5-1-2004

Enhancing Expressiveness of Speech through Animated Avatars for Instant Messaging and Mobile Phones

Joseph E. Pechter
Dartmouth College

Dartmouth College Computer Science Technical Report TR2004-503

Enhancing Expressiveness of Speech through Animated Avatars for Instant Messaging and Mobile Phones

A Senior Honors Thesis
Submitted to the faculty
in partial fulfillment of the requirements for the
Degree of Bachelor of Arts in Computer Science

By
Joseph Pechter
Dartmouth College
Hanover, NH
May 2004

____
Lorie Loeb

____
Hany Farid

____
Stephen Linder

Abstract

This thesis aims to create a chat program that allows users to communicate via an animated avatar that provides believable lip-synchronization and expressive emotion. Currently many avatars do not attempt to do lip-synchronization. Those that do are not well synchronized and have little or no emotional expression. Most avatars with lip synch use realistic looking 3D models or stylized rendering of complex models. This work utilizes images rendered in a cartoon style and lip-synchronization rules based on traditional animation. The cartoon style, as opposed to a more realistic look, makes the mouth motion more believable and the characters more appealing. The cartoon look and image-based animation (as opposed to a graphic model animated through manipulation of a skeleton or wireframe) also allows for fewer key frames resulting in faster speed with more room for expressiveness.

When text is entered into the program, the Festival Text-to-Speech engine creates a speech file and extracts phoneme and phoneme duration data. Believable and fluid lip-synchronization is then achieved by means of a number of phoneme-to-image rules. Alternatively, phoneme and phoneme duration data can be obtained for speech dictated into a microphone using Microsoft SAPI and the CSLU Toolkit.

Once lip synchronization has been completed, rules for non-verbal animation are added. Emotions are appended to the animation of speech in two ways: automatically, by recognition of key words and punctuation, or deliberately, by user-defined tags. Additionally, rules are defined for idle-time animation.

Preliminary results indicate that the animated avatar program offers an improvement over currently available software. It aids in the understandability of speech, combines easily recognizable and expressive emotions with speech, and successfully enhances overall enjoyment of the chat experience. Applications for the program include use in cell phones for the deaf or hearing impaired, instant messaging, video conferencing, instructional software, and speech and animation synthesis.

Table of Contents

Introduction

Our motivation for developing this program is the need for an improved method of “real-time” communication that can be effective, clear, and enhanced by emotionally expressive images. Current chat and instant messaging programs rely on text alone, voice alone, or marginally effective avatars. We create an interactive chat program, X-Speech, that combines all three by displaying messages and combining speech with an expressive, understandable avatar.

Secondly, we realize the need to include images or video in phone technology and the overall necessity of developing methodology that is small enough to be “real-time” and interesting enough to hold the attention of the user. Our program animates speech input quickly by using simple yet effective lip-synchronization and emotion rules to produce recognizable emotions and an appealing avatar.

Our third motivation is the need to make these applications more accessible for the hearing impaired and to improve the understandability of avatars to increase the range of potential uses for the technology. We create a versatile chat program that can be easily accessed and used with a variety of applications and devices without the installation of additional software; this is achieved by using highly supported programming languages. Additionally, our program is flexible because it is able to interchange components, such as speech technologies, which continue to develop.

Finally, we realize that effective and expressive avatars can save time and improve overall communication. Images are used as a vehicle for communication in order to express emotions that can otherwise only be conveyed by many words. Our program makes it easy to use emotion by providing high functionality and an interactive user interface, and we use simple methods that allow selected emotions to be added seamlessly.

Our primary objective is to create a “real-time” avatar chat program that achieves better lip-synchronization than that of currently available software and the ability to incorporate emotions. In Chapter 1, we describe speech technology along with features of animated speech systems. Chapter 2 describes the rationale for emotion used in our program. The context of our program and its modules are explained in Chapter 3 (Fig. 1).

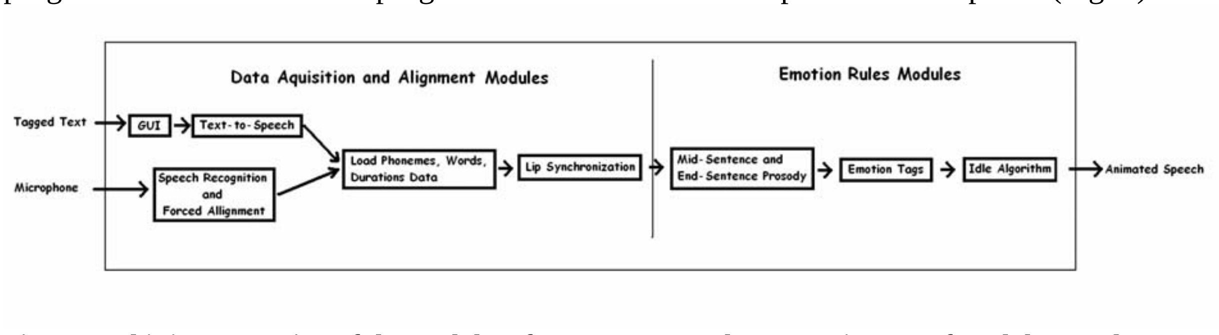

Data Aquisition and Alignment Modules
Emotion Rules Modules
Tagged Text
GUI
Text-to-Speech
Microphone
Speech Recognition and Forced Allignment
Load Phonemes, Words, Durations Data
Lip Synchronization
Mid-Sentence and End-Sentence Prosody
Emotion Tags
Idle Algorithm
Animated Speech

Figure 1. This is an overview of the modules of our program. The two major sets of modules are the Data Acquisition and Alignment Modules and the Emotion Rules Modules.

Figure 1 describes the data acquisition and alignment modules, which take the input from either text from the GUI or the microphone and load it onto the file system in the form of phoneme and phoneme duration data. The data is then synchronized and transferred to the emotion rules modules, which add automatic and user-defined emotions. The data, results, and conclusions are presented in Chapters 4 and 5.

Chapter 1: Utilizing Speech Technology as an Integral Part of Chat Programs

Chat programs, instant messaging, and many other real-time applications rely on current state of the art speech technology. Text-to-Speech (TTS) programs generate speech from text input using phoneme and syllable-based methods. At the most basic level, a phoneme is the smallest unit of sound in a language. Between 39 and 45 phonemes exist in the English language depending on which phoneme alphabet is used. To produce understandable speech, text-to-speech algorithms convert text into phonemes using letter-to-phoneme rules that map text into a corresponding representation of phonemes. Additionally, phonemes are combined into diphones (combinations of two phonemes), triphones (combinations of three phonemes), or syllables, and speech is produced by concatenation. The use of a database of diphones and triphones to produce speech is an improvement over the use of a large database of commonly used words, because less memory is required to store the speech segments, and the program has the ability to pronounce nonsensical words or words that are not in the dictionary (such as names). By utilizing diphone, triphone, or syllable based methods, text-to-speech programs create understandable speech.

Since the advent of text-to-speech technologies in 1939, development of software, hardware, and linguistic analysis techniques have been used to modify speech using prosodromic variation. Mid-sentence and end-of-sentence prosody alter the speech file by inflecting pitch and tempo, and these changes create more realistic-sounding speech. The corresponding positions in the speech where prosody is added are determined by context and punctuation. Words that end in a question mark are recognized by increased pitch and tempo, while exclamations have an increase in volume and intensity. Homograph discrimination is used to distinguish words that are spelled the same but are pronounced differently depending on context. The word wind can be pronounced “wind the clock” or “the wind blew,” and computers are able to differentiate the two by using part-of-speech rules to predict whether, based on the context of the sentence, the word is a noun or a verb.

Current text-to-speech software has strengths and weaknesses. AT&T software uses prosody at the end of sentences and homograph discrimination, and speech sounds more fluid than do the other text-to-speech programs that we test. The most understandable text-to-speech program is Hybrid Text-to-Speech, which uses a diphone-based method, prosody at the end of sentences, homograph discrimination, and boasts a 98% understandability. The Whistler system, which operates by a different mechanism, is a trainable TTS system developed by Microsoft that uses text-analysis and speech recognition to produced high-quality speech output (Huang 4) (Hon 1). While the performance and quality of text-to-speech programs have improved over the past ten years, software is weak in the areas of understandability and prosody. Speech produced for large text input (such as novels) does not hold the attention of the listener and is not expressive. However, for more simple applications, TTS is widely used today for reading internet text, email, telephone systems, bus stations, and for the blind.

Utilizing Speech-to-Text for Real-time Applications

All chat programs that animate speech from a human speaker incorporate speech recognition, and this is an integral part of our program. Speech-to-text is much harder to implement than TTS. Digital signal processing (DSP) uses statistical pattern recognition and acoustic language models to recognize speech by analyzing its digital representation (Matousek 2). It is difficult to obtain a "clean" representation of speech because factors such as background noise, echo, more than one speaker, and inadequate microphones can distort recorded speech. Noise filters are used to eliminate some errors, while techniques such as Hidden Markov Models (HMM), Gaussian mixture models (GMM), linear and nonlinear approximations, and speech smoothing aid in modeling for speech recognition. Recently, speech recognition technology has developed software that utilizes optical-flow algorithms that analyze speech and track lip movements simultaneously to improve recognition (Matousek 8).

Speech recognition technology has much room for improvement. For example, even with recognizing connected digits, human understandability error rate is over 5 times less than that of computers (Huang 12). Better recognitions program will be created that utilize huge databases of speech utterances and create better rules for speech recognition. One project started in Japan, Corpus of Spontaneous Japanese, aims to analyze over 700 hours of speech for use in recognition systems (Matousek 4). Automatic speech recognition can be achieved by using this corpus with acoustic and language models. Additionally, sentence extraction and sentence compaction can be utilized to produce a text representation of speech (Matousek 10).

