Gesture-Based/Tactile Music Composition

Team 75 - Adam Marcott and Tyler O’Neill
ECE 445 Project Proposal - Spring 2017
TA: John Capozzo

Table of Contents

Introduction

1. Objective

Musicians face a unique problem as artists. When inspiration strikes, expensive recording equipment is not always readily available, especially on-the-go. It is possible, for example, to use a mobile drum sequencer application to “jot down” an idea for a drum pattern, but users are confined to a small space and an often unintuitive interface. Instead of having a dedicated application for a step sequencer, there needs to be a cost-effective way to capture rhythmic taps and smooth gestures that can control any MIDI-based device wirelessly.

The aim of this project is to build a scalable, wireless glove that can capture motion and tactile data from a musician and generate Musical Instrument Digital Interface (MIDI) signals and sounds wirelessly on a mobile platform more intuitively than a traditional graphical user interface (GUI) and without the need for more expensive, specific hardware.

2. Background

Many current solutions exist, such as Air Beats[1], a glove that allows musicians to compose music digitally without interacting with a screen. There are simpler alternatives, such as the Xkey[2], but a MIDI keyboard requires a stable surface and ample space for movement in order to be an effective composition tool. Our device needs neither, as the gestures are compact and only need as much space as your hands occupy. There is yet to be a commercially successful gesture-based product as there is a high reliance on either expensive pre-existing technology with questionable reliability (digital paper technique), or a lack of throughput capability due to inexpressive language during pattern recognition. Our solution aims to solve both of these problems with current solutions by using a minimal set of viable hardware and an expressive language that can be used to quickly translate motion into synthesizable sound.

This expressive language uses simple movements of the hand (right and left, up and down, forward and backward) and rotations (rotate up and down, rotate left and right) in combination with finger placement to enable over 30 different control functions to be available at once. Both hands will be able to generate MIDI notes and controls simultaneously to enable versatile composition. For example, when a user holds their left thumb to their index finger and moves their hand up or down, it will control volume, whereas if they were to make the same motion with their left thumb held to their ring finger it would control a modulation wheel. The most interesting part is that these MIDI control signals can be easily remapped by the user to control any MIDI-compatible device.

3. High-Level Requirements

- Glove must be able to stay powered with a Li-ion battery for more than 2 hours.
- Glove must be able to process sensor data (switch presses/taps and gestures from a defined library) and map it to MIDI data.
- Glove must be able to wirelessly transmit MIDI data over Bluetooth LE.

Design

Our design will satisfy our project requirements for the following reasons:
First, our Bluetooth transmitter will be able to transmit MIDI data by leveraging the MIDI over Bluetooth protocol. We will implement this protocol on the Adafruit Trinket microcontroller. Second, our sensors will be able to recognize gestures with the IMUs providing orientation and acceleration data. The sensors also recognize taps, thanks to the mechanical switches located on the tip of the finger and the fingerprint of each finger. Third, our power subsystem will include a voltage regulator to provide a consistent 3.3V well over our 2 hour requirement (we plan on using a Li-ion battery with 1000mAh capacity).

