Symphonic Harmonics

- Neereja Sundaresan
- Surekha Muralidharan
- Vikram Rajkumar
- Wayne Chiang
Objectives

- Portable, interactive music keyboard
- Many instruments with a variety of ways to play
- Composing tool that keeps track of notes played
- Interfaces with Android for visual display
- Roll-able, various sensors (pressure, accelerometers etc)
Competitive Analysis

- Yamaha Keyboards
  - Multiple instruments, sound recording
  - Reviews: Bulky, limited mobility, no phone interfacing

- Roll-able piano keyboards
  - Highly portable, popular
  - Reviews: limited sounds, hard to press, limited/no composing aids or feedback, no phone interfacing

- Smartphone piano applications
  - Good sound quality and recording capabilities
  - Reviews: Lacks tactility, not realistic size for composing, not standalone
Hardware/Parts

- Arduino Mega 2560
- Bluetooth Mate Silver
- Multiple tactile button switches
- Speakers
- Foam sheets
- Plastic mesh
System Architecture

User Input

Digital I/O

Arduino Mega 2560

Serial

Bluetooth

Bluetooth Mate Silver

Android

Sound Output

Tactile Switches
## Experimentation Plan

<table>
<thead>
<tr>
<th>Factor</th>
<th>Expectation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android application loading time</td>
<td>User should not wait too long for application to load</td>
</tr>
<tr>
<td>Sound file loading time</td>
<td>User should not wait too long while switching sounds</td>
</tr>
<tr>
<td>Bluetooth connecting time</td>
<td>User should not wait too long to connect to keyboard</td>
</tr>
<tr>
<td>Bluetooth packet sending time</td>
<td>Sounds should play as close to button presses as possible</td>
</tr>
<tr>
<td>Bluetooth disconnection behavior</td>
<td>User should be notified when a failure occurs with the option to attempt reconnect</td>
</tr>
<tr>
<td>Concurrent sound playback</td>
<td>User should be able to reliably play as many sounds as they want simultaneously</td>
</tr>
<tr>
<td>Sound switching</td>
<td>User should be able to reliably switch instrument sounds whenever they want</td>
</tr>
</tbody>
</table>
## Initial Data

- **Android app startup:** 1061 ms (avg)
- **Establishing Bluetooth connection:** 3894 ms (avg)
- **Receiving Bluetooth packet:** ~20-60 ms
- **Bluetooth disconnection:** Android and Arduino both detect and alert user
- **Concurrent sound playback and sound switching:** Both work reliably by using Android SoundPool library
- **Sound file loading time:**

![Graph](image)
Lessons Learned

- Test and debug hardware frequently as assembly occurs
- Loading sounds & establishing Bluetooth connection takes a few seconds
- Let user initiate these & use threading to ensure interface doesn’t lock
- Receiving BT packets takes time – streamline receiving procedure
- Sound playback works fine with built-in Android Soundpool library
- Make sure to properly disconnect BT connection on exit
TimeLapse
TimeLapse
TimeLapse
TimeLapse
TimeLapse
TimeLapse
TimeLapse
Possible Extensions

- Windows application
- Rock-Band type matching game implemented in software
- More sounds
- ???
- Profit!