Effective Animation of Speech

Real-time chat and video conferencing systems are enhanced by using expressive avatars that animate speech. A viseme is a visual representation of a phoneme, and the concatenation of visemes using keyframes and morphing algorithms is similar to phoneme-based methods of text-to-speech. By synchronizing visemes with speech, animated avatars can be created that aid in non-verbal and verbal communication.

Animated avatars are generally chosen to best relate to the personality of the user, and human-like actions are created by synchronizing lips with the speech produced by a text-to-speech engine. Studies have shown that communication is enhanced for chat programs using animated avatars without text-to-speech or lip-synchronization, thereby suggesting that the addition of lip-synchronization will further improve communication (Persson 1).

Table 1 is a chart of a variety of avatar systems and their features.

Table 1. Avatar Systems and Features

System	Web Based	Real-Time	MPEG-4 Standard	Training or Wireframes	Lip. Synch.	Verbal Emotion	Non-Verbal Emotion	End-Sentence Emotion	Mid-Sentence Prosody	XML
Albrecht's Avatar		X					X	X	X	
BodyChat		X					X	X	X	
BEAT		X					X	X	X	X
Ruttkay's 2D System		X					X		X	X
Russel's Avatar		X		X	X					
CrazyTalk				X	X		X			
3D Tennis			X	X	X					
Tessa System				X	X					
Esher's 3D Virtual		X	X				X			

Human Director										
Visual TTS			X		X	X	X			
Mobile Framework		X	X		X	X	X			X
Song Project		X	X			X	X			
Medusa		X			X	X	X			X
Microsoft SpeakMouth		X			X					
Text-to-Speech for SitePal	X	X			X		X			
Famous3D	X	X			X		X		X	
Ananova	X				X	X	X			
X-Speech	X	X			X	X	X	X	X	X

Non-verbal Communication

Many avatar systems use solely non-verbal communication as a vehicle for communication because it is easy to implement. Model-based animation systems are created that use a set of basic expressions and corresponding facial deformations to control separate animated components of a cartoon face and body, and points are specified with linear interpolation between values (Ruttkay 4). For example, BEAT is a program that conveys non-verbal animation by changing lips, eyebrows, gazes, and posture to convey emotion. Another example is BodyChat, a non-verbal avatar chat that uses high-level user controls that allow pre-defined facial expressions and behaviors to convey emotion for its cartoon avatars in a virtual environment (Vihjalmsson 7). A third example is a chat program where avatars meet and converse using facial expressions and gestures to provide non-verbal chat in a Collaborative Virtual Environment (CVE) (Salem 4). Furthermore, some non-verbal chat programs recognize a set of emoticons such as ";-)", acronyms such as LOL, and "performative words" such as, *kiss* (Salem 4).

Training or Wireframes

Avatar systems that do not use predefined rules create animation rules by training or manipulating a wireframe. For example, Russel's program creates an avatar by creating 17 visemes from a 2D photograph of the user by using prototype transformations. It uses model-based rules to warp the image and blend colors, and the animation is combined with speech using the Microsoft Speech API. (Russel 1-2). Similarly, CrazyTalk allows a user to specify a 2D image and manually place a wireframe on top of the picture. Once in place, the program performs 20 different animations and speaks using NaturalVoice speech technology.

Model-based rules for facial animation can be produced by analyzing motion capture data (Lee 1). To record motion, animators use tracking markers on the face and software such as ProFace to help map, adjust and reproduce motions for animation. For example, a complicated 3D tennis playing virtual human uses predefined action and facial animation rules that are produced by analyzing video of points on the face and a digital speech signal (Kalra 19). The Tessa system for the deaf uses the Entropic speech recognizer (that takes an hour to train) to convert speech into Sign Language, and their system is created from a database of recordings of a person signing with multiple sensors attached to the hands, arms, wrists and face.(Cox 7).

MPEG-4 Avatars

Real-time animated avatars that produce realistic animation use model-based animation and the MPEG-4 standard. One example is 3D Virtual Human Director implemented by Escher, which allows Facial Animation Parameters (FAPs) that are supported by the MPEG-4 standard to be specified for the program to perform facial deformations that manipulate specially placed points on the face (Escher 3). A project entitled Visual TTS, by AT&T Labs Research, uses 84 "feature points" that are fitted to a face to produce animation by extracting phonemes and timing from a TTS system and synchronizing with speech by setting bookmarks with predefined FAPs (Ostermann 2). A more realistic animation is produced by adding facial expression by calculating interpolation functions to vary the amplitude of each FAP.

Mobile framework implementation of avatars has been tested using MPEG-4 facial animation that uses low-level and high-level FAPs (Pandzic 2). The user controls the avatar either by providing a table of instructions for expressions and visemes, or by using XML Markup language, and high and low level functions are achieved. An example of an animated avatar system that uses the MPEG-4 standard is the Song project, which uses predefined facial expressions created from FAPs for animation (Kshirsagar 3).

Avatars without MPEG-4

Animation systems that do not use the scalable MPEG-4 standard require other model-based or imaged-based animation techniques. For example, the Medusa program is a model-based animation program that animates German speech by mapping phonemes to keyframes and using muscle contraction parameters that manipulate specific muscles in the face (Albrecht 3). For this system, non-verbal and verbal emotions are generated by manipulating muscles so that they correspond with the pitch-data of the generated speech. Additionally, if an emoticon such as ":-)" is recognized, the intensity of the emotion is increased for a short duration (Albrecht 7).

Web-Based Avatars

Real-time web-based avatars must perform calculations quickly to perform animations over the web. There are multiple web-based avatars currently available, for example, Text-to-speech for Sitepal, which animates speech using Flash media, famous3D, which allows emotion markup tags, and Ananova, a non-interactive virtual newscaster that is updated daily.

Our Method

One of the major objectives in our program is to utilize current text-to-speech and speech-to-text technologies and combine them with simple cartoon images to create fast, expressive, visual images that can be used for multiple applications and multiple platforms. Our avatar differs from the avatar systems described above because it is image-based, does not require training, and does not use the MPEG-4 standard. Instead, we create a database of 28 strip images (Fig. 2) which contain 2 keyframes each and in-betweens which we use to create visemes for our morphing algorithms.

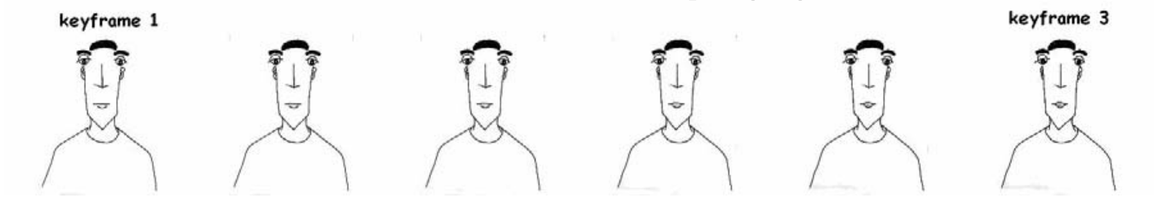


Figure 2. Strip image contains keyframe 1, keyframe 3, and in-betweens.

Simple emotion strip sequences are created that use between 1 and 6 keyframes to convey recognizable emotion, and this small number reduces the response time for our chat program. By using 41 strip images together totaling less than one megabyte, we create a character that can accurately synchronize speech with lips and produce 8 recognizable emotions. Three different characters are created, and one of the characters is tested. Furthermore, we use a markup language which provides enhanced functionality. Because our program uses few keyframes and simple rules, it loads as quickly as an instant messaging program, and it does not need the full installation and training that existing programs require.

Chapter 2: Rationale for Emotion

This chapter describes the rationale for emotions used in X-Speech, which stands for “expressive speech.” First, we present drawn images and techniques used to create believable animations, and then we explain how animation is added to X-speech to convey emotion. Facial expression and the perception of emotion are a result of cognitive processes in the brain that produce instinctive impulses on the face. By looking at the expression on a person’s face, we can immediately tell what the person is thinking or feeling. Only eleven muscles control facial expression, and the specific affected areas of the face are the eyes, eyebrows, and mouth (Gautier 77-78). The animations in Figure 3 are used in X-Speech because they are easily recognizable. In Figure 3, emotions are produced by shifting, shaping, or resizing facial components. Each emotion has characteristics that set it apart from a “neutral” expression.

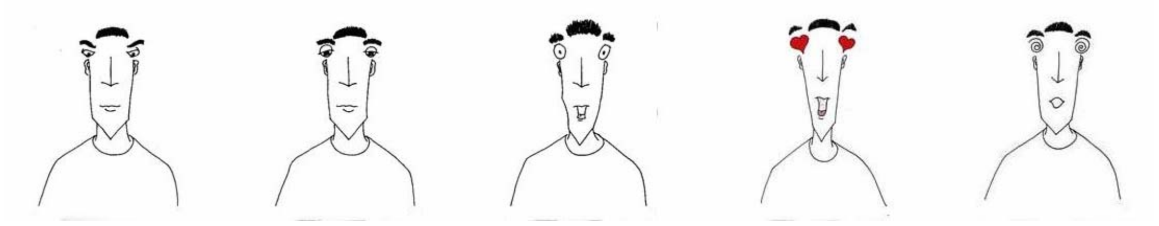


Figure 3. Recognizable emotion images are created by manipulating facial components.

In the anger emotion, the eyebrows are lowered and the eyes glare out from under them. In the bored/uncaring emotion, the eyelids are lowered and the eyes appear droopy and downcast. In the scared expression, the lower jaw of the mouth is dropped and the eyes are widened. The pupils are smaller and the white part of the eye is clearly visible. Even the hair is standing on end, and this adds to the overall effect of the expression.

