-D human cartoon avatar that is focused on the educational aspect of speech technology [25]. Baldi has been proven a success in classrooms of profoundly deaf children, and it has been very influential in the speech animation field. Primarily developed by Massaro, Baldi is an avatar system that can engage with users in simple structured dialogues. Depending on the domain of the underlying dialogue system that Baldi is built upon, Baldi's speech recognition can recognize a limited set number of words the user says that is in the domain of the system, and it rejects other words not in its domain's vocabulary.

Baldi's animation is controlled by a wireframe model. Parameters can be changed to move the vertices on the face by geometric functions. Interpolation is also used between face shape parameters (e.g., cheek and neck).

Any image from a person can be mapped onto Baldi's 3D polygon surface to make Baldi into a familiar face. In addition, to help facilitate visual speech recognition, Baldi's skin can be made to be transparent in order to see the movements of the inner mouth. Simple emotions can also be displayed by Baldi (e.g., sad, angry, surprised, etc.). In order to create speech, Baldi can use pre-recorded speech from a person or use the FESTIVAL text-to-speech synthesis system [3].

Since Baldi is aimed at the educational community that is primarily comprised of students and teachers, Baldi works with the CSLU Toolkit, which is a set of tools that

allows for the development of spoken language systems [26]. In this toolkit, people can easily create dialogue systems for BALDI using the toolkit's Rapid Application Developer that provides an easy graphical interface to drag and drop objects to design interactive dialogues.

2.2.2 Speech Driven 3-D Animation

There has also been several recent attempts at creating non-photorealistic, cartoon-like 3-D avatars. A system developed by Hofer, Yamagishi, and Shimodaira creates a human avatar with lip motions from a speech signal [10]. Hofer et al.'s system predicts the lip motions of the avatar from the speech signal by generating animation trajectories using a Trajectory Hidden Markov Model [16]. There is a reason they chose a Trajectory Hidden Markov Model to predict the lip motions. The reason is because it would give a smooth output for such lip motions. A regular HMM would provide very disjointed motions that would have been visually unsatisfying.

To get the corpus that the Trajectory Hidden Markov Model would be trained on, Hofer et al. had an actor read 500 newspaper sentences. Markers were placed on the face, and only two distinct lip features were calculated: the mouth opening and the lip pucker. The Trajectory Hidden Markov Models were trained on speech features and the lip tracking data simultaneously using the Maximum Likelihood Criterion. The synthesized trajectory follows the original trajectory relatively close. The major drawback Hofer et al. admit is that in the current implementation, only four points around the lips were tracked, which gave impoverished models. However, Hoffer et al. assert that more than four points can easily be tracked. Using more than four points around the lip for training would give more types of lip motions and overall better video quality.

2.2.3 Text-to-Speech 3-D Animation

Research by Microsoft Research Asia, also has some similarities to the research presented in this thesis. In their paper, Wang et al. present a real-time text to audio-

visual speech synthesis system [18]. The key difference, however, between Microsoft Research Asia's work and the work presented in this thesis is that Wang et al.'s focus is on creating a 3D cartoon avatar, while the research presented in this paper focuses on a photorealistic/videorealistic avatar.

In order to go from text to speech, Wang et al. use a HMM approach in generating a speech synthesizer [17]. The HMM models for the synthesizer are pre-trained for spectrum, pitch, and duration. To give the cartoon avatar natural head movements while talking, an HMM is also used to generate the head movements of the cartoon avatar, which would correlate with the speech prosody features. The head movement HMM was trained from a female announcer on a video broadcasting show. Of key interest is Wang et al.'s work in creating facial animations for the avatar.

To create realistic looking facial movements (lips, eyes, etc.) key-frames were used. Key-frames are the primary frames used in the animations, which the rest of the 3D cartoon animations are derived from. For the lip synchronization, Wang et al. grouped each set of phonemes into a viseme category [20], where each group of phonemes in a certain viseme category has the same lip structure when that particular phoneme is pronounced. From the synthesized speech transcription, the lip-synchronized timing can be synched with the speech. There are also several key-frames used for both the eye blinking and the facial emotions. From these key-frames, spline interpolation is used to create the animation. The animation is controlled by using a skeleton-muscle based physical model that is deformed by changing 3-D model parameters.

Chapter 3

Speech Technology

A variety of speech technology is used in the real-time generation technology and prototype systems. All of the technology discussed in this section was developed by the Spoken Language Systems group, with the exception of DECtalk. Several types of technology are discussed in this chapter: a web-based multimodal speech toolkit, a speech recognizer, and two text-to-speech synthesizers.

3.1 Web-Accessible Multimodal Interfaces toolkit

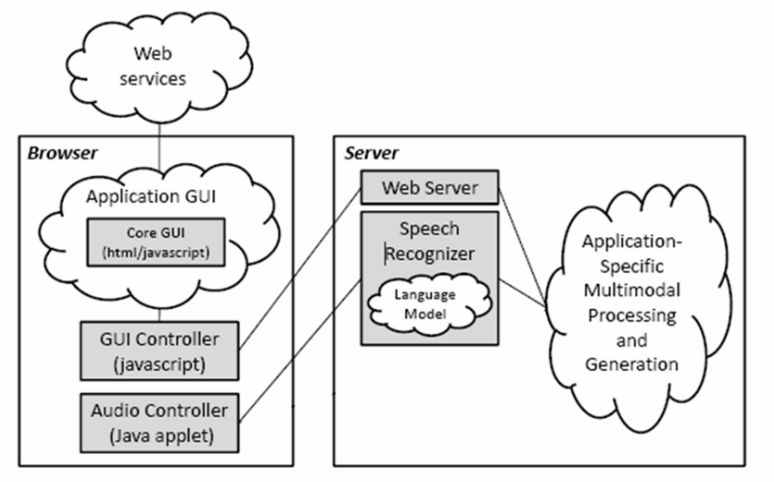

Figure 3-1: Diagram of the WAMI architecture [4]

Created at MIT by Gruenstein et al., WAMI consists of several components that are split up across both the server and the client, which is running a web browser [4]. The server side provides the speech recognizer, and the applications written through WAMI for the server side is written in Java. Each application on the server side must provide a language model for the speech recognizer and code that must handle any user input coming from the client-side. In addition, the speech recognition on the server-side is based on a n-gram language model. As the user interacts with the WAMI-based application, the speech recognition system can dynamically send speech recognition hypotheses. These hypotheses can be encoded in XML and sent to the client-side, which will handle the message accordingly.

WAMI allows, on the client-side, to easily make a GUI, which allows messages to be exchanged with the server through a GUI controller created by AJAX. An audio controller component also runs on the client-side that allows user speech to be sent back to the server, and for audio to be streamed to the client. The audio controller component is a Java applet.

3.2 Speech Recognition

SUMMIT is the speech recognizer used in this system that was developed at MIT. SUMMIT is a feature-based recognizer that takes a speech utterance, and from the feature vectors in the acoustic observation space, a segment based Viterbi search is used to find a hypothesized word sequence [13].

3.3 Speech Synthesis

Two text-to-speech synthesizers can be used to create the synthesized audio and phonetic transcription needed for the real-time Generation system discussed in Chapter 5. One option is ENVOICE, which is a concatenative text-to-speech system developed at MIT [15]. The DECtalk system is another text-to-speech synthesizer used in the real-time generation system. DECTalk is a more generalized speech synthesizer that

can produce any utterance that can be typed [29]. The trade-offs between the two is that DECtalk sounds less human and more robotic, but is very efficient at handling any type of sound that can be typed. ENVOICE, on the other hand, is audibly more appealing, but because it is trained on a limited domain, its vocabulary is limited.

3.3.1 ENVOICE

ENVOICE is a system that is trained on a pre-recorded speech corpus, and from this speech corpus, it can concatenate segments of speech to create newly synthesized phrases and sentences [15]. Finite State Transducers are used to efficiently search the pre-recorded speech corpus to find the best segments that fit the inputted phrase or sentence that needs to be synthesized. As mentioned previously, the downside of using ENVOICE is that its vocabulary is limited to the pre-recorded speech corpus.

3.3.2 DECtalk

DECtalk was created largely by Dennis Klatt in the 1980s, and it intially ran on machines specifically built for DECtalk [29]. As time progressed and computational power increased, instead of taking up the majority of a CPU's load, DECtalk could run on any average computer. Highly efficient and streamlined, DECtalk has the ability to speak in several voices (both male and female) and say practically any known sound that can be expressed by the English alphabet.

Chapter 4

Mary 101 and Morphable Models

This chapter presents the work done by Ezzat et al. on Mary101, a system that can create videorealistic and photorealistic facial animations, which can utter any phrase desired. The main theoretical foundation of Mary101, Morphable Models, is also discussed in detail.

4.1 Mary 101

The backbone of this thesis relies on the work done by Ezzat et al.'s research that focused on creating videorealistic and photorealistic speech animations [9]. For a video to be photorealistic, it must have the correct visual facial structure of a person. For a video to be videorealistic, it must have the correct motion, dynamics, and co-articulation effects [21]. Ezzat et al.'s research achieves both videorealism and photorealism.

To yield new video from a set of pre-defined videos, novel use of the multidimensional morphable model (MMM) [6] was employed to create a vector space where new mouth configurations were made from a pre-defined set of mouth images. The images from the video corpus are used to create the shape and appearance axes, which are used as the basis vectors in the MMM space. In addition, a trajectory synthesis technique was created to synthesize trajectories in the MMM space from the video corpus for any new mouth movement. The phone sequence from the user-provided

transcription was utilized as input for the trajectory synthesis in order to map to a trajectory of parameters in the MMM space. More details about multidimensional morphable models are discussed in Section 4.2.

Ezzat et al.'s training for the system worked in the following manner: recorded video of pre-determined speech, which was pre-processed, was converted to a sequence of video frame images. Then, these images were normalized to remove the head movement, which were subsequently aligned with the audio spoken. The MMM space would be created from a set of these video frame images. The images would be analyzed to produce the necessary phonetic models that are needed for the trajectory synthesis.

In the creation of new video, when new audio is given, which is phonetically aligned, a trajectory in MMM space is synthesized, and from this space, a new sequence of mouth movements can be created. Once the new sequence of mouth movements is obtained, the novel mouth images are composited onto a background video sequence, which contains the subject (the subject is the person that the animation is based on) having natural facial movements.

At the time this system was created, the MMM synthesis took about seven seconds per frame. With today's computational power, the time can be cut rather considerably, but even given today's technology, the generation of novel, new video could not be in real-time using this technique. Optimizations were made by a group at the Industrial Technology Research Institute (ITRI) to make the video generation in real-time using the same method used by Ezzat et al. These optimizations are used in creating the real-time generation system discussed in Chapter 5.