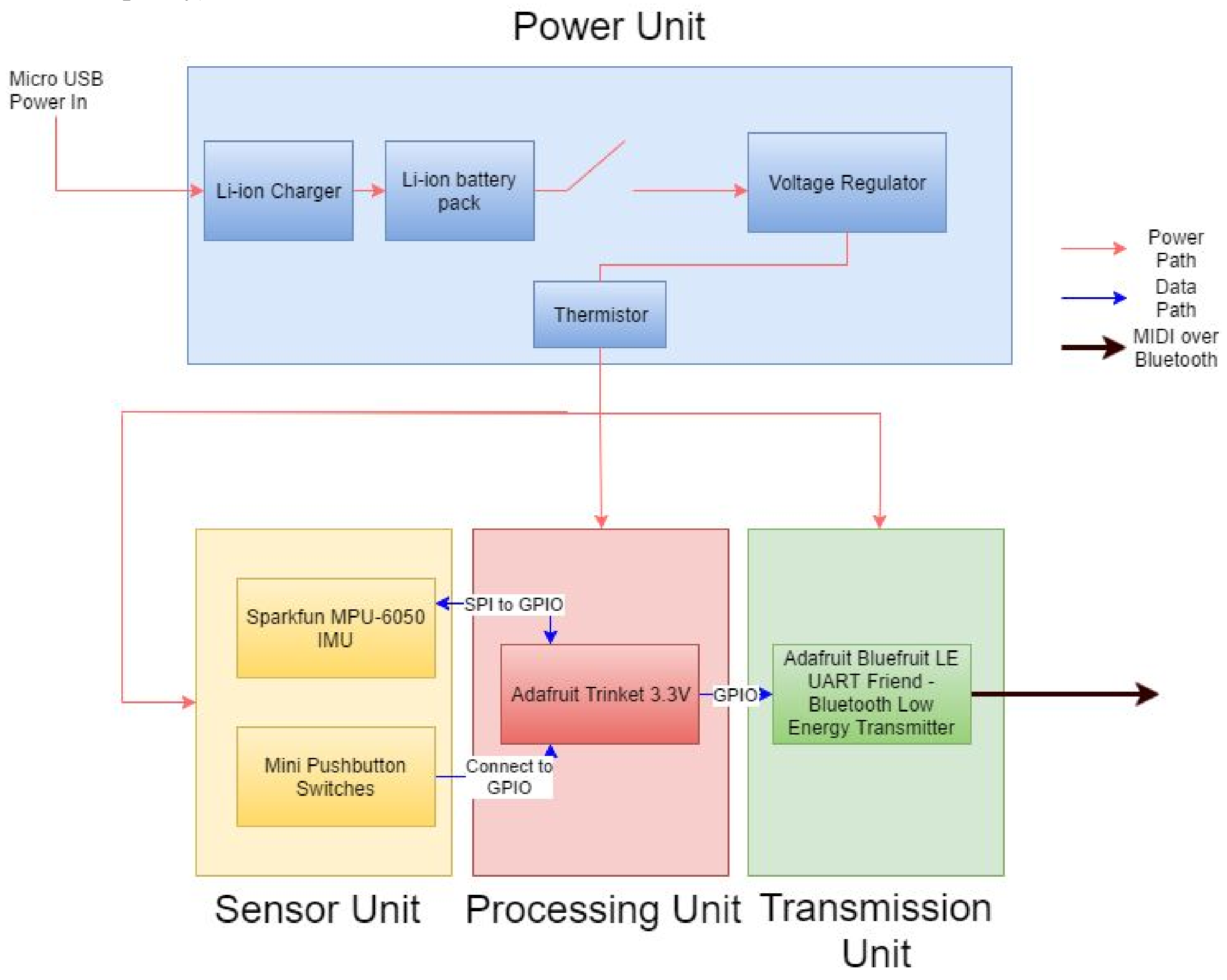

Figure 1. Block Diagram

Our physical design diagram shows the key placements of the devices we will use in our project. Notably, we will consolidate most of the crucial elements -- the microcontroller, transmitter, and power supply -- together on the back of the glove. This placement will vastly simplify connecting components and keep the design concise. We opt to use two

IMUs -- one on the wrist, and one on the middle finger's first knuckle. This decision gives us the flexibility we need to recognize all of the gestures we require.

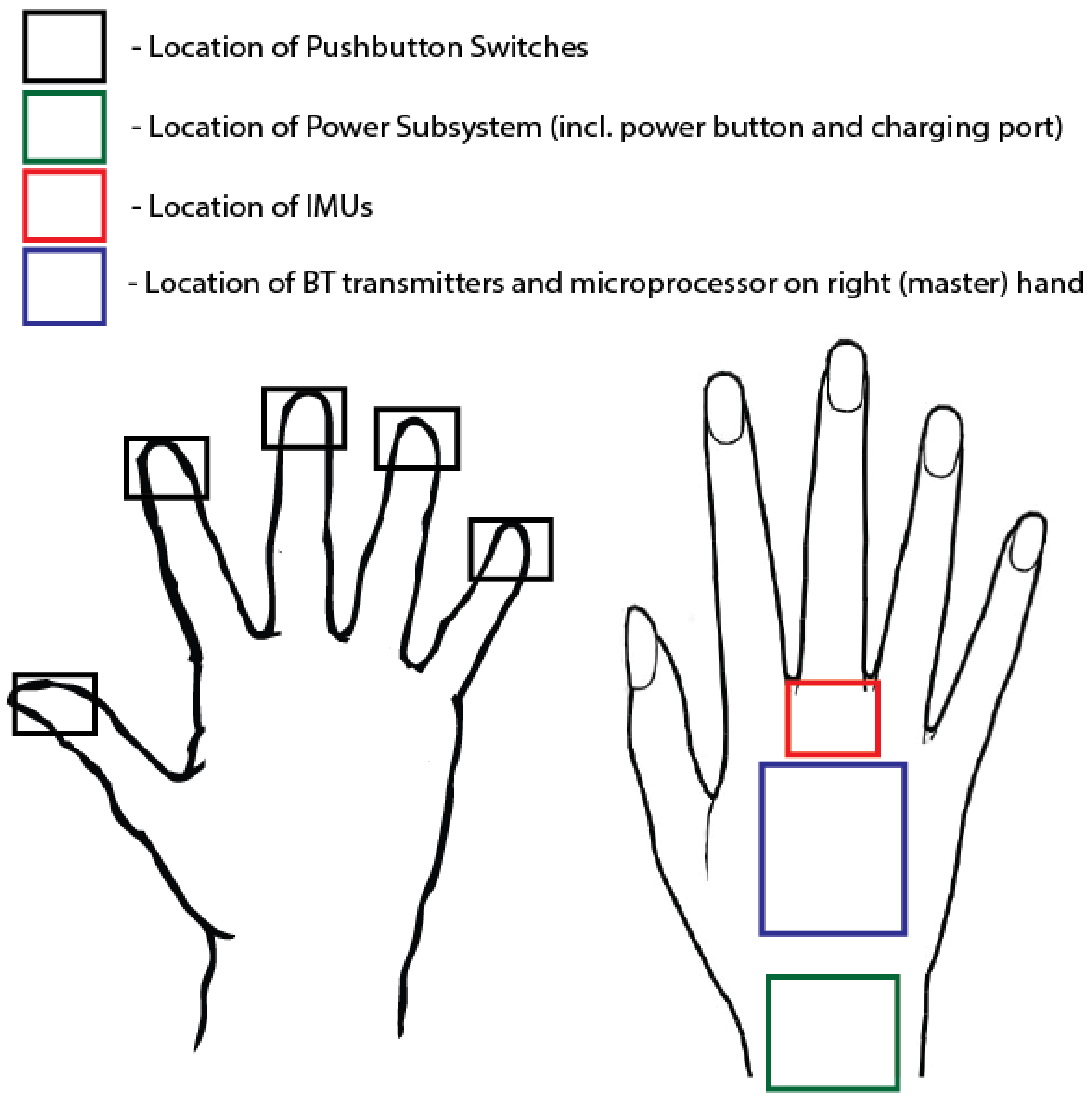
- Location of Pushbutton Switches
- Location of Power Subsystem (incl. power button and charging port)
- Location of IMUs
- Location of BT transmitters and microprocessor on right (master) hand

