Automatic Lip Synching using Non-LPC Analysis

Term Project

By: Anil Rohatgi
Gtg618b
Georgia Institute of Technology

1. Introduction

As digital entertainment and technology advances, the importance of computer generated characters is growing. In order to make these characters believable to the general public, they must exhibit human like behavior in order to draw the audience in. Lip synching to dialog is one way of making these computer generated characters become more lifelike and realistic. This paper proposes a new technique to automatically perform lip synching for a computer generated character to fit a segment or real speech.

2. High level Algorithm

The key to performing accurate lip synching to real speech is a technique known as phoneme extraction. A phoneme is the smallest discreet element of sound typically voiced by a human. These phonetic representations of speech can be concatenated to describe the sounds comprising any speech segment.

Many of these phonemes can be mapped to a given mouth motion key frame that corresponds to the facial expression produced when voicing the sound. Also, the majority of these phonemes, especially vowels, have a unique frequency spectrum characterized by the resonance frequencies (formants) present in the vocal tract at the time of voicing. Using these two pieces of information, an approximation to the act of voicing a speech segment can be generated as follows.

The first step is to break apart the speech in to small pieces in time and take the Fourier transform of each windowed speech signal. Doing this for the entire segment produces a spectrogram representation of speech. Then, for each segment's frequency plot, perform an analysis to extract the formants present at that instant. Next, map the formants to the closest phoneme, and a therefore the closest mouth position. Finally, interpolate between these positions at each time interval to produce the overall mouth motions for the segment. Figure one shows a high level flow diagram of this algorithm.

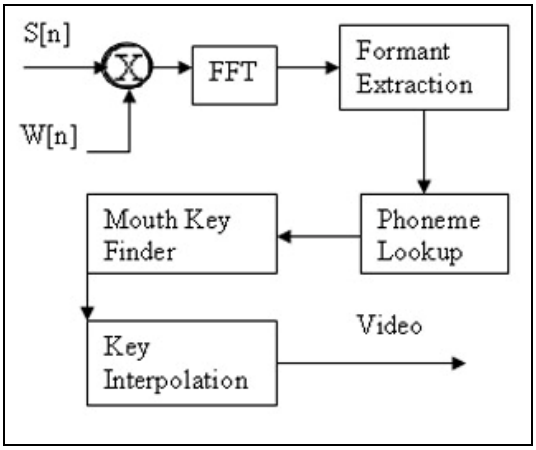

Figure 1. High level flow diagram

3. Formant Analysis

The first crucial step to performing the algorithm proposed above is to extract the formants of each small windowed segment of speech. To do this, the first stage is to create a spectrogram representation of the speech signal. Figure 2 below shows an extracted spectrogram from the corresponding input speech signal. The window typed used in this analysis is 512 point Kaiser shaped window, with 475 non overlapping samples. The sampling frequency was assumed to be 8 kHz.

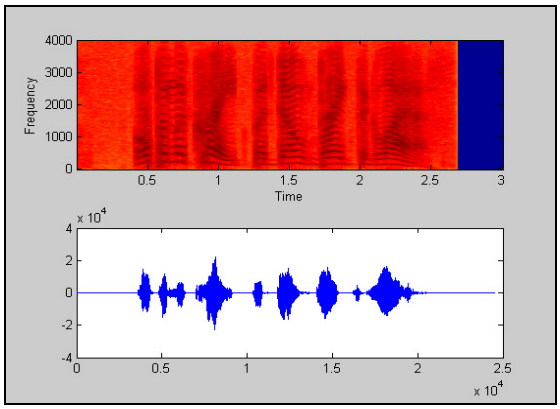

Figure 2. Spectrogram vs. input speech signal "Open the crate, but don't' break the glass"

Each set of windowed speech was then processed individually to extract the formant frequencies. Let's examine one such segment, at time step 1.5 sec and walk through the analysis. The frequency plot of the signal at this time is shown below as figure 3.

