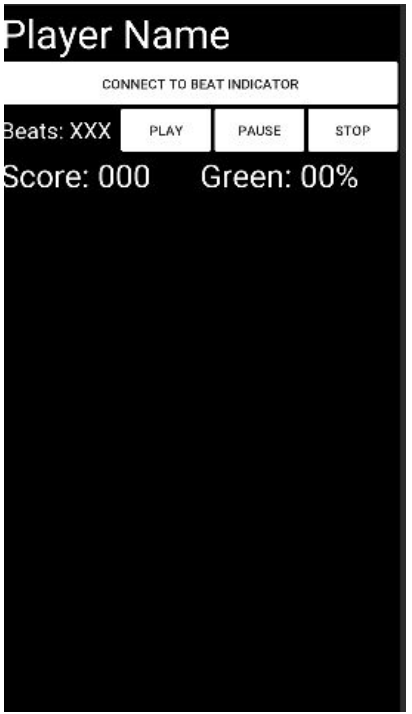
Sound Hero

Michael O’Dell, Jonathan Solheim

Objective

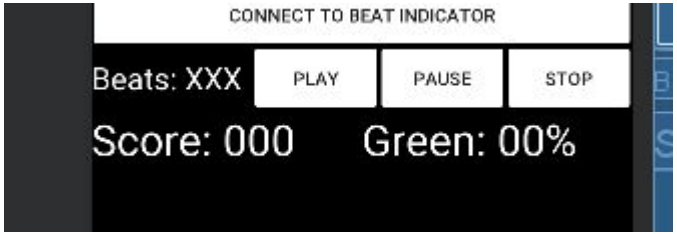
- To develop a “Guitar-Hero” style Android app using methods learned in class
  - Buttons, Intent, Accelerometer, Bluetooth LE, Arduino, Logging
- User attempts to match the kick drum by shaking the phone to the beat
- App connects to Arduino Circuit playground, which changes color corresponding to the kickdrum’s proximity to the user’s attempt to match it
  - Green: on beat
  - Yellow: almost on beat
  - Red: off beat
- Keep score

App Layout(Title Screen Vs Play Screen)



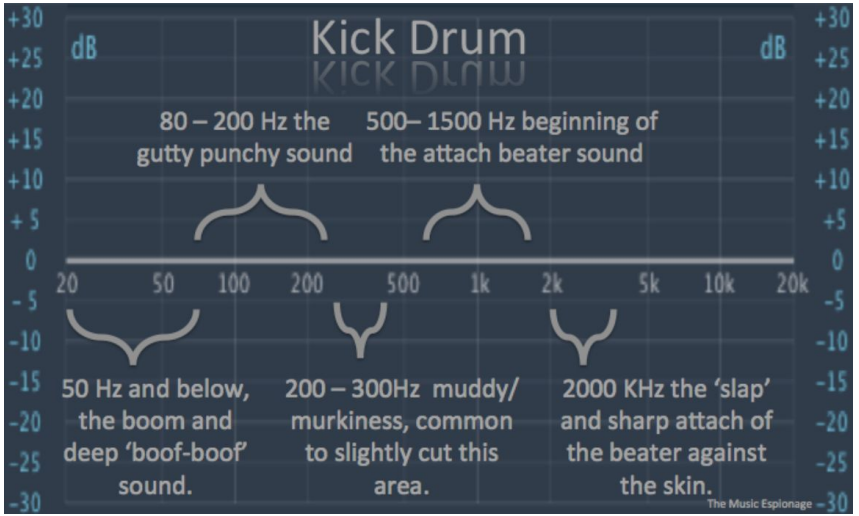
Scoring

- Maximum score is 2 times beats shown
- Green = 2 points
- Yellow = 1 point
- Red = No points
- Green percentage shown to be percentage of point earning beats detected

