Wearable Devices for Software Instruments

Team 17 — John Born, Kuang Wang, and Miguel Torres
ECE 445 Project Proposal — Fall 2019
TA: Kristina Miller

1 Introduction

1.1 Objective

Choreographed movement is known to enhance the expressivity of a musical piece. This is most evident in dance performances but can also be seen in musical performances spanning genres from pop to experimental. It is customary for music and choreography to be completed separately. In the case of a pop performance, choreography enhances the music. Contrastingly, dance performances use music to highlight choreography. Even though movement contributes to the experience of a musical performance, the dancer has no influence in the composition process of the music. Contributing factors to this separation are:

1) the performer must interact with a physical instrument during the performance, thereby limiting the performers range of motion.

2) the playability of typical instruments requires trained dexterity and coordination from its performer (e.g. the flute requires remembering note fingerings).

With the advancement of computing technology, creating music has become more accessible to non-musicians via electronic instruments and computers. However, interacting with these newer instruments still requires the capability of pushing buttons, twisting knobs, and moving sliders during a performance. Unfortunately, this continues to exclude dancers from being active participants in the composition process.

In order to include dancers in the process of composing music, this project aims to make a set of wearable wrist and ankle devices that will use the performer's movement to control a software instrument. The orientation data of these devices will be used to control sonic characteristics of the software instruments such as pitch, volume, modulation, filter cutoffs, and panning. The orientation data will be transmitted wirelessly through Wi-Fi using the Open Sound Control (OSC) protocol (a URL style transmission) to a laptop.

1.2 Background

Both musical and dance performances are increasingly incorporating various forms of technology to captivate their audience through the use of projections, custom lighting, and lasers. These productions involve dancers performing to pre-arranged multimedia, which inadvertently excludes the dancers from the compositional aspect of said productions. Regarding music, the direction of inspiration between composer and choreographer tends to be one way, where the composition influences the choreography. Attempts to make this relationship more bidirectional have been explored as early as 1965 with *Variations V*, a collaboration between John Cage and Merce Cunningham which debuted at the New York Philharmonic's French-American Festival. This performance involved twelve antennas and photocells set throughout the stage and used to sense the proximity of seven dancers in order to trigger sound [1]. Experiments such as these change the dynamic between composition and choreography to be more interactive. The term Interactive Dance is now used to refer to such productions where the dancer, through the use of technology, is able to influence the musical composition [2]. Through this project, we seek to continue the evolution of such technology while making it more accessible to non-musicians.

1.3 High-level Requirements

To successfully accomplish this project, the following requirements must be met:

1) The virtual instrument output sound should be continuous without any jumps in pitch or volume, for instance. The orientation data should have minimized any spikes and discontinuities such that they do not detract from musical performance.
2) At least four musical characteristics of the virtual instrument must be controllable. These will include pitch, filter cutoff frequencies, volume, modulator frequencies, and potentially more characteristics.
3) Latency between the controller and virtual instrument will be limited to 10 ms [3] so the controller acts as an instrument.

2 Design

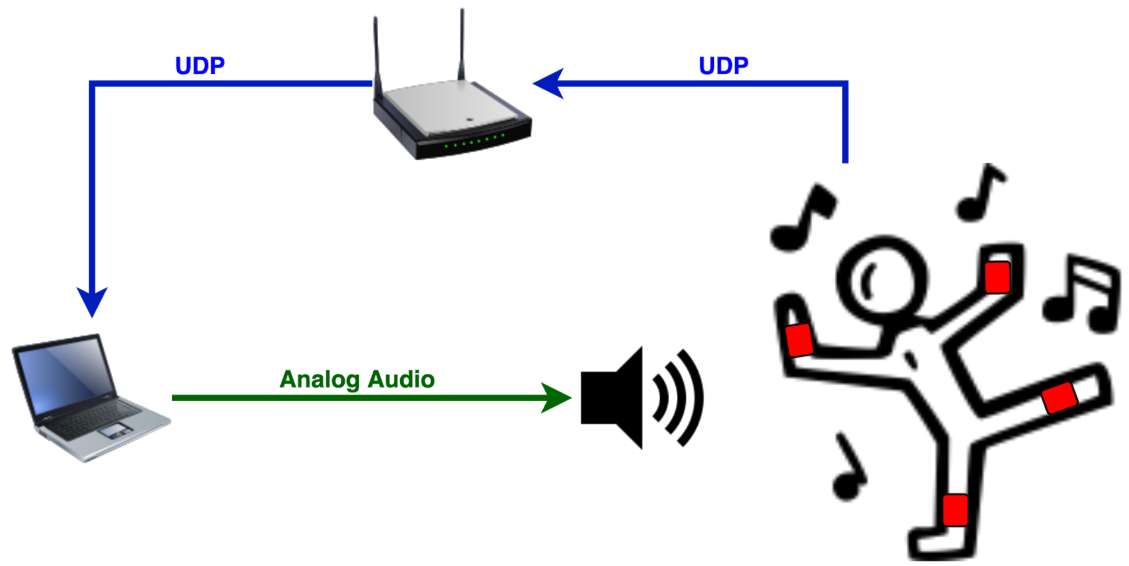

Figure 1. Macroscopic view of the whole system.

Figure 1 depicts the closed loop system of music creation facilitated by these wearable devices where the dancer is now a participant. The music will influence the dancer's movements and, in turn, the dancer's movements will influence the music.