Figure 2. Physical Design

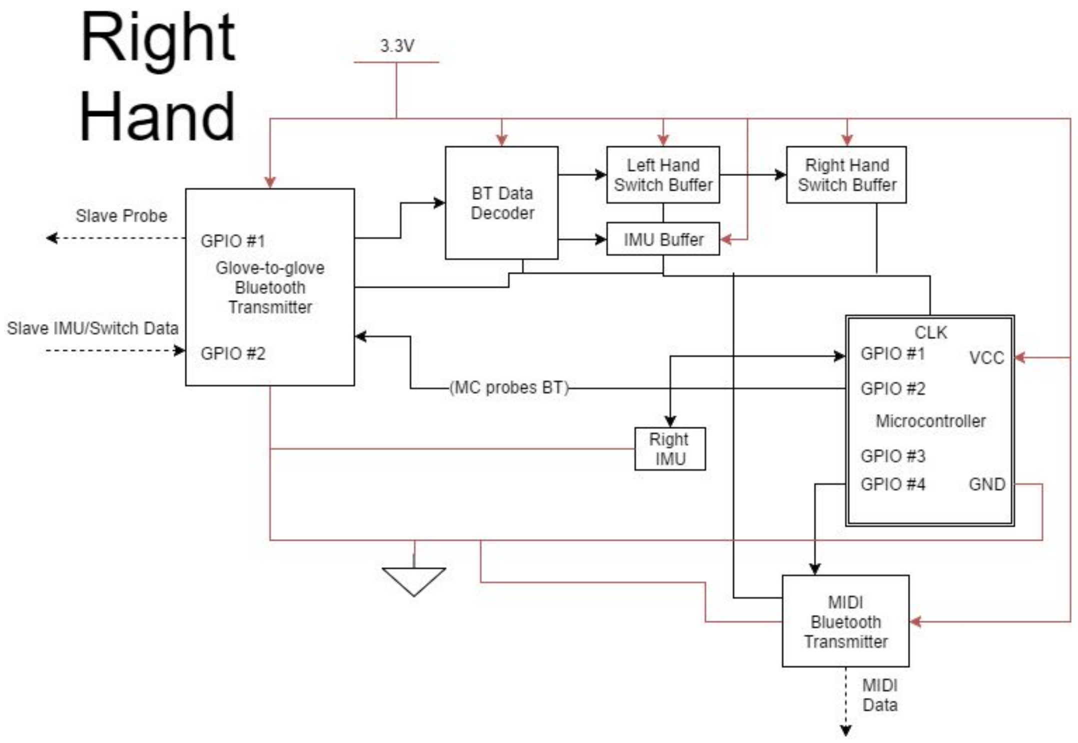

Figure 3. Right Hand High-Level Circuit Diagram

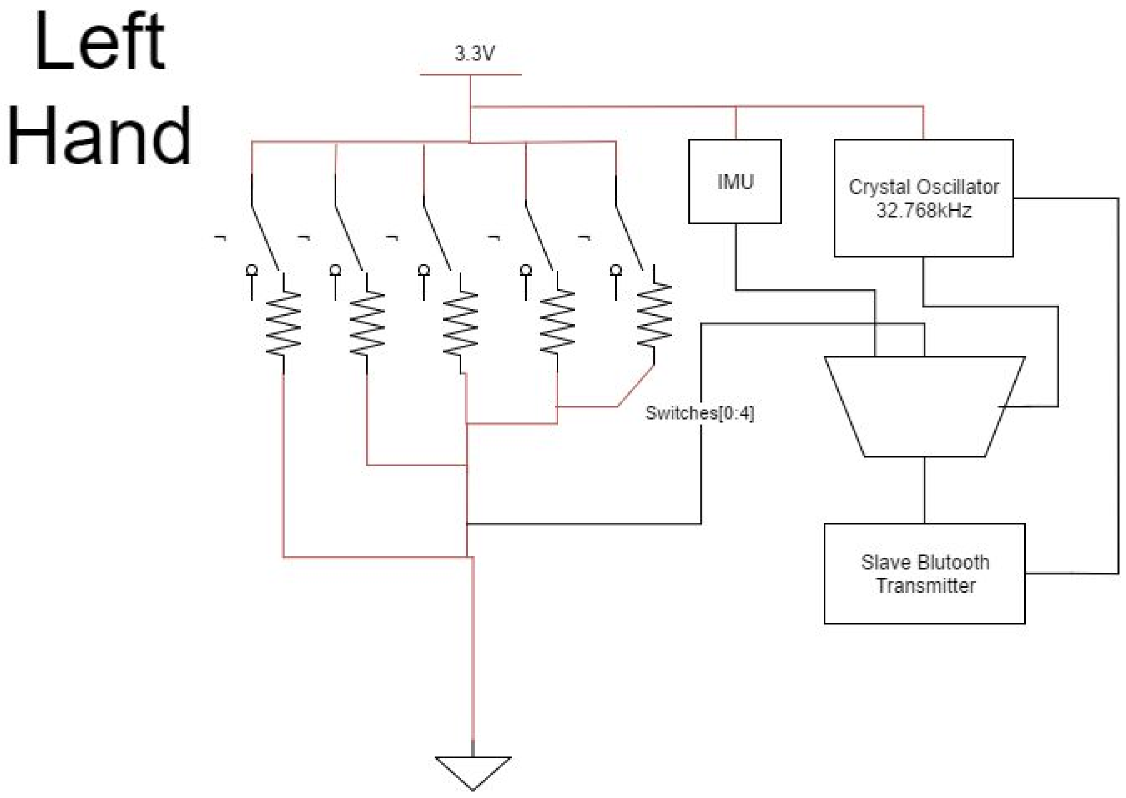

Figure 4. Left Hand High-Level Circuit Diagram

1. Power Subsystem

A power supply is required to safely power all components of the system.