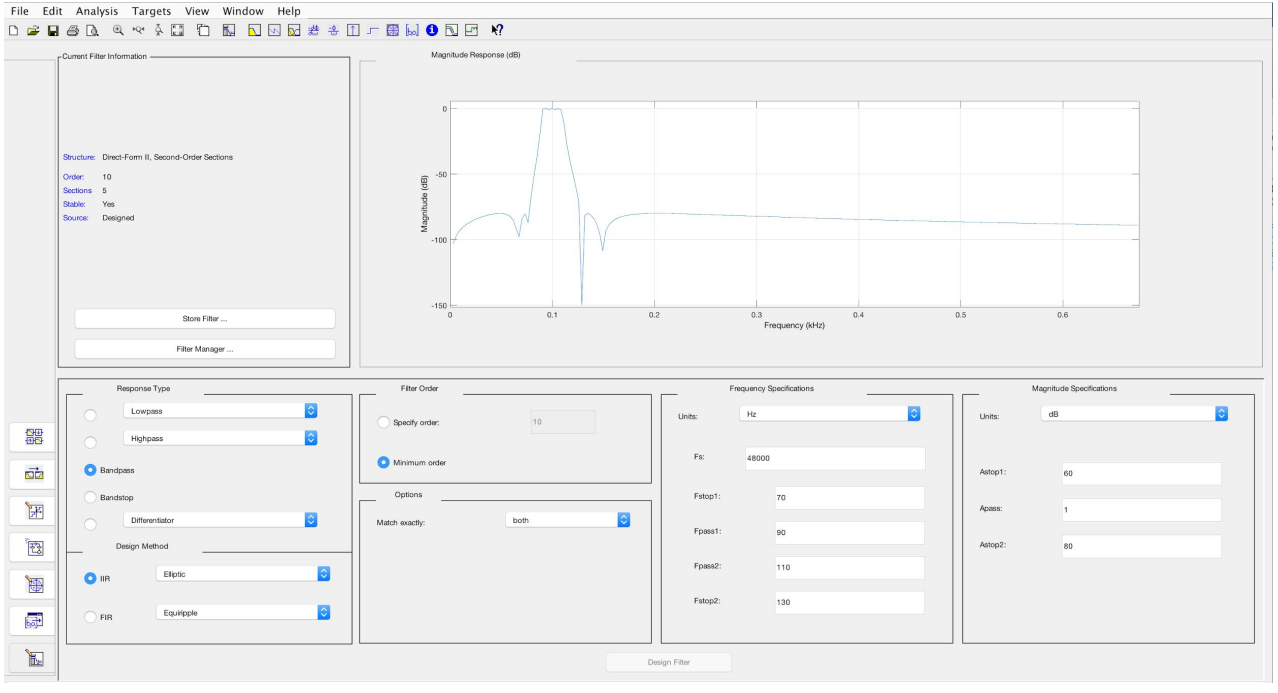


Kick Drum Detection

- Using a techno song
  - Loud Punchy kick
  - Standardized kick
- Find characteristic kick frequency using MATLAB animation
- 100 Hz



MATLAB's Filter Designer



- 100 Hz
- Default settings gave 6k+ sections
- Elliptic gave 5

Filtering Using Difference Equation

- Implementable in Kotlin
- No frequency domain manipulations, only addition and multiplication in a loop
- Coefficients provided by filter designer
- 5 sections

```
21      % Section 1
22 -    k1 = 0.21043727385169030963396608058246783912 2;
23 -    num1 = k1*[1,-1.99971527915500035810225654131500050425 5,1];
24 -    den1 = [1,-1.99955689404341763193428960221353918314, 0.99976386187908339309160510310903191566 5];
25      % Difference equation
26 -    y1d = zeros(length(wav(:,1)),1);
27 -    for n = 3:length(wav(:,1))
28 -        y1d(n) = (num1(1)*wav(n,1) + num1(2)*wav(n-1,1) + num1(3)*wav(n-2,1) - den1(2)*y1d(n-1) - den1(3)*y1d(n-2))/den1(1);
29 -    end
```