Joy is expressed in a variety of ways of varying degrees. A joyful smile can be achieved by curving the corners of the mouth and widening the eyes (Gautier 80). Gleeful joy is expressed in Figure 3 by opening the mouth, widening the eyes, replacing the eyes with hearts, and raising the eyebrows. Additionally, color adds to the emotion because the color red is symbolic of love. The dazed look on the final image is achieved by opening the mouth to an “oh” sound and changing the eyes into a spiral pinwheel. Since pinwheels are associated with hypnotism, the character immediately appears dazed and confused.

Besides the major facial expression components including eyes, eyebrows, and mouth, there are other body parts that can be used to convey emotion, such as the ones displayed in Figure 4.

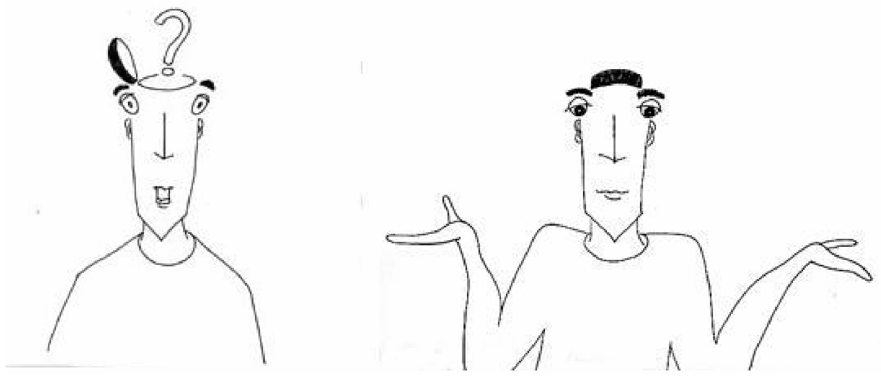


Figure 4. Besides using facial expression, non-verbal emotions are created by drawing behaviors and gestures.

In Figure 4, the question emotion is created by opening the mouth and raising eyebrows. The head with a question mark popping out clearly conveys an inquisitive expression. The shrug emotion is enhanced by raising the shoulders and pointing the avatar's hands upwards with palms exposed.

An avatar character can express emotion with controlled facial movements. With only a few keyframes, understandable emotions can be created. This requires fewer calculations and less memory than techniques used in other systems, and the result is more desirable and recognizable.

Keyframes and Morphing Algorithms

Avatars that convey emotions require smooth transitions for fluid animation. We find that morphing algorithms with few keyframes are more advantageous than ones with many keyframes and in-betweens. For example, the emotion conveying anger needs just one keyframe, a stern look, to convey the emotion. The transition to the angry face does not need any in-betweens because the slow transition would not be noticeable; we find that using more in-betweens has no desirable effect on the emotion. By using expressive keyframes, morphing algorithms become less complex. Whereas morphing between concatenated viseme keyframes and mouth movements require many in-betweens for the transitions to appear smooth, facial expressions can change more rapidly and require few keyframes.

It is advantageous to delete expressive keyframes that do not enhance the desired emotion. For example, in Figure 5, the love emotion can loop through 6 keyframes, or it can just alternate between the two more desirable keyframes 5 and 6. We find that the latter produces a better emotion.

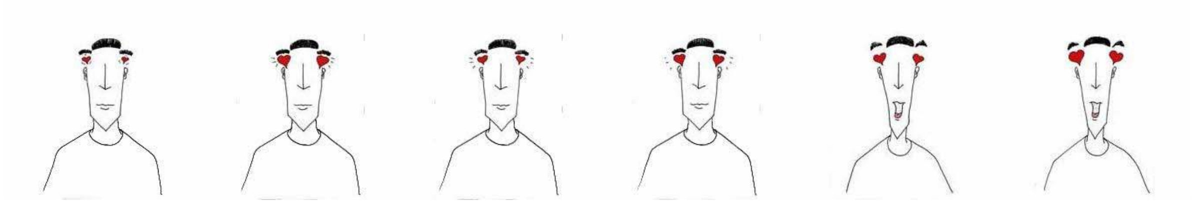


Figure 5. Desirable keyframes for the love emotion.

By utilizing expressive keyframes and creating a smooth morphing algorithm, a character produces easily recognizable emotions.

Timing and In-Betweens

Effective emotions are produced by using precise keyframe timing. For example, by using only 2 keyframes, we create over 4 degrees of emotion. We find that subtle changes in timing enhance emotion. Intensity is produced by manipulating the duration that a keyframe is held in an emotion sequence. For example, if the eyes are shifted from left to right, and back slowly, the avatar glances around carelessly. However, if the eyes shift quickly, jerking left, jerking right, and looking ahead again, the avatar appears very suspicious. Even though the same keyframes are used in each sequence, the timing affects the intensity and the resulting emotion.

For verbal emotions, our animation loops through the desired emotion images in synchronization with the audio. The avatar animation loops through the emotion for the duration of emotional speech. For non-verbal emotions, our algorithm holds emotion keyframes for an adequate duration so that the emotion is recognizable.

Emotion Blending

When blending from one emotion to the next, we maintain smooth transitions so that the avatar does not appear jerky while speaking. Synchronization is achieved by overlapping the speaking face with the desired emotion. We find that transitions from verbal to non-verbal emotions and no emotion to verbal emotions do not require morphing.

Adding Emotion to Speech

There are four areas where emotion can be added to the animation to convey non-verbal emotion: in between sentences, after sentences, while speaking, and when the animation is idle. Both the user and X-Speech itself possess the means to manipulate the animation. The computer controls animation automatically by analyzing the context of the sentence, and the user controls animation by tagging a sentence to produce desired emotions.

Automatic Mid-Sentence Speech and Intonation

While the avatar is speaking, the emotions that convey love, questioning, confusion, and fear are implemented by modifying the eyes of the avatar and the background of the frame. The mouth remains synchronized with the audio, while our emotion algorithm analyzes the sentence and adds emotion to frames that correspond with emotion words and emotion phrases. A small dictionary of emotion phrases, emoticons, and acronyms are recognized, and the algorithm automatically updates the specific corresponding frames with an emotion.

Duration = Phrase length in frames

Animation prosody is used at the end of sentences. This is similar to text-to-speech prosody, where pitch and tempo are changed to make a sentence sound less robotic. Our algorithm recognizes punctuation such as question marks and exclamation points, and it changes frames at the end of a sentence so that they convey the correct emotion:

Duration = Sentence Length in Frames * .25
Start of emotion = Sentence Length in Frames – Duration
End of emotion = End frame of Sentence

The emotions conveyed in blinks and shrugs are implemented in between sentences, during pauses, and when the avatar is not speaking. These emotions are automatically inserted into the animation, and they do not disrupt the flow of the sentence because they occur only at times where there is no verbal activity. Furthermore, insertion of non-verbal emotions produces more realistic results.

Duration = Number of Pause Frames

Automatic Idle Algorithm

The Idle algorithm provides automatic emotions for the avatar when it is not speaking. When the animation is left idle, X-Speech periodically performs an animation so that the avatar appears alive, personable and realistic. In the beginning, the avatar blinks, gazes at the user, and looks around thoughtfully. After a sentence has been animated, the idle algorithm remembers some of the previous conversation's animations. For example, if a question is recently used, the avatar shrugs every 5 seconds that the animation is idle.

Tagged End of Sentence

X-Speech increases functionality by allowing users to create non-verbal emotions at the end of sentences using tags. X-Speech splices the specified emotions to the end of the animation. The love strip, for example, is created by switching between the final two keyframes (holding each for 3 frames) for 2 seconds. At the end of the animation, the avatar resumes the neutral expression. The dazed strip is used to convey the confused emotion by looping through keyframes 1 through 5 repeatedly for 2 seconds, and it returns to the neutral expression.

Tagged Mid-Sentence Speech

In addition to tagging emotion at the end of sentences, users can create emotions for specific words and phrases. These emotions are conveyed while the avatar is speaking. This option in our GUI allows users to enhance the meaning of messages, and it is as simple to use as pressing the bold or italics key. The tag feature allows the user to control the duration of an emotion and determine where it occurs in the sentence. The tagged mid-sentence feature overrides our automatic emotion functions such that the user can maintain control of the animation.