1.1. Li-ion Charger

This is an IC that will allow us to safely charge the Li-ion battery pack. It will act in redundancy to the battery's built-in overvoltage and overcurrent protection to keep the battery in nominal ranges while going through a charge/discharge cycle.

Requirement	Verification
The charger will allow the battery pack to be charged to 3.6 - 3.8V within 4 hours while staying below the maximum charging temperature of 45ºC[4].	From the fully discharged state, we will monitor battery cell voltage over the time period to ensure that the required voltage is reached. We will also use an IR thermometer to ensure the battery does not exceed the maximum temperature during this charging cycle.

1.2. Li-ion Battery

The battery will be able to power all components of the system through the voltage regulator mechanism. It will have 1000mAh capacity which should be more than sufficient in this application.

Requirement	Verification
Must power all components stably for at least 2 hours from a full charge (which we will define as within 50mAh of its 1000mAh listed capacity).	From the fully charged state we will use the glove for 2 hours of continuous operation and ensure that the device is still on after that period.

1.3. Voltage Regulator

Transforms the battery's output voltage of 3.7V to the voltage needed by the hardware components of the glove, which is 3.3V. Also acts as a protection mechanism for the mechanical switches, which do not have built-in voltage regulation, and as a redundancy for all the other components which do have built-in protection circuits.

Requirement	Verification
Must provide 3.2-3.4V from the 3.6-3.8V source continuously while glove is on.	Using a voltmeter we will measure the output voltage for 1 minute to ensure it is within the nominal range for that period.

2. Sensor Unit

Sensors are required to enable the user to perform gestures and taps, which will be recognized by the glove and converted into a MIDI signal.

2.1. Inertial Measurement Unit (IMU)

An IMU is needed to detect and quantify hand movement and orientation for gesture recognition. We will use two units - one placed on the glove where the first knuckle of the middle finger is on each hand. The microcontroller's control function will step through and probe linear acceleration on all three axes and angular velocity on the x- and y-axis to make gesture decisions. The IMU will use a 1MHz SPI protocol and input to a GPIO pin on the Bluetooth Unit or the Processing unit, depending on the hand in question.

Requirements	Verification
Gyroscopic functionality can distinguish an angle on all three axes to an accuracy within +/- 0.1 rad.	1. Make sure that in the duration of 10 seconds, the drifting effect of the gyroscope will not make the accuracy unacceptable 2. Run the gyroscope for four hours 3. Set the gyroscope on a constant rotation rate of 60 degrees/second about all three axes in the phone coordinate system, and record data 4. Compute the accuracy of the gyroscope by comparing the percent error between the measured data and the ground truth
Accelerometer functionality can distinguish acceleration on all three axes to an accuracy within +/- 0.1g.	Testing Accelerometer: 1. Test the accuracy of the gravitational acceleration 2. Use measurements of another accelerometer with known superior

	performance (with accuracy within 1%) as the ground truth and compare it with the measurements with our accelerometer readings to obtain the accuracy for our accelerometer.

2.2. Mechanical Switches

Mechanical switches are needed to detect taps on a surface. These sensors work by measuring the deformation of a physical material[5]. In this application, they will measure glove deformation, and we will determine a threshold of measurement that we will consider a "tap". These sensors will be placed on the fingertips of the glove. In Figure 5, a simulation shows a range of force from a tap that can be expected. The 20N requirement is in place to make sure the glove does not break if there is too hard of a tap. Figure 6 shows the circuit that will be used to ensure a tap is registered, whether or not the impact comes from the fingertip or fingerprint region.

Requirements	**Verification**
Mechanical switch can detect a force of up to 20N without malfunctioning.	Pressing down with a force of 20 cN should send a high signal

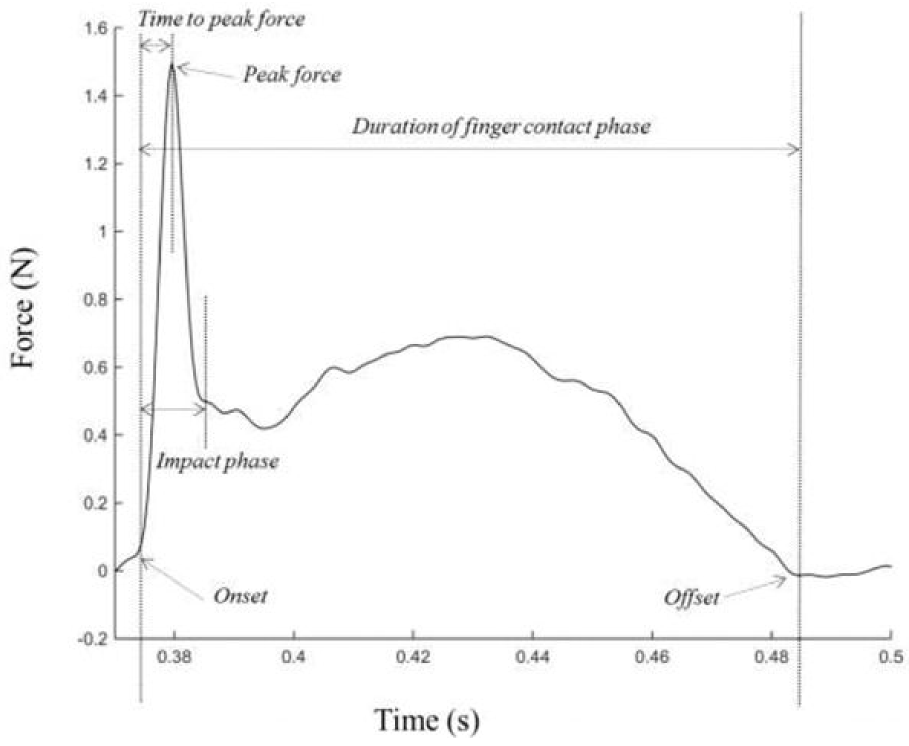

Figure 5. Recorded force over time of a finger tap[7]