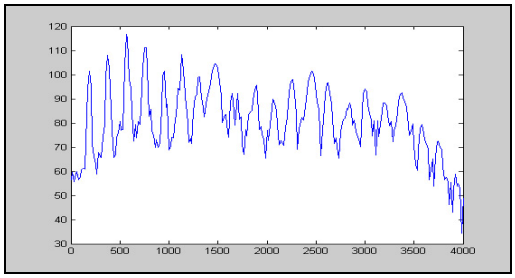

Figure 3. Frequency plot of speech at time t=1.5 sec

As you can see, there are harmonic peaks at nearly every pitch period, but there are also overall global peaks in the graph. These global maximums correspond to the formant frequencies comprising this instant of speech. In order to extract these global maxima, an envelope detection algorithm was run on the spectrum.

3.1 Envelope Detection

The procedure to generate the overall spectral envelope works as follows. Firstly, the derivative of the frequency spectrum was taken, and the zero crossings were located to find the local maximums and minimums. However, performing this procedure alone did not yield accurate harmonic peak resolution because noise in the signal produced unexpected small local minimums and maximums that do not follow the spectral trends as shown in figure 4.

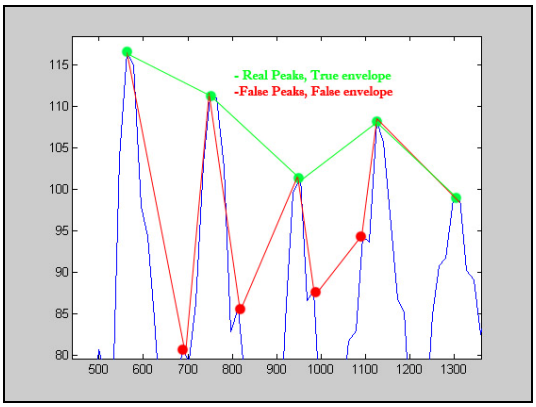

Figure 4. Peak error demonstration. Green points and lines correspond to true peaks and the true envelope, while red peaks and lines correspond to the false peaks and incorrect envelope

Allowing these peaks to be connection points for the envelope would drastically change the shape of the envelope and thus give false formant frequencies. Therefore, the newly located peaks need

to be sorted in order to distinguish harmonic peaks from noise.

Through much trial and error, it was determined that the distinguishing characteristic between false peaks and true harmonic peaks is the value transition between the peak and its surrounding troughs. Important harmonic peaks have a large jump in value from both its previous and following troughs when compared to its value. Smaller peaks however, did not have these drastic value fluctuations from one or both of these sides. The threshold used to make this distinction was determined by using the difference between the maximum peak and the mean value of the signal. Using these criteria, a filter was written to only located and label the correct harmonic peak frequencies as shown in figure 4.

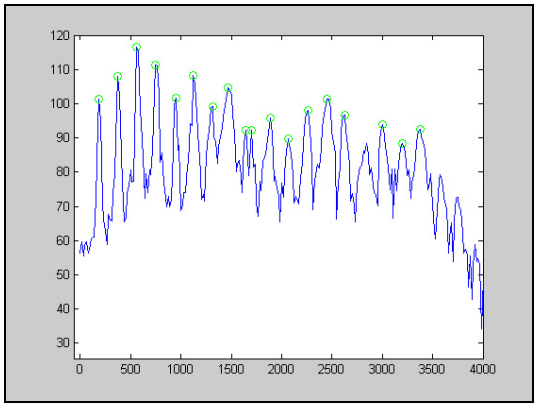

Figure 5. Location of the correct peaks

Now that the correct harmonic peaks are located and labeled, determining the overall envelope of the spectral signal is simply a matter of interpolating the samples between these peaks and essentially connecting the dots. Then, in order to locate the formant frequencies of this envelope, the derivative operation was performed on the envelope signal and the zero crossings were found. These local maximum values correspond to the formant frequency locations. It should also be noted that any formant frequency below 200 Hz was disregarded as glottal effects. Unvoiced sounds were assigned the value of zero for both frequencies. These unvoiced segments were identified by the ratio of maximum peak to the signal mean. If a signal had a low ratio, the speech at this instant in time was classified as unvoiced. Figure 6 below shows the final result of the envelope detection and formant extraction for a voiced segment.