2.1 Block Diagram

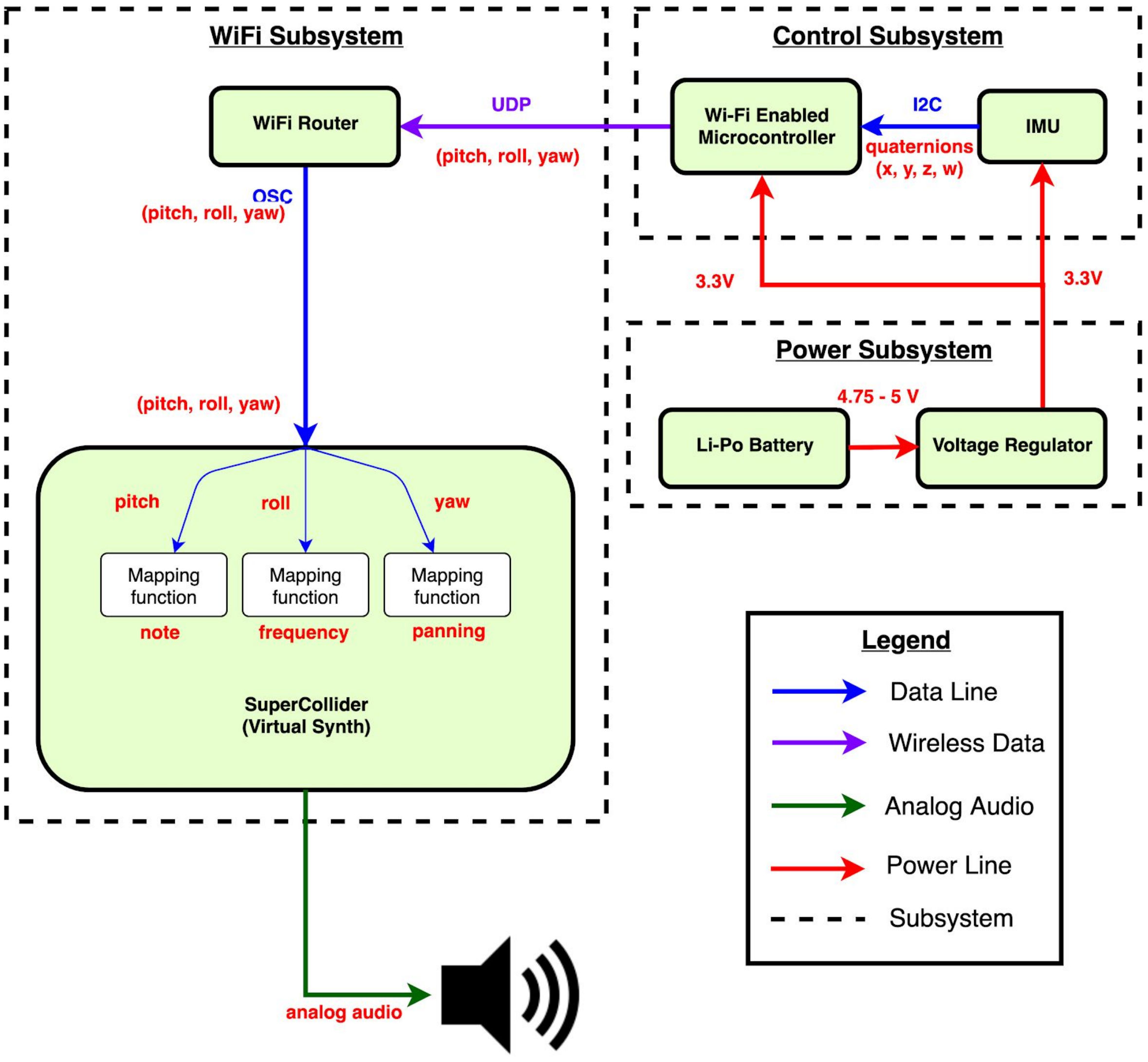

Figure 2. Block Diagram of the whole system.

The wearable devices require three subsystems to function properly: Power, Sensing, and Wi-Fi subsystems. The Power Subsystem will be comprised of a LiPo battery that will supply the necessary 3.3 V to the IMU and the Wi-Fi microcontroller and a voltage regulator to ensure there are no voltage fluctuations. The Sensing Subsystem, consisting of a 9 DoF IMU, will provide pitch, yaw, and roll data via I^2C to the Wi-Fi microcontroller. The Wi-Fi Subsystem will link the wearable device to a computer via a local Wi-Fi network adhering to the IEEE 802.11b/g/n standards [4]. The sensor data will be packaged for transmission by the MCU using the OSC protocol.

External devices receiving the transmitted data consist of a wireless router and a computer. The wireless router will facilitate the wireless network, connecting each wearable device to the computer. The computer will receive the transmissions from the network and process the data in two steps: 1) identifying the device of origin of the data, and 2) mapping the data to synthesizer parameters. The program on SuperCollider will identify each device using their respective IP addresses and will forward the received data to the synthesizer. The Supercollider IDE will host the customized software synthesizer to produce the audio. Optional external devices will include speakers to amplify the computer's audio output.

