Formant Plotting on iOS Devices

We are investigating the use of an iOS device (iPhone, iPad, or iPod Touch) to measure the resonant frequencies of human speech and plotting of the formant frequencies on the screen of the iOS device. This is a preliminary report that serves as a Proof-of-Concept (PoC) study as well as a demonstration of capabilities of the freelancer (Dr. Muhammad Akmal Butt).

Due to the challenging nature of the project, the whole project will be executed in three stages. The first stage includes MATLAB-based processing of a few vowel sounds, documentation of results, and a detailed plan for next two stages. The second stage includes real-time speech capturing from an iOS device and real-time measurement of the energy in the captured speech. The third stage includes measurement of formant frequencies from the captured speech and displaying of these frequencies.

As a first step to conduct this PoC study, we picked 8 sound samples with different vowels in them and combined them to create a speech signal that is more than 13 seconds long. The time-domain waveform of the signal is plotted below in Figure 1.

sounds_plot.emf

Figure 1. Time-domain waveform of 8 vowel sounds

Next, we perform energy detection on this long speech signal and only take those parts of the signal that have energy above a threshold. The selection of this threshold is a manual and interactive operation. To measure energy, we break the signal into 0.1 second long segments and find sum of squares of samples in the segment. The results of energy measurement operation are shown below in Figure 2 and MATLAB code to implement this algorithm is given on next page.

energy_plot.emf

Figure 2. Energy plot of the sound sample with 8 vovels

FS = 44100; % Sampling frequency

seg\_length = FS/10;

energy\_threshold = 1;

data\_length = length(sounds);

segments = floor(data\_length/seg\_length)

energy\_flag\_vector = zeros(1,segments);

for seg\_idx = 1:segments

sound\_seg = sounds((1+(seg\_idx-1)\*seg\_length):(seg\_idx\*seg\_length));

energy = sum(sound\_seg .\* sound\_seg);

if (energy > 1)

energy\_flag\_vector(seg\_idx) = 1;

end

end

subplot('position',[0.03 0.44 0.962 0.53])

time\_v = [1:length(sounds)]/fs;

plot(time\_v,sounds)

axis([0 max(time\_v) -1 1])

set(gca,'XTick',[1:13])

set(gca,'YTick',[-1 -0.5 0 .5 1])

grid on

subplot('position',[0.03 0.01 0.962 0.32])

plot(energy\_flag\_vector,'LineWidth',3)

axis([0 length(energy\_flag\_vector) -0.1 1.2])

grid on

print -dmeta energy\_plot

The next step would be to discard 0.1 seconds of speech from each detected sound burst and process the remaining portion to measure its format frequencies.

We followed the procedure given at <http://www.phon.ucl.ac.uk/courses/spsci/matlab/lec10.html> to find format frequencies of 8 vowels. Since there are more than one segments (of length 0.1 second) in every vowel, we find the linear prediction coding model for all segments in a vowel and take the average of the models. The frequency responses of 8 average model filters are plotted in Figure 3 below.

ARModel_plots.emf

Figure 3. Frequency responses of average linear-prediction model filters for 8 vowels.

Since we are going to plot the location of first two formant frequencies on a 2D image of vowel map, the locations of two formant frequencies of all speech segments, as well as those from average filter are plotted in Figure 4 below.

average_formant_cluster.emf

Figure . Cluster plot of two formant frequencies of all speech segments (blue legend) and the formant frequencies obtained from average model (green legend).

By processing of given speech samples using MATLAB, we learned valuable things, particularly from Figure 4. We can see that if a speech signal does not change over the duration of observation, the formant frequencies remain stable and we get a tight cluster (4th cluster). On the other hand, if speech sample is short and surrounded by consonants, we get poor results (3rd cluster). In general, the clusters are well spread and we need to perform post processing on the results of all speech segments present in a vowel sound.

Now, we are in a good position to estimate the effort required for real-time processing of human speech using an iOS device. Our findings can be summarized as:

1. FFT-based Cepstrum of a speech sample can be computed easily using readily available software provided by Apple in AurioTouch sample application.
2. There is no readily available iOS library available for implementation of linear prediction based measurement of formant frequencies. But we can find an open-source C code and include it in our project. One example of such open source code is  
   <http://www.koders.com/c/fidE29F4CF19B7A1413AA3CDF9633466CAF9EA18A9A.aspx>
3. It is not advisable to re-invent the wheel; hence we should not write our own objective-C routines to implement linear prediction coding. But it will not be straight forward to take another person’s code and integrate it with our project. We need to understand the flow of the code to be able to test and verify its performance. Hence, the third phase of the project remains challenging.
4. The second stage of this project can be easily implemented using CoreAudio framework, as it is implemented in AurioTech and other examples provided by Apple. It will take 5-7 days to implement that phase for a fixed fee of $400 (including vworker.com fees).
5. The last stage is challenging but do-able. It is proposed that instead of fee-for-deliverables we should go for fee-for-time model to implement that phase. We can devise a plan that pays a portion of the hourly rate during development and the remaining component is only paid if reasonable progress is made. We can negotiate this in near future over a skype call.
6. Due to inherent challenges in the third stage, we can only have a rough estimate of the cost for that phase. The cost will be in $1500 - $2500 range.

This work was performed as a demonstration of capabilities to develop, test, and document speech processing work using MATLAB. A similar approach will be adopted if this work is to be implemented on an iOS device.