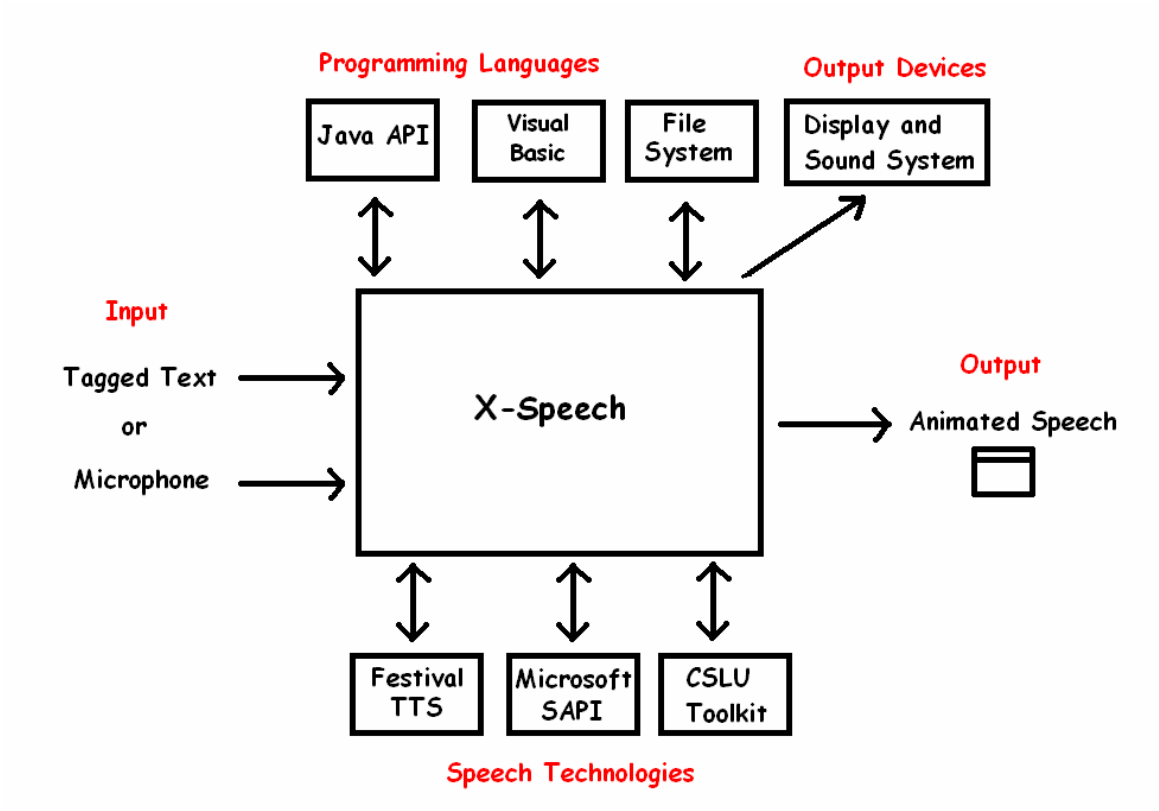


Figure 6. The context diagram displays X-Speech software and the components used. It shows the domain of our program, and arrows indicate interaction and data flow.

Chapter 3: Software Overview

This chapter gives an overview of our software. First, a high level overview is given, and then the context of our program is explained. Animated speech is produced by utilizing the Java and Visual Basic programming languages, file systems, speech technologies, and output devices.

As shown in the context diagram in Figure 6, input comes from either a microphone or "tagged text" (text with optional emotion tags) typed into the textfield of our GUI. Output is animation displayed in the GUI and speech from the sound system.

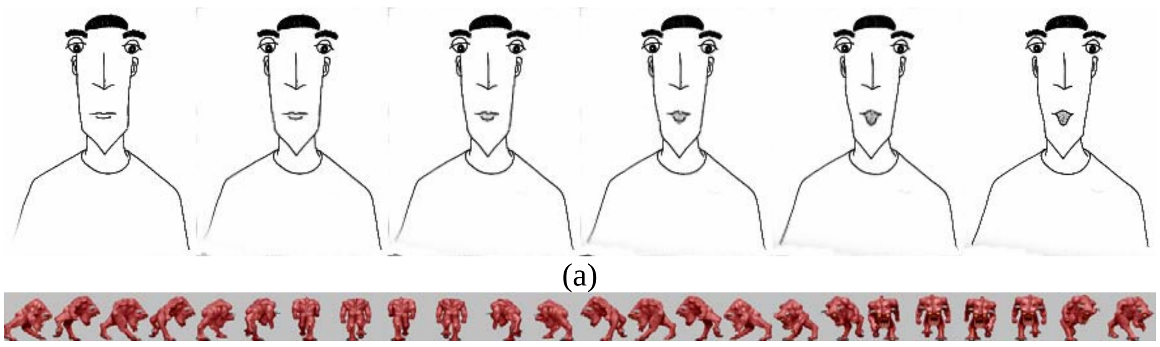
We use the Festival Text-to-Speech engine to convert text input into speech because it is readily available, cost efficient, and produces phoneme, phoneme duration, and word duration data. However, Festival can be easily interchanged with other text-to-speech software. X-Speech interacts with Festival TTS by creating a text file script that directs Festival to create a speech file and extract phoneme and phoneme duration data for any text that is entered. The script file may be modified for use with a variety of synthesized voices provided by Festival, and the resulting speech is animated.

We incorporate Microsoft SAPI (Speech Application Program Interface) for speech input by creating a program that automatically recognizes speech and displays animation. First, we direct SAPI to convert a speech file into text. The CSLU Toolkit provides the resources to create a program that aligns a wave file. A speech file and corresponding dictation are specified, and a forced alignment produces the phonemes, words, and durations needed to perform animation.

Additionally, we use Visual Basic for microphone input because it is easy to record speech and incorporate Microsoft SAPI for speech recognition. First, the Java animated avatar program is initialized, and Microsoft Media control is utilized to record the sound from the user. Next, the SAPI interface is set to recognize the recorded speech file and output a text file, such that both fed into the CSLU Toolkit for a forced alignment. An SOB file, containing phonemes and word durations necessary for animation, is analyzed by our Java program. Finally, the Visual Basic program directs the Java program to analyze the SOB file data, and recorded speech is animated.

The Java programming language is the backbone that provides the functions necessary to create an interactive chat program that synchronizes animation with sound output. Java is used to animate speech both quickly and efficiently on almost any computer or PDA. By utilizing strip images, double buffering, threads, Java's media tracker, and constant frame-rate algorithms, we create animation that runs with a small limit of response time.

Imaged-based animation is performed by using a database of strip images saved as JPEG images on the file system. Strip images are multiple pictures contained in a single file and are loaded using Java's media tracker. Figure 7 is an example of a strip image.


(a)

(b)

Figure 7. a) A viseme animation is created by selecting and displaying frames from the avatar strip image. b) The creature walks across the screen by displaying frames from the strip image in succession.

Instead of loading six separate images with six different file names, only one file is loaded. In a technique called "morphing," the first, third, and fifth frames can be selected and added to a movie to animate a viseme, yet only one strip image is required for this task. Since frames can be selected and displayed in any order, more than one viseme can be created from a single strip image. For example, Figure 7a can be used to open the mouth by selecting frames one three, and five, and the same strip image can be used to close the mouth by adding the fifth, third, and first frames to the animation.

Double buffering is used to reduce image flicker by creating each frame of the animation in an offscreen buffer before it is displayed. In Figure 7b, we can make a character walk by using double buffering and selecting key frames from the strip image.

Audio and video are synchronized for playback by using multiple threads. One thread is used to play the speech, while another displays the images onto the screen. A constant frame rate of 30 frames per second is maintained by adjusting frame delay so that it matches the system time, and this is demonstrated in the last step of Figure 8.

.

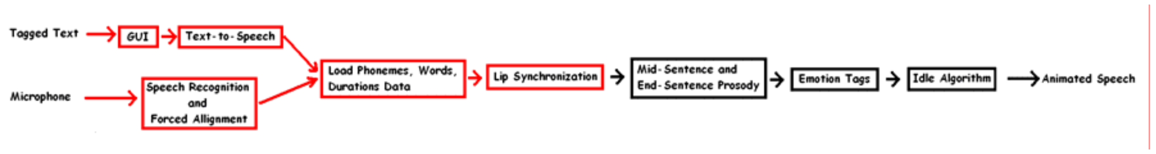

Tagged Text
GUI
Text-to-Speech
Microphone
Speech Recognition and Forced Allignment
Load Phonemes, Words, Durations Data
Lip Synchronization
Mid-Sentence and End-Sentence Prosody
Emotion Tags
Idle Algorithm
Animated Speech

Figure 8. Data Acquisition and Alignment Modules are highlighted in red.

Data Acquisition and Alignment Modules

Chat and instant messaging programs require user input as a prerequisite for animation. In the data acquisition and alignment modules, input is either "tagged text" entered by the user or speech recorded via the microphone. According to Figure 8, tagged text enters X-Speech through our GUI module. "<#> I love what you are wearing <#> <<#>>" is an example of tagged text. When the "send" button is pressed, our GUI receives the text input and removes the tags. As described above, we interact with Festival by using a script input file that specifies the speech voice and the message. Festival outputs three files: two text files which give phoneme/phoneme duration data, and word/word duration data; and a speech file that corresponds with the text file data.

X-Speech records speech input through the microphone, and we use Visual Basic for the reasons described above. According to Figure 8, a module directs Microsoft SAPI and the CSLU Toolkit (described above) to output an SOB file, which corresponds with the recorded speech.

In the next module, phoneme/phoneme duration data and word/word duration data are analyzed and loaded into the data structure described in Figure 9. This data structure is important because, once in place, emotions are easily added for specific words and phrases.

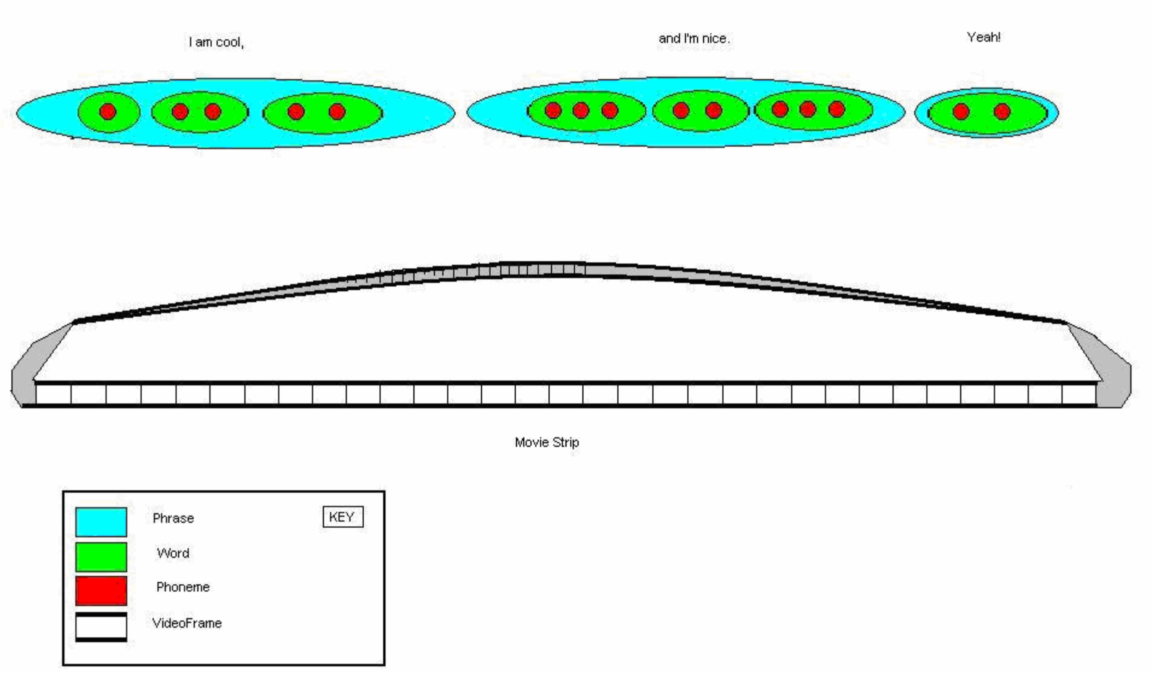


Figure 9. This is an example of the data structures used in X-Speech. The test sentence is "I am cool, and I'm nice. Yeah!" This is broken down into phrase, word, and phoneme objects which are colored in blue, green, and red respectively. The animation strip is a circular linked list. Each