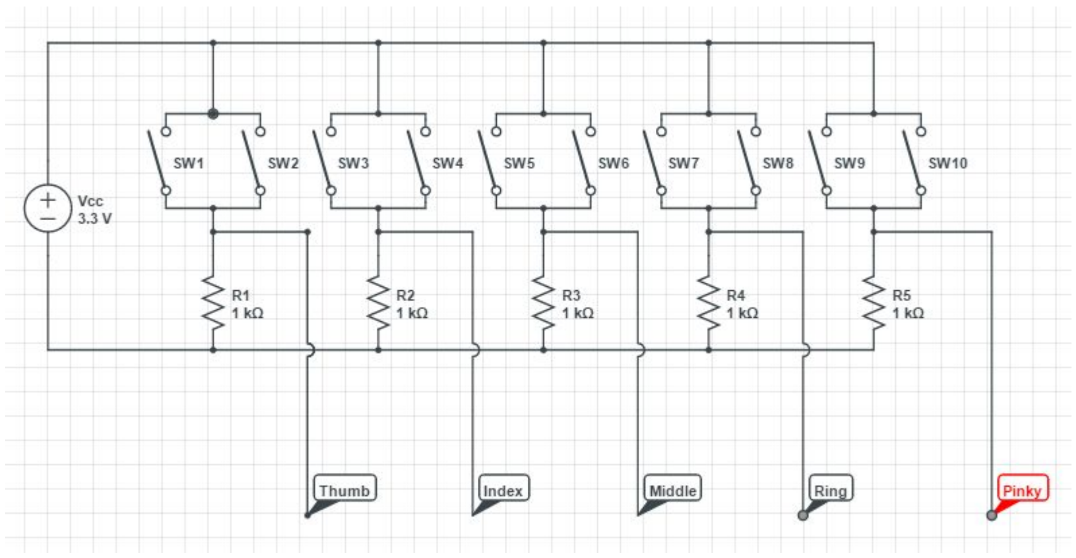

Figure 6. Circuit Diagram of Mechanical Switch Sensing

3. Processing/Transmission

Microcontrollers and Transmission are required to generate the MIDI signal and wirelessly send the signal over Bluetooth to a compatible receiver.

3.1. Microcontroller

We need a Microcontroller to process all sensory input data and convert it into a meaningful MIDI signal. The microcontroller is the heart of this project: all paths go into or out of this device. It will act reliably and consistently to meet our project requirements, and it will serve as the clock for all synchronous components on the right hand (8MHz).

The microcontroller will determine the current state (Normal, Gesture, Calibration) based on the combinations of mechanical switches. These states are defined later in the document.

We will be using all 5 GPIO pins on the microcontroller, defined as follows:

1. Probe Left Hand
2. Left Hand IMU Data
3. Switch Data Buffer
4. Right Hand IMU Simultaneous Probe/Data
5. MIDI Out to Bluetooth Transmitter

Requirements	**Verification**
Translates gestures and taps into a MIDI signal with greater than 90% accuracy.	From the defined set of gestures and taps we will perform a series of 100 randomly chosen actions and expect the microcontroller to generate the correct MIDI signal for at least 90 of those actions.

3.2. Bluetooth Low Energy (BLE) Transmission Unit

We need a Bluetooth LE unit because it will enable us to use our glove with any MIDI controller supporting MIDI over Bluetooth. The unit will receive a MIDI signal from the microcontroller and transmit it wirelessly to any (arbitrary/outside project scope) receiver which we assume will be a MIDI controller that will enable playback.

The Bluetooth unit we have chosen has 32kB RAM, 256k Flash Memory and a 9600 baud transmission rate.

We will be using three Bluetooth units. The first will be placed on the left hand, sending either bytes of switch data or IMU data to the left hand. The second receiver will send probe bytes to the left hand BLE transmitter and receive either IMU or mechanical switch bytes in return. The third unit will take the output of the microcontroller and transmit MIDI data to any compatible MIDI device.

Requirements	**Verification**

The Bluetooth LE connection allows communication between the two gloves and a receiver within 0 to 5.5m.	We will place a receiver at the minimum and maximum distance and ensure that signals are still received.

4. Modes of Operation

There are three distinct modes of operation for the device depending on user input. These modes are:

- Normal mode
  - Generates MIDI notes from finger switch taps
  - State machine will switch to Gesture or Calibration mode based on combinations of finger switches.
- Gesture mode
  - Enabled by pressing the thumb against another finger on the same hand
  - Allows IMU data to pass through to microcontroller for gesture recognition
  - Can only be enabled on ONE glove at a time -- if both thumbs are pressing other fingers only the master's IMU data is recognized
  - Automatically disabled when thumb stops being pressed
  - There will be a slight delay in software to make sure a MIDI note in the two fingers isn't generated when a user is trying to enter gesture mode.
  - In gesture mode, each clock cycle will probe x-, y-, and z-axis translational acceleration and x- and y-axis rotation. This data will be sent to the processing unit, which will take 50 samples of each movement and decide which gesture to begin tracking by comparing the magnitude of each signal.
  - For hand rotation gestures, both the angular velocity and the linear acceleration are relevant in gesture decision. During a rotate left/right gesture, the x-axis acceleration will have a high magnitude, along with angular velocity about the y-axis. Similarly, the z-axis acceleration will be used together with the x-axis angular velocity to decide if a rotate up/down gesture is in effect.
  - Once a gesture decision has been made, the IMU will only be probed for the relevant data to that gesture until the thumb switch is disengaged.
- Calibration mode
  - Entered by holding all switches on both hands down for 3 seconds.
  - Defines a bounding box for recognizing gestures and determining scale.
  - For example, if the user in calibration mode when prompted to move left to calibrate X translation, moves 1 meter, then 1 meter will be the range of X movement needed to change control from 127 to 0.
  - Thumb to index finger calibrates Z translation, thumb to middle finger calibrates Y translation, thumb to ring finger calibrates X translation, and thumb to pinky finger calibrates rotation.

