PERFECT POSTURE

By

Apoorva Josyula (josyula3)
Julianna Gecsey (jgecsey2)
Rohan Kanianchalil (rohansk2)

Final Report for ECE 445, Senior Design, Spring 2021
TA: Daniel Vargas

May 5th, 2021
Group 38

Abstract

The primary goal of our design was to propose a solution to the challenges brought by prolonged sitting posture due to the COVID-19 pandemic. We aim to help users to train and become aware of when they start developing bad posture through their workday by leveraging an easy to wear device that will inform and track throughout the day when they start to lose their good posture. We also included an app interface that allows users to see their progress over time as they regularly use this device. Ultimately, our product helps to minimize the risk of serious posture related injuries by gradually improving and training a user's back to be in a stronger sitting position.

Contents

1. Introduction

1.1 Objective

In the time of Covid-19, we are seeing a rise of the "WFH" lifestyle as more and more non-essential workers are having their offices be shut down and their work be moved to a remote setting. This sudden change of workplace is creating a new sort of health crisis to occur. According to an April Facebook survey from the American Chiropractic Association, 92 percent of chiropractors (out of 213 respondents) said that patients report more neck pain, back pain or other musculoskeletal issues since the stay-at-home guidance began [1]. The most common posture related issues that are arising are coming from the upper back to neck area, Scott Bautch, the president of the American Chiropractic Association's Council on Occupational Health, says that as screen time has exploded, we're more at risk of "Text Neck" (bending our necks too close to our screen) and "Selfie Elbow" (strained use of the elbow being held up for long time) [1]. We recognize that this issue will only continue to get worse as more jobs continue to become remote, requiring more time to be spent sitting in front of a desk. Our solution is to introduce our device "Perfect Posture".

1.2 Background

The products for creating "perfect posture" devices are varied, from back braces to even single system wearable devices that one can wear right on their back. These products however all lack in holistically assessing a person's posture. When it comes to physical braces, the external mechanics of a back brace force a person's body and muscles to rely on the physical weight of the brace to hold up their back posture. This can be harmful as the body will become dependent on that physical brace to help keep them upright and could led to in some cases muscle dystrophia with the lack of use of muscle in the back region [2]. Other wearable smart devices, such as the "Upright Go" are helpful in eliminating the need for a physical back brace however, they typically only focus on one aspect of a person's spine. As studies have shown, the result of bad back posture is not just isolated to one specific area of a person's spine but deal with the holistic approach of maintaining one's spinal curvature [3]. These posture devices don't consider how a person's spine can be held "upright" but fail to maintain a healthy curvature of the total spine [3].

This is one of the many reasons why we have created our device to not just be a posture sensor that detects whether a person's upper spine is being held upright, but to calculate a person's spinal curvature to see if they are maintaining the healthy amount of tension and bend in their upright position [3].

When we talk about curvature, we are looking at how the spine looks, from a side viewpoint. The whole spine itself, from the top of the cervical spine to the bottom of the lumbar spine, all form an important side view curvature that resembles an S-shape [4]. For the most part, there is a lot of focus on worrying about lower back problems, and thus most products are looking at the lower back angles to fix spinal curvature shape and position. What many products neglect to realize that is there are very few products that focus on the upper spine curvature and shape. We are specifically talking about the shape and position of the thoracic spine, one's upper-middle spinal column. The thoracic kyphotic curve has the shape of a normal C, which is proportional to all humans. Many of the modern-day posture issues such as text-neck, hunching, and selfie elbow stem from the lack of maintaining that thoracic kyphotic curve [4].

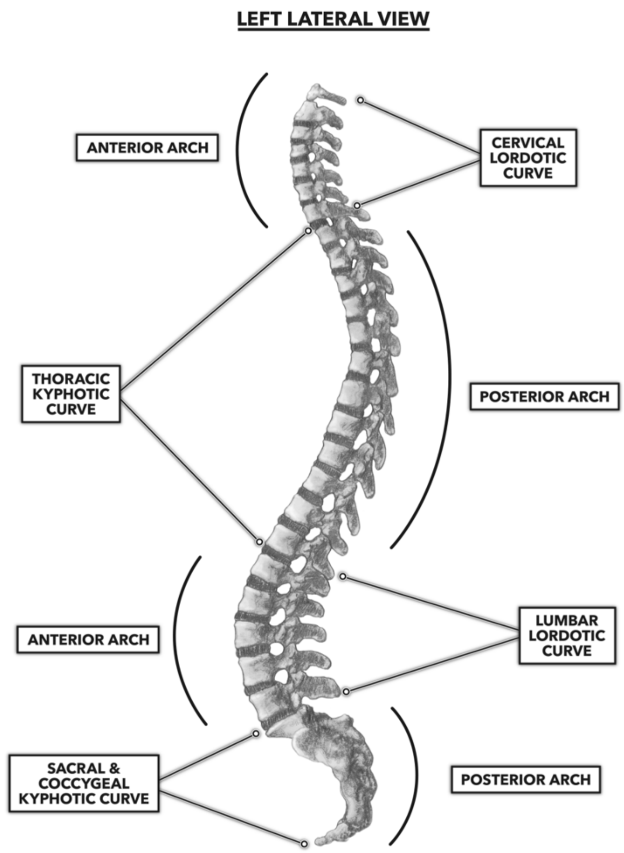


Fig 1. Kyphotic and lordotic curves of the vertebral column [5]

The thoracic (rib cage) portion of the spine has a normal forward curvature, which has a normal range (20 to 50 degrees). Any exaggerated rounding of the forward curvature in the upper back is called hyper kyphosis (meaning too much kyphosis), but the term "kyphosis" is commonly used to refer to the clinical condition of excess curvature of the upper back (greater than 50 degrees), leading to a stooped forward posture [5].