2.2 Physical Design

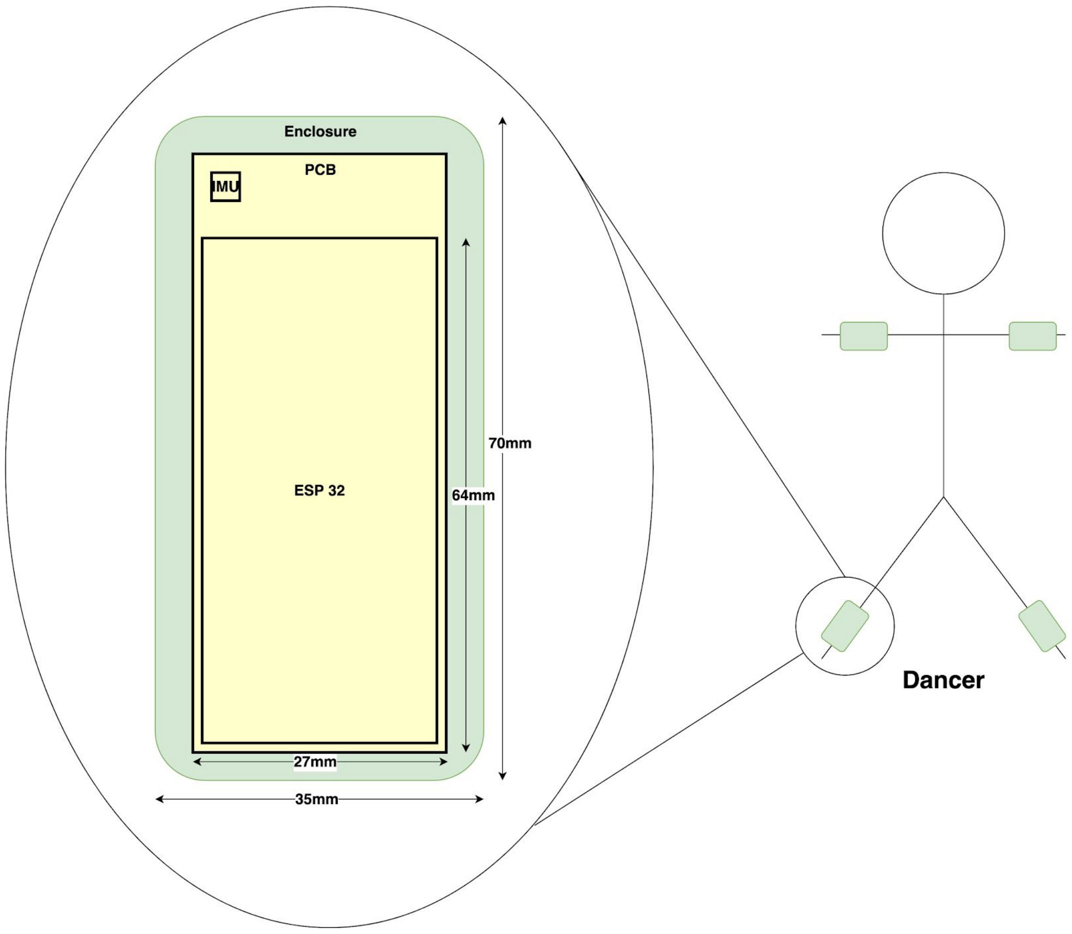

Figure 3. This is a prototype model of the wearable devices

The physical design shown is an approximation based on the dimensions for components currently in the design. The ESP 32 will be the main Wi-Fi controller mounted on a breakout board. The IMU sensor and microcontroller that communicates with the Wi-Fi controller will be mounted on a PCB connected to the Wi-Fi controller using wires. These boards will be attached to wrists and ankles using adjustable bands made of either fabric or elastic. Since these devices will need to be comfortable yet secure, they must be enclosed in a box with soft edges and a backing made of foam or some other soft material. Initially, one of these devices will be built as a proof of concept. Once a working prototype is finished, we expect to duplicate the work and produce at most 4 devices.

The devices will be worn on the wrists and ankles. To ensure that the enclosure fits comfortably and is easy to attach, we will be looking into nylon straps with velcro ends.

2.3 Power Subsystem

The power subsystem consists of a Li-Po battery and voltage regulator. The Li-Po battery will power the voltage regulator which will output a regulated 3.3V for all circuit components.

2.3.1 Li-Po Battery

A lightweight Lithium-Polymer (Li-Po) battery will be used to power the components of the wearable device. The battery should be able to power the entire device for at least 2 hours since that is the approximate duration of a typical concert. However, since the device is wearable, weight should be minimized. The ESP32 Microcontroller has a typical power consumption of 190mA at 3.3V using the 802.11g Wi-Fi standard [4]. The MPU-9250 IMU has a typical power consumption of 3.7mA at 3.3V in normal operating mode [5]. Allotting for about 50mA of current for other circuit component, we estimate about 250mA for 2 hours is needed.

2.3.2 Voltage Regulator

The voltage regulator will convert the slightly variable Li-Po battery voltage to a regulated 3.3V in order to supply power to the MPU-9250 and ESP32 which operate within ranges of 2.4-3.6V and 2.3-3.6V respectively [4][5]. It must be able to handle the range of input battery voltages from 4.75-5V at an estimated current draw of 250mA [6]. The voltage regulator being used is the LM1117 3.3V regulator which operates between 0-125°C [6].

Requirement	Verification
1) Supply regulated 3.3V output from 4.75-5V battery voltage inputs.	1) a) Connect voltage regulator circuit to a power supply of 4.75V b) Measure output voltage of regulator for 1 minute and ensure it is 3.3V +/- 5%. c) Repeat for power supply of 5V.
2) Voltage regulator should remain below 43 °C, the maximum threshold for safe temperature of wearable devices. [13] Our devices are supposed to have operating temperature around 30 °C).	2) a) Connect voltage regulator circuit to a power supply of 5V (nominal battery voltage). b) Measure regulator temperature using an IR thermometer ensuring it stays below 43 °C.

2.4 Control Subsystem

The control subsystem consists of the IMU and Wi-Fi-enabled microcontroller. This is the device that takes in the user's orientation data and sends it to the virtual synthesizer.

2.4.1 IMU

The IMU being used is the MPU-9250 which is a 9-Axis Gyroscope, Accelerometer, and Magnetometer chip. It collects inertial data (angular velocity, angular acceleration, and geomagnetic field strength) from each subsensor and compiles these into orientation data output as time-stamped quaternions. The output quaternions are computed using a proprietary on-chip sensor fusion algorithm called MotionFusion. This orientation output will be used to control the virtual synthesizer.

Requirement	Verification
1) IMU should detects correct orientation within +/- 10 degrees after a 180 degree flip.	1) a) Setup the IMU to display orientation on a computer monitor. b) Record the orientation with the unit laying flat on a table. c) Flip the unit and ensure the orientation is flipped by 180 +/- 10 degrees.
2) IMU limits noise and drift to +/- 1 degree while not moving.	2) a) Setup the IMU to display orientation on a computer monitor. b) Hang the IMU from a string in a still environment. c) Ensure the orientation stays within a +/- 1 degree range.

2.4.2 Wi-Fi Enabled Microcontroller

The microcontroller being used will be the ESP32. This is a Wi-Fi capable MCU for use with IoT devices. This microcontroller will be used to take in data from the IMU sensor, package the data for transmission according to OSC protocol requirements, and transmit that data with its onboard antenna. The ESP32 complies with the IEEE 802.11b/g/n standards allowing us to theoretically transmit data at 150 Mbps [4]. We will use I^2C to communicate between microcontroller and IMU.

Requirement	Verification
1) Is able to apply smoothing algorithms and maintain total latency under 10ms [3].	1) a) Load microcontroller code to output both smoothed and unsmoothed data. b) Plot 1 second of smoothed data and unsmoothed data to ensure it smooths properly. c) Load microcontroller code to output smoothed and unsmoothed data separately. d) Calculate orientation packet frequency for both cases. Ensure smoothed packet is no less than 50% slower than unsmoothed packet frequency.
2) Can package pitch, yaw, and roll data according to OSC protocol standards.	2) a) Load code on ESP32 that sends orientation data from IMU to a serial monitor according to the OSC protocol. b) Read the output data from the monitor and ensure all three orientations of yaw, pitch, and roll are output as 32-bit float values.

2.5 Wi-Fi Subsystem

2.5.1 Wi-Fi Router

A consumer grade wireless router will be used to facilitate a local wireless network for communication between the wearable devices and the computer. The router is capable of transmitting data at rates closest to the theoretical limit of 150 Mbps and ensure the transfer rate is greater than 100 +/- 10 Mbps.