- ○ Second-order approximation on each time slice will be used on the range of movement divided by 128 to determine the "bucket size". This bucket size is the relative distance or angle a user's hand must move in order to increase or decrease a control value by 1.

5. MIDI Standard Overview

MIDI has been a long established standard in the world of electronically generated music, owing to its simplicity and versatility. MIDI works by sending messages about musical notes, pitch, velocity, and a wide variety of other parameters that are encoded as control signals. These messages are encoded as packets in a simple format. Notes are sent as three bytes -- the first byte determines which MIDI channel to use (we use exclusively Ch. 1), the second byte determines which note was generated, and the third byte encodes velocity (how hard the note was pressed). The other type of messages are called control bytes and they follow a similar format. Again the first byte determines channel, the second byte, called the control byte, determines which parameter is being modified, and the third byte specifies the value for the parameter.[9]

6. Gesture Table

Symbol	**Function**		**Range**	**Requirement**
	(L/R) Thumb +	**Function/Control Byte Value**		
Z Translation (move up/down)	L Index	Volume control/#7	[0.5g, 8g]	● Discretizes movement into 128 bins. ● From any given position a positive delta followed by the exact same delta opposite in sign should map to the same bin.
	L Middle	Modulation Wheel/#1		
	L Ring	Breath Controller/#2		
	L Pinky	Foot Controller/#4		
	R Index	Expression Controller/#11		
	R Middle	Effect Control #1/#12		
	R Ring	Effect Control		

		#2/#13		
	R Pinky	General Purpose Controller 1/#16		
Y Translation (move left/right)	**(L/R) Thumb +**	**Function/Control Byte Value**	[0.5g, 8g]	● Same requirements as Z Translation
	L Index	Balance/#8		
	L Middle	Pan/#10		
	L Ring	General Purpose Controller 2/#17		
	L Pinky	General Purpose Controller 3/#18		
	R Index	General Purpose Controller 4/#19		
	R Middle	General Purpose Controller 5/#80		
	R Ring	General Purpose Controller 6/#81		
	R Pinky	General Purpose Controller 7/#82		

X Translation (move in/out)		[0.5g, 8g]	● Same requirements as Z Translation

(L/R) Thumb +	Function/Control Byte Value
L Index	Granular volume control/#39
L Middle	Granular modulation wheel/#33
L Ring	Granular breath controller/#34
L Pinky	Granular foot controller/#36
R Index	Granular expression controller/#43
R Middle	Granular effect control #1/#44
R Ring	Granular effect control #2/#45
R Pinky	Granular general purpose controller 1/#48

X Rotation (roll left/right)		[10º/s, 90º/s]	● Same requirements as Z Translation

(L/R) Thumb +	Function/Control Byte Value
L Index	Sound

		Controller 1/#70		
	L Middle	Sound Controller 2/#71		
	L Ring	Sound Controller 3/#72		
	L Pinky	Sound Controller 4/#73		
	R Index	Sound Controller 5/#74		
	R Middle	Sound Controller 6/#75		
	R Ring	Sound Controller 7/#76		
	R Pinky	Sound Controller 8/#77		
Y Rotation (pitch up/down)	**(L/R) Thumb +**	**Function/Control Byte Value**	[10º/s, 90º/s]	● Same requirements as Z Translation
	L Index	Sound Controller 9/#78		
	L Middle	Sound Controller 10/#79		

L Ring	Effects 1 Depth/#91
L Pinky	Effects 2 Depth/#92
R Index	Effects 3 Depth/#93
R Middle	Effects 4 Depth/#94
R Ring	Effects 5 Depth/#95
R Pinky	Portamento Control/#84

Figure 8. Gesture Table

7. Simulations

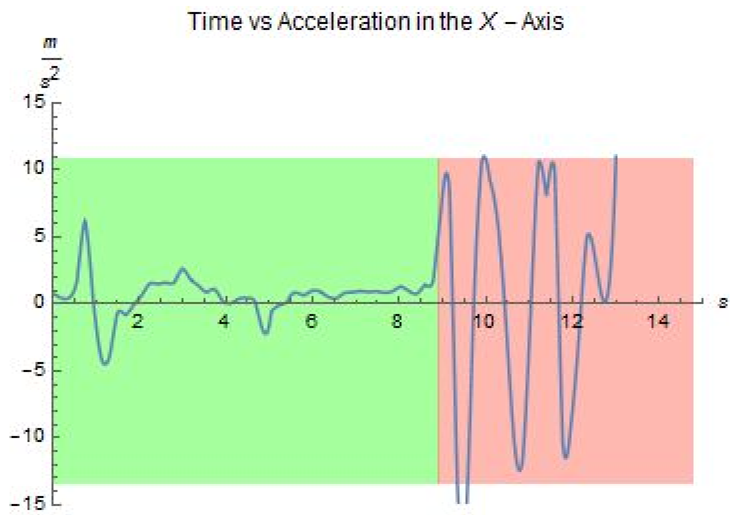

Figure 9. X-Axis Accelerometer Data

This is a demonstration of the accelerometer which will be very similar to the one we will use in our project. In the green region from 0 to about 9 seconds we moved the sensor at what we'd consider the lower bound for gesture recognition, and we recorded this as about +/- 1m/s^2. Our simulation verified our requirements for the lower bound for the accelerometer. In the red region we moved the sensor as quickly as possible and recorded about +/- 10m/s^2. Again, this is satisfactory for our requirements.