phrase, word, and phoneme object contains a start and end pointers that indicate where frames begin and end. For example, the phrase object "I am cool," contains a "start of phrase" variable that points to the 1st frame in the animation strip, and it contains a "end of phrase" variable that points to the 27th frame in the animation strip. For end-sentence emotion, the animation strip is spliced, and for mid-sentence emotions, frames are modified so a different picture can be superimposed.

The data structure pictured in Figure 9 represents an exposure sheet (X-Sheet), which contains phonemes, words, and animation frame data. We use exposure sheets to synchronize the avatar's lips with speech by using phoneme-to-keyframe mapping and simple keyframe selection rules. These rules modify the data structures at the phoneme and videoframe levels. As described in Figure 9, pointers are shifted to improve fluidity of the animation. Our empirical data shows that the lip synchronization rules succeed in aiding the understandability of speech.

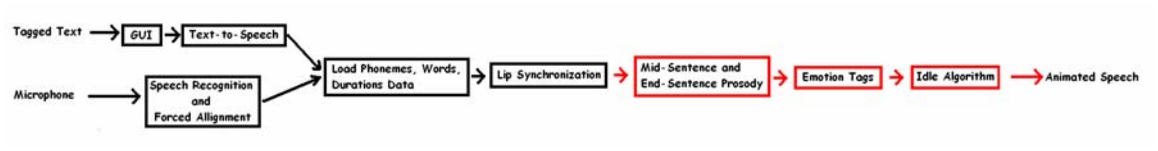

Tagged Text
GUI
Text-to-Speech
Microphone
Speech Recognition and Forced Allignment
Load Phonemes, Words, Durations Data
Lip Synchronization
Mid-Sentence and End-Sentence Prosody
Emotion Tags
Idle Algorithm
Animated Speech

Figure 10. Red indicates the emotion rules modules used by X-Speech prior to displaying the animation.

Emotion Rules Modules

The goal of our emotion rules modules is to enhance the meaning of messages by performing rules quickly, using visual simplicity, and using few keyframes to produce recognizable emotions. Our emotion algorithm uses a small set of picture files, but it conveys a large set of emotions. Each emotion keyframe is versatile. For example, a single keyframe that conveys the scared emotion can be used in the following ways: 1) it can be used while speaking to highlight specific words or phrases within in a sentence, 2) it can be used at the end of a sentence to give a frightened look, 3) it can be used in-between sentences to give looks of concern, and 4) our idle algorithm can replay the scared emotion during idle periods so that the user knows what had been said in the previous sentence. When using more than one keyframe for an emotion, the number of emotions that can be produced grows exponentially.

Parsing algorithms

Parsing for the emotion algorithm is accomplished by utilizing the data structures described in above and the tagged user input phrase. Tagged phrases are first removed from the phrase systematically. The end-of-sentence emotion tags are removed first, and the corresponding emotion strip is spliced to the end of the animation. To achieve prosody, automatic emotions are applied to the animation on frames that correspond with emoticons, in-between sentences, and at the end of the sentences. Next, mid-sentence tags are removed, and the corresponding frames are updated with the correct facial expression. Finally, the data structure is passed to the animation algorithm where it is displayed.

XML Markup

XML Markup language can be used to create the data structures needed to perform animation. The tags we use are in Table 2.

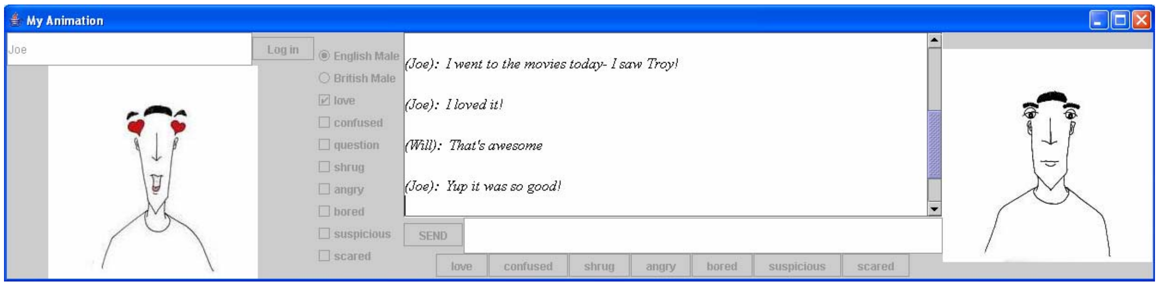
Table 2. Markup Language

Emotion	**Tag**
<<#>>	Love
<<%>>	Confused
<<?>>	Question
<<&>>	Shrug
<<*>>	Angry
<<$>>	Scared
<<@>>	Suspicious
<<+>>	Bored

An XML data structure can be analyzed and converted into the markup tags used for our animated avatars. Other tags implemented include choice of character and voice.

Chapter 4: Data and Results

The data and results are primarily intended to test the effectiveness of our program. Observers evaluate 350 emotional sentences. Non-verbal and verbal emotions are both automatic and tagged. The user views the animated avatar and reports whether or not the animation aids in the understanding of the sentence, determines the emotion that is being portrayed, and reports whether or not he or she enjoyed watching the avatar. A fourth aspect of our data involves testing speech with animation for understandability. X-Speech is compared with other real-time animated avatars.



(Joe): I went to the movies today- I saw Troy!

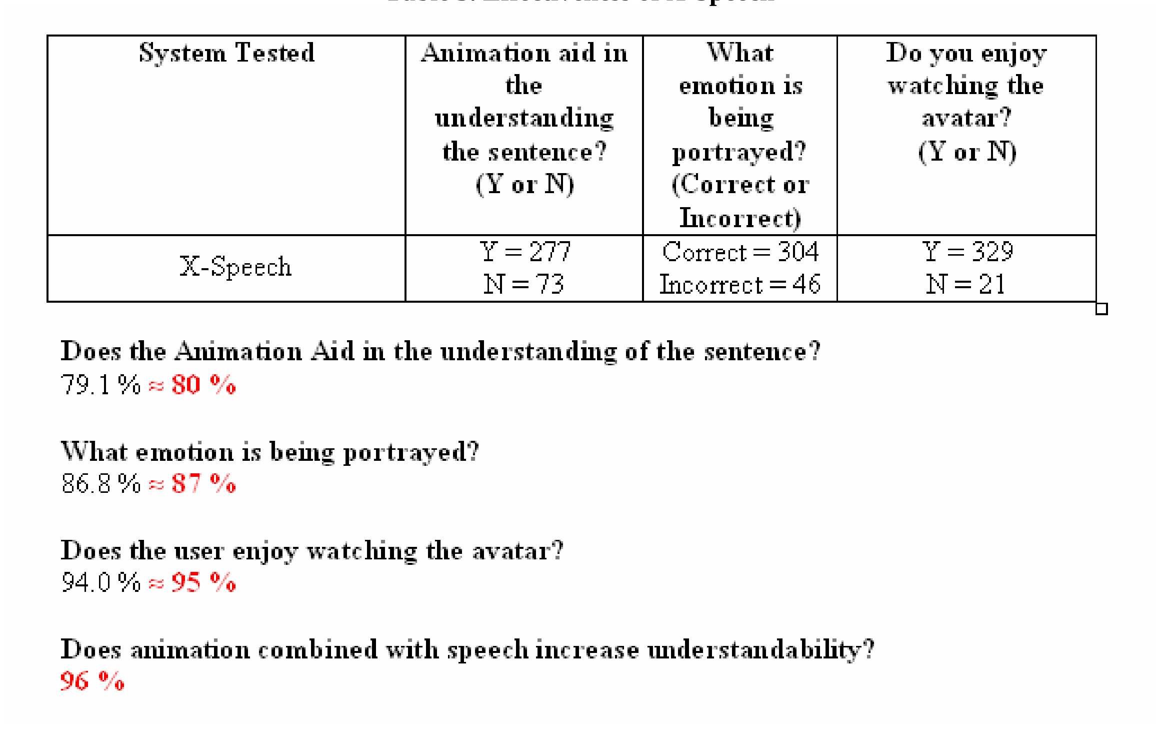
(Joe): I loved it!

(Will): That's awesome

(Joe): Yup it was so good!

Figure 11. Chat messages are displayed in the center text box. Text input is transmitted to X-Speech by typing a message and pressing the "send" button. The left avatar in the GUI is the displays the sender's animations, and the right avatar displays the received messages.