4.2 Multidimensional Morphable Models

The foundation of Ezzat et al.'s video generation is based on multidimensional morphable models. The main assumption made when creating this MMM space is that the mouth movements can be described fully in a low-dimensional space where the axes are the mouth shape and mouth appearance. From the video training a set of

prototype images are extracted. A small number of prototype images are extracted from the pre-recorded training corpus using K-means clustering. For Mary101, 46 prototype images were extracted. From these prototype images, the mouth appearance axis is represented. Optical flow vectors computed from the prototype images represent the mouth shape axis.

The optical flow vectors from the prototype images are computed in the following manner: First, one of the prototype images must be designated as the reference image. This reference image is usually an image where the person's mouth is closed and eyes are open. The optical flow is then computed with respect to the reference image, and the data is stored into two displacement vectors. The optical flow of the reference image would be zero because there is no displacement from itself.

The entire MMM space can be parameterized by shape parameters α and appearance parameters β. The trajectory synthesis outputs the (α, β) parameters computed from the speech transcription. Given that there are 46 prototype images, there are also 46 optical flow correspondences. Therefore, α is a 46-dimensional parameter vector. Analogous to the dimensionality of α, since there are 46 prototype images, β is a 46-dimensional parameter vector. This gives a total dimensionality of 92 for the MMM space. Given a set of input parameters from the trajectory synthesis module (α, β), synthesis is performed by morphing the prototype images to a corresponding shape-appearance configuration described by α and β. For a more detailed explanation of this process, please refer to [9].

Chapter 5

Real-Time Generation

The main contribution of this thesis lies in the creation of the real-time text-to-speech video generation system. Given English text, this technology can, in real-time, generate video of the avatar speaking the text. This chapter discusses, in detail, the design of this technology. The issues encountered when creating the technology are also presented, and the solutions devised to solve these issues are discussed.

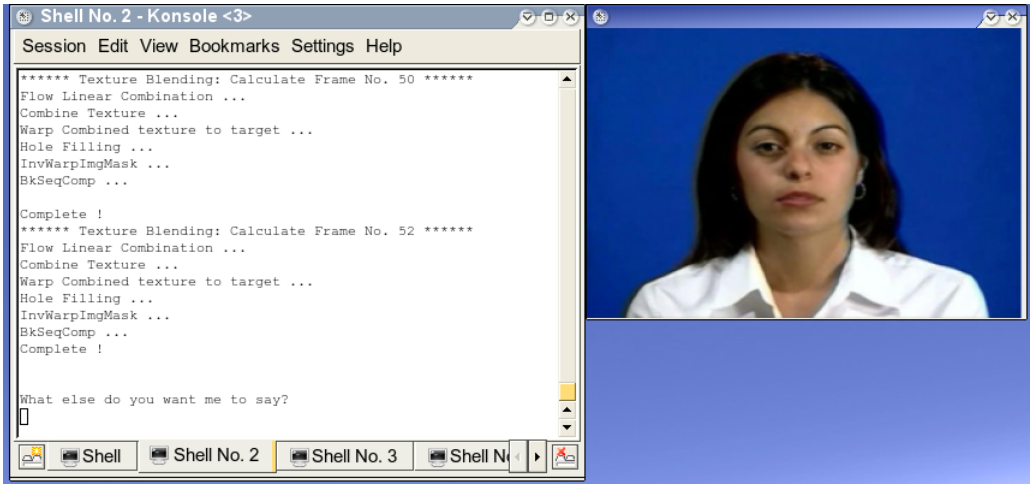

Figure 5-1: Real-Time System Screenshot

The core functionality of this technology was designed using C++ in conjunction with the SDL [1] and FFMPEG libraries [8]. The system consists of three components: the Text-to-Speech Synthesis module, the Video Generation module, and the Video Player module. The Text-to-Speech Synthesis module takes in English text as input

and outputs the phonetic transcription of the text and the corresponding computer synthesized audio. The Video Generation module receives the phonetic transcription from the Text-to-Speech Synthesis module, and from the Text-to-Speech Synthesis module, video frames are created which correlate with the utterance from the phonetic transcription. As the Video Generation module creates the video frames in real-time, the frames are passed along and stored into the video buffer of the Video Player module. The Video Player module pulls the frames stored from the Video Generation module, and tries to synchronize those frames with the synthesized audio received from the Text-to-Speech Synthesis module. The synchronized video and audio are displayed on the screen and played through the speakers accordingly. Figure 5-2 illustrates a diagram of the real-time text-to-speech video generation system.

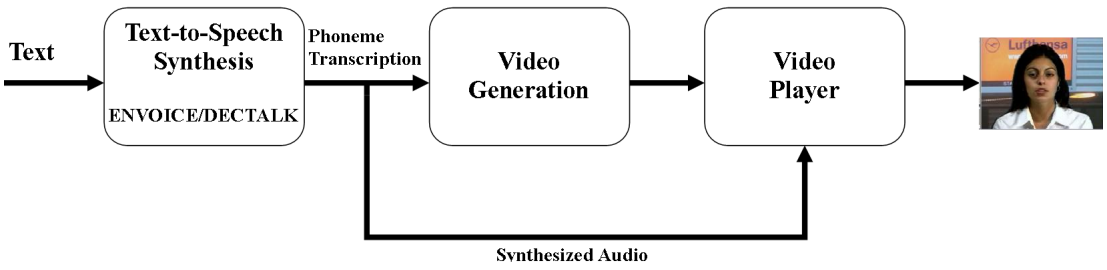

Figure 5-2: High-Level Diagram of the Text-to-Speech Real-Time Generation System

5.1 Text-to-Speech Synthesis

The Text-to-Speech Synthesis module works in the following manner: given text from the user, the Text-to-Speech Synthesis module converts the text to computer synthesized audio, and it also outputs a phonetic transcription. The phonetic transcript consists of a sequence of phone tuples; where each tuple is a phone label, a start time and a end time.[1] Once the phonetic transcription and synthesized audio are created, the phonetic transcription is sent to the Video Generation module.

[1] A phone is the basic unit of sound used in speech.

The user has the option of using two different Text-to-Speech synthesizers for this system: DECtalk and ENVOICE. Care must be taken because both DECtalk and ENVOICE do not output a phonetic transcription in the exact form expected by the Video Generation module. Therefore, a conversion process occurs before the transcription is sent to the Video Generation module to convert the phonetic transcription into the proper expected format. The Video Generation Module expects the phonemes to be in the CMU SPHINX phone set [30], and the time intervals must be in seconds.

5.1.1 ENVOICE

Interfacing with ENVOICE is rather simple. Using just one command, ENVOICE can deliver a phonetic transcription of the text inputted, and also a sound WAV file containing the spoken synthesized text. However, the phonetic transcription outputted by ENVOICE uses a slightly different phone set and time measurement than what is expected by the Video Generation module. A simple look-up table is used to convert the phonemes outputted by ENVOICE to the phonemes expected by the Video Generation module. In addition, because the time measurement by ENVOICE is based on 8000 samples per second, a division by 8000 converts the times outputted by ENVOICE into the proper time expected by the Video Generation module (seconds).

5.1.2 DECtalk

Interfacing with DECtalk to get a proper phonetic transcription and synthesized WAV file was more challenging than interfacing with ENVOICE. DECtalk's API must be used directly and low level manipulation of the audio buffer had to be done in order to obtain both a WAV recording of the synthesized audio and a phonetic transcription. Getting the WAV recording through DECtalk can be done with relative ease, but one needs to manually force DECtalk to go through the synthesized audio buffer properly to create the necessary phonetic transcription.

Moreover, DECtalk outputs a numerical value instead of an actual phoneme, and DECtalk expects this numerical value to be used in its look-up table to convert it to a phoneme in the ARPABET phone set. However, as mentioned previously, the Video Generation module expects the phonetic transcription to be using the CMU SPHINX phone set. Thus, instead of passing along this numerical phonetic represention to DECtalk's ARPABET look-up table, a separate look-up table was created, and the numerical phonetic values were intercepted and passed along to this CMU SPHINX look-up table. Furthermore, in order to create more natural lip movements for the avatar, several phonemes were split apart into two separate phonemes. In order to create coherent consistency, when a selected phoneme was designated to be split, the time interval was also halved and distributed equally.

5.2 Video Generation and Video Player

Both the Video Generation module and the Video Player module interact with each other very closely. The Video Generation module must constantly output frames, both at times when the avatar is talking and when the avatar is silent. Otherwise, a still image is displayed onscreen. Therefore, when the avatar is supposed to be silent (because there is no inputted text), the Video Generation module needs to constantly create silent video frames to store in the Video Player module's video buffer. Furthermore, knowing that the Video Generation module must constantly output video frames, the Video Player module constantly tries to pull video frames from its video buffer and audio from its audio buffer to synchronize and play through the computer.

The Video Generation and Video Player module are comprised of four primary threads, which communicate extensively with each other. The "Main" thread is the fundamental thread of both the Video Generation and Video Player modules. This thread spawns the "Video Display" thread, which handles the Video Player module's video functionality, and the "Audio" thread, which handles the Video Player module's audio functionality. In addition, the "Main" thread does the essential work for the

Video Generation module by generating the frames of the avatar talking. Moreover, the "Main" thread spawns a helper thread called "Video Loop," which does the video frame generation of the frames when the avatar is silent. Figure 5-3 illustrates the inner workings of the threads.

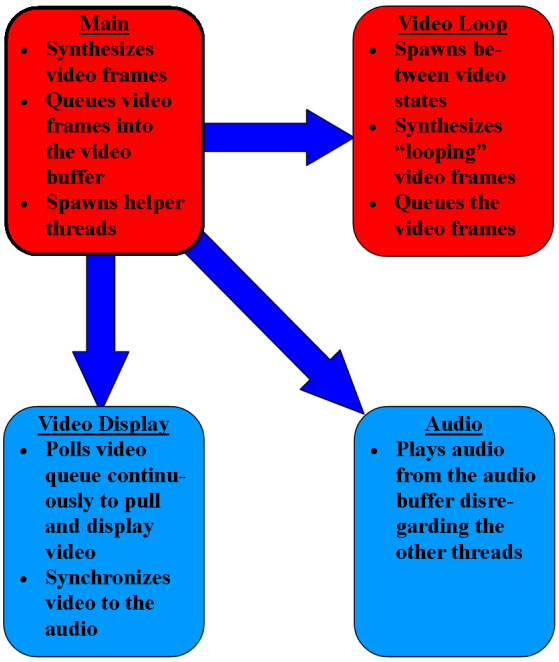

Figure 5-3: Primary Threads Diagram

5.2.1 Video Generation

A synthesis text file needs to be stored beforehand before running the real-time generation system. This file contains the necessary location-based information that points to the various pieces of data needed to perform the video generation. The synthesis text file contains information like the locations of the phonetic transcription, the synthesized WAV file, the prototype images, and other information that is necessary to generate video. Whenever new video needs to be generated for a new utterance, the data locations written in the synthesis text file are referenced.