2.5.2 SuperCollider (Virtual Synth)

This is an audio synthesis IDE that uses class-based, object oriented programming to facilitate algorithmic processing of audio. We will use this IDE to custom build a software synthesizer whose sonic characteristics will be controlled by IMU data. The software will allow us to directly map pitch, yaw, and roll information received from each device to the software synthesizer’s note frequency, volume, modulation, filter cutoffs, panning, and other possible controls. Currently, it is difficult to tell what mapping will work best without experimenting, but we will aim to map to the controls mentioned above. An example of how we anticipate the mapping to work is depicted in Figure 4.

Requirement	Verification
1) Synthesizer object responds to external inputs to manipulate parameters.	1) Setup synthesizer to respond to laptop's trackpad to adjust note with x-axis and volume with y-axis, such that moving the mouse pointer right will increase the pitch, moving left will decrease the pitch, moving up will increase the volume and moving down will decrease the volume.
2) Parameter mapping ranges make perceivable sonic adjustments as dictated by movement. Assign each parameter to its required range as follows: a) Note range spanning at least 2 octaves b) Low-Pass Filter cutoff frequencies between 50 Hz and 15 kHz c) Amplitude ranges from value of 0 to 0.6 d) Modulator frequencies from 0.01 Hz to 220 Hz e) Panning from left to right	2) a. Open level meters, scope and frequency scope. b. Create a synth object to respond to laptop's trackpad. For each parameter, map the value to the x-axis and y-axis of the mouse pointer as follows: ○ Set x-axis to respond to note range. ○ Set y-axis to respond to cutoff frequency c. When the mouse pointer has reached the far left side of the screen, the note pitch should be 2 octaves higher. d. When the mouse pointer reaches the top of the screen, all low frequencies should no longer be heard. e. Repeat steps a. through d. for the remainder of the parameters. ○ For amplitude, watch the level meter to see the maximum amplitude reached. ○ For modulator frequencies, use the frequency scope to see that multiple side-bands appear ○ For the panning, listen for the sound to pan from left to right.
3) Synthesizer receives and responds to incoming OSC data.	3) Print input messages to the SuperCollider built in terminal monitor called the Post Window.

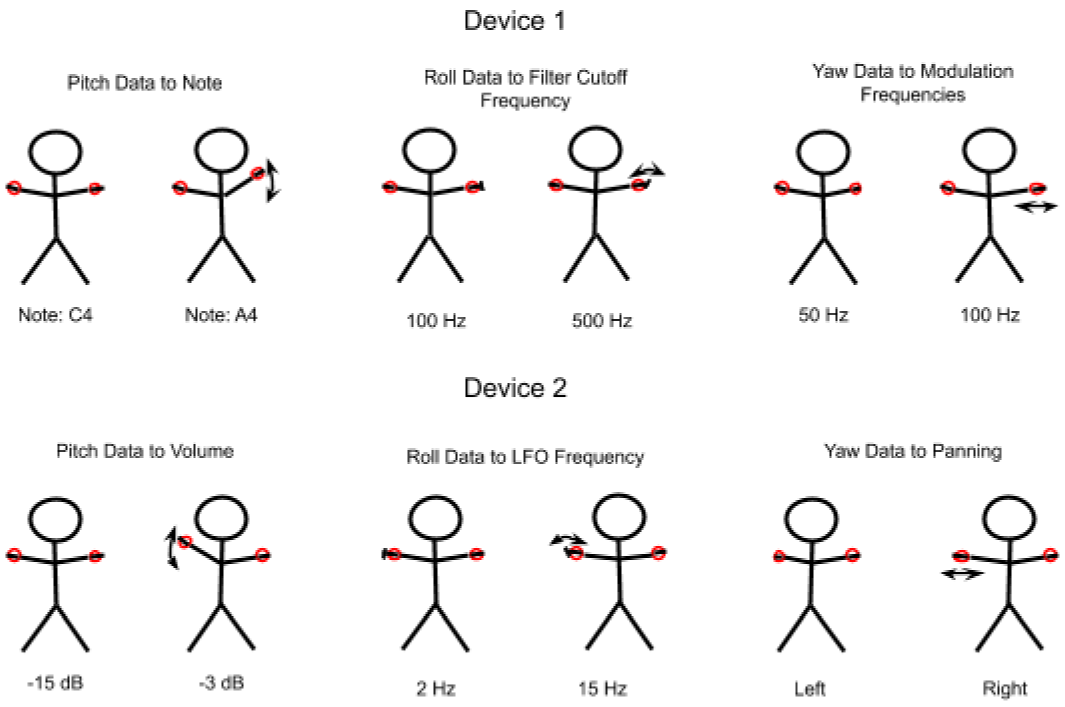

Figure 4. One example of the possible mapping of IMU data to synthesizer parameters

Possible mapping functions we will implement in SuperCollider include pitch data to note, roll data to filter cutoff, and yaw data to frequencies. Initially we will apply linear mapping for each of the functions (similar to what we did for range testing). We will linearly map pitch with range [-180 degree, 180 degree] to note with range [C4, C6], linearly map roll with range [-180 degree, 180 degree] to filter cutoff frequency with range [50 Hz, 15 kHz], and linearly map yaw with range [-180 degree, 180 degree] to modulation frequencies with range [0.01 Hz, 220 Hz].

After we are able to test the mapping functions with actual orientation data, we may adjust the method of mapping, targeted parameters and range of parameters.

2.6 Circuit Schematics

In order to expediently get a working proof of concept, we will start our project with breakout boards designed specifically for the IMU and MCU we have chosen. Once we have a working prototype we will use the breakout board schematics as a template to finalize our design. The following breakout board schematics for the used for the prototype were developed by Sparkfun under the Creative Commons Attribution Share-Alike 4.0 License.

Sparkfun ESP32 Thing schematics [7].