Our product will help to maintain a healthy range of curvature and bend in this area by ensuring that a user does not exceed the range of 50 degrees or below the range of 20 degrees.
We recognize that it is humanly impossible to never move from your given posture, there will be times that will require a person to slouch for a bit. Our device shouldn't be going off every second if a person were to just bend down to pick up a pencil. Thus, there is leeway in how long a person can be in a certain position without it being considered "detrimental" to that person's given posture.

Another factor to consider is how long a user should sit at their desk with a device attached to them. What most researchers have found is that a low-risk time spent sitting at your desk is typically around 30 minutes with a max interval of 4 hours all together [5]. For our device to encourage holistic posture health, we will allow for our device to remain operational for a max of 4 hours.

With the knowledge of how we need to be maintaining these angles and for how long can time elapse before a person engages in bad posture, we as ECE students created a device that will accurately be able to detect deviances from these important angles and create a system that will help alert the user on when they have been continuously in a bad posture.

1.3 Overall System

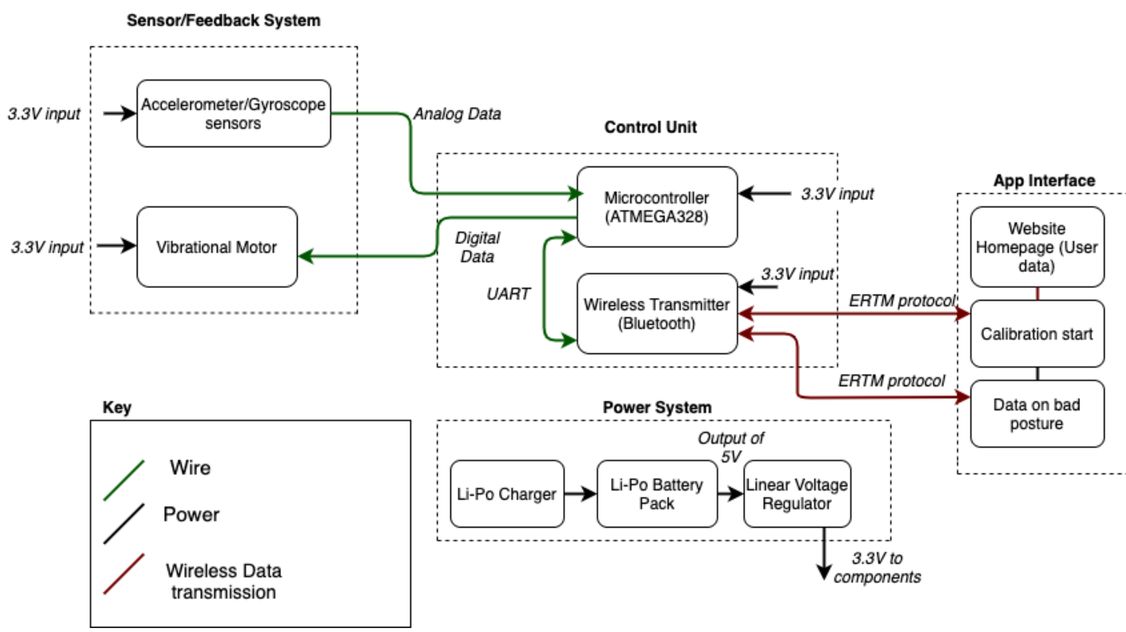
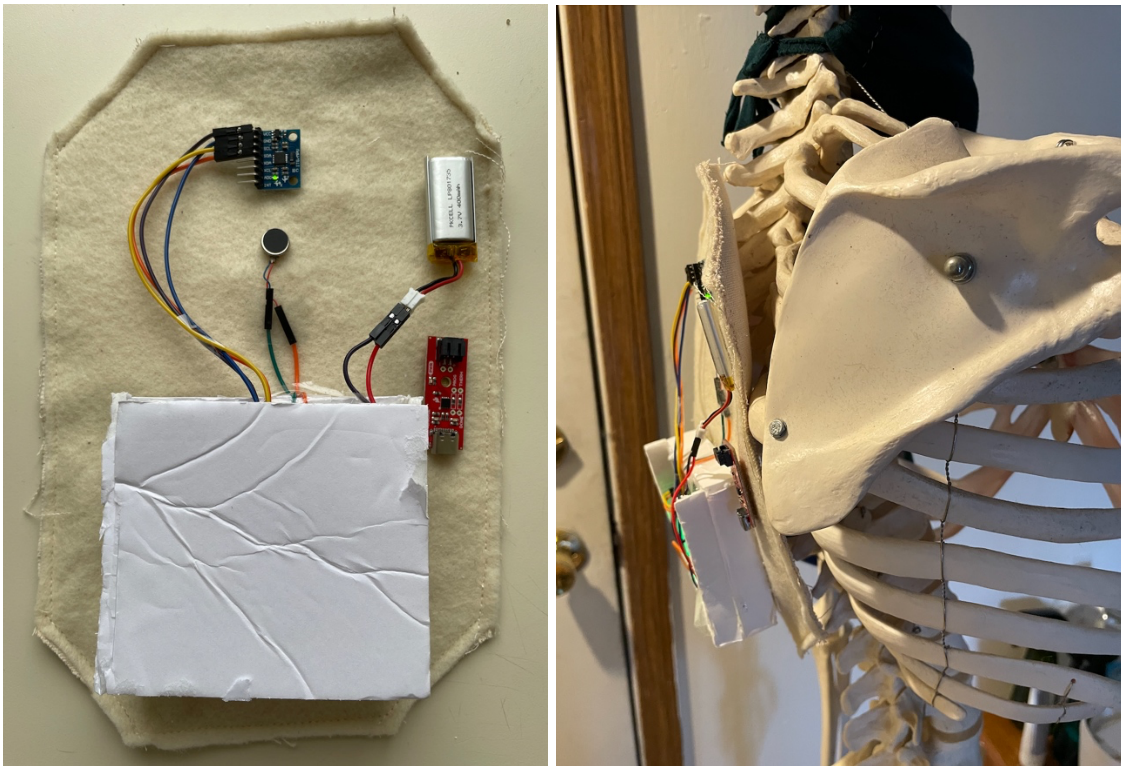