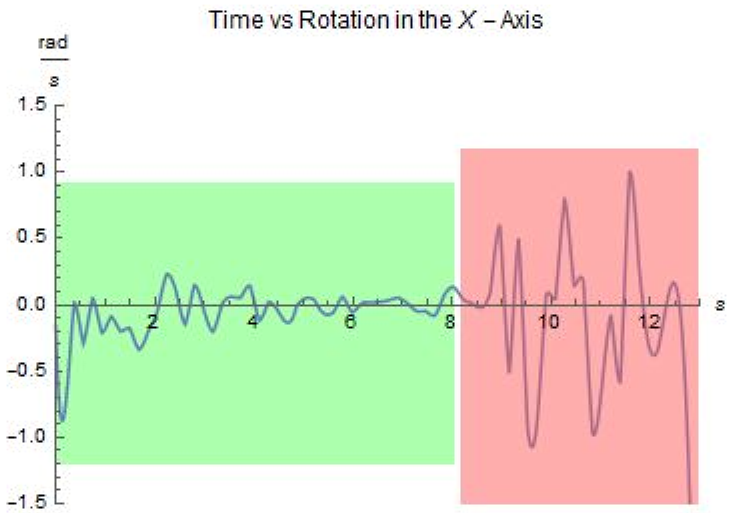

Figure 10. X-Axis Gyroscope Data

We also tested a gyroscope to simulate lower and upper bounds for rotational gestures. In the green region where we rotated the sensor very slowly we achieved a rotation of about 0.1rad/s = 6deg/s which agrees with our requirements. Similarly in the red region where we rotated the sensor very quickly we recorded rotation of about 1rad/s = 57deg/s in magnitude. Our requirements are much greater in range and so this simulation is a good reassurement that our requirements are well within scope.

These simulations are the result of an experiment to see how the data stream of IMU data might very well look. In that regard, the demonstration of using the X axis accelerometer and gyroscope data provides a substantial "proof of concept" of the kind of ranges we will be working with. We expect the vast majority of our user input to be in the middle of the ranges shown above, which makes our processing more reliable.

8. Risk Analysis

The part of our project we are most concerned about is the availability of MIDI controllers supporting MIDI over Bluetooth. It's a very new standard with scant adoption, but we are willing to take the risk of supporting it because it will vastly simplify our design. If we were not to use a Bluetooth transmitter on the glove to directly transmit MIDI, we would still have to transmit to a generic Bluetooth receiver which we'd have generate MIDI over USB. This decision is due to the glove being wireless is a key design requirement.

9. Tolerance Analysis

The sensor unit is the most critical component of our system. It is essential for capturing the user's gestures and it is the bottleneck for the performance of the recognition algorithm. The accuracy of the entire system depends on the accuracy of the sensor unit. If the IMU and mechanical switch failed to give data within a reasonable accuracy, the later pattern recognition would be impossible and thus defeat the purpose of the entire system.

There are two sensors within the sensor unit: the inertial measurement unit (IMU) and the mechanical switch unit. The accuracy of the sensor unit is a function of the accuracies of the IMU and mechanical switches.

The acceptable tolerance for the sensor unit should be within 10%, since this is the maximum error rate for the recognition algorithm to be able to function. If the accuracy for the sensor unit is less than 90%, then the recognition algorithm will not be able to distinguish different gestures, and thus greatly impair overall performance of the system and not meet the requirements set out.

The accuracy of the sensor unit can be calculated as a function of the accuracies of IMU and mechanical switches in EQ1.

$$Accuracy(Sensor\ Unit) = Accuracy(IMU) * Accuracy(Switches) \quad \text{EQ. 1}$$

We assume that since the switches can be modeled as a basic SPST circuit and furthermore, since the output for the switches is all-or-nothing/binary, we will assume that the accuracy of this component is essentially 100%. The only time the switches would not be 100% accurate would be they are mechanically broken, which we obviously cannot handle in our design. The IMU sensors, on the other hand, are prone to error -- especially in a noisy environment with many micro-vibrations (for instance, when user is riding a car). The measurements of the IMU sensors are still usable as long as its error is within 10% of our required operating bounds to meet the overall gesture recognition requirement. From the datasheet given for the IMU, the gyroscope has an error range of +/- 20deg/s and the accelerometer has an error range of +/-80 m*g* = 0.7848 m/s^2. [8]

If we assume the IMU error follows a normal distribution, then the probability that the sensor error is within 2 standard deviations is 95%. Since within 2 standard deviations the range for error is still within our required recognition value we have that the sensor will recognize gestures above the minimum tolerance as given.

Cost and Schedule

1. Cost

1.1. Labor

We assume that based on the average UIUC Computer Engineering graduate salary for the 2015 graduating class of $84,250 [7] and for an average workload of 10 hours per week for each partner that we will be looking at $421.25/week for a total of $3791.25 for the remaining 9 work weeks.