After reading in the necessary data needed to generate the video of a new utterance, the real-time generation system performs the same morphable model procedure

that is discussed in detail in Chapter 4. While the video frames are being generated as quickly as possible, they are stored into the Video Player module's video frame buffer. When the Video Generation module is finished with generating new video of the avatar saying a phrase, it will spawn a new "Video Loop" thread that will continue to create silent video frames, while the "Main" thread waits for more input from the Text-to-Speech synthesis module. When the Video Generation module receives new input from the Text-to-Speech synthesis module, the "Video Loop" thread is joined back with the "Main" thread, and the whole process described in this section repeats itself.

5.2.2 Video Player

The Video Player module's main role is to properly synchronize the audio and video and display and play them accordingly. In order to do so, this module must properly access both the video frame buffer and audio buffer. Whenever a new video of the avatar saying something needs to be played, the Video Generation module fills the Video Player module's video buffer with frames, as it produces them; however, the "Main" thread completely stores all of the audio data into the audio buffer all at once when it intially opens the audio WAV file. This is done so that the video can be synchronized based on the audio, since it would be more difficult to synchronize the audio to video that is being generated in a unreliable, non-uniform manner.

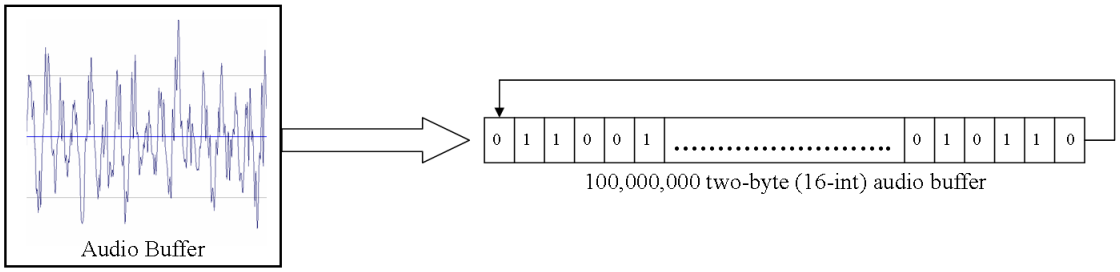

Figure 5-4: Low-level internal representation of the audio buffer

As Figure 5-4 shows, the audio data is loaded completely into the audio buffer. The Video Player module constantly pulls a portion of audio data from the audio

buffer to play, and from knowing the sample rate of the audio, the time according to the audio can be easily calculated. Using this audio-based time, the Video Player module can synchronize the video frames to the audio. As a side note, if the audio buffer is filled with zeros (i.e., the avatar is not saying anything) the Video Player module will still read the buffer thinking there is valid data in it; however, if for some reason the Video Player module reaches the end of the buffer (which can only happen if the avatar is silent for an extended period of time) it will loop back to the beginning of the audio buffer.

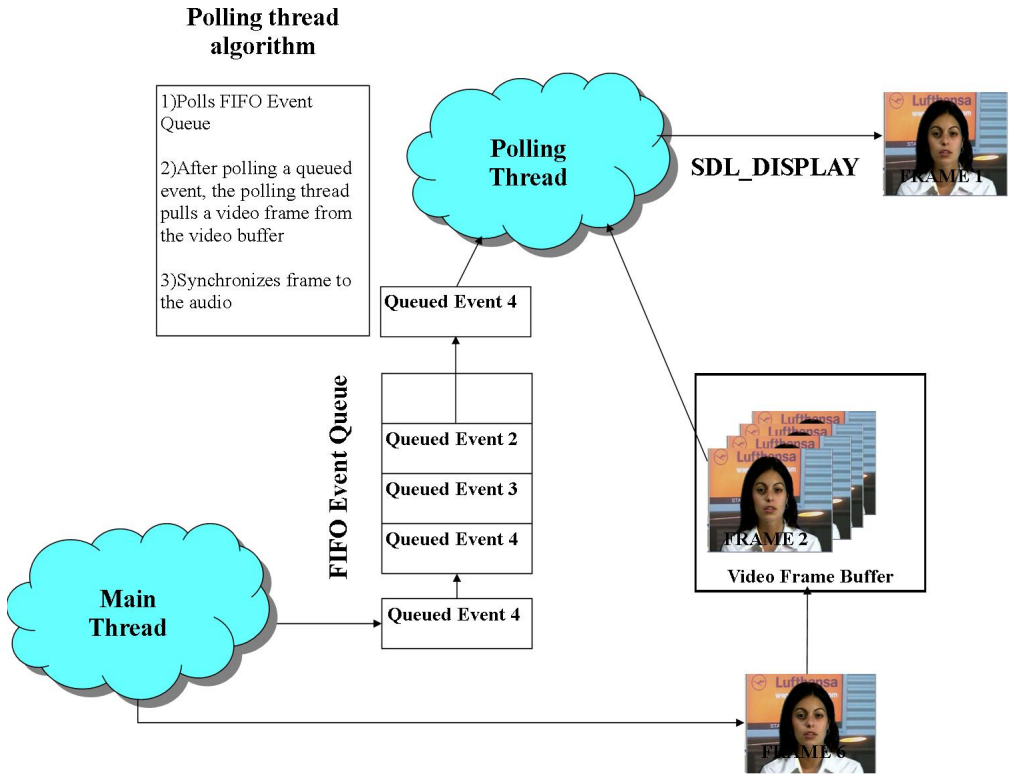

Figure 5-5: Graphical representation of the polling algorithm

When the "Main" thread of the Video Generation module loads a video frame into the Video Player module's video buffer, the Video Generation module queues an "event" into a queue telling the Video Player module that a video frame is ready to be pulled from the buffer, synchronized with the audio, and displayed. As soon as a video frame is pulled from the video buffer, the time according to the video frame must be determined. Using the frame number of the video frame and the frames per

second that the video was sampled, the video-based time can be calculated. If the video frame is determined to be too far ahead of the audio, the next video frame is delayed from being obtained from the video buffer. If the video frame is determined to be too far behind the audio, the current video frame is instantly displayed, and the next video frame is instantly pulled from the video buffer. If the video frame is determined to be close enough to the audio time (within a certain threshold), the video frame is displayed according to its current video-based time. Given this back and forth nature of delaying or advancing video frames, the Video Player module always keep the video frames as synchronized as possible to the audio.

5.3 Issues Encountered

5.3.1 Looping Issue

A major issue that was encountered was how to properly transition from the state where the avatar is talking to the silent, "looping" state where the avatar is silent and waiting for the next thing to say. For the first iteration of this system, there was no "Video Loop" thread, and the "Main" thread did the looping video generation. This caused problems because while the "Main" thread was generating the silent frames, it could not listen for any new input from the Text-to-Speech synthesizer. Therefore, it was decided upon that the "Main" thread would spawn a helper thread that would create the silent frames, while the "Main" thread listens for more input. This solution solved the problem partially, but it also created a new problem. The silent videos would be input into the buffer too early before the active, talking video frames were finished being displayed, which caused audio synchronization problems. In addition, when the audio in the buffer was finished playing, leftover data in the memory would be accidently read, which would cause noise to be emitted from the speakers.

In order to rectify these unforeseen problems, the active, talking video frames had to be close to being completely flushed out of the video buffer before inserting the new silent, "looping" video frames. Once the video buffer is almost empty, the audio

is paused and no more audio would be pulled from the buffer for a brief moment. The silent video frames start to get loaded into the video buffer, and all zeros are stored into the audio buffer. Once the audio buffer is reloaded and re-initialized with silence, the audio is un-paused, and the system goes back to pulling audio data from the buffer. In addition, the frame number of the looping video frames are re-initialized back to zero, in order to get the silent video frames properly aligned with the new silent audio. When the system needs to stop producing silent video frames, the "Video Loop" helper thread is joined back into the "Main" thread. The same process of reinitializing the audio and video buffer at the appropriate times is also executed when the system transitions back to generating active, talking video frames.

5.3.2 Audio Buffer Issue

It became apparent after running the real-time generation system on other computers that a synchronization problem was occurring where the mouth of the avatar was slightly ahead of the audio. The problem was narrowed down to the audio buffer; not the software audio buffer discussed in this chapter, but the actual hardware implementation of the audio buffer. Because each computer contains a differently sized audio buffer that the operating system can access, a fencepost error type problem occurs. It is hypothesized that the audio data from the software audio buffer gets read into the hardware audio buffer, and then played through the computer's speakers. The audio does not get played directly from the memory accessed by the software, as what was initially assumed. Because of this issue, if a computer's hardware buffer is sufficiently large, there can be an inherent delay and an inconsistency between the audio-based time in the software audio buffer and the time according to the audio in the hardware buffer. To solve this subtle issue, the size of the hardware audio buffer is dynamically determined when the system starts, and depending on the size of the actual hardware buffer, a delay τ is added to the audio-based software time to compensate for the added delay caused by the large hardware audio buffer.

Chapter 6

Prototype Avatar-based Speech Interfaces

Two prototype systems are presented in this chapter. One system is built to operate on the web, and it is called Mary101.NET. The other system, CGI.Mary101, runs locally on a computer. Both systems perform similar functions that allow the user to interact with an avatar to modify a flight itinerary.

6.1 Mary101.NET

Much of what people do on the computer today is through an internet browser. It is only natural to create an avatar that works through a web browser. Mary101.NET presents a scenario that allows the user to interact with an avatar to modify a flight itinerary by voice. The ease of use and the responsiveness of the system were key interests that were taken into heavy consideration when implementing Mary101.NET. The system is based on the infrastructure provided by the Web-Accessible Multimodal Interface (WAMI) toolkit.

The details in this section are organized in the following manner: First, a description of how the user would interact with the system is presented. Then, the actual implementation of Mary101.NET and the issues encountered when designing this system are discussed.

6.1.1 System Use

Figure 6-1: Screenshot of Mary101.NET

Mary101.NET is a web-based speech interface that uses a directed dialogue system and the pre-generated video system that was created by Ezzat et al. When the user loads the web page containing the system, he or she is greeted with two buttons that lets him or her choose between two different flight scenarios: Flight Canceled and Options Menu. Clicking on the Flight Canceled button mimics a scenario where the user's flight is canceled, and clicking on the Options Menu button mimics the scenario where the user just wants to modify his or her flight itinerary. Clicking on either button leads to identical options the user can choose to interact with the system. The only thing that is different between the two initial scenarios is the introduction avatar video that is streamed through the browser after the user clicks on either button. Furthermore, a Push to Talk button in the upper left hand corner

allows the user to interact with the avatar by voice, after he or she clicks on either of the two initial buttons. This speech button is always visible to the user throughout the use of Mary101.NET.