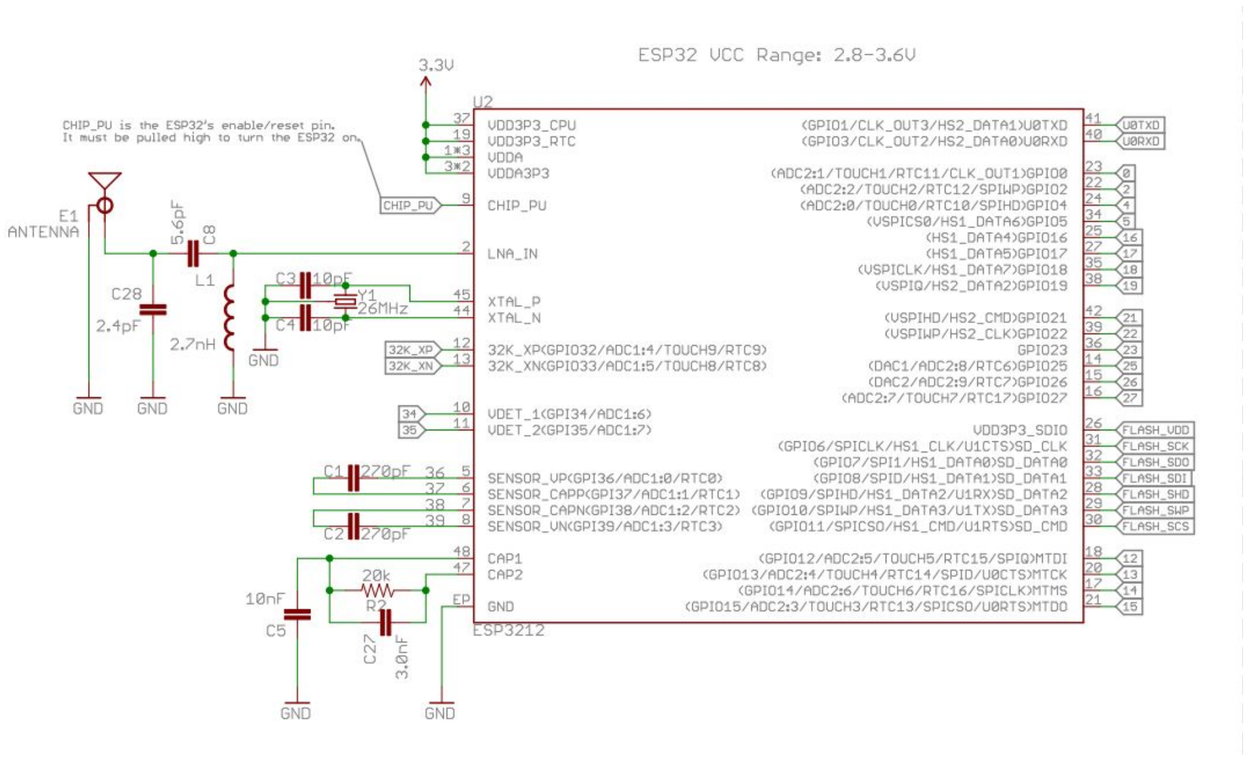
ESP32 VCC Range: 2.8-3.6V
3.3V
U2
CHIP_PU is the ESP32's enable/reset pin.
It must be pulled high to turn the ESP32 on.
E1
ANTENNA
5.6pF
C8
L1
2.7nH
C28
2.4pF
C3 10pF
C4 10pF
Y1
26MHz
VDD3P3_CPU
VDD3P3_RTC
VDDA
VDDA3P3
CHIP_PU
LNA_IN
XTAL_P
XTAL_N
32K_XP(GPIO32/ADC1:4/TOUCH9/RTC9)
32K_XN(GPIO33/ADC1:5/TOUCH8/RTC8)
VDET_1(GPI34/ADC1:6)
VDET_2(GPI35/ADC1:7)
SENSOR_VP(GPI36/ADC1:0/RTC0)
SENSOR_CAPP(GPI37/ADC1:1/RTC1)
SENSOR_CAPN(GPI38/ADC1:2/RTC2)
SENSOR_VN(GPI39/ADC1:3/RTC3)
CAP1
CAP2
GND
C1 270pF
C2 270pF
10nF
C5
20k
R2
C27
3.0nF
ESP3212
GND
(GPIO1/CLK_OUT3/HS2_DATA1)U0TXD
(GPIO3/CLK_OUT2/HS2_DATA0)U0RXD
(ADC2:1/TOUCH1/RTC11/CLK_OUT1)GPIO0
(ADC2:2/TOUCH2/RTC12/SPIWP)GPIO2
(ADC2:0/TOUCH0/RTC10/SPIHD)GPIO4
(VSPICS0/HS1_DATA6)GPIO5
(HS1_DATA4)GPIO16
(HS1_DATA5)GPIO17
(VSPICLK/HS1_DATA7)GPIO18
(VSPIQ/HS2_DATA2)GPIO19
(VSPIHD/HS2_CMD)GPIO21
(VSPIWP/HS2_CLK)GPIO22
GPIO23
(DAC1/ADC2:8/RTC6)GPIO25
(DAC2/ADC2:9/RTC7)GPIO26
(ADC2:7/TOUCH7/RTC17)GPIO27
VDD3P3_SDIO
(GPIO6/SPICLK/HS1_CLK/U1CTS)SD_CLK
(GPIO7/SPI1/HS1_DATA0)SD_DATA0
(GPIO8/SPID/HS1_DATA1)SD_DATA1
(GPIO9/SPIHD/HS1_DATA2/U1RX)SD_DATA2
(GPIO10/SPIWP/HS1_DATA3/U1TX)SD_DATA3
(GPIO11/SPICSO/HS1_CMD/U1RTS)SD_CMD
(GPIO12/ADC2:5/TOUCH5/RTC15/SPIQ)MTDI
(GPIO13/ADC2:4/TOUCH4/RTC14/SPID/U0CTS)MTCK
(GPIO14/ADC2:6/TOUCH6/RTC16/SPICLK)MTMS
(GPIO15/ADC2:3/TOUCH3/RTC13/SPICSO/U0RTS)MTDO
U0TXD
U0RXD
FLASH_VDD
FLASH_SCK
FLASH_SDO
FLASH_SDI
FLASH_SHD
FLASH_SWP
FLASH_SCS

