Due date: Sep 17th 2015

Project1

Project1.1

Requirement

Write a program to capture speech data. It must include the following:

- Actual speech capture. The captured speech signals must have **16-bit resolution**, **mono channel** and be captured at a **sampling rate of 16000 samples per second**. If you're one of the unfortunates stuck with working on a Mac, you may use a sampling rate of 44100 instead.
- Endpointing, with hit-to-talk. Recording must begin at a keyboard hit, and stop automatically when end of speech is detected. You may use one of the endpointing schemes mentioned in class 2 to find the trailing endpoint, or any other method you may come up with.
- The endpointed segment must be written to file in Microsoft PCM wav format.

Suggestion

1. You can use [portaudio](http://portaudio.com/) for the audio capture. Portaudio is a well established cross-platform audio capture package.

2. You can use `readwave.h` to read and write wave. Please refer to the `ReadWave` and `WriteWave` function.

Project1.2

Requirement

Write a routine for computing MFCC from audio

- Record multiple instances of digits
    - Zero, One, Two etc.
    - 16Khz sampling, 16 bit PCM
    - Compute log spectra and cepstra
        - Use 40 Mel spectral filters. They must cover the frequencies between 133.33Hz and 6855.4976Hz (you may use a different setting if you choose).
        - No. of features = 13 for cepstra (use first 13 DCT coefficients)
    - Visualize both spectrographically (easy using matlab)
        - Note similarity in different instances of the same word
    - Modify number of filters to 30 and 25 (over the same frequency range).
        - Patterns will remain, but be more blurry

Suggestion

You are allowed to refer to other people's code and implement it by yourself.

- Dan Ellis has nice matlab code on his website.
  http://www.ee.columbia.edu/~dpwe/resources/matlab/rastamat/
- The "wav2feat" code in CMU sphinx is good.
  wav2feat.c, fe_sigproc.c, etc

However, we recommend doing your own code if you can.

Regardless of what you use, the feature computation code must be integrated with the audio capture routine.

- Assume keyboard hit for start of recording. Stop of recording is obtained via automatic endpointing.

How to visualize the spectrogram represented by cepstra

The Mel-log spectrum can be directly visualized as a matrix.

However, the cepstrum is a dimensionality-reduced and transformed version of the log spectrum. It is not visually meaningful. However, the truncated cepstrum can be converted back to a log spectrum by zeropadding it to 64 or 128 poitns and computing an inverse DCT (if you used a DCT to derive cepstra from log spectra). The IDCT-derived logspectrum is what the cepstrum really represents.

You can use matlab to visualize the IDCT-derived logspectrum offline.

> please refer to following functions

[`wavread`](http://cn.mathworks.com/help/matlab/ref/wavread.html),

[`dct`](http://cn.mathworks.com/help/signal/ref/dct.html?searchHighlight=dct),

[`idct`](http://cn.mathworks.com/help/signal/ref/idct.html?searchHighlight=idct),

[`plot`](http://cn.mathworks.com/help/matlab/ref/plot.html)

In the demo and report, the students must plot the original audio waveform, the power spectrum (y axis 256 bins), the log mel-scale filter bank energy (y axis 40 dimension), the cepstrum after DCT (13 dimension), the log mel energy after IDCT (y axis 40 dimension).

> If you finish the project correctly, `log mel-scale filter bank energy` and `log mel energy after IDCT` sould be similar when you plot them.