Table 3. Effectiveness of X-Speech



System Tested	Animation aid in the understanding the sentence? (Y or N)	What emotion is being portrayed? (Correct or Incorrect)	Do you enjoy watching the avatar? (Y or N)

X-Speech	Y = 277
	N = 73
		Correct = 304
		Incorrect = 46
			Y = 329
			N = 21

Does the Animation Aid in the understanding of the sentence?
79.1 % ≈ 80 %

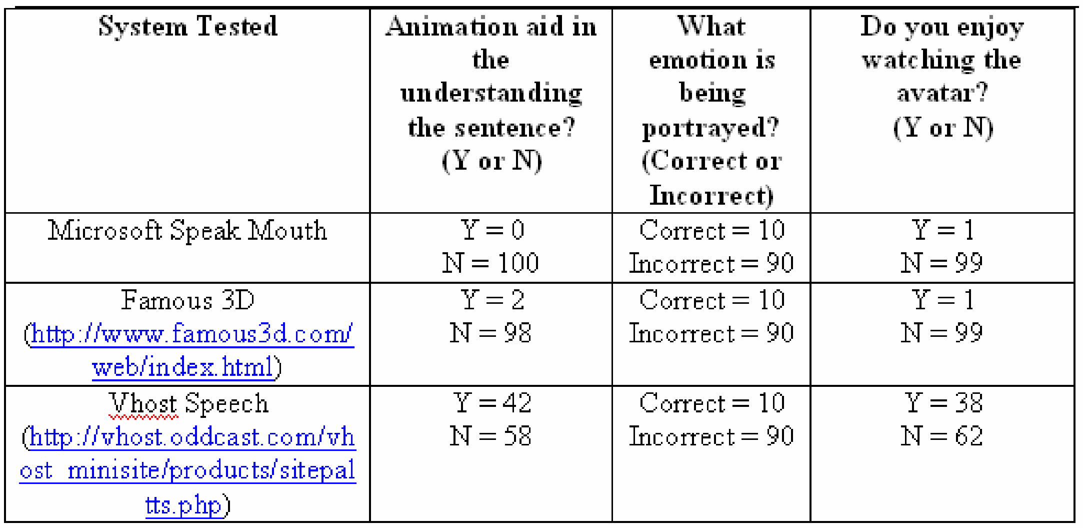
What emotion is being portrayed?
86.8 % ≈ 87 %

Does the user enjoy watching the avatar?
94.0 % ≈ 95 %

Does animation combined with speech increase understandability?
96 %

Comparison with Other Animated Avatar Programs

Table 4. System Comparison



System Tested	Animation aid in the understanding the sentence? (Y or N)	What emotion is being portrayed? (Correct or Incorrect)	Do you enjoy watching the avatar? (Y or N)

Microsoft Speak Mouth	Y = 0
	N = 100
		Correct = 10
		Incorrect = 90
			Y = 1
			N = 99

Famous 3D (http://www.famous3d.com/web/index.html)	Y = 2
	N = 98
		Correct = 10
		Incorrect = 90
			Y = 1
			N = 99

Vhost Speech (http://vhost.oddcast.com/vhost_minisite/products/sitepal tts.php)	Y = 42
	N = 58
		Correct = 10
		Incorrect = 90
			Y = 38
			N = 62

Chi Squared Test

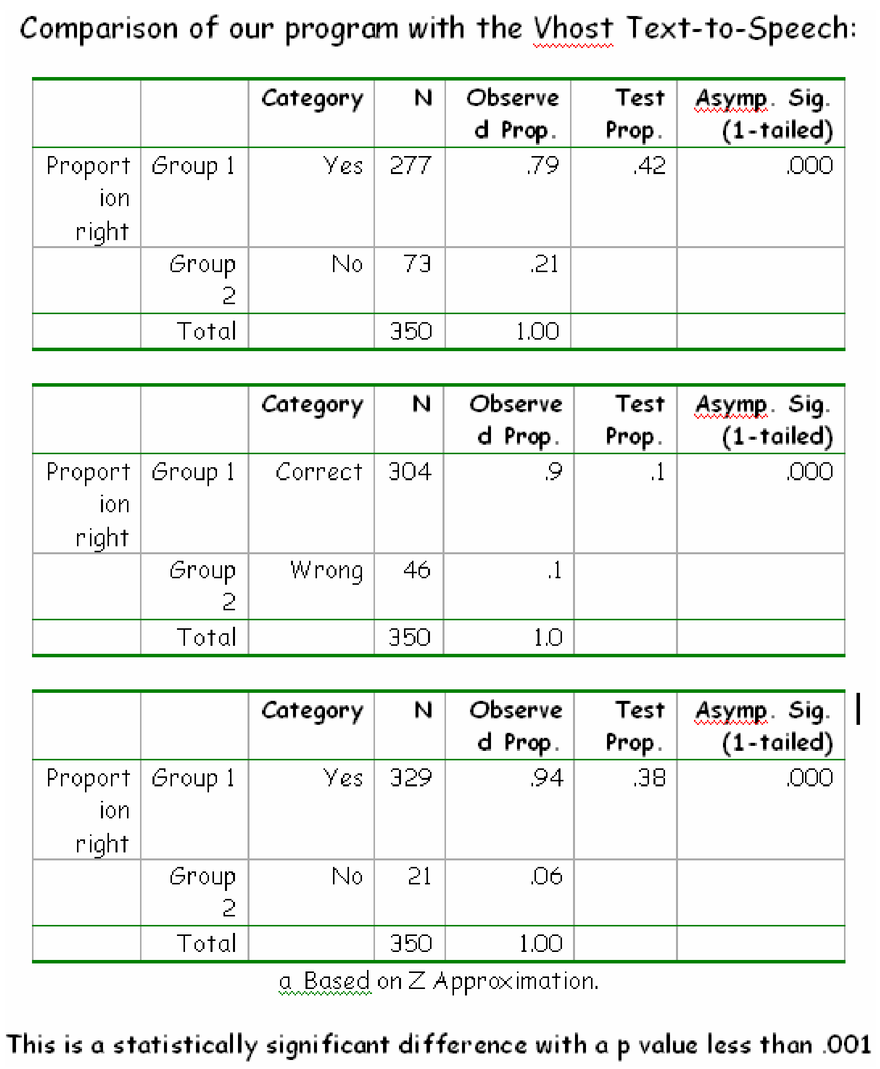
$$\left(\frac{(277-175)^2}{175}\right)+\left(\frac{(73-175)^2}{175}\right) = 119\text{, which is significant at the .001 level}$$

$$\left(\frac{(304-175)^2}{175}\right)+\left(\frac{(46-175)^2}{175}\right) = 270\text{, which is significant at the .001 level}$$

$$\left(\frac{(304-39)^2}{39}\right)+\left(\frac{(46-311)^2}{311}\right) = 2025\text{, which is significant at the .001 level}$$

This means that there is a less than 1/1000 chance that the program performed this well by chance.

Binomial Test



Comparison of our program with the Vhost Text-to-Speech:

		Category	N	Observed Prop.	Test Prop.	Asymp. Sig. (1-tailed)
Proportion right	Group 1	Yes	277	.79	.42	.000
	Group 2	No	73	.21		
	Total		350	1.00		

		Category	N	Observed Prop.	Test Prop.	Asymp. Sig. (1-tailed)
Proportion right	Group 1	Correct	304	.9	.1	.000
	Group 2	Wrong	46	.1		
	Total		350	1.0		

		Category	N	Observed Prop.	Test Prop.	Asymp. Sig. (1-tailed)
Proportion right	Group 1	Yes	329	.94	.38	.000
	Group 2	No	21	.06		
	Total		350	1.00		

a Based on Z Approximation.

This is a statistically significant difference with a p value less than .001

Chapter 5: Conclusions and Applications

Our original software demonstrates increased functionality, enhanced expressiveness of messages, and heightened enjoyment with the product. In our study, observers are asked to evaluate our program. Their results demonstrate that the animation aids in understandability 79.1% of the time. Further data reveals that the observers are able to correctly identify the emotion portrayed in the sentences 86.8% of the time. Additionally, the observers report that their enjoyment level is 94.0%. These results demonstrate that the animation aids in understandability, utilizes easily recognizable emotions, and successfully enhances overall enjoyment.

In three other currently available animation programs, observers are asked to evaluate whether or not the animation aids in the overall understandability of sentences. The results show that the rates of understandability are 42%, 2%, and 0%. This is a dramatic difference when compared with our program, which has a 79.1 percent rate of understandability. In the next set of data, the observers are asked to determine the emotions portrayed in the sentences. The data shows that the other programs have 0 % rate of recognizing emotions. In comparison, our program demonstrates identifiable emotion 86.8% of the time. In the third aspect of the data, observers test sentences for overall enjoyment. Other currently available software demonstrates only 1%, 1%, and 38% enjoyment, while our program demonstrates an impressive 94% enjoyment.

A fourth aspect of our data tests whether the animation combined with speech increases understandability. Our results show that our lip synchronization and automatic emotions enhance the understandability of spoken messages 96% of the time.

On all levels of comparison, our software is more effective than other currently available programs. Additionally, since this program can be activated quickly, it is ideal for applications such as cell phones for the deaf or hearing impaired, instant messaging, video conferencing, voicemail, instructional software, and speech and animation synthesis. Not only is our program more versatile than other currently available software, but the hearing impaired can benefit from better lip synchronization and enhanced emotion, which provides for better understanding of messages. Additionally, our data supports the fact that our program is more exciting, stimulating and user friendly than other real-time animated avatar programs currently available.

Future Work

Further work involves testing with more observers, creating more of a basis of comparison by using a larger number of software samples, increasing the rate of understandability to even higher levels, and combining our chat program with other text-to-speech software.