Fig 2. Block Diagram of Perfect Posture

Our wearable device has the following four subsystems: A sensor/feedback system, a control unit, a power system, and an app interface. The layout is shown below in Fig. 2. The power system ensures that the system can be powered for a healthy 4-hour period the user might be spent working with the proper 3.3V output. Our sensor system reads a person's spinal position, and our feedback system alerts the user when that spinal position is off and repeats that alert in a steady 2-minute interval. The control unit consists of our microcontroller that parses and processes the sensor data and sends bad posture alert to the feedback system as well as sending processed data via a Bluetooth module to the user app interface. Our app interface has the users posture data in an interactive way for the user to see the trends in their posture.

2. Design Details

2.1 Physical Design



a. Layout of our device b. Placed on Skeletal structure

Fig 3. Final Prototype of device

As we mentioned before, our goal is to ensure upper back posture, so our placement will be along the cervical/thoracic spinal column. The curve of the thoracic kyphotic curve can be measured anywhere from the range of the C7 spinal disk to the T10 spinal disk, so for our sakes we will be focusing on placing the sensors from the expanse of the C7 to the T8. That roughly translates to placing our sensors at bottom nape, the middle of the shoulder blades, and at the middle of the back.

An important thing to note about our physical design is that we want to keep this relatively small. The individual sensor components and vibrational motor will be no large than a 4x4cm design. For ease of attaching our device without shifting the sensors around too much, we attached it to a felt/duck canvas pad with reusable fabric adhesive to be able to remove off clothing whenever possible. A makeshift Styrofoam case was built to protect our control unit.

The weight to not affect the individual posture will be no larger than 150 grams.

2.2 Subsystem Descriptions

2.2.1 Sensor/Feedback System

In making our sensor subsystem, we needed to measure the angular data of how a person's spine would curve throughout the day. To accomplish this goal, we chose to use a combination of a gyroscope and accelerometer.

Our choice design for the sensor system was to use the "Triple Axis Accelerometer and Gyroscope Breakout from Sparkfun" (MPU-6050).

Our sensors were powered by a 3.3V input, communicating with our microcontroller via the I2C bus. The sensors would return back raw g force and acceleration readings, in which the embedded programming from the microcontroller would convert the raw readings and filter them to be x,y, and z angles to help determine a tilt position. The code can be found in Appendix B.

For our feedback system we had a vibrational motor that was attached alongside of our control unit. Whenever the sensors would detect a slight slouch in the users' posture, the vibrational motor was set to vibrate for a 3 second period.

The vibrational motor was set off through the microcontroller's logic as it was attached to one of the digital I/O pins.

2.2.2 Control Unit

Our control unit consisted of our microcontroller chip that read our sensors data and wrote to our vibrational motor when posture was off. It also consisted of a Bluetooth module that could send data to our front-end application for users to see their progress.

The ATMega328p was the chip we used to be our microcontroller. We included our microcontroller into our overall design of our PCB design for ease of connecting all the different components of our device. To support reading multiple sensor devices to our I2C bus, we designed our PCB to have additional 10k Ohm pull up resistors along the SDA/SLA lines. The main purpose of our microcontroller was to hold our embedded program that determined if the sensor angles being read were passed the certain range that was needed to maintain good posture.

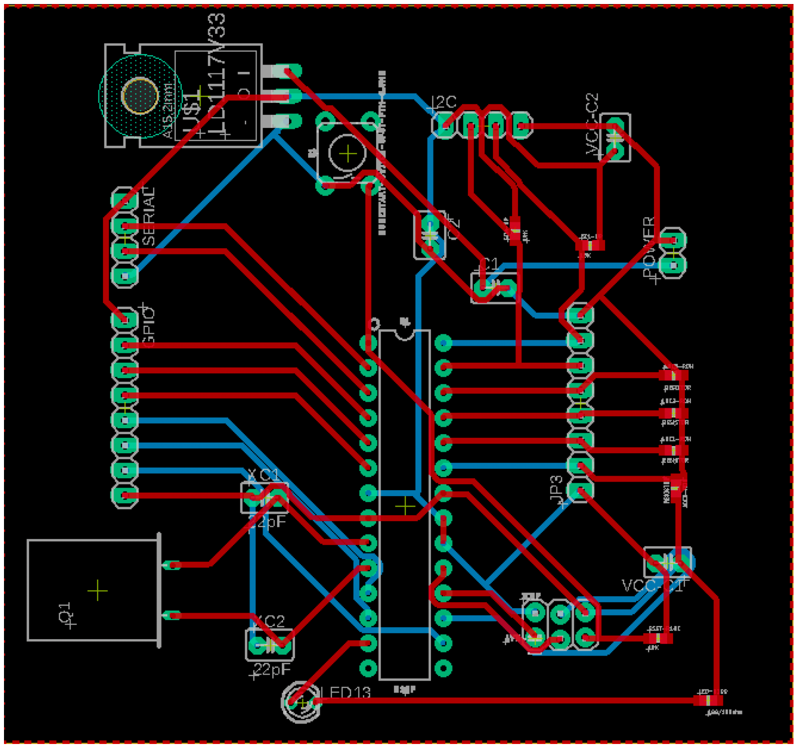


Fig 4. PCB design of microcontroller unit

The Bluetooth module we used was the Sparkfun RN-42 Bluetooth Module breakout. This was externally connected to our microcontroller communicating via UART. Our Bluetooth module was needed to take our user data that is being processed by our microcontroller and have it sent to our front-end application for users to graphically see their results.