1.2. Parts

Part	Quantity	Unit Price
Adafruit Trinket 3.3V Microcontroller	1	$6.95
SparkFun MPU 6050 IMU	2	$39.95
SparkFun Mini Pushbutton Switch	20	$0.35
SparkFun Li-Ion Battery 1Ah	2	$9.95
SparkFun LiPo Charger Micro-USB	2	$7.95
SparkFun Voltage Regulator - 3.3V	2	$1.95
Mouser 871-B59219J130A20 Thermistor	2	$4.62
SparkFun BLE Mate 2	3	$29.95

1.3. Grand Total

Summing the values calculated in the labor and parts section we reach a grand total of $3791.25 + $144.77 = $**3936.02**

2. Schedule

Week of	Adam	Tyler	Both
February 27	• Create power submodule circuit schematic • Finalize the “tap” switch design • Make cost chart	• Revise block diagram • Hammer down requirements for sensor output resolution mapping • More specific physical design	• **Design Review** • **Requirements and Verification**
March 6	• Create Chart of all MIDI signals to be transmitted over Bluetooth	• Determine inputs and outputs of dynamic time warping module	• Complete Soldering Assignment • Order all needed parts
March 13	• Evaluate performance of two gloves working together and simultaneous receiving / sending with two different Bluetooth transmitters	• Entire physical layout/placement	• Simulation of all submodules • Determine needed glove materials
March 20	SPRING BREAK	SPRING BREAK	SPRING BREAK
March 27	• Test Bluetooth transmission of MIDI data	• Test sensor/button to microcontroller communication	• Work on Microcontrolle r specific layout
April 3	• Build Project	• Build Project	• Build Project
April 10	• Build Project	• Build Project	• Build Project
April 17	• Test and Debug completed Power and Transmission	• Test and Debug completed Sensor and Control	• Mock Demo • Sign Up for Demonstration

	modules	modules	• Sign Up for Mock Presentation • Sign Up for Presentation
April 24	• Test and Debug completed Power and Transmission modules	• Test and Debug completed Sensor and Control modules	• Demonstration • Mock Presentation
May 1			• Final Papers • Lab Notebooks • Lab Checkout

Safety and Ethics

Our design and product will comply with the IEEE Code of Ethics. The following two points from the Code are especially relevant for our design:

1. "*to accept responsibility in making decisions consistent with the safety, health, and welfare of the public, and to disclose promptly factors that might endanger the public or the environment*" [3]
   We considered the safety and health risks of our design and have provided a list of precautions to take when using our hardware. We also designed our hardware to reduce any potential risks to the users.
2. "*seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others*;" [3]
   We will properly cite the contributions from previous research and similar products.

The hardware operates at a low voltage, so there is no prominent risk of electric shock from normal use. However, the sensor unit contains fragile components that can be broken when the hardware is not used properly. For example, the mechanical switch can be damaged when too much force is applied, and the IMU can be damaged if the hardware is dropped to the ground from high places. Also, since we do not need any validation data to generate a library of calibration data, human subjects will not be needed, thus data integrity will not be an issue.

If the Lithium-ion battery is not charged properly, it can explode. During the build process, we will use a thermometer to monitor the temperature of the power subsystem to make sure the cell temperature does not reach unsafe levels. To make sure the battery does not get overcharged, all of the charging circuitry will be tested thoroughly so the maximum threshold is not exceeded.

A positive thermal coefficient (PTC) thermistor will be used as a resettable device that creates high resistance when there is an excess current and it goes back to its normal state when the current/temperature levels are back to normal[6].

We have signed the following battery safety sheet:
https://courses.engr.illinois.edu/ece445/documents/GeneralBatterySafety.pdf

List of precautions to take when using the wearable hardware:

- Avoid using the hardware with wet hands or near water.
- Avoid dropping or squeezing the hardware.
- Use the hardware within the safe temperature range: 0°C to 45°C (32°F to 113°F).
- Be cautious for the potential electrostatic discharge.
- Before using, check that there is enough space around you. Also, make sure the hardware cannot slip out of your hand. If the hardware hits a person or object, this may cause accidental injury or damage.
- Be cautious of the hardware overheating. Remove the hardware immediately if it feels too hot.

References

1. Shah, Utsav, Abhishek Kannekanti, Pinaj Basutkar, Yebai Zhao, and Paula Garcia. "Air Beats: A System for Eyes Free Music Composition." Rochester Institute of Technology, 15 June 2016. Web. 13 Oct. 2016.
2. Xkey Technical Specifications, 2016. Web. 27 Nov. 2016
3. "Code of Ethics IEEE", Institute of Electrical and Electronics Engineers, Inc. (2006). http://www.ieee.org/. Retrieved Web. 20 Oc. 2016 from the website temoa : Open Educational Resources (OER) Portal at http://www.temoa.info/node/23284
4. Adafruit Li-Ion Battery 3.7V 2200mAh Datasheet. Available: https://cdn-shop.adafruit.com/product-files/1781/C2253_-_ICR18650_2200mAh_3.7V_with_PCM_20140728_APPROVED_8.18.pdf. Accessed: Feb. 9, 2017
5. "What is a strain gage?," 2003. [Online]. Available: http://www.omega.com/prodinfo/straingages.html. Accessed: Feb. 9, 2017.
6. I. Buchmann, "Battery safeguards; protection circuits – battery university," 2016. [Online]. Available: http://batteryuniversity.com/learn/article/safety_circuits_for_modern_batteries. Accessed: Feb. 20, 2017.
7. "Salary averages: ECE ILLINOIS,". [Online]. Available: https://www.ece.illinois.edu/admissions/why-ece/salary-averages.asp. Accessed: Feb. 24, 2017.
8. http://43zrtwysvxb2gf29r5o0athu.wpengine.netdna-cdn.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf
9. The Complete MIDI 1.0 Detailed Specification. 1st ed. 1996. Print.