Appendix A: Diagrams

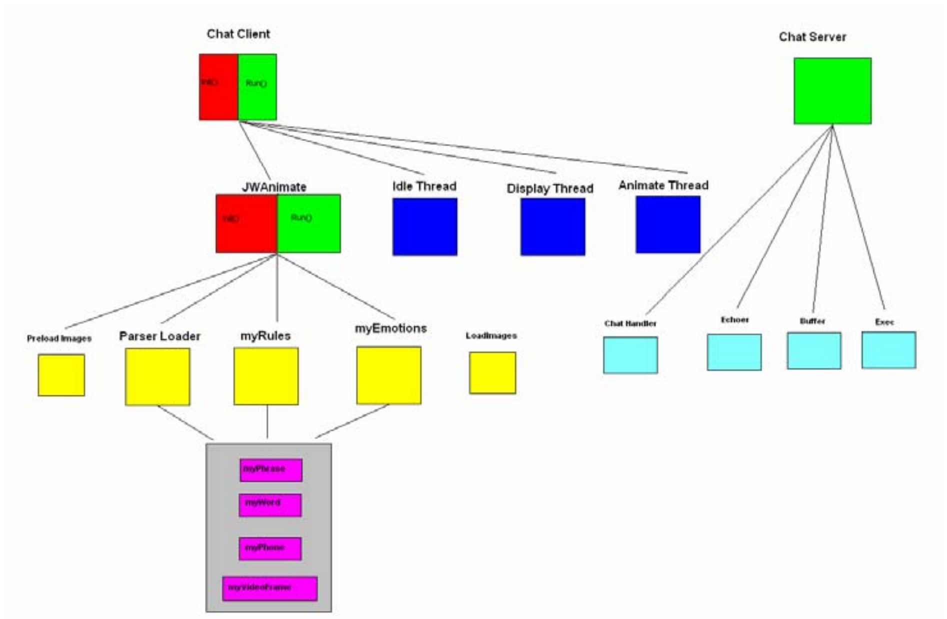


Figure 12. The class hierarchy of X-Speech is illustrated. The chat client provides a GUI for input, and it outputs animated speech. The chat server connects chat clients and synchronizes messages.

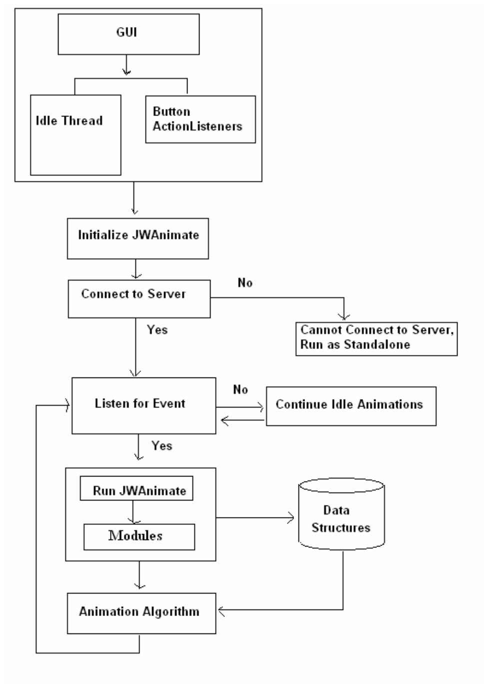


Figure 13. The chat client connects to the server, continuously receives text input, and performs animation. Intermediate file data are stored on the file system.

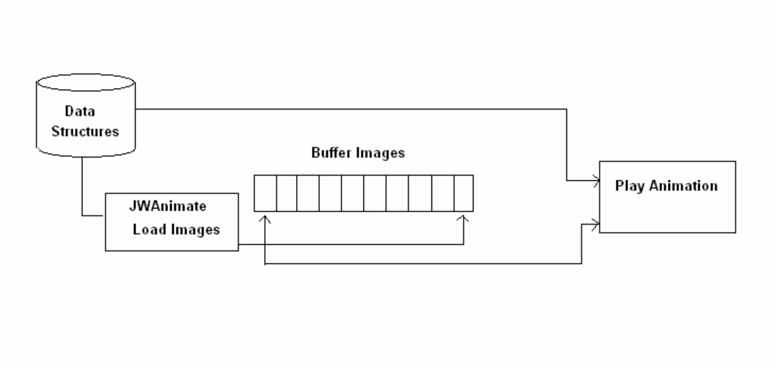


Figure 14. The animation algorithm outputs images sequentially to X-Speech's GUI. It stores offscreen buffer images to provide seamless animation.

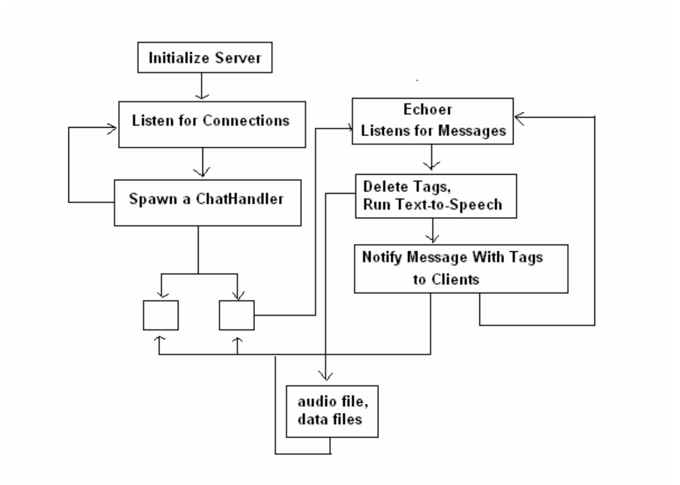

Initialize Server
Listen for Connections
Spawn a ChatHandler
Echoer Listens for Messages
Delete Tags, Run Text-to-Speech
Notify Message With Tags to Clients
audio file, data files

Figure 15. The chat server listens for connections and spawns a thread for each chat client that connects. It also synchronizes messages and interacts with the text-to-speech engine.

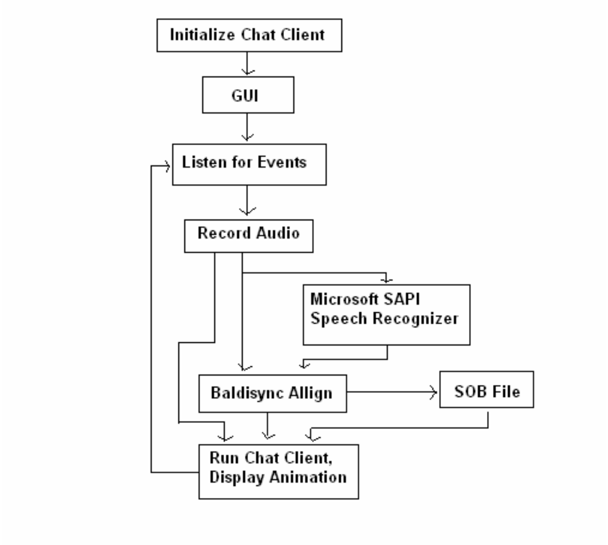


Figure 16. Speech input is converted into animation by recognizing and aligning the recorded speech.

Appendix B: Evaluation of X-Speech

Bank of possible emotions
None
Love
Confused
Shrug
Question
Angry
Bored
Suspicious
Scared

#	Sentence	Animation aids in the understanding the sentence? (Y or N)	What emotion is being portrayed?	Do you enjoy watching the avatar? (Y or N)
1	Would you like to see a movie with me? <<#>>			
2	I am not pleased with you right now. <<*>>			
3	Where are you going? Can I come? <<&>>			
4	Is there something under there? <<$>>			
5	Did you bring the beach towel or not? <<?>>			
6	The dog jumped and ran away.			
7	It is very hard to stay awake.			
8	Do you really want to go? <<+>>			
9	Have you seen the new movie yet? <<?>>			
10	I am pleased to finally get to meet you.<<#>>			
11	Have one on me, ok? <<&>>			
12	I went to Alaska where I baked in the sun and I loved it!			
13	I surf in California almost every day.			
14	I drank a lot of orange juice.			
15	Oh no I missed the basket. <<&>>			

16	I take karate 3 times a week! <<#>>			
17	<*> I hate it when you turn on the radio very loudly<*><<*>>			
18	<+>Four score and seven years ago, text to speech was first dreamed of.<+> <<+>>			
19	<$>There is a 50 car accident on highway 1. <$> <<$>>			
20	<#>Romeo, oh romeo, where are you romeo<#> <<#>>			
21	<%>I did a keg stand and drank a lot of beer last night<%> <<%>>			
22	<%> Where the heck did I put my glasses. <%> <<%>>			
23	<*>Get over here right now, and do not take your time <*> <<*>>			
24	Can you please clarify the meaning of this question? <<?>>			
25	He is a very loyal animal, and he is a member of our family. <<#>>			
26	I understand <*> part <*> of the question, but not the entire thing. <<*>>			
27	I like to watch scary movies in the dark. <<$>>			
28	Is it in here? <<@>>			
29	I parked my car in the wrong spot! <<*>>			
30	This is a bunch of crap. I want one too. <<*>>			
31	I have four brothers and two sisters. <<$>>			
32	When I talked to her, she seemed interesting! <<#>>			
33	<+> The formal presentation takes place at 6 oclock <+>			
34	I want to use windows, but first I have to download a serial number.<<%>>			
35	<#>South Carolina has great fishing!<#>			
36	Should we meet at the movies or the bowling alley. <<&>>			
37	The Lakers are the best basketball team! <<#>>			
38	Lunch was great, wasn't it? <<?>>			

39	The computer talks to me is that normal? <<%>>			
40	I am going to the Zoo after school! <<#>>			
41	Graduation is a foolish process. <<+>>			
42	K mart is having a sale? <<?>>			
43	Do you see the dangerous gang? <<@>>			
44	<$> He was murdered outside burger king <$>			
45	Strawberry cake is my favorite kind of dessert!<<#>>			
46	<*>Carlos took my bicycle<*> <<*>>			
47	Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto <<&>>			
48	<+>Happy Anniversary President Clinton. <+> <<+>>			
49	I am hungry, so let's go to Publix and buy some food.			
50	My sister won the science fair. <<#>>			

References

Adamson, Chris. *A Gentle Re-Introduction to QuickTime for Java*. June 2003. May 2004. <http://www.onjava.com/pub/a/onjava/2003/05/14/qtj_reintro.html?page=2>.