2.2.3 Power Supply

Our power supply consisted of a light-weight lithium polymer rechargeable battery, a lithium battery charger, and a linear voltage regulator.

Our battery had a supply voltage output of 3.7V at 2500mAh. Since we didn't want a user to have to continuously remove and replace a battery, we attached a Sparkfun Li-Po battery charger to our device. The charger would feature an LED indicator to when the battery was fully charged back up to 3.7V.

Since most of our components required a 3.3V input to operate at their peak performance, we attached a linear voltage regulator to our PCB board to help take in our inputs from the battery and correctly adjust to an appropriate input voltage across other parts.

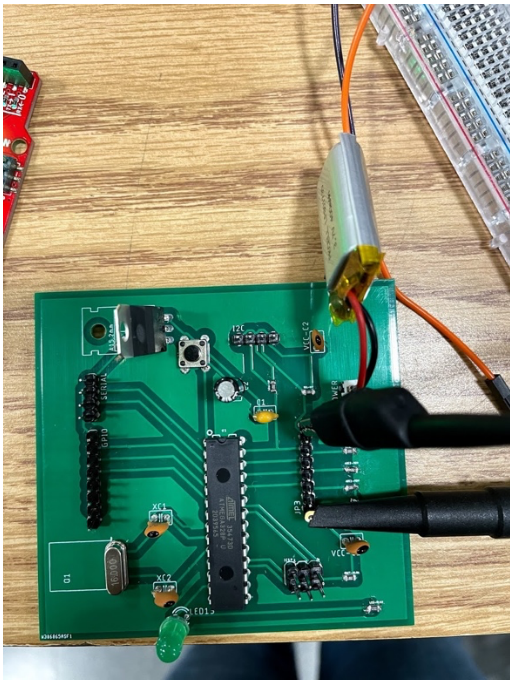


Fig 5. PCB board with attached Lin Regulator

2.2.4 User Interface

Alongside having a physical reminder of when a user's posture is off, we also wanted to include a mobile application that would help a user physically see their progress. The following figure shows the frontend design of what a user would see on their mobile device after one session with the device.

The storage of the incoming data the application would receive from the Bluetooth module would be stored in a MongoDB database, and displayed from a Flask/React mobile application. The main purpose of this user interface is track progress of how their posture has improved in a long-term setting.

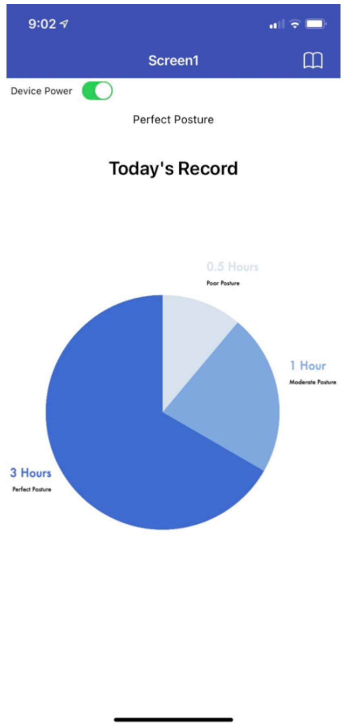


Fig 6. User Interface Mockup

2.3 Design Alterations

From our original design to our final physical implementation, there were certain design choices we had to alter to meet our goals. From a sensor standpoint, we had originally chosen two sensors to be placed alongside the

3. Calculations and Simulations

3.1 Sensor Angular Readings

To read the raw data that was coming in from our sensors, we needed to add a filtering equation to understand the actual tilt angles.

$$A_x = tan^{-1}\left(\frac{x}{\sqrt{y^2+z^2}}\right)$$

$$A_y = tan^{-1}\left(\frac{y}{\sqrt{x^2+z^2}}\right)$$

Eq 1: Angles about the x and y axis

The following equation was needed to understand how to take the raw values of the pitch and yaw values that are getting read from the sensors themselves. This equation was added into our overall code that would be embedded into our control unit. We chose to opt out of reading the z-axis reading as that angular value didn't necessarily give us useful information on understand what spinal position looked like.