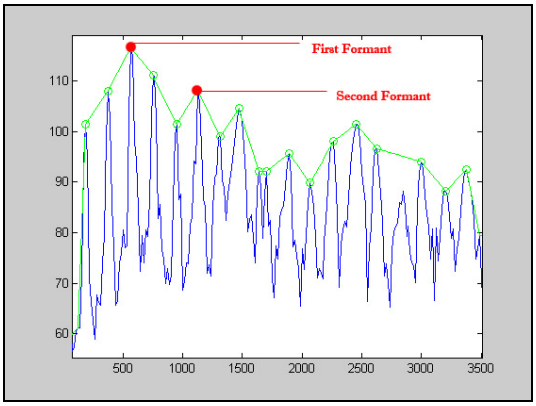

Figure 6. Overall envelope and formant graph

The first two formants for each time step are stored and passed to the next stage of the algorithm that produces the video sequence. A graph of formant frequencies vs. time is shown below superimposed over the spectrogram plot.

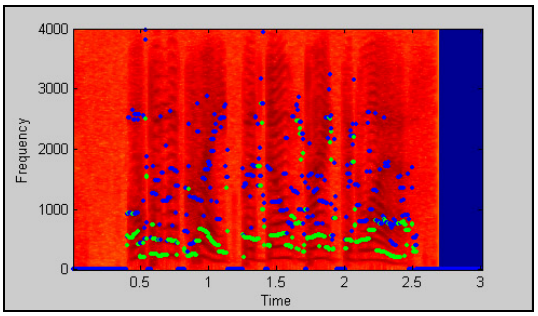

Figure 7. Formants vs. time. The green dots are the first formants, and the blue dots are the second formants

3. Graphics Creation

The output of the last function is now sent into the graphics creation algorithm that takes the first two formant frequencies at each time interval, and maps them to a phoneme. For this project a set of 11 phonemes were used and are described in the chart below.

Typical formant frequencies for vowels [4]

vowel phoneme	F_1	F_2
IY	270	2290
IH	390	1990
EH	530	1840
AE	660	1720
AA	730	1090
AO	570	840
UH	440	1020
UW	300	870
ER	490	1350
AX	500	1500
AH	520	1190

Figure 8. Phonemes and their corresponding formant frequencies

To map a set of formants to a given phoneme the Euclidian distance in two dimensions was used to find the closest phoneme to each set. A separate key frame was assigned to accommodate unvoiced speech where the formants were both mapped to zero.

Next, to map these phonemes to mouth positions, a set of mouth expressions were generated to mimic the lip positions created while voicing the sound. These positions were generated by creating a polygon from eight vertices that mimic articulation points of the mouth. Figures 9-16 demonstrate all of the phonemes and their corresponding mouth positions.