After the user clicks on either scenario button, the avatar introduction video is streamed. Text always accompanies the spoken dialogue of the avatar in the video. In the introduction video, the avatar either says that the person's flight has been canceled and that he or she must choose one of the several options, or the avatar immediately introduces the options he or she can make. As mentioned previously, this introduction video is dependent on the user's initial button click. The user is given the option to either rebook his or her flight, schedule a return flight, or request a refund.

When the user verbally selects one of the three options, a new avatar video is streamed through the web browser asking if the user wants to confirm his or her choice. If the user decides to cancel the current option, he or she will be reverted back to the initial option menu, and the avatar will reintroduce the menu to him or her. If the user, on the other hand, confirms the current option, the state will change and a new avatar video will be streamed confirming the user's choice. Confirming his or her choice will end the operation of the system; however, the user can at any time reload the page and restart the system from the very beginning.

6.1.2 Mary101.NET Design

Using the WAMI toolkit as the foundation of its design, Mary101.NET is primarily composed of a speech recognizer, a Flash video player, and an AJAX-powered GUI Controller, which is controlled by the web server. In this section, the design of Mary101.NET is discussed. In addition, the problems that arose during the construction of Mary101.NET and the solutions created for these problems are discussed.

Speech Recognition

WAMI's speech recognition module is based on SUMMIT [13], which was developed at MIT and is discussed in more detail in Chapter 3. In order to use the WAMI toolkit's speech recognition module, a grammar file written in the JSGF (Java Speech Grammar Format) standard needed to be created. The grammar file created for Mary101.NET detects whether the user says the following base commands: yes, no, rebook, return flight, or refund. The user can say more complicated phrases, but only the base command will be parsed. The speech is sent from the audio controller on the client side to the incremental speech recognition module located on the server. When the incremental speech recognition module has confirmed that it has recognized and hypothesized the utterance sent from the user, Java code written for the server determines which state the system should be in, given the speech hypothesis provided by the speech recognition module.

```
#JSGF V1.0;
grammar FlightRebook;

public <top>    = (<command>)
        ;

<command>       = <yes>
| <no>
| <rebook>
| <return>
| <refund>
;

<yes>= yes {[command=yes]} [ please ]
;

<no>= no {[command=no]} [ thanks | thank you ]
;

<rebook>= rebook {[command=rebook]} [ please ]
;

<return>= return flight {[command=return]} [ please ]
;

<refund>= refund {[command=refund]} [ please ]
;
```

Figure 6-2: Mary101.Net's Grammar File

Finite State Machine

The finite state machine (FSM) devised for Mary101.NET is composed of seven states. The states are used to determine which avatar video should be streamed and which text should be displayed. When the user starts the system and is presented with the two button choices (canceled flight or options menu), either button click will take the user to the main state of the system, which is the Option Menu state. This is the hub state where all the secondary states can return. From the Option Menu state, the introductory video of the avatar is streamed, and from the user's speech command, the FSM will transition to one of the three option states provided by the option menu

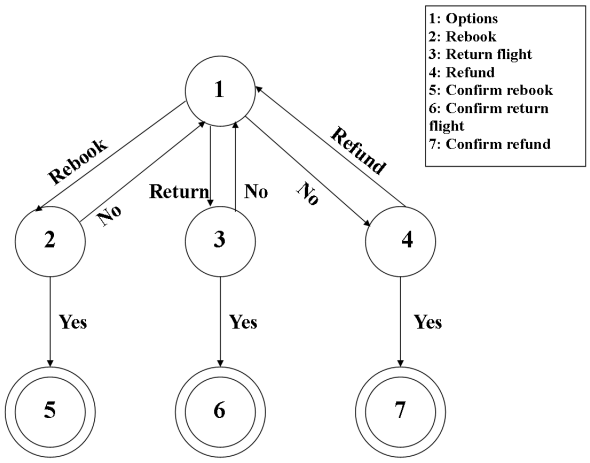

Figure 6-3: Diagram of the Finite State Machine used in Mary101.NET

(Rebook, Return Flight, Refund). When the FSM is at any of the three option states, the machine can either transition back to the main Option Menu state or transition to a final confirmation state. The system concludes once it has transitioned to any of the final confirmation states.

Client-Server Interaction

Using the WAMI toolkit, the audio from the user's voice on the client side is sent to the speech recognizer on the server side. The server determines the appropriate state transition, and a XML message is sent back to the client's browser telling the browser what video and text to display. The client's browser is updated by the use of Javascript, which updates the video/text on the client's screen and also receives the messages from the server.

Avatar Video

The videorealistic avatar videos were created beforehand, and for this system, the avatar is female. The videos were pre-generated using a MATLAB script. Given a sound wave file for the voice and the phonetic transcription of the sound file, the video files are created by running the MATLAB script. The video frames created are under the assumption that the video frame rate will be according to the NTSC format, 29.97

frames per second. FFMPEG [8], an open source, cross-platform video utility, was used to take the outputted image frames and the sound wave file and transform them to Flash video. Once all the Flash avatar videos were created, they were embedded into Mary101.NET using HTML and Javascript. The videos produced were 640 pixels wide by 480 pixels high.

Because the video is being streamed through a client's browser over limited bandwidth, the avatar video's file size was an issue. Streaming uncompressed video and audio would make Mary101.NET unusable. With a large video size, users would have to wait seconds, if not minutes, for the system to interact with them, and the interaction would seem unnatural at best. One reason Flash was chosen for the video format is because of its nice compression aspects. The audio for the Flash video is encoded in the MP3 format, and the video stream is encoded with a variant of H.263/H.264. The video's audio was compressed more heavily than the actual video, since audio fidelity was an extremely low priority when compared to having acceptable video quality. The only priority for the audio quality is that it is comprehensible and clear.

Moreover, Flash was chosen because of its compatibility between different browsers (Internet Explorer, Firefox, etc.) and operating systems (Linux, OS X, Windows). Several other video formats were explored before settling on Flash, specifically Windows Media Video and MPEG. Windows Media Video and MPEG would work in one environment but not the other, and not all the web-based players could play these two formats properly without the user obtaining the proper add-on. It would be difficult to force some users to track down a web video player that plays the video for their system properly. For the Flash video, all the user needs is any flash plug-in for his or her browser, which is extremely easy and painless to obtain because Flash is relatively ubiquitous on the web.

Unlike other web video players available, the Flash video API was much simpler to use. Using the Flash video API, starting and stopping the avatar videos were simple, and modifying the video player itself was extremely straightforward. Taking all of these positive, unique qualities of Flash into consideration, it should be readily apparent why Flash was the video type of choice in implementing Mary101.NET.

Flash would provide a consistent user experience across all major platforms.

6.2 CGI.Mary101

This section presents CGI.Mary101, which is a standalone prototype system that runs locally on one's machine. Like Mary101.NET, it presents a scenario allowing the user to change his or her flight itinerary. There are times when having a standalone system running locally is beneficial. For instance, the data bandwidth limitation that is inherent with using a web-based system is much less of a issue when running on a standalone, local machine. In addition, delay between the user's input and the system's output would be kept to a minimum.

This section is organized as follows. First, an explanation of how the system works and an explanation of how the user would interact with CGI.Mary101 is presented. Afterwards, the multimedia library is introduced. In conclusion, the implementation of CGI.Mary101 is described in depth.

6.2.1 System Use

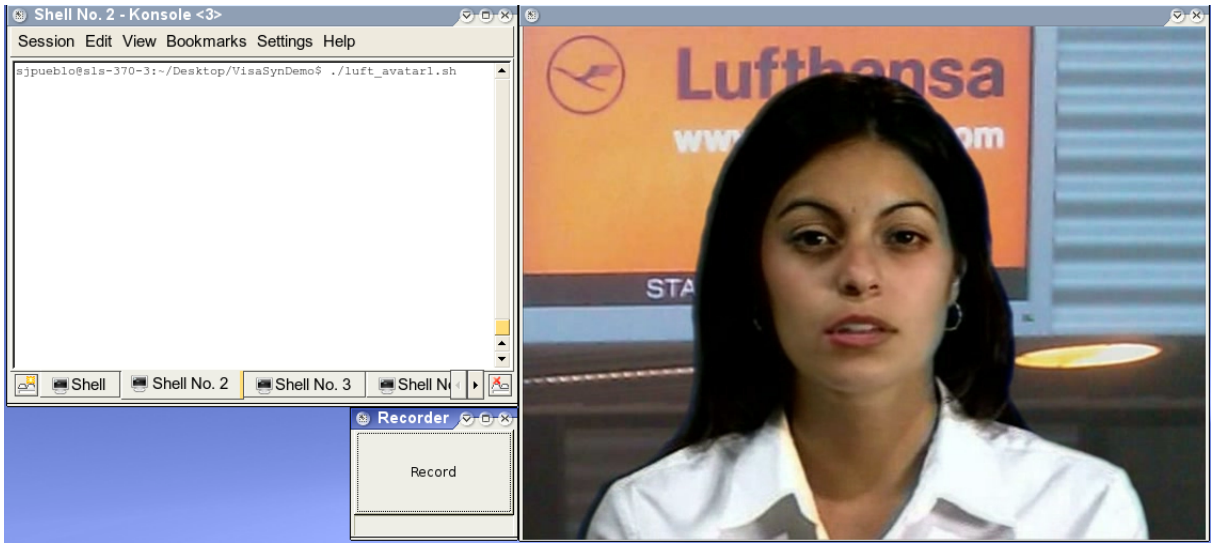

Figure 6-4: Screenshot of CGI.Mary101

Using CGI.Mary101, the user has the option of loading two different scenarios. The first scenario the user can load is practically identical to the Flight Canceled scenario in Mary101.NET. The avatar mimics the situation of the user's flight being canceled and allows the user to change his or her flight itinerary. The second scenario the user can load is different. In this second scenario, the user's flight still gets canceled, but instead of just changing his or her flight itinerary, the user can also change his or her itinerary to involve traveling by train. The only difference in presentation between Mary101.NET and CGI.Mary101 is that Mary101.NET includes text with the video; however, CGI.Mary101 does not include text, and the video is displayed in a standalone window.

For information on how scenario one works, please refer to Section 6.1.1. Scenario two works in the following manner: When the user loads the second scenario, the system starts, and the avatar informs the user that his or her flight has been canceled and he or she has the option to either rebook another flight or travel by train to his or her destination. When interacting with this system, the user must click and hold the Record button, and then say the appropriate command. If the user confirms by voice

that he or she is willing to take the train, the system transitions to another video informing the user that he or she has booked to travel by train, and the system ends. On the other hand, if the user confirms by voice that he or she wants to rebook to fly by plane, the avatar will confirm the request. After confirming the user's request, the avatar asks the user whether he or she would like to print his or her boarding pass or have the boarding pass sent to his or her mobile phone. The user chooses between the two print options, and then the avatar confirms the option and thanks the user, which subsequently ends the system. Once the system ends, the user has the option to restart it by saying something to the avatar; the system will restart to its initial state.