Original signal v. Filtered

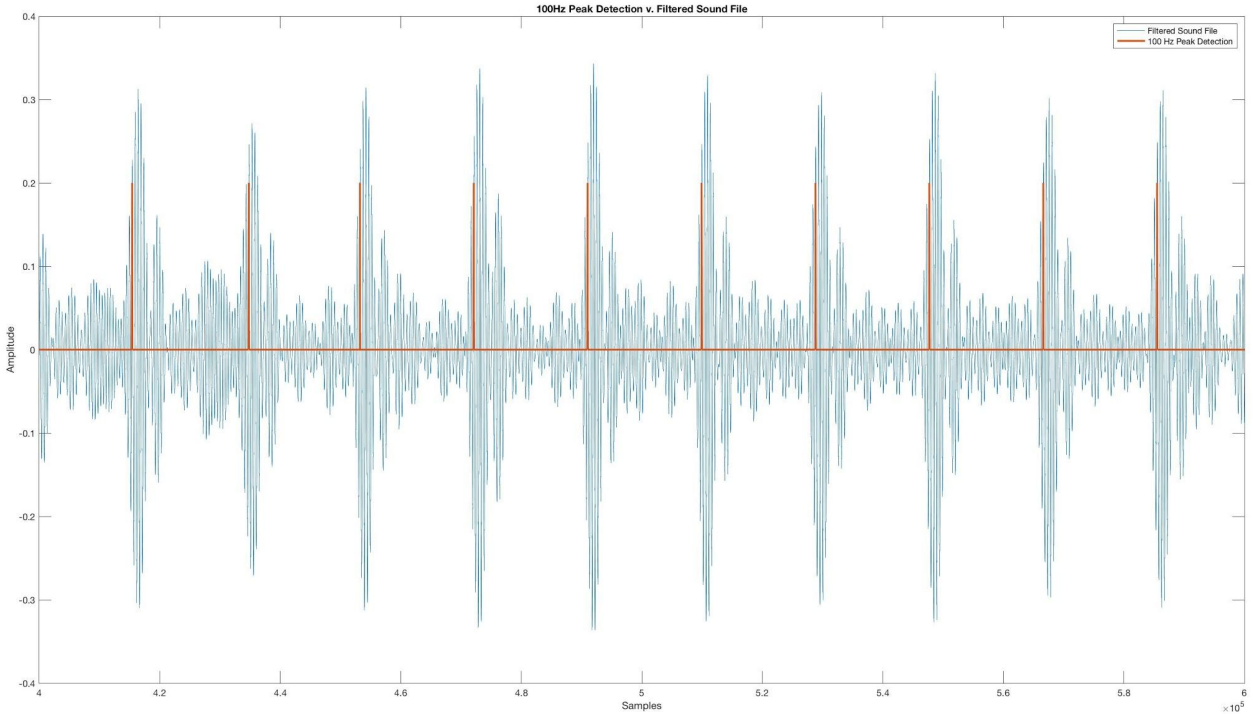
https://www.youtube.com/watch?v=HpdixGRUkIo&feature=youtu.be