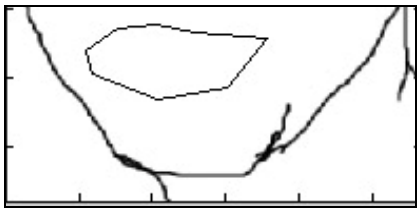

Figure 9. AA

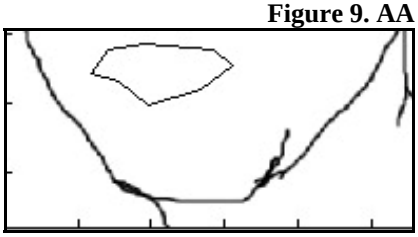

Figure 10. AE

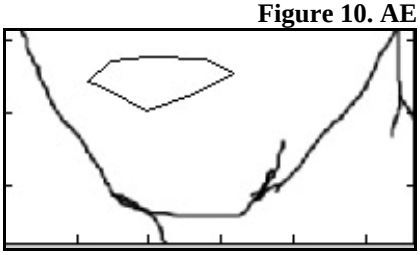

Figure 11. EH

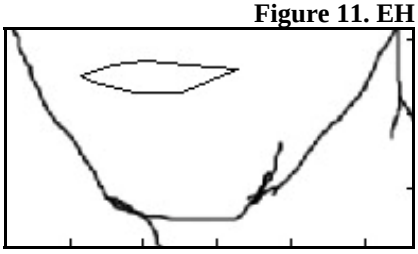

Figure 12. AH

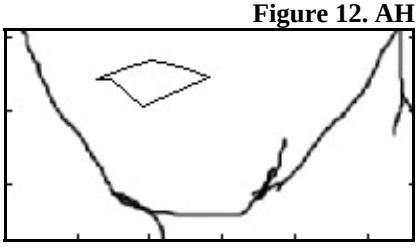

Figure 13. AO

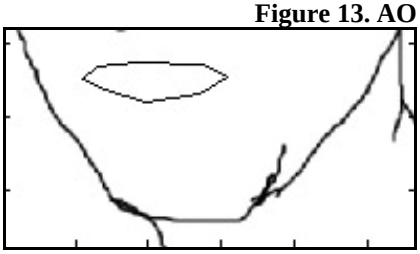

Figure 14. ER

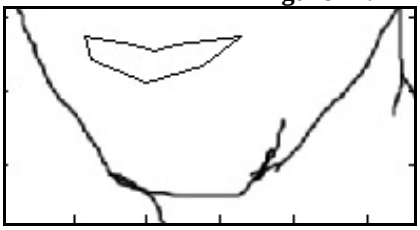

Figure 15. IH

Figure 16. IY

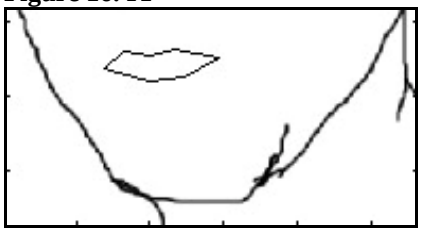

Figure 17. UH

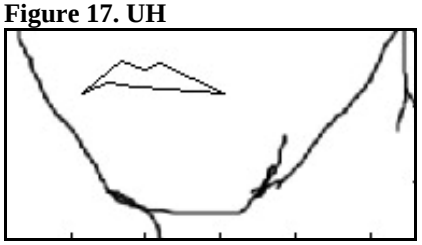

Figure 18. Un Voiced

These position keys each contained the x and y Cartesian coordinate for each of the eight vertices. A linear interpolation of each vertex position was performed to fill in the samples between key frames. The result is a smooth mouth motion following the voiced phonemes in the speech signal. These key frames are placed over the image below, to provide a facial reference superimpose the mouth on top of.

Figure 19. Face template

Aside from simply overlying this polygon on top of the face image, extra features such as blinking the eyes were added to make the sequence seem more believable and look more realistic.

4. Conclusions and Future Work

Through this project, it was proved that that ability to perform fairly accurate lip synching is possible. Undertaking this project in one semester was very ambitious, but the results provided proof of concept. Given more time, it would be trivial to extend this implementation into three dimensional characters and morph keys. Another extension of this algorithm would be to create key frame representations for more phonemes including unvoiced speech and consonants. Also, given more time the procedure could be optimized to run on individual frame packets in real time. However, given the goal of this project, the overall results proved to be a success.

5. References

[1] http://ai.korea.ac.kr/papers/ideal2002.mudwall.pdf

[2] http://sips03.snu.ac.kr/pub/conf/c56.pdf