Figure 5. The ESP32 Microcontroller Unit

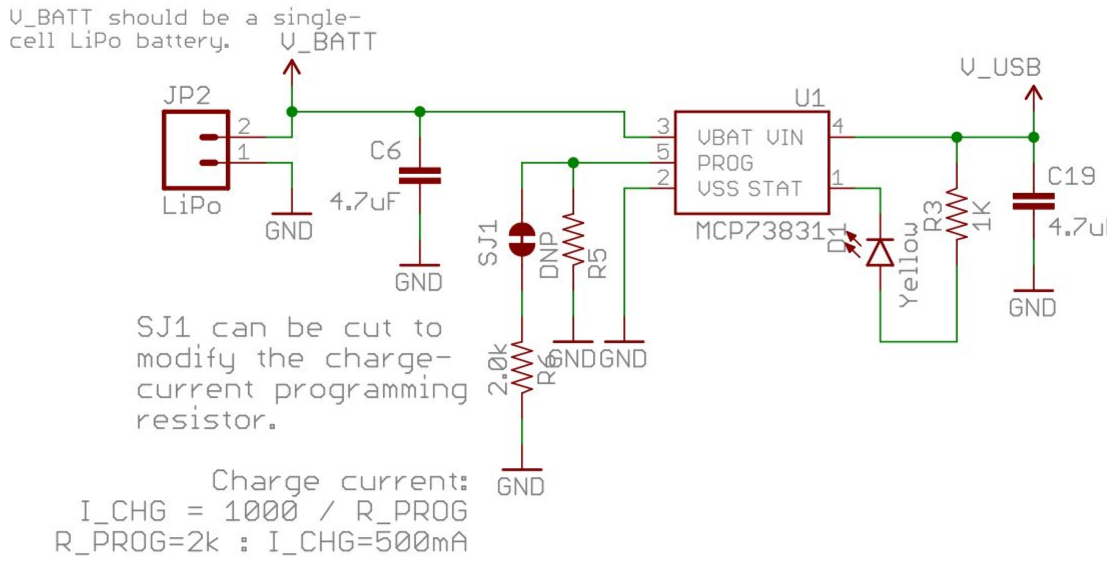

Figure 6. Lithium-Polymer Battery Charger (1-cell)

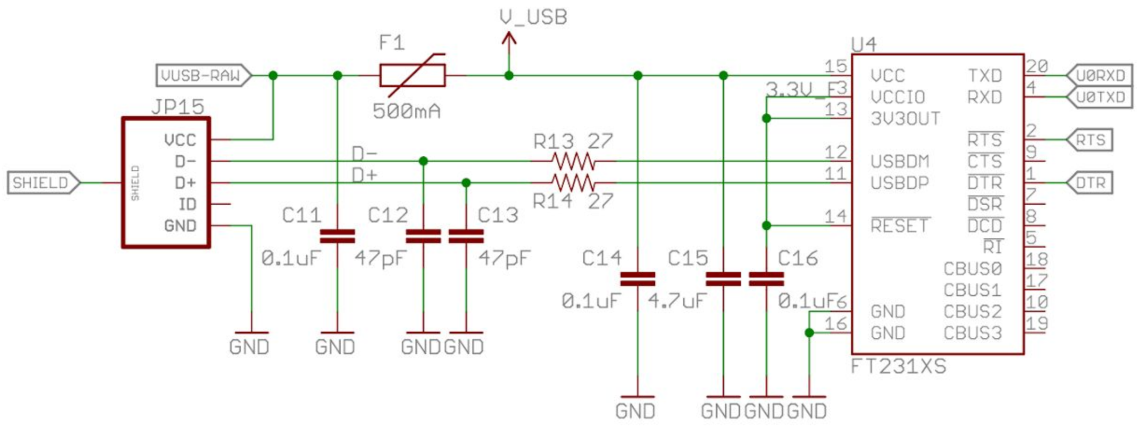

Figure 7. USB-to-Serial Converter

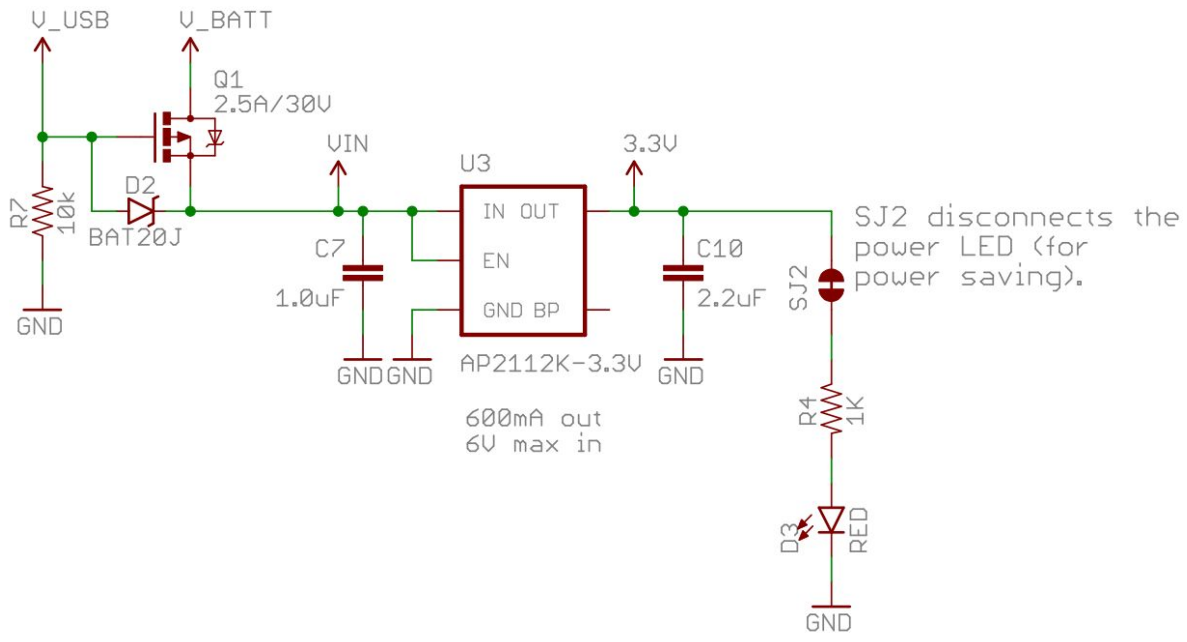

Figure 8. Voltage Regulator and Battery Charger

Sparkfun IMU Breakout - MPU-9250 [8].