Albrecht, I., J. Haber, and H.P. Seidel. "Automatic Generation of Non-Verbal Facial Expressions from Speech." *Proceedings Computer Graphics International (CGI) 2002, 3-5 July 2002*. 283-293.

Ananova. 2002. 20 May 2004. <http://www.ananova.com/video/>.

Bartlett, Sandra L. *Java Images*. Oct. 1998. 20 May 2004. <http://www.eecs.umich.edu/~bartlett/f98java-images.html>.

Beskow, Jonas, and Giampero Salvi. *The Teleface Project*. 27 June 2002. 20 May 2004. <http://www.speech.kth.se/teleface/>.

Brenton, G., C. Bouville, D. Pele. FaceEngine. "A 3D Facial Animation Engine for Real Time Applications." *In Proceedings of the sixth international conference on 3D Web technology, Feb, 2001.*

Cassell, J., H. Vihjalmsson, and T. Bickmore. "BEAT: the Behavior Expression Animation Toolkit." *Proceedings of the 28th annual conference on Computer graphics and interactive techniques, Aug, 2001.*

Cox, S., M. Lincoln, J. Tryggvason, M. Nakisa, M. Wells, M. Tutt, and S. Abbot. "Tessa, a system to aid communication with deaf people." *Proceedings of the fifth international ACM conference on Assistive technologies, July, 2002.*

CrazyTalk: Create animated messages from any digital image or photo. 20 May 2004. <http://crazytalk.reallusion.com/>.

CSLU Toolkit. 21 Aug. Oregon. 2003. 20 May 2002. <http://www.cslu.ogi.edu/toolkit/index.html>.

Donohues, John. *JDAnimate: Animation with sound Applet*. 28 Sept. 1997. 20 May 2004. <http://www.serve.com/wizjd/java/JDAnimate/JDAnimate.html>.

Dougherty, Bob. *Quicktime Grab Movie Frames*. 31 July 02. 20 May 2004. <http://www.optinav.com/ImageJplugins/QT_GrbMvFrms.htm>.

Dwight, Jeffry and Michael Erwin. *Special Edition Using CGI*. 1996. 20 May 2004. <http://medialab.di.unipi.it/doc/CGI/toc.htm>.

Eisert, P., S. Chaudhuri, B. Giord. "Speech Driven Synthesis of Talking Head Sequences." *3D Image Analysis and Synthesis*. pp. 51-56, Erlangen, Nov, 1997.

Escher, M., G. Sannier, N. Thalmann. "Real-Time Interactive Facial Animation." *WSCG'99, Pilzen, 1999*.

Famous3D. 12 May 2004. <http://www.famous3d.com/web/index.html>.

Frédéric, Tyndiuk. *Basic Graphical Counter Version 1.0*. 3 Aug. 2000. 20 May 2004. <http://www.ftls.org/en/examples/cgi/GraphCounter.shtml>

Gautier, Dick. *Drawing and Cartooning 1001 Faces: Step-by-step techniques for Mastering Features, Heads, and Expressions*. New York: The Putnam Publishing Group, 1993.

Hon, H., A. Acero, X. Huange, J. Lui, and M. Plumpe. "Automatic generation of Synthesis Units for Trainable Text-to-Speech Systems." *Microsoft Research, 1998*.

Huang, X., A. Acero, J. Adock, H. Hon, J. Goldsmith, J. Liu, and M. Plumpe. "Whistler: A Trainable Text-to-Speech System." *Microsoft Research, 1996*.

Huang, Xuedong, Alex Acero, and Hsiao-Wuen Hon. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. New Jersey: Prentice Hall Inc, 2001.

Images and Sound. 1997. MageLang Institute. 20 May 2004. <http://www.rz.fhtw-berlin.de/javakurs/magelang/image/images.html>.

Inovani. 20 May 2004. <http://www.inovani.no/ThePlayer.htm>.

Johnson, Elizabeth. *Threads/Animation in Java*. 15 March 2000. 20 May 2004. <http://206.21.170.2/csci260/00s/java/threads/>.

Kainr. *Double Buffering in Java*. 12 June 2002. 20 May 2004. <http://kainr2.tripod.com/DoubleBuffering.html>.

Kalra, P., N. Thalmann, L. Moccozet, G. Sannier. "Real-time animation of realistic virtual humans." *IEEE Computer Graphics and Applications, 1998*. Vol.18, No.5, 42-55.

Klatt, Denis. History of Speech Synthesis. 6 Aug. 2002. 20 May 2004. <http://www.cs.indiana.edu/rhythmsp/ASA/Contents.html>.

Kurniawan, Agus. Converting Text-to-Speech and using mouth motion animation. 2001. 20 May 2004. <http://www.codeproject.com/audio/speakmouth.asp>.

Kshirsagar, S., A. Vuilleme, K. Kamyab, N. Thalmann, D. Thalmann, and E. Mamdani.

"Avatar Markup Language." *Proceedings of the workshop on Virtual environments 2002,*
May, 2002.

Laybourne, Kit. *The Animation Book*. New York: Three Rivers Press, 1998.

Lee, J., J. Chai, P. Reitsma, J. Hodgins, N. Pollard. "Interactive Control of Avatars Animated with Human Motion Data." *ACM Transactions on Graphics (TOG), Proceedings of the 29th annual conference on Computer graphics and interactive techniques, July 2002*. Volume 21 Issue 3.

Massaro, D., Speech and gaze. "A Computer Animated Tutor for Spoken and Written Language Learning." *Proceedings of the 5th international conference on Multimodal interfaces, Nov 2003.*

Matousek, Vaclev, and Pavel Mautner. Text, Speech and Dialogue. New York: Springer -Verlag, 1998.

Maxey, David. Smithsonian Speech Synthesis History Project. 1 July 2002. Washington. 20 May 2004. <http://www.mindspring.com/~ssshp/ssshp_cd/ss_home.htm>.

Morozov, Michael. Guard Java Applet. 1996. 20 May 2004. <http://morozov.adelaida.net/>.

Morse, Gary. *The Java Game Programming Tutorial.* 1997. 20 May 2004. <http://modena.intergate.ca/personal/iago/javatut/jtp4.htm>

Ortiz, I. Aizpurua, and J. Posada. "Some Techniques for Avatar Support of Digital Storytelling Systems." *Technologies for Interactive Digital Storytelling and Entertainment. Darmstadt, 2003*.

Ostermann, J., M. Beutnagel, A. Fischer, Y. Wang. "Integration of Talking Heads and Text-to-Speech Synthesizers for Visual TTS." *International Conference on Speech and Language Processing, Sydney, Dec 1998.*

Pandzic, Igor. "Facial Animation Framework for the Web and Mobile Platforms." *3D technologies for the World Wide Web, 2002*. 27-34.

Pandizic, Igor. "An XML Based Interactive Multimedia News System." *Conference on Human - Computer Interaction HCI International 2003, Crete, Greece*.

Penn, Stirling. Sterlz Web-Design. 23 Oct. 2003. 20 May 2004. <http://www.sterlz-web.pwp.blueyonder.co.uk/Tutorial/Animation0.html>.

Persson, P., "ExMS: an Animated and Avatar-based Messaging System for Expressive

Peer Communication." *Proceedings of the 2003 international ACM SIGGROUP conference on Supporting group work, Nov 2003*.

Facial Animation Software – proFace complete. 2000. San Francisco. 20 May 2004. <http://www.metamotion.com/software/proFACE_complete.htm>.

Roberts, Ken. *Java*. 20 May 2004. <http://www.infocellar.com/java/right.htm>.

Robinson, Mike. How to communicate from a Java Applet to a CGI script. 21 July 2002. 20 May 2004. <http://users.frii.com/michael/how_to_communicate.htm>.

Russell, E. and B. Tiddeman. "A Text to Visual Speech Instant Messaging System." *Post Graduate Networking Conference (PGNet2003), Liverpool, June, 2003*.

Ruttkay, Z., H. Noot. "Animated Cartoon Faces." *Proceedings of the first international symposium on Non-photorealistic animation and rendering, June 2000*.

Salem, B., and N. Earle. "Designing a Non-Verbal Language for Expressive Avatars." *Proceedings of the third international conference on Collaborative virtual environments, Sept 2000*.

Smith, John B. Java Multimedia. 30 Sept. 2002. 20 May 2004. <http://www.cs.unc.edu/Courses/comp117/docs/lessons/java/java_multimedia/>.

Taylor, Richard. *Encyclopedia of Animation Techniques*. Oxford: Focal Press, 1996.

Text-to-Speech for SitePal. 20 May 2004. <http://vhost.oddcast.com/vhost_minisite/products/sitepaltts.php>.

Van Hoff, Arthur and Kathy Walrath. Animation in Java Applets. Jan. 1996. May 2004. <http://www.javaworld.com/javaworld/jw-03-1996/animation/index.html>.

Vihjalmsson, H., and J. Cassell. "BodyChat: Autonomous Communicative Behaviors in Avatars." *Proceedings of the second international conference on Autonomous agents, May 1998*.

Wutka, Mark. Faster Image Downloads. 1997. 20 May 2004. <http://www.wutka.com/hackingjava/ch22.htm>.

Yabe, J., S. Takahashi, E. Shibayama. "Automatic Animation of Discussions in USENET." *Proceedings of the working conference on Advanced visual interfaces, May 2000*.