```
 1 //
 2 // Created by daran on 1/12/2017 to be used in ECE420 Sp17
   for the first time.
 3 // Modified by dwang49 on 1/1/2018 to adapt to Android 7.0
   and Shield Tablet updates.
 4 //
 5
 6 #include "ece420_main.h"
 7 #include "ece420_lib.h"
 8 #include "kiss_fft/kiss_fft.h"
 9
10 // JNI Function
11 extern "C" {
12 JNIEXPORT float JNICALL
13 Java_com_ece420_lab4_MainActivity_getFreqUpdate(JNIEnv *env
   , jclass);
14 }
15
16 // Student Variables
17 #define F_S 48000
18 #define FRAME_SIZE 1024
19 #define VOICED_THRESHOLD 3e9  // Find your own threshold
20 #define START 40
21 #define END FRAME_SIZE/4
22 float lastFreqDetected = -1;
23
24 void ece420ProcessFrame(sample_buf *dataBuf) {
25     // Keep in mind, we only have 20ms to process each
   buffer!
26     struct timeval start;
27     struct timeval end;
28     gettimeofday(&start, NULL);
29
30     // Data is encoded in signed PCM-16, little-endian,
   mono
31     float bufferIn[FRAME_SIZE];
32     for (int i = 0; i < FRAME_SIZE; i++) {
33         int16_t val = ((uint16_t) dataBuf->buf_[2 * i
   ]) | (((uint16_t) dataBuf->buf_[2 * i + 1]) << 8);
34         bufferIn[i] = (float) val;
35     }
36
37     // ********************** PITCH DETECTION
    ********************** //
38     // In this section, you will be computing the
   autocorrelation of bufferIn
```

```
39      // and picking the delay corresponding to the best
    match. Naively computing the
40      // autocorrelation in the time domain is an O(N^2)
    operation and will not fit
41      // in your timing window.
42      //
43      // First, you will have to detect whether or not a
    signal is voiced.
44      // We will implement a simple voiced/unvoiced detector
    by thresholding
45      // the power of the signal.
46      //
47      // Next, you will have to compute autocorrelation in
    its O(N logN) form.
48      // Autocorrelation using the frequency domain is given
    as:
49      //
50      //  autoc = ifft(fft(x) * conj(fft(x)))
51      //
52      // where the fft multiplication is element-wise.
53      //
54      // You will then have to find the index corresponding
    to the maximum
55      // of the autocorrelation. Consider that the signal is
    a maximum at idx = 0,
56      // where there is zero delay and the signal matches
    perfectly.
57      //
58      // Finally, write the variable "lastFreqDetected" on
    completion. If voiced,
59      // write your determined frequency. If unvoiced, write
     -1.
60      // ********************* START YOUR CODE HERE
     ********************** //
61  //    def ece420ProcessFrame(frame, Fs):
62  //    freq = -1
63  //    if np.sum(np.square(frame))>threshold:
64  //    auto=np.fft.ifft(np.fft.fft(frame)*np.conjugate(np.
    fft.fft(frame)))
65  //    peak=np.argmax(np.abs(auto[start:len(frame)//2]))
66  //    freq=Fs/peak
67  //    return freq
68      float energy=0;
69      for (int i=0;i<FRAME_SIZE;i++){
70          energy+=bufferIn[i]*bufferIn[i];
71      }
```

```
 72     if(energy>VOICED_THRESHOLD){
 73         kiss_fft_cfg cfg=kiss_fft_alloc(FRAME_SIZE,0,NULL,
    NULL);
 74         kiss_fft_cfg cfg_inv=kiss_fft_alloc(FRAME_SIZE,1,
    NULL,NULL);
 75
 76         kiss_fft_cpx in[FRAME_SIZE];
 77         kiss_fft_cpx fft[FRAME_SIZE];
 78         kiss_fft_cpx ifft[FRAME_SIZE];
 79
 80         for (int i=0;i<FRAME_SIZE;i++){
 81             in[i].r=bufferIn[i];
 82             in[i].i=0;
 83         }
 84         kiss_fft(cfg,in,fft);
 85         for (int i=0;i<FRAME_SIZE;i++){
 86             in[i].r=(fft[i].r*fft[i].r+fft[i].i*fft[i].i);
 87             in[i].i=0;
 88         }
 89         kiss_fft(cfg_inv,in,ifft);
 90         float max_value=-1;
 91         int peak=0;
 92         for (int i=START;i<END;i++){
 93             float mag=ifft[i].r*ifft[i].r+ifft[i].i*ifft[i
    ].i;
 94             if(mag>max_value){
 95                 max_value=mag;
 96                 peak=i;
 97             }
 98         }
 99         lastFreqDetected=F_S/peak;
100         free(cfg);
101         free(cfg_inv);
102     } else{
103         lastFreqDetected=-1;
104     }
105
106     // ********************* END YOUR CODE HERE
     ************************ //
107     gettimeofday(&end, NULL);
108     LOGD("Time delay: %ld us",  ((end.tv_sec * 1000000 +
    end.tv_usec) - (start.tv_sec * 1000000 + start.tv_usec)));
109 }
110
111 JNIEXPORT float JNICALL
112 Java_com_ece420_lab4_MainActivity_getFreqUpdate(JNIEnv *
```

```
112 env, jclass) {
113     return lastFreqDetected;
114 }
```