Detecting Kick in Filtered Signal

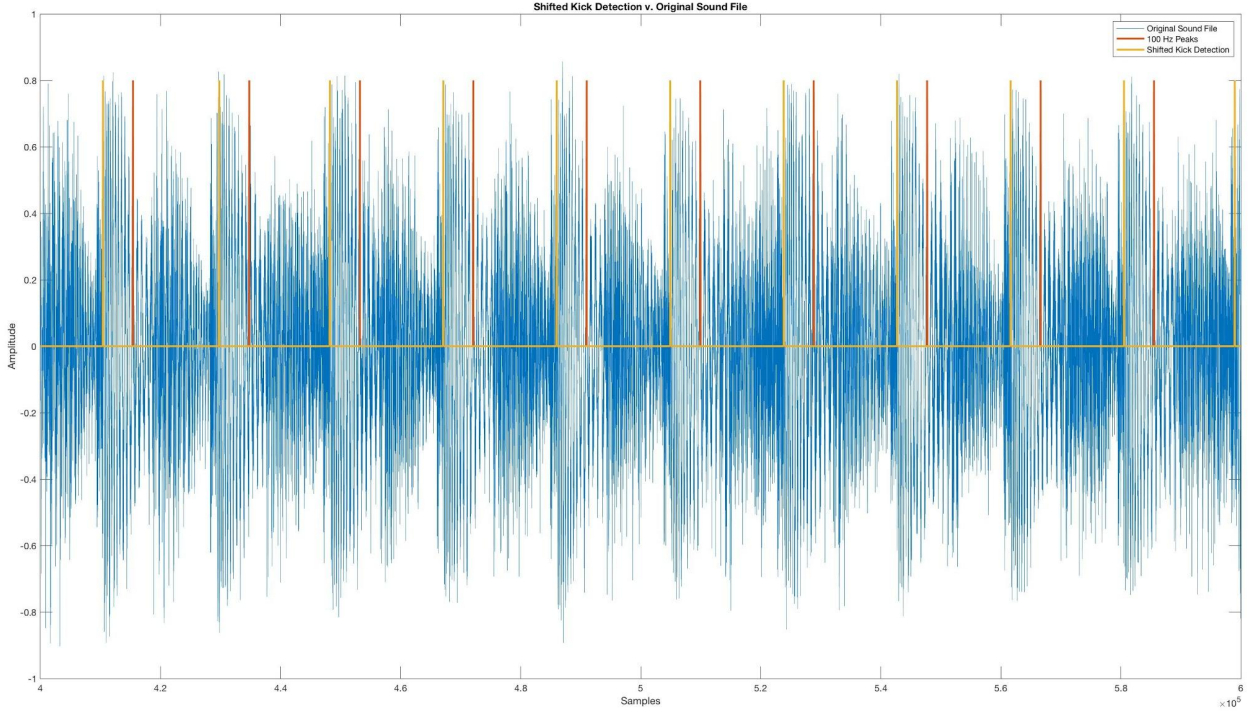
- Implementable in Kotlin
- If signal surpassed threshold, that location is logged
- Cannot log another kick for 4000 samples (~90 mS)
- Create impulse train of hit locations

```
157     % Finding the kick
158 -   hits = zeros(1,length(y5d));
159 -   grace_samp = 4000;
160 -   count = -1;
161 -   hit_location = zeros(1,1000);
162 -   i = 1;
163 -   shift = 5000; % Number of samples to shift
164 -   for n = 1:length(y5d)
165 -       if((y5d(n)>th) && (count < 0))
166 -           hits(n) = th;
167 -           count = grace_samp;
168 -           hit_location(i) = n-shift;
169 -           i = i + 1;
170 -       else
171 -           count = count-1;
172 -       end
173 -   end
```