6.2.2 CGI.Mary101 Design

CGI.Mary101 uses the real-time infrastructure described in Chapter 5. However, transcriptions of the speech are pre-written and stored, and whenever CGI.Mary101 needs to change states, a different transcription is fed to the real-time infrastructure in order to create video of Mary101 saying a new utterance. CGI.Mary101 also uses a SUMMIT-based speech recognizer like Mary101.NET, but this one runs locally on the machine and is wrapped in Python [13]. More information on Summit can be found in Chapter 3.

Speech Recognition

When the user clicks on the Record button, the Python-based speech recognizer uses the already stored JSGF (Java Speech Grammar Format) file to parse the command from the user's speech. When the speech recognition module figures out which command the user said, it sends that command to the main processing module that contains the state machine.

Finite State Machine

The first scenario follows the same state machine as shown in Figure 6-3 and described in Section 6.1.2. The second scenario works in the following manner: The first state (state 0) outputs the introductory video, and it waits for the plane or train command from the speech recognition module. If the first state receives the plane command it transitions to state 1 and outputs the corresponding video, and it waits for either the print or phone command. When state 1 receives the phone command, it transitions to state 4 and outputs the corresponding video, or if state 1 receives the print command, it transitions to state 3. On the other hand, if state 0 receives the train command it transitions to state 2. If the FSM is in either states 2, 3, or 4 and it receives any other command, it will revert back to its initial state 0. Figure 6-5 details the FSM.

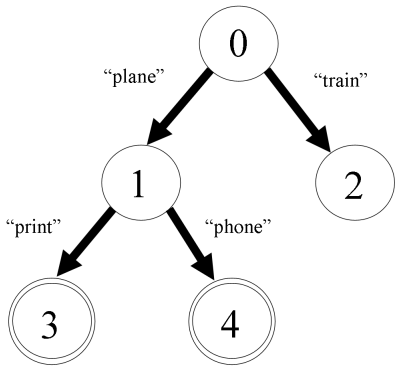

Figure 6-5: Diagram of the Finite State Machine used in CGI.Mary101 Scenario #2

Chapter 7

User Study

In order to assess whether adding speech animation to a system enhances the end user's experience, a user study was performed. Of keen interest for doing this study was to figure out whether people felt comfortable having a human-like video avatar conversing with them. Similar to studies done with human-robotic interactions, we wanted to see whether the video avatar reached an uncanny valley [22] for people, where people would not use a system with such an avatar that looked eerily like a human being but lacked human-like qualities.

7.1 User Study Design

Forty-one people were subjects for this study. The study was done in a public area of MIT's campus. For around five minutes of their time, they were given a chocolate bar. There were no requirements for taking the study, and anyone walking by the user study booth was allowed to participate.

After consenting to take part in the survey, instructions were presented to the user. There was no time limit given to any of the subjects, and the subjects were not told beforehand whether the video of the avatar was computer generated. The instructions directed the user to click through the three buttons on the screen. These three buttons present the user with three distinct scenarios. These three scenarios are called baseline, cartoon, and video. Each scenario is a mockup of a form of

media composited to a screen shot of the Flight Browser system developed here at MIT [11]. The baseline scenario presents the screenshot of Flight Browser along with audio reading the text onscreen saying that your flight has been canceled. The cartoon scenario presents the same screenshot as the baseline scenario, but unlike the baseline scenario, in addition to audio reading the text, a static cartoon image of a human was also presented. The video scenario was the final scenario the subjects could click on, and with this scenario, video of the avatar was presented reading the text onscreen from the screenshot.

There is a reason the three scenarios were chosen. Several sites and applications use either audio only or a still image with audio (e.g., Alaska Airlines) [2]. Designing two scenarios that are similar to actual web sites and applications in use today would provide an excellent baseline for comparison with the video avatar. After clicking on

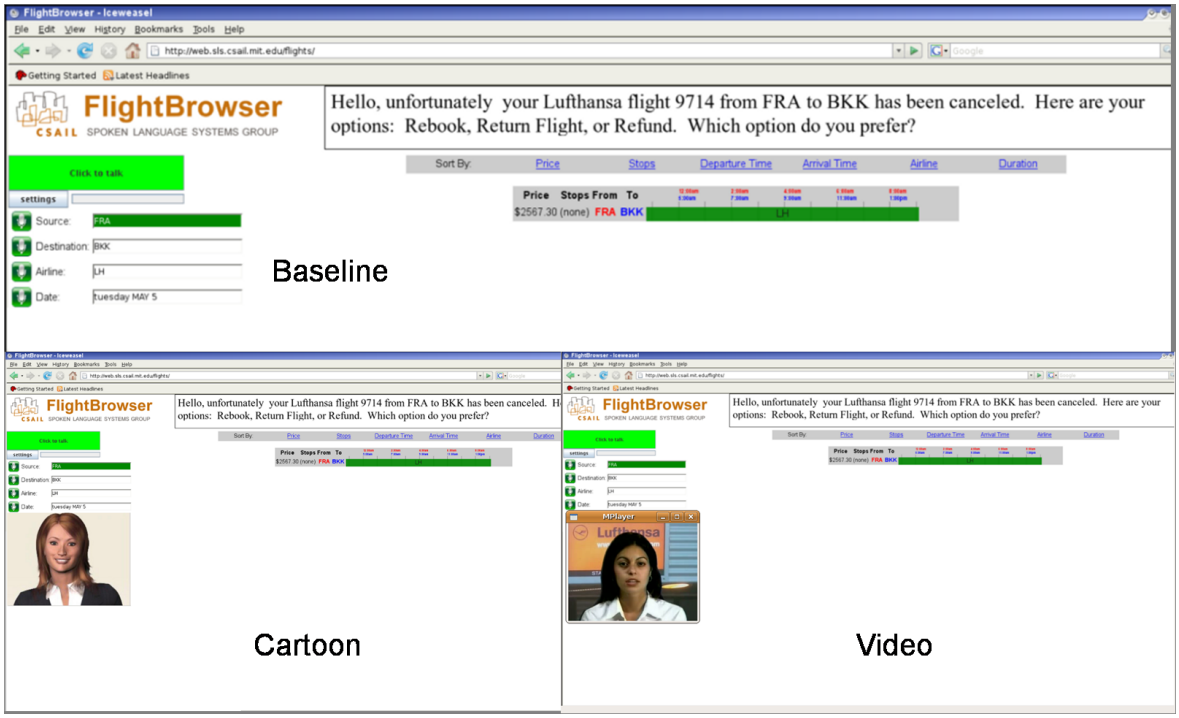

Figure 7-1: Scenarios used in the User Study: Baseline, Cartoon, and Video

the three scenario buttons and observing the different scenarios, the subjects were forced to take a brief survey detailing their preferences for each scenario. For each scenario, they were asked how intelligible and natural the scenario was. In addition,

for each scenario, subjects had to answer whether they would use a system based on such a scenario, and the overall rating for that particular scenario. In the last section of the survey, the subjects had the opportunity to rank the best scenario out of all three scenarios.

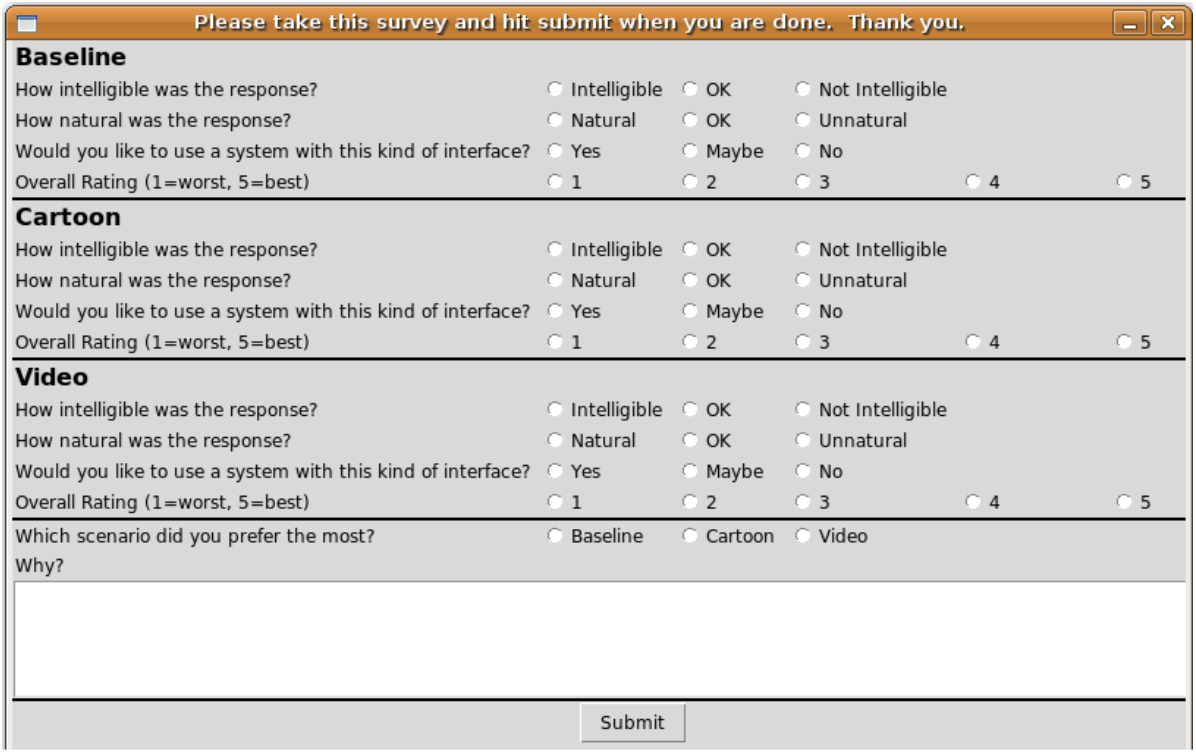

Figure 7-2: Screenshot of the User Study Survey

7.2 User Study Results

The user study provided insightful information on how people perceive computer generated media. Numerous users had responses to the different scenarios that were not hypothesized beforehand. Five categories from the three scenarios are analyzed: Intelligibility, Naturalness, Overall Quality, System Use, and User's Favorite. The data analyzed in the following sections were taken from the users through the survey that they submitted after completing the assigned task [1].