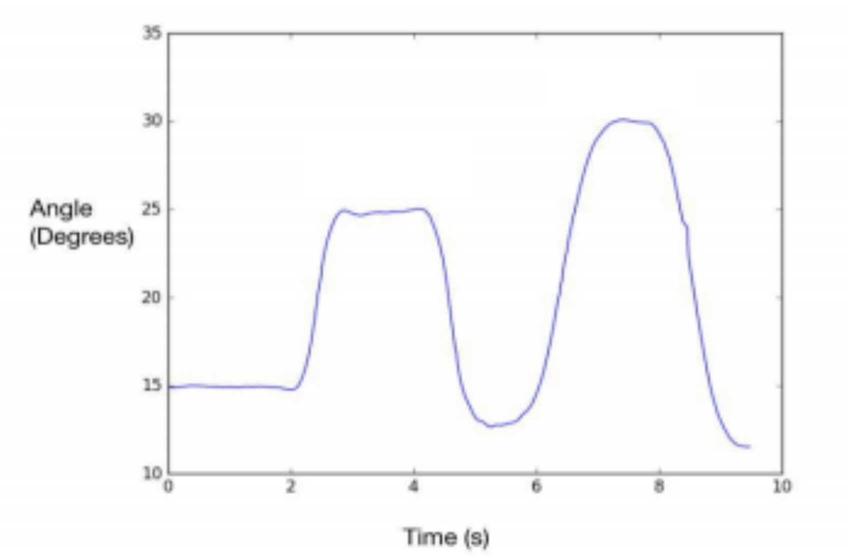


Fig 7. Graph taken of a user slouch in 10 second period.

The graph above depicts two instances when a user was slouching. Given what we know what a healthy range of a spine curving is, the sensors were able to accurately detect at the range we needed to see what an unfit angle is. We had an allowance of 5 to 7-degree difference off what was said to be a proper angle reading.

One thing we had to account for as we were developing our code to read sensor data was the ability for the gyroscope to drift after a certain period.

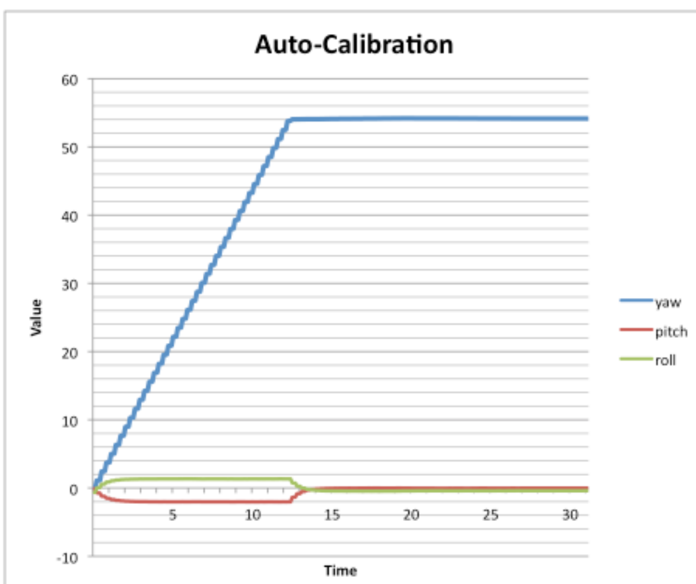


Fig 8. Graph of Gyroscope drift

We tested to see the value of how the gyroscope would drift initially by lying it flat on the surface where pitch, yaw, and roll values would have been 0. However, the yaw value had initial drift of a period of 10 seconds before it was steady. To counteract the drift, we were receiving we added error code to help process out the noise the gyroscope was receiving initially in the beginning found in Appendix C.

3.2 Power Supply Strength

One of the tests we ran for our power supply was to see if the linear voltage regulator was properly able to output 3.3V given a power source. We ran it with first our 3.3V li-po battery. We noticed that it would keep a steady 3.3V output initially but it would very quickly drop down to 3.0V output, as shown in figure 7.

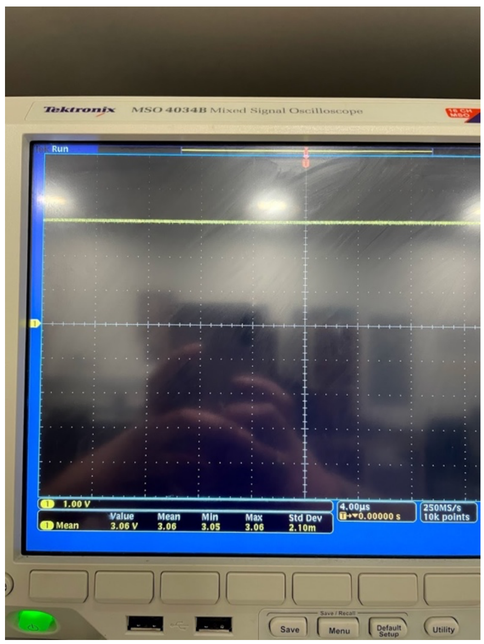


Fig 9. Voltage Output reading on PCB of 3.7V

Since we want a consistent voltage output of 3.3V for about a 4-hour long period, we decided to switch to a 5V power output and ran the test for about 20 minutes, the figure below shows a consistent reading with a 5V supply battery after running this test.

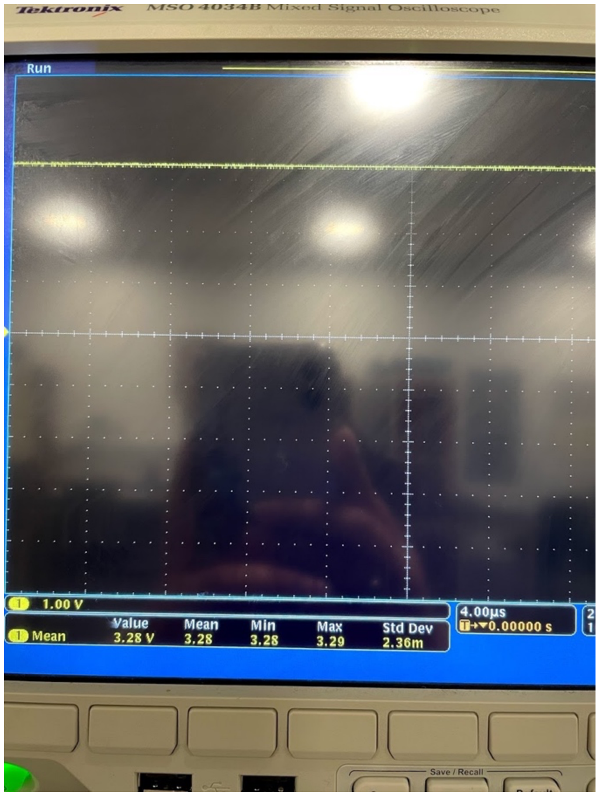


Fig 10. Voltage Output reading on PCB of 5V

Thus, to keep our system being able to run without constant recharging of the battery we opted to use our 5V battery instead.