Original Filtered Kick Detection



5000 Sample Phase Correction (~113 mS)



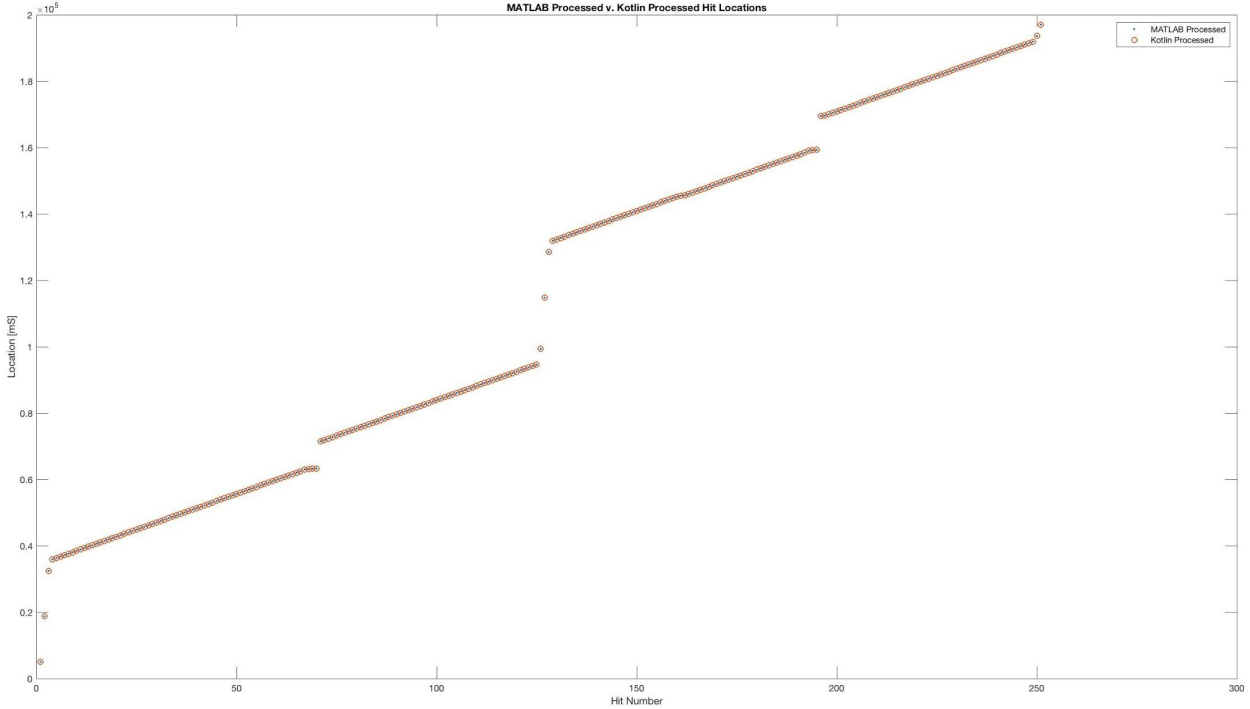
Checking Accuracy of Kick Detection

- Place indicator sound at each hit location by convolving “beep” with impulse train of hits
- Play original sound file and hit location “beeper” at once

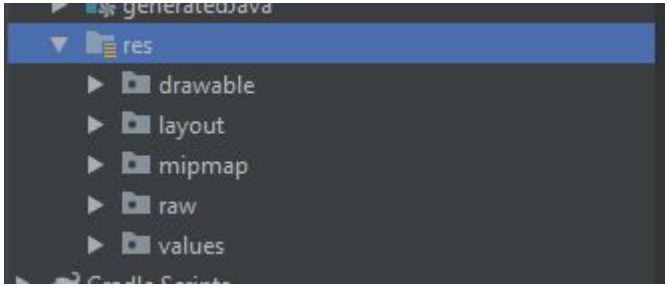
https://www.youtube.com/watch?v=CDhKtbxBQmo&feature=youtu.be

Implementing in Kotlin

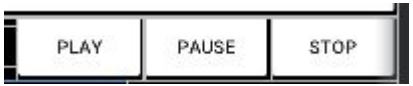
*Using WavFile.java by A.Greensted to read wav file



Media Player



- Media player is Kotlin class that can be used to play a sound file
  - File needs to be in “raw” folder in res directory
- Player will return value of current position when playing, this is useful to determine timing for kick beat
- Pause button implemented to allow game to be paused and maintain score
- Stop button clears game and score



Shake Detection

- Magnitude of sensor data found using all 3 coordinate directions
- Looking for “shakes” that are approximately 2G’s
- If magnitude is greater check detected beat against actual
- Only allow a beat to be sent every 250 ms
- This is to eliminate the backward shake detected from forward

Shake Detection

- If accelerometer passes threshold, Kotlin function compares timestamp of music from media player to know hit locations
- If Mediaplayer timestamp is within +/-30 mS of known hit location, ble sends “green” to Arduino
- If Mediaplayer timestamp is outside +/-30 mS, but within +/-60 mS of known hit location, ble sends “yellow” to Arduino
- Otherwise “red” is sent

Video