[1] In order to understand the subsequent graphs that are going to be presented, refer to Appendix B to see the numerical mappings from the descriptive ratings to their actual numerical scores that

In order to test the significance of the results from the user study, McNemar's Significance Test was used [2]. Determing significance is of key importance in order to determine which results are substantial and valid and which results are most likely influenced by noise and chance.

7.2.1 Intelligibility

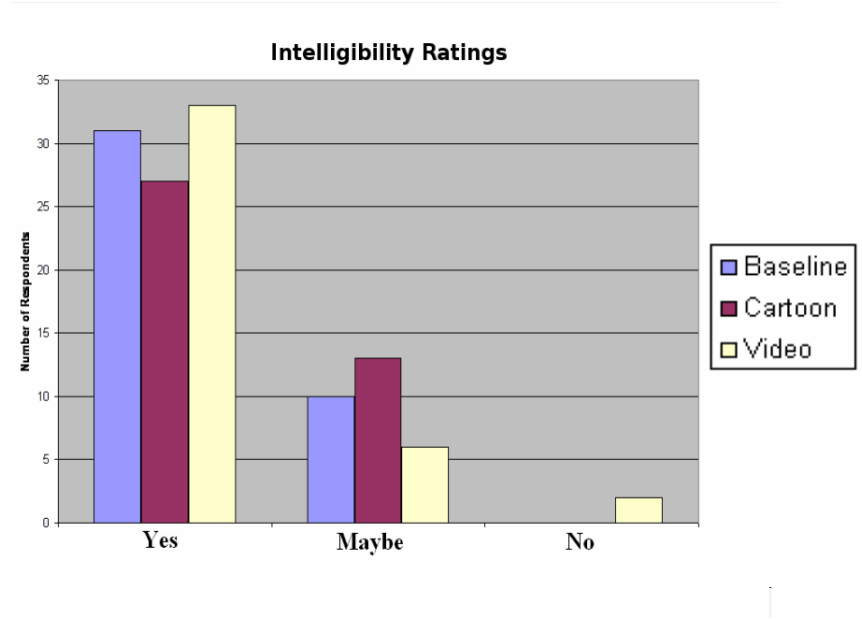

Figure 7-3: User responses on the intelligibility of each scenario

were used in determing the averages.

[2]In order to deem results as significant, the probability threshold from the McNemar's Significance Test was decided upon to be less than .05. For more information on McNemar's Significance Test, please refer to Appendix A

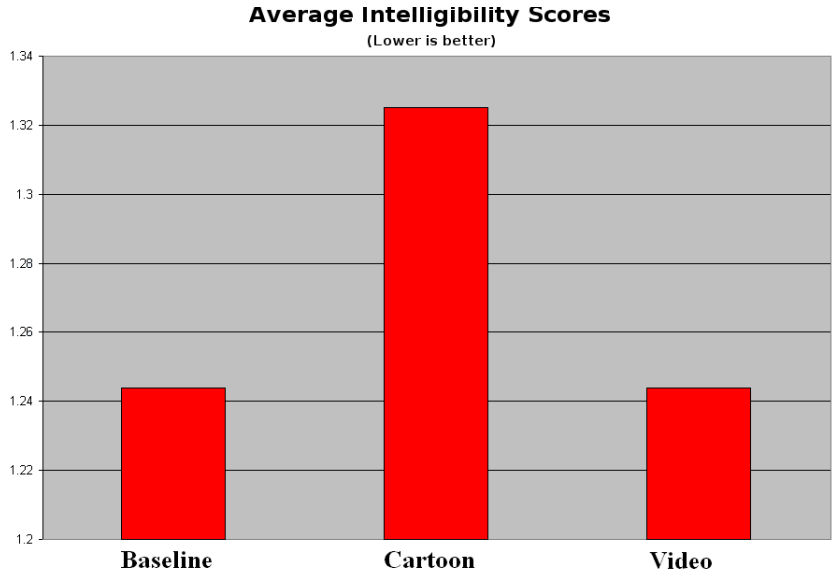

Figure 7-4: Average Intelligibility Scores (Lower is Better)

With regards to intelligibility, it was expected that the intelligibility scores should have been the same between all three scenarios, since the audio was identical between all three. However, as one can see from the graphs, that was not the case. Analyzing the above graphs, the video avatar was significantly better than the cartoon when it came to intelligibility. This result makes intuitive sense because one would hope that a static cartoon image with no moving lips would be less intelligible than a video. With regards to comparing the video to the baseline of audio only, there is no discernable difference between the two when it comes to intelligibility. One could hypothesize that since people are used to hearing audio only in a multitude of areas, the addition of video would have a negligible effect on intelligibility; however, one should take notice that the addition of video to the baseline did not adversely affect intelligibility, which is a plus.

7.2.2 Naturalness

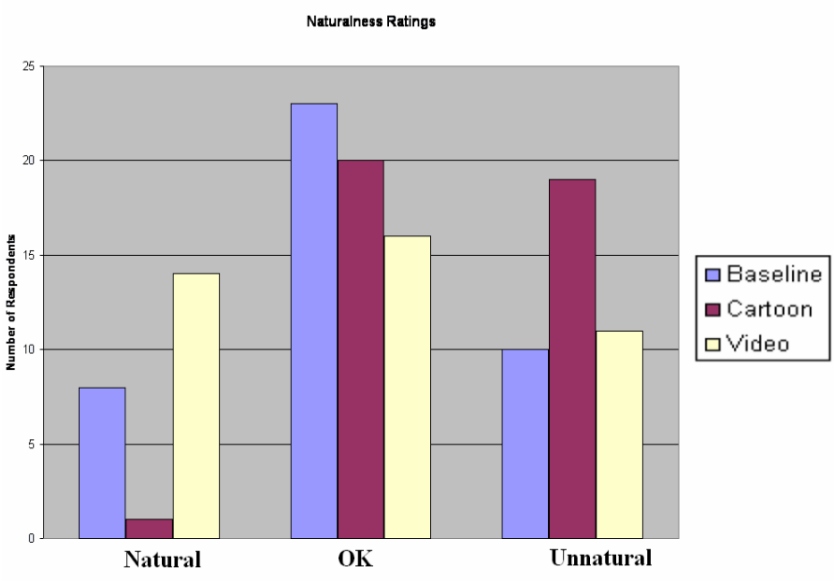

Figure 7-5: User responses on the naturalness of each scenario

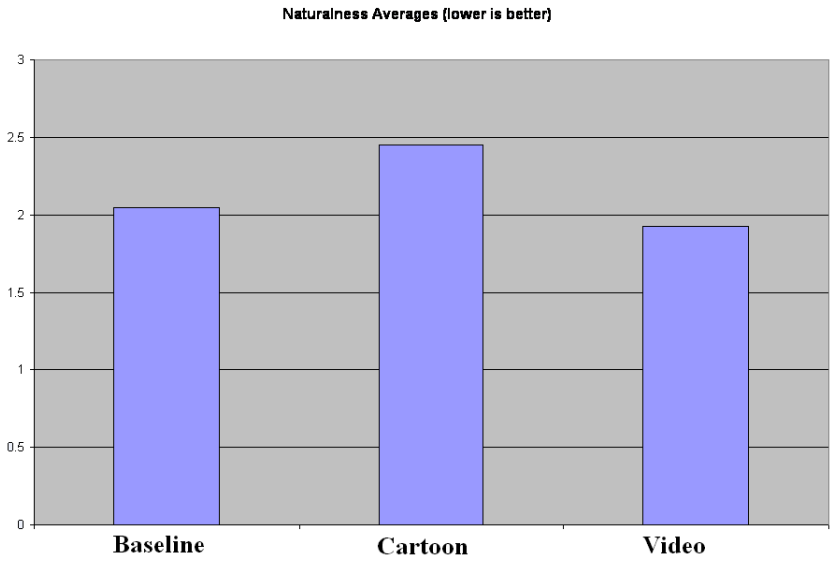

Figure 7-6: Average Naturalness Scores (Lower is Better)

Users were asked to rate naturalness of each scenario in order to determine whether the avatar video disturbed people in an uncanny manner. The subjects of this user study were asked how natural each scenario was. The results are rather interesting.

As expected, the video scenario was deemed the most natural, and the cartoon was deemed as the most unnatural. This makes perfect sense because there is nothing natural about having a static image talking to you, where the lips are not moving. The baseline and video were both significantly more natural than the cartoon. These results all fall in line with what was hypothesized beforehand. Nothing out of the ordinary stands out in these results.

7.2.3 Overall Quality

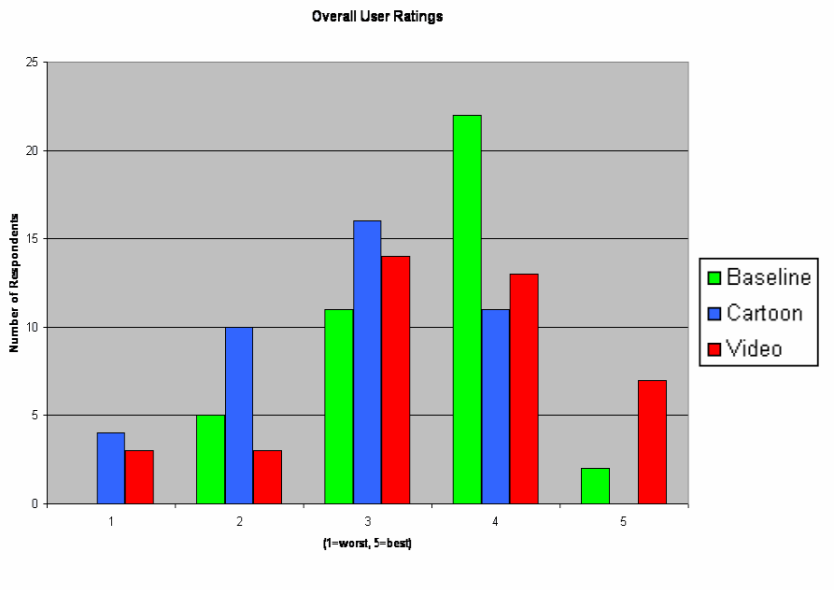

Figure 7-7: User responses on the quality of each scenario

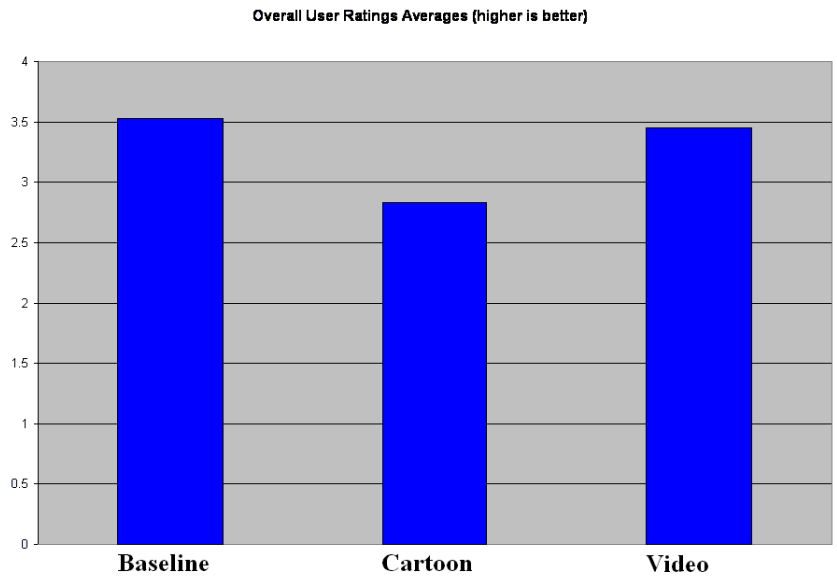

Figure 7-8: Average Overall Scores (Higher is Better)

The Overall ratings were less clear in determing the best scenario. Both the baseline and video were considered significantly better than the cartoon scenario. The baseline and video perform rather well in people's opinions, which is slightly surprising. The video did not outperform the baseline.