3.3 Reliability of Algorithm and Sensors

One thing we had to check for to see the reliability of our system was to check for how often we would get results where the user would get a vibrational feedback even though they were in correct posture, or they would get no vibration even while being in bad posture.

We had 6 different people try on our device, we gave them about 15 minutes to move around in the device and could change positions for about 5-6 times during that time. We were able to acquire 34 different position readings from the users.

A false positive in our case is defined as when a user detects a vibration even though their posture is correct. False negative is defined in our case as someone who doesn't detect a vibration even though their posture is not correct.

False Positive	4	False Negative	6
True Positive:	12	True Negative	12

Table 1. Test results of wearing device

With the results given above we calculated our sensitivity rate based on how often the slouching position was correct and got a result of 66.6%. One reason we believe our sensor system was not as responsive as we wanted it to be was only relying on 1 sensor. Since we are looking at the curve, to get a more accurate depiction of the curve at the upper back two sensors would have reduced that extra false positive number.

Another we noticed as we tracked the false negative rate was when users were moving around side to side, it would not necessarily vibrate. We defined incorrect posture within our code as a change around the x axis of the plane, so in future testing redefining bad posture to include the y axis reading as well would help to reduce that false negativity rate.

4. Cost Analysis and Schedule

4.1 Final Cost of Labor

When organizing an analysis on the cost of our labor we find that assuming we work an approximal 10 hours a week, with a team of 3 students, at a standard rate of $35 an hour, our labor would equate to $16,800 over the course of the 16-week semester.

$$3\ Group\ Members * \frac{\$35}{hour} * \frac{10\ hours}{week} * 16\ weeks = \mathbf{\$16,800}$$

Eq 2. Labor Calculation

4.2 4.2 Final Cost of Parts

The price of labor is not the only cost in this project, we will also have equipment to be purchased to build out our project. A summary of the costs has been calculated below in Appendix D. Our total cost came out to be $**101.28**.

4.3 4.3 Schedule

Week	Julianna	Apoorva	Rohan
1/25	Come up with an initial posting for a senior design proposal.		
2/1	Establish senior design group - Perfect Posture		
2/8	Help build out the validation of the importance of upper back posture in initial project approval.	Start drafting Project Proposal and getting it submitted for approval	Researched more individually on what was feasible for our project

2/15	Continue to validate and research upper back posture for our project proposal.	Collect research data and solutions discussed in team meetings and draft up proposal	Found comparable projects and started getting ideas of possible implementations.
2/22	Attend Eagle CAD workshop and complete individual Eagle Assignment. Craft the introduction for the design document and help contribute to validation where needed.	Watch Eagle CAD workshop and finish individual assignment on Eagle CAD Compile together Requirements and Verifications for different subsystems	Research on how to measure the curve of the back
3/1	Prepare for design document check. After presenting, discuss with the team what should be changed with project design and make the corresponding edits to the design document. Solidify the components that we will be using: ATMEGA328PU and RN-42 bluetooth module.	Creating Draft of Design Document, mostly focusing on creating Block Diagram, Subsystem descriptions and Requirements and Verification	Set up the design document, we then presented, taking notes on what we needed to improve on.
3/8	Prepare for design review presentation. Begin designing the PCB by building out the schematic. Look at components' recommended schematics and determine what parts are required for our application based on pinouts.	Take feedback from Design Review and adjust components accordingly	Searched up what components would be necessary and what was needed to be ordered.
3/15	Begin placement of components on the PCB and calculate necessary trace width to carry specific voltages and currents.	Create first round design of PCB of Microcontroller and submitted it for approval.	Drew up how the components would be connected, creating an overall schema of the project
3/22	Order the components to be soldered onto our PCB board.	Order all parts that have been agreed upon, look at revisions needed for PCB.	Ordering of the components
3/29	Begin my individual progress report and submit for feedback before actually due.	Submit Individual Progress Report, Start compiling code for testing MPU-6050 sensors	Completed the individual progress report.
4/5	Learn TinkerCad through online tutorials to build three 3D	Send in revisions for Third PCB design. Collect parts that	Soldered the pieces together for our PCB

	enclosures with a sliding top to encase all the sensors for ease of placement on the user.	have come in and breadboard for testing.	Initial test of our device on an anatomically correct skeleton
4/12	Help try and create a mockup of what the app interface should look like for the bluetooth module to connect to using MIT App Inventor.	Finalize testing done on sensors on Arduino and start creating code for vibrational motor.	Began testing on other individuals in order to have more variety of test cases Finishing touches on the device in order to be ready for the final demo
4/19	Help virtually demo the project. Begin working on the slide deck for the final presentation and mock presentation.	Testing of Sensor/Feedback subsystem and Vibrational finished. Third PCB arrival so start solder components for control unit. Revert to backup plan as certain components were not functioning properly.	Demoed in person And helped work on the slides for the final presentation
4/26	Present the introduction, control unit, and successes and challenges for our final project presentation.	Present Demo during class and prepare for presentations	Presented portions of the slides at the mock demo and took notes for improvements
5/3	Present final project, write sections of each part of the final paper	Present final project and finalize paper.	Demoed the final presentation and finished up the final paper.