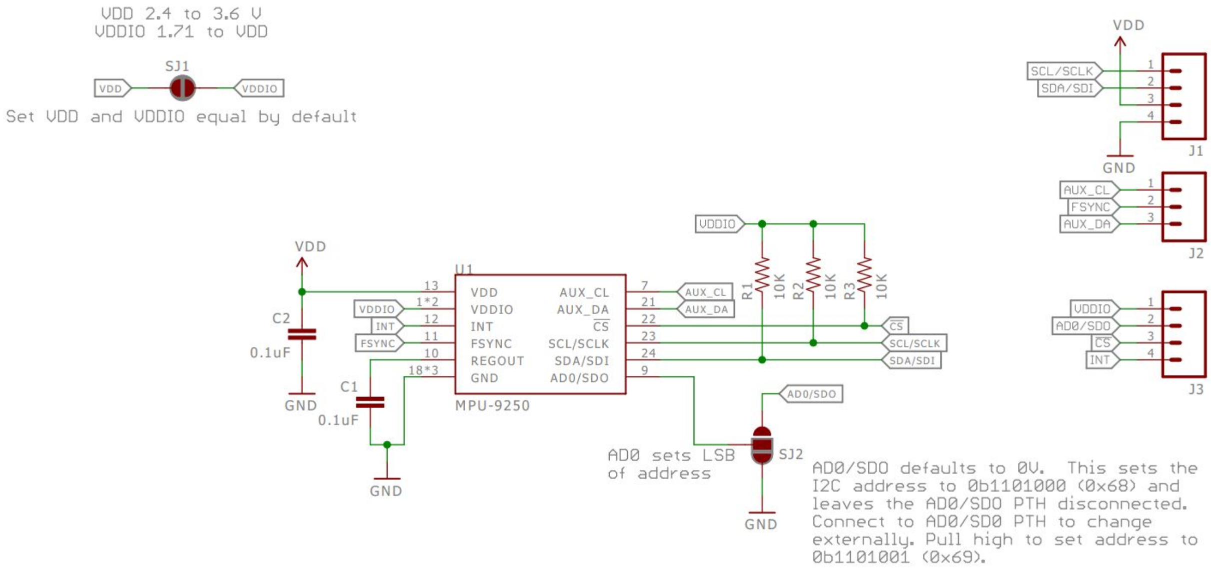
VDD 2.4 to 3.6 V
VDDIO 1.71 to VDD
SJ1
Set VDD and VDDIO equal by default
VDD
U1
MPU-9250
VDD
VDDIO
INT
FSYNC
REGOUT
GND
AUX_CL
AUX_DA
CS
SCL/SCLK
SDA/SDI
AD0/SDO
C2
0.1uF
C1
0.1uF
R1 10K
R2 10K
R3 10K
AD0 sets LSB of address
SJ2
AD0/SDO defaults to 0V. This sets the I2C address to 0b1101000 (0x68) and leaves the AD0/SDO PTH disconnected. Connect to AD0/SD0 PTH to change externally. Pull high to set address to 0b1101001 (0x69).
J1
J2
J3

Figure 9. MPU-9250 IMU

2.8 Tolerance Analysis

The most critical aspect of our design is the latency between physical movement and the resulting change in sonic characteristics of the software synthesizer. According to the MIDI Manufacturers Association, delays of less than 10 ms are practically imperceptible [3]. Once this threshold is exceeded, the user experience will be affected negatively due to the noticeable delay between movement and sonic change. Our goal is to maintain the amount of latency in communication below this 10 ms threshold. Therefore, our focus will be on the performance of the MCU as this component will need to 1) receive the IMU data, 2) smooth out any jarring spikes or discontinuities in the data, 3) and transmit the data over Wi-Fi.

Analyzing the data transfer of the MPU-9250, we find that the IMU sends between 29 and 39 bits per package of single-data streams through I^2C. Transmissions to and from the IMU's registers operate in fast mode, at 400 kHz [5]. Fast mode commonly transmits at a rate of 400 kbps, allowing for expedient transfer of data [9]. Taking how many bits are in a single-data package into consideration along with the speed of the I^2C bus, we can estimate how much time it will take for one package to be transmitted.

$$\frac{39}{400}\text{ bits/kbps } = 0.0975\text{ ms (with 1 data packet)}$$
$$\frac{47}{400}\text{ bits/kbps } = 0.1175\text{ ms (with 2 data packets)}$$

Furthermore, multi-data streams—meaning an additional 9 bits per data packet— will only increase the delay by about 0.020 ms.

$$\text{The increase of delay = } 0.1175\text{ ms - } 0.0975\text{ ms } \approx 0.020\text{ ms}$$

From these calculations we can determine that delays caused by I^2C transmissions between the IMU and MCU are negligible.

Moving on to smoothing algorithms, the best candidate for noise reduction is the Kalman Filter. This is a predictive, recursive algorithm that uses measurements and estimations in a feedback loop to increase the accuracy of a sensor. The basis of this algorithm is the calculation of a Kalman Gain (KG): which is used to calculate estimates and errors based on previous calculations, the current estimate (EST_t): which relies on the deviation of the previous estimate from the current measurement weighted by KG, and the error in the estimate (E_{EST}). These parameters are calculated as such:

$$KG = \frac{E_{EST}}{E_{EST}+E_{MEA}}$$

$$EST_t = EST_{t-1} + KG\left[MEA - EST_{t-1}\right]$$

$$E_{EST,\ t} = [1 - KG]\, E_{EST,\, t-1}$$

Where MEA is the measurement, E_{MEA} is the error in the measurement, and EST_{t-1} represents previous estimates [10]. The calculation of the Kalman Gain allows for this algorithm to minimize the amount of iterations it takes to produce an accurate estimate of a measurement. Considering that the ESP32 is rated at 600 DMIPS, which comes to about 2 ns per instruction, we do not foresee much gain in latency due to computation load [4]. Additionally, the MPU-9250 features a proprietary algorithm called MotionFusion that is implemented in their Digital Motion Processor (DMP) [11]. We anticipate this algorithm to help reduce the amount of noise in the sensor readings before sending the data off, thereby decreasing the amount of processing conducted by the MCU.

Lastly, the ESP32's Wi-Fi radio supports IEEE 802.11b and 802.11g data rates of up to 150 Mbps [4]. This indicates that wireless transmissions should pose no meaningful contributions to signal delays.

3 Cost and Schedule

3.1 Cost Analysis

We will calculate the cost of labor based on a $40 per hour rate using the following project phase to estimated hours. As seen below, this brings the today labor cost to 230hrs * $40/hr = $9200.

Project Phase	**Estimated Hours**
Design and Preparation	60
Ordering and Initial Prototyping	20
PCB Design	20
Enclosure CAD Design	10
Finalize PCB and CAD Designs	15
Software Synth Coding	40
Testing and Debugging	40
Poster and Presentation Preparation	15
Final Design Documents	10
TOTAL	230

The following part costs are estimated based on the devices we have chosen to use. The current estimated parts cost for 1 device is $137.83, so for 4 devices (one for each limb) the cost would be $551.32.