7.2.4 System Use

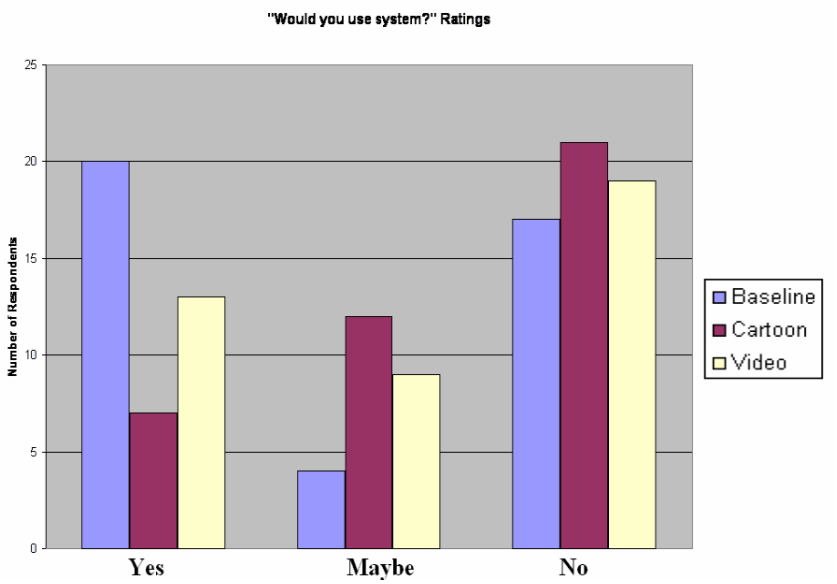

Figure 7-9: System Use Ratings

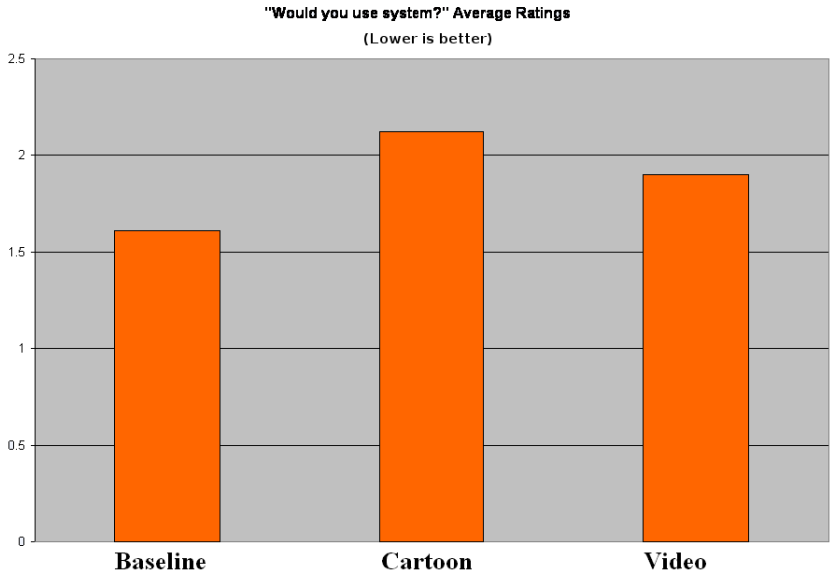

Figure 7-10: Average System Use Scores (Lower is Better)

The graphs above detail users' preference over which scenario they would use in real life. The results were roughly right-sided, which was initially a surprise, but after some thought it makes sense. The scenarios presented to the subjects were

overly simplistic, and the real description of what the mock scenarios would actually do was never explained to the subjects. It would make sense why a considerable amount of people would not prefer to use either of these scenarios presented, since they did not know what a full system would actually do. In retrospect, this question presented to the subjects in the survey was poorly worded, and it should have been modified.

7.2.5 User's Favorite

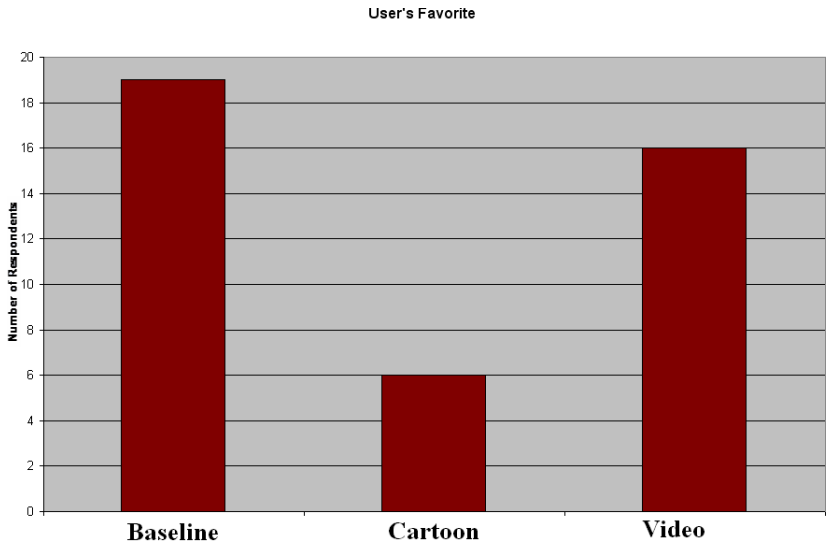

Figure 7-11: Favorite Scenario Graph

The baseline scenario was the users' favorite scenario. The video scenario was the users' second favorite, and the cartoon scenario came in last. It is difficult to pinpoint why the Baseline scenario was the most preferred; however, one can assume that maybe users are more comfortable with audio only, since audio is more prevalent in web systems than computer generated avatars. Another reason might be that some users noticed uncanny qualities with the avatar in the video. Regardless it is tough to pinpoint the exact reasons why the Baseline were preferred more than the Video. Still, the number who preferred the Video was not too far behind. Maybe, with a larger sample size, the results would have been different, but it is hard to tell. A difference of only three subjects between the Baseline and Video scenarios out of a

total of forty-one subjects, does not conclusively say that Video scenario was inferior. It is easier to conclude that people preferred both the Baseline and Video scenarios relatively equally.

Chapter 8

Discussion and Future Work

The use of the real-time generation technology has the potential to be integrated in a variety of systems that could greatly enhance the user's experience. Imagine an instance where someone is heading to the airport kiosk to retrieve and print their itinerary and tickets. Normally, one would interact with the kiosk by pushing a touch sensitive screen with no interaction from the kiosk other than text on the screen. Now imagine a new, futuristic kiosk that has the ability to converse with the user through speech with the use of speech recognition and a videorealistic avatar. For the future work with this real-time generation technology, we are indeed trying to make this vision come true.

There has been web-based technology developed at MIT where, using speech recognition and natural language understanding, one can easily, in real-time, find information about current flights by using his or her voice as input. This system is called FlightBrowser [11]. The real-time generation technology could be integrated into such a system, so that instead of hearing solely audio when the system finds your requested flight information, a videorealistic avatar could appear. The use of the videorealistic avatar could, if Shechtman and Horowitz are correct in their study [23], provide users with the necessary environment to interact with the system on a more natural and personal level.

The user study described in Chapter 7 provided very interesting insights into the use of videorealistic facial animations in potential everyday systems. The user study

results showed that the computer generated video performed rather well and that a good portion of people would be willing to use it. However, user comments about the video of the avatar brought to light issues with the current version. Several users noticed that something was not entirely human with the video. For instance, some users detected that the eye blinking was at a constant rate, while others noticed that the head and neck had limited to no movement. It can be argued that the uncanny valley was somewhat reached with the use of the videorealistic avatar [22]. In order to better assess the comfortableness the user has towards a system that uses an avatar, in the future, a more in-depth user study could be performed where participants could actually evaluate a complete system that has integrated the avatar. Performing such a user study with a full system would give a better sense of users' preferences.

Along with integrating the real-time generation technology into other systems, the addition of emotions to the facial animation could greatly enrich the current system. Instead of the avatar looking constantly stoic and emotionless, the addition of even a small smile during the avatar's silence could greatly enhance the current system. Other modifications to deliver more emotion and add more humanity to the avatar could also be researched.

The user study has proven that the addition of facial animations to systems has the potential viability to make the overall user experience more enjoyable. Currently, it seems that some users still cling to what they feel at ease using (i.e., text or audio) when interacting with a computer, but as facial animation technology progresses, it is only a matter of time before people start to feel comfortable enough to interact with such animations on a daily basis.

Appendix A

McNemar's Significance Test Tables

McNemar's Significance Test is used to determine the significance between two correlated and related proportions. It was first given by Quinn McNemar, who published the first tests in 1947 [24].