Table 2. Schedule broken down by week and team member

5. Conclusions

Our group, at the end of the 16 weeks of class, was able to present a functioning device at our final demo. We encountered a few issues throughout the process; however, we were able to show a physical implementation of our idea. We had our control unit and sensor that successfully extrapolated data from a user's movement and was able to interpret whether the individual’s posture was in a good or poor position.

5.1 Accomplishments

Based on the final device shown, our group was able to achieve a lot of goals that were set at the beginning of the semester. Our device was able to correctly implement a sensor/feedback system that detected bad posture and after a two-minute interval, notifying the user using the vibrational motor. The device also had a functioning control unit that ran Arduino code from the sensors. Our final device as well fell into our weight requirement and it was not a hindrance to the user when placed on their back.

5.2 Uncertainties

Although our device was complete and functioning, there were a few portions of the project that were unable to be completed due to time constraints as well as other issues that came along the way. One was that when constructing the physical device, our microcontroller was not properly working. For that reason, we had to change to an Arduino by the last week due to the timing constraints of ordering another. Due to the fact, instead of having 2 gyro/accelerometers, our device was only able to support one and not able to attach our Bluetooth module. Therefore, the code used to extrapolate the data had to be shifted slightly to account for the lack of another gyro/accelerometer. Another issue that came about was the fact that another individual was needed to place the device on one's back. This issue was difficult to circumvent without changing how we created our device altogether.

5.3 Ethical Considerations

When creating our project, we considered a few safety issues as well as some ethical as well.

5.3.1 Safety

Starting with the issues that could possibly occur during the development of the project is the usage of 2 accelerometers and gyroscopes. With that many being in use, if we are not careful with how things are wired and powered, it may cause a malfunction in our project. Continuing, with the usage of a microcontroller and a Bluetooth module, if we miscode the microcontroller, we may cause an adverse effect in our project. We must also consider how the accelerometer and gyroscope may affect the wearer and the rpm of the vibrational motor so that it doesn't hurt the user. Finally, a factor to consider is the type of adhesive used when placing the device on the user.

Regarding the physical issues with the device regarding the user, we have to consider many factors. We must consider the voltage of the battery when designing our implementation so that it may not cause harm to the user. With a power supply being physically placed onto the user, we must ensure that the amount of power that is being used will not cause any harm. Although we are only planning on using a 3.7V battery, which is too low for shock (~60V), skin and membrane damage(~600V) or permanent damage(~20,000V), we must consider the current that our design is using. Anything more than 16 mA, will cause discomfort to an individual, so to air on the side of caution, we should not even reach 10mA, to ensure the utmost safety to the user [6]. Even with these precautions, we can ensure the utmost safety to the user by encasing the battery pack as well as the rest of our circuits in non-conducting material, which will be done by 3D printing using material such as carbon fiber.

Another factor that we need to consider is the type of adhesive that will be used on the user's skin. We must make sure that it's hypoallergenic, so that there's no adverse reaction to the user and we must make sure that we pick an adhesive that can be reused a multitude of times.
A key ethical issue we must concern ourselves with is that our project is attempting to prevent a medical condition. Therefore, if we are not accurate with our readings and data, we may mislead the user with incorrect information and may cause more harm than good to the user. We must account for every possible scenario to ensure that we do not cause a detriment to the health of

our user. With this invention being placed on our user, we must also make sure that it is not too heavy (> 150 grams) or else it will cause strain to the user's neck and upper back.

5.3.2 Ethics

There are few ethical codes that we must worry about when we design our project. Because our implementation is physically being put on a person, we must make sure that we follow code 1.2 of the ACM Code of Ethics: Avoid Harm [6]. We must make sure the adhesive and the components of our invention that are being placed on a person will not harm them in any way and cannot be used in such a way that it negatively impacts the user.

With our project including an app interface, we must also consider code 1.6 of the ACM Code of Ethics: Respecting Privacy [7]. Our project constantly monitors an individual's posture as well as many calibration factors, such as weight, height, age, etc., so we must keep the user's information confidential as those factors are private to oneself.

One more ethical issue that we should highlight and make sure to follow is the IEEE code of ethics 7.8 subsection 10: to support colleagues and co-workers in following this code of ethics, to strive to ensure the code is upheld, and to not retaliate against individuals reporting a violation. [8]. This is a code that we have to uphold because we are making something from scratch, which entails 100's of hours of potential work. We must make sure that although we must put that amount of effort into our project, that everyone in our group is making sure that they are both physically and emotionally well and ensure that there is a balance between the work needed for this project as well as a healthy off-work life.

5.4 Further Work

With how the device is now, it can improve immensely with a few changes. Firstly, a switch from the Arduino to the microcontroller like intended, to implement our second sensor and make the device's reading even more accurate. Furthermore, it would be imperative to have the user interface completed so that it could be used and understood by an individual who may not have the knowledge that our group or someone with a medical profession would have. Another consideration would be to make the device itself smaller or even embedded in an undershirt so that it can be more easily placed without much error.