Part	**Part #**	**Cost**
Li-Po Charger (Micro-USB)	PRT-10217	$8.95
Li-Po Battery - 850mAh	PRT-13854	$4.95
Voltage Regulator - 3.3V	LM1117-3.3V	$1.14
MPU-9250 Breakout	SEN-13762	$14.95
ESP32 Thing Breakout	PID-13907	$22.95
PCB	(Estimated)	$15
Jumper Wire Kit (140 Pieces)	PRT-00124	$5.95
Wi-Fi Router	Linksys - AC1000	$39.99
Enclosure	(Estimated)	$15
Wrist / Ankle Bands	Griffin Sport Case for iPod	$8.95
TOTAL		$137.83

This makes the estimated total cost of the project $9751.32 for a set of four devices.

3.2 Schedule

Week	**Task**
9/29	1) Estimate cost of parts and labor - JB 2) Order breakout boards for testing - JB 3) Create first draft of SuperCollider synth - MT, KW 4) Order Router - MT
10/6	1) Interface IMU with MCU to monitor orientation data -All 2) Analyze IMU data and implement smoothing filter - All
10/13	1) Implement conversion from quaternion to EulerAngles - KW 2) Test MCU Wi-Fi communication by sending IMU data to SuperCollider -All 3) Start working on PCB design - MT, JB
10/20	1) Research wrist/ankle band materials - MT, KW 2) Order first draft of PCB - JB
10/27	1) Design 3D CAD of enclosure - JB 2) Order Wrist/Ankle bands - MT 3) Mapping IMU data to music characteristics on SuperCollider synth - MT, KW
11/3	1) 3D print enclosure (first draft) - JB 2) Order final draft of PCB - MT 3) Adjust music characteristics mapping - KW
11/10	1) 3D print enclosure (final draft) - JB 2) Construct the device with enclosure - All 3) Prepare for the music performance - MT
11/17	1) Testing and debugging - All
11/24	Fall Break
12/1	1) Final testing for demo - All 2) Prepare for presentation - All 3) Start working on posters - All
12/8	1) Print posters - JB 2) Final paper due - All

4 Ethics and Safety

4.1 Ethics

Our design is such that the project complies with all of the Ethics requirements mentioned in the IEEE Code of Ethics [12]. There are several aspects we are primarily focusing on for the project. The IEEE Code of Ethics #5 states, “to improve the understanding by individuals and society of the capabilities and societal implications of conventional and emerging technologies, including intelligent systems” [12]. We believe our project satisfied this requirement because our project goal is to develop wearable devices for software instruments, which let not only dancers but also people without music composition experience to enjoy and compose music easier. We hope that with a combination of hardware and software technology, we can provide a better way for individuals to interact with music composition. The IEEE Code of Ethics #7 and #10 mentioned about how we accept criticism from others and assist co-workers in their professional development [12]. Just like how the ECE455 course designed, we worked as a group with assistance from the course’s staff. We will accept any honest criticism from the course’s staff, to acknowledgment and to credit each person’s contribution to the project. Moreover, we will make sure group members can assist each other to make sure we all get progress in professional development and get the project accomplished.

4.2 Safety

We think there are minimal chances to have safety issues for this project. However, we still have two potential safety issues to keep in mind for both the design and development process of the project. First, since we will use a LiPo battery as a power supply, we need to make sure the battery is connected correctly with each component requiring a power supply. This is crucial in order to avoid conditions such as a short circuit or where too much current is being drawn where the potential of causing severe burns becomes inevitable. According to the International Consumer Electronics Show (CES), the damage threshold for skin damage is defined as 43°C, so our device should keep its operating temperature much more below the threshold [13]. Second, we need to make sure the enclosure of the devices can protect users’ limbs from scratching or cutting as well as ensuring comfort for extended wear.

References

[1] L. E. Miller, “Cage, Cunningham, and Collaborators: The Odyssey of Variations V,” *The Musical Quarterly*, vol. 85, no. 3, pp. 545–567, 2001.

[2] W. Siegel and J. Jacobsen, “The Challenges of Interactive Dance: An Overview and Case Study,” *Computer Music Journal*, vol. 22, no. 4, p. 29, 1998.

[3] *The complete MIDI 1.0 detailed specification: incorporating all recommended practices*. MIDI Manufacturers Association, 2006.

[4] Espressif Systems, “ESP32 Datasheet,” ESP32 Datasheet, Oct. 2016

[5] InvenSense, “MPU-9250 Product Specification Revision 1.1,” MPU9250 datasheet, Jun. 2016

[6] Texas Instruments, “LM1117 800-MA Low-Dropout Linear Regulator.”, LM1117 Datasheet, January 2016

[7] A. Allar and K. Lobo, “SparkFun ESP32 Thing,” *DEV-13907 - SparkFun Electronics*. [Online]. Available: https://www.sparkfun.com/products/13907. [Accessed: 2 - Oct. - 2019]

[8] J. Steele, “SparkFun IMU Breakout - MPU-9250,” *SEN-13762 - SparkFun Electronics*. [Online]. Available: https://www.sparkfun.com/products/13762. [Accessed: 2 - Oct. - 2019]

[9] Circuit Basics, “Basics of the I2C Communication Protocol,” *Circuit Basics*, 11-Apr-2017. [Online]. Available: http://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/. [Accessed: 28 - Sept. - 2019]

[10] M. van Biezen, “Special Topics - The Kalman Filter (4 of 55) The 3 Calculations of the Kalman Filter,” *YouTube*, 15-Sep-2015. [Online]. Available: https://www.youtube.com/watch?v=X9cC0o9viTo. [Accessed: 2 - Oct. - 2019]

[11] TDK InvenSense, “Motion: TDK,” *InvenSense*, TDK Corporation 2019. [Online]. Available: https://www.invensense.com/motion/. [Accessed: 2 - Oct. - 2019]

[12] “IEEE Code of Ethics”, Ieee.org. (2019) Online https://www.ieee.org/about/corporate/governance/p7-8.html [Accessed: 17 - Sept - 2019]

[13] “Design Safe Wearable Technology with Heat Transfer Modeling” C. Brianne . [Online]. Available: https://www.comsol.com/blogs/design-safe-wearable-technology-with-heat-transfer-modeling/ [Accessed: 3 - Oct. - 2019]