A.1 Intelligibility

Baseline \Cartoon	**1**	**2**	**3**
1	24	7	0
2	3	6	0
3	0	0	0

(a) Baseline vs. Cartoon

Cartoon \Video	**1**	**2**	**3**
1	25	0	2
2	27	6	0
3	0	0	0

(b) Cartoon vs Video

Baseline \Video	**1**	**2**	**3**
1	29	1	1
2	3	5	1
3	0	0	0

(c) Baseline vs Video

Table A.1: Intelligibility Tables

A.2 Naturalness

Baseline \Cartoon	**1**	**2**	**3**
1	0	5	3
2	1	13	8
3	0	2	8

(a) Baseline vs. Cartoon

Cartoon \Video	**1**	**2**	**3**
1	0	0	1
2	6	8	6
3	7	8	4

(b) Cartoon vs Video

Baseline \Video	**1**	**2**	**3**
1	3	2	3
2	9	9	6
3	3	5	2

(c) Baseline vs Video

Table A.2: Naturalness Tables

A.3 User Ratings

Baseline \Cartoon	**1**	**2**	**3**
1	5	4	10
2	0	2	2
3	2	6	9

(a) Baseline vs. Cartoon

Cartoon \Video	**1**	**2**	**3**
1	4	2	1
2	5	2	5
3	4	5	12

(b) Cartoon vs Video

Baseline \Video	**1**	**2**	**3**
1	6	6	8
2	2	0	2
3	5	3	9

(c) Baseline vs Video

Table A.3: User Ratings Tables

A.4 Quality Ratings

Baseline \Cartoon	**1**	**2**	**3**	**4**	**5**
1	0	0	0	0	0
2	0	0	5	0	0
3	2	3	3	2	0
4	2	6	8	7	0
5	0	1	0	1	0

(a) Baseline vs. Cartoon

Cartoon \Video	**1**	**2**	**3**	**4**	**5**
1	0	0	2	2	0
2	1	1	3	2	2
3	1	1	4	8	2
4	1	2	5	0	3
5	0	0	0	0	0

(b) Cartoon vs Video

Baseline \Video	**1**	**2**	**3**	**4**	**5**
1	0	0	0	0	0
2	0	0	0	5	0
3	1	1	1	4	3
4	2	3	11	3	3
5	0	0	2	0	0

(c) Baseline vs Video

Table A.4: Quality Ratings Tables

Appendix B

User Responses

The table below shows all of the users' responses from the user study conducted.

Intelligibility	
1	Intelligible
2	OK
3	Not Intelligible

(a) Intelligibility Legend

Naturalness	
1	Natural
2	OK
3	Unnatural

(b) Naturalness Legend

Quality	
1	Poor
2	Below Average
3	Average
4	Above Average
5	Excellent

(c) Quality Legend

User's Favorite	
1	Baseline
2	Cartoon
3	Not Intelligible

(d) User's Favorite Legend

Table B.1: User Responses Legend Tables

User	Baseline Intelligibleness	Baseline Naturalness	Baseline User Rating	Cartoon Intelligibleness	Cartoon Naturalness	Cartoon User Rating	Video Intelligibleness	Video Naturalness	Video User Rating	Overall Rating Baseline	Overall Rating Cartoon	Overall Rating Video	User's Favorite
1	2	2	3	2	3	3	2	1	3	4	3	5	3
2	2	2	3	2	3	2	2	2	3	3	1	4	3
3	1	3	3	1	3	3	1	2	1	3	3	4	3
4	2	2	3	1	2	1	1	1	1	3	4	5	3
5	1	2	1	1	3	2	1	1	1	4	3	4	3
6	1	2	1	1	3	2	1	2	3	4	1	3	1
7	1	1	1	1	2	3	1	1	3	5	4	3	1
8	2	3	1	2	3	3	2	2	1	4	2	4	1
9	1	3	3	1	3	3	1	3	2	4	3	2	1
10	1	3	2	2	3	2	1	1	3	2	3	4	3
11	1	2	1	1	2	3	3	3	2	4	4	2	2
12	1	2	3	1	2	3	1	2	3	3	3	3	1
13	1	2	1	1	2	3	1	2	3	4	3	3	1
14	1	3	2	1	2	2	1	2	3	2	3	4	3
15	2	3	2	1	2	3	1	1	1	N/A	4	5	3
16	2	3	3	2	3	2	2	3	2	4	2	1	1
17	1	2	3	1	3	2	1	1	1	3	1	4	3
18	1	1	3	1	2	2	1	2	3	4	3	4	1
19	2	2	3	1	1	1	3	3	2	3	2	5	2
20	1	2	1	1	3	2	1	2	1	4	1	3	1
21	2	3	1	2	3	1	1	1	1	2	3	4	3
22	1	2	3	1	2	3	1	1	3	4	4	5	3
23	2	2	3	N/A	N/A	2	1	1	1	3	3	5	3
24	1	3	2	1	3	3	1	2	1	2	3	4	3
25	1	1	1	2	3	N/A	1	3	3	5	2	3	1
26	1	1	1	1	3	3	1	1	3	4	2	3	1
27	2	2	3	2	2	3	2	3	3	4	4	3	2

User	Baseline Intelligibleness	Baseline Naturalness	Baseline User Rating	Cartoon Intelligibleness	Cartoon Naturalness	Cartoon User Rating	Video Intelligibleness	Video Naturalness	Video User Rating	Overall Rating Baseline	Overall Rating Cartoon	Overall Rating Video	User's Favorite
28	1	2	3	1	2	3	1	3	3	4	4	3	1
29	1	1	1	2	3	3	2	3	2	3	2	2	1
30	1	2	1	1	2	3	1	2	3	4	2	3	1
31	1	1	1	2	2	3	1	2	2	4	4	3	N/A
32	1	3	3	1	3	3	1	2	3	2	3	4	3
33	1	2	1	1	2	1	1	3	2	3	4	1	2
34	1	2	1	1	2	1	1	2	3	4	4	3	2
35	1	1	1	1	2	1	1	3	1	4	3	3	1
36	1	2	1	2	3	2	1	1	2	4	2	N/A	1
37	1	1	1	2	2	1	1	1	1	4	2	5	3
38	1	2	1	1	2	3	1	3	2	4	3	1	1
39	1	2	3	1	2	3	1	2	3	3	4	4	2
40	1	2	1	1	3	3	1	2	3	4	3	3	1
41	1	2	3	2	2	2	1	1	1	3	2	4	3

Table B.2: Table of User Responses

Bibliography

[1] http://www.libsdl.org/.

[2] Alaska Airlines. http://www.alaskaair.com.

[3] A. Black and P. Taylor. Festival Speech Synthesis System: System Documentation (1.1.1). *Human Communication Research Centre Technical Report HCRC/TR-82*, 1991.

[4] I. McGraw A. Gruenstein and I. Badr. The WAMI Toolkit for Developing, Deploying, and Evaluating Web-Accessible Multimodal Interfaces. In *Proc. ICMI*, Crete, Greece, October 2008.

[5] S. Liu S. Roberts J. Zabel B. Reimer B. Mehler S. Seneff J. Glass J. Coughlin A. Gruenstein, J. Orszulak. City Browser: Developing a Conversational Automotive HMI. In *Proc. CHI*, pages 4291–4296, Boston, Massachusetts, April 2009.

[6] V. Blanz and T. Vetter. A Morphable Model for the Synthesis of 3D Faces. In Alyn Rockwood, editor, *Proc. SIGGRAPH 2001*.

[7] M. Covell C. Bregler and M. Slaney. Video Rewrite: Driving Visual Speech with Audio. In *Proc. SIGGRAPH 1997*, Los Angeles, CA, August 1997.

[8] Dranger. FFMPEG Tutorial. http://www.dranger.com/ffmpeg/, December 2003.

[9] T. Ezzat. *Trainable Videorealistic Speech Animation*. PhD thesis, Massachusetts Institute of Technology, 2002.

[10] J. Yamagishi G. Hofer and H. Shimodaira. Speech-Driven Lip Motion Generation with a Trajectory HMM. In *Proc. Interspeech*, Brisbane, Australia, September 2008.

[11] G. Matthias. Incremental Concept Understanding in a Multimodal Web-Based Spoken Dialogue System. Master's thesis, Massachusets Institute of Technology, June 2008.

[12] H. Dubberly and D. Mitch. The Knowledge Navigator. Video Tape.

[13] J. Glass. A Probabilistic Framework for Segment-Based Speech Recognition. *Computer Speech and Language*, 17:137–152, 2003.

[14] J. Spool. Knowledge Navigator Deconstructed: Building an Envisionment, June 2007.

[15] J. Glass J. Yi and L. Hetherington. A Flexible, Scalable Finite-State Transducer Architecture for Corpus-Based Concatenative Speech Synthesis. In *Proc. ICSLP*, Beijing, China, October 2000.

[16] T. Masuko T. Kobayashi K. Tokuda, T. Yoshimura and T. Kitamura. Speech parameter generation algorithms for HMM-based speech synthesis. In *Proc. ICASSP 2000*, page 1315, June 2000.

[17] K. Tokuda, H. Zen, J. Yamagishi, T. Masuko, S. Sako, A. W. Black, and T. Nose. The HMM-based speech syntheseis system (HTS). http://hts.ics.nitech.com.

[18] L. Ma Y. Qian Y. Chen L. Wang, X. Qian and F. K. Soong. A Real-Time Text to Audio-Visual Speech Synthesis System. In *Proc. Interspeech*, Brisbane, Australia, September 2008.

[19] K. Liu and J. Ostermann. Realistic Facial Animation System for Interactive Services. Brisbane, Australia, September 2008.

[20] M. Brand. Voice puppetry. In *Proc. SIGGRAPH 1999*, pages 21–28, New York, New York, 1999.

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

[22] M. Mori. Uncanny valley. *Energy*, 7(4), 1970.

[23] N. Schechtman and L. Horowitz. Media Inequality in Conversation: How People Behave Differently When Interacting with Computers and People. In *Proc. CHI2003 Conference on Human Facts in Computer Systems*, New York, New York, 2003.

[24] Q. McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12:153–157, 1947.

[25] J. de Villiers B. Rundle K. Shobaki J. Wouters M. Cohen J. Beskow P. Stone P. Connors A. Tarachow R. Cole, D. Massaro and D. Solcher. New tools for interactive speech and language training: Using animated conversational agents in the classrooms of profoundly deaf children. In *Proc. ESCA/SOCRATES Workshop Method and Tool Innovations Speech Science Education*, page 45, London, United Kingdom, 1999.

[26] Y. Yan P. Vermeulen R. Cole, S. Sutton and M. Fanty. Accessible technology for interactive systems: A new approach to spoken language research. In *Proc. ICSLP*, page 3221, Sydney, Australia, November 1998.

[27] S. Seneff, R. Lau, J. Glass, and J. Polifroni. The Mercury System for Flight Browsing and Pricing. *MIT Spoken Language System Group Annual Progress Report*, pages 23–28, 1999.

[28] V. Zue, S. Seneff, J. Glass, J. Polifroni, C. Pao, T.J. Hazen, L. Hetherington. Jupiter: A Telephone-Based Conversational Interface for Weather Information. *IEEE Transactions on Speech and Audio Processing*, 8(1):100–112, January 2000.

[29] W. I. Hallahan. DECtalk Software: Text-to-Speech Technology and Implementation. *Digital Technical Journal*, 7(4):5–19, April 1995.

[30] X. Huang, F. Alleva, H. W. Hon, M. Y. Hwang, K. F. Lee, and R. Rosenfeld. The SPHINX-II speech recognition: An Overview. *Computer Speech and Language*, 7(2):137–148, 1993.