References

[1] nytimes.com, “The Pandemic of Work From Home Injuries”, 2020. [Online]. Available: https://www.nytimes.com/2020/09/04/well/live/ergonomics-work-from-home-injuries.html?fbclid=IwAR3y4TwqmDq2ZLAoYVMlpikjyPZH-7xRMcqfMyD-eQoSS3FaI5UZ14nc5zU [Accessed 1-March-2021]

[2] spine-health.com, “How to Measure and Fix Head Posture”, 2018. [Online]. Available: https://www.spine-health.com/conditions/neck-pain/how-measure-and-fix-forward-head-posture.com [Accessed 18-Feb-2021]

[3] spine-heath.com, “Forward Head Posture’s Effect on the Cervical Spine”, 2018. [Online]. Available: https://www.spine-health.com/conditions/neck-pain/forward-head-postures-effect-cervical-spine.com [Accessed 18-Feb-2021]

[4] Cedars Sinai, “Curves of the Spine”, 2021. [Online]. Available: https://www.cedars-sinai.org/health-library/diseases-and-conditions/c/curves-of-the-spine.html [Accessed 4-March-2021]

[6] ncbi.nlm.nih.gov, "Conduction of Electrical Current to and Through the Human Body", 2009. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2763825/ . [Accessed: 4-March- 2021].

[7] acm.org, "ACM Code of Ethics", 2018. [Online]. Available: https://www.acm.org/code-of-ethics. [Accessed: 18- Feb- 2021].

[8] Ieee.org"IEEE Code of Ethics", 2016. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html . [Accessed: 18- Feb- 2021].

Appendix A Requirements and Verification Tables

Requirement	Verification	Successes and Challenges
1. Accelerometer/ Gyroscope must be able to send raw x,y,z and g force data to the microcontroller via I2C. 2. MPU-6050 must be able to sample data at a rate of at least 100Hz.	1. See if microcontroller receives packets of sensor data from the MPU-6050. a. Test accelerometer by first powering the device to a 5.0V power in the VCC pin, ground to GND pin, and the data pins onto our ATMega328p chip. b. Connect Microcontroller to computer and let it be connected to an Arduino Serial Monitor. Check to see if connection has been made between the two parts. c. Write program that will print out the raw g values and degree values from sensors onto the monitor. d. Add algorithm for reading filtered angle values and check on serial monitor 2. Check 100 samples per second on average is being collected from MPU-6050. a. Load program for reading timestamps onto ATMega328p b. Connect MPU-6050 to microcontroller. c. Assure power connections. d. Start recording data for 20 seconds. e. Check data collected on microcontroller. If the number of samples that was collected was greater than 2000 samples.	1. Accelerometer was able to send raw x,y,z and g force data to the arduino. 2. Ardiuno was able to sample at a rate of 9600 BAUD.

Table 3: R&V table for MCU-6050 sensor

Requirement	Verification	Successes and Challenges
1. When microcontroller sends HIGH signal to motor, the motor will vibrate for 3 seconds to indicate bad posture.	1. Connect vibrational motor to a 5V power source and connect to our microcontroller chip as well. a. Write code that will switch digital pin from low to high	1. Arduino can send HIGH signal to motor and will vibrate for 3 seconds to indicate bad posture. Additionally, a waiting period of 2 minutes is instilled to prevent user from constant vibration

	Hold vibrational motor to feel for vibrations when switched onto c. Record the time the vibrational motor is held on high to see if it stays on for 3 seconds.	

Table 4: R&V table for Vibrational Motor

Requirement	Verification	Successes and Challenges
1. At full charged capacity, battery should be outputting a voltage charge of 3.7V +/- 0.2 V at 400 mA for maximum current	1. Connect an oscilloscope to the battery (one end to the positive terminal of battery, other end to the negative terminal of battery). a. Graph the voltage readings to ensure, the output voltage is around 3.7V +/- 0.2V. b. To measure current output, with the oscilloscope already connected, connect the battery to a 10ohm resistor, and check current output on graph.	1. To provide steady output to the PCB components we decided to change the battery to 5v to handle all the components being held at 3.3v through the linear voltage regulator

Table 5: R&V Table for Li-Po battery

Requirement	Verification	Successes and Challenges
1. Given a voltage of 3.7 +/- 0.4V, our linear voltage regulator must output a voltage of 3.3 +/- 0.05V.	1. Use a function generator to supply the given voltage output. a. Attach an oscilloscope where one end is attached to the positive terminal of the voltage regulator, and the other end is attached to the negative terminal of the voltage regulator. b. Check the graph results from the oscilloscope to see what the output voltage from the linear voltage regulator is showing. The results should be around 3.3V +/- 0.05V.	1. Using our 5v battery we were able to provide a steady 3.27V signal to all the necessary components

Table 6: R&V Table for Linear Voltage Regulator

Requirement	Verification	Successes and Challenges
1. Be able to read raw sensor data and calculate an angle approximation via I2C at BAUD rate of 9600.	1. Attach our sensors by the analog pins on the microcontroller.	1. We were able to read raw sensor data and calculate the angle of approximation to the nearest degree.

	a. Flash user defined code onto microcontroller that will take inputs from sensors as inputs to angle calculator function. c. On the Arduino serial monitor, wait to see if the function code will return the predetermined angles. d. Determine on graph to see frequency rate.	

Table 7: R&V Table for Microcontroller (ATMEGA328)

Requirement	Verification	Successes and Challenges
1. Be able to communicate with microcontroller via UART at a speed of 1.125kBd.	1. Open terminal on computer and attach microcontroller to a USB UART bridge a. Have terminal receive at 1.125kBd. b. Have set characters to be sent and return. c. Check to see if all characters are received from and verify.	1. We were able to get our arduino communicating with our arduino serial monitor at a rate of 9600 BAUD.

Table 8: R&V Table for Bluetooth Module (RN-42)

Appendix B Sensor Algorithm and Flowchart

Flowchart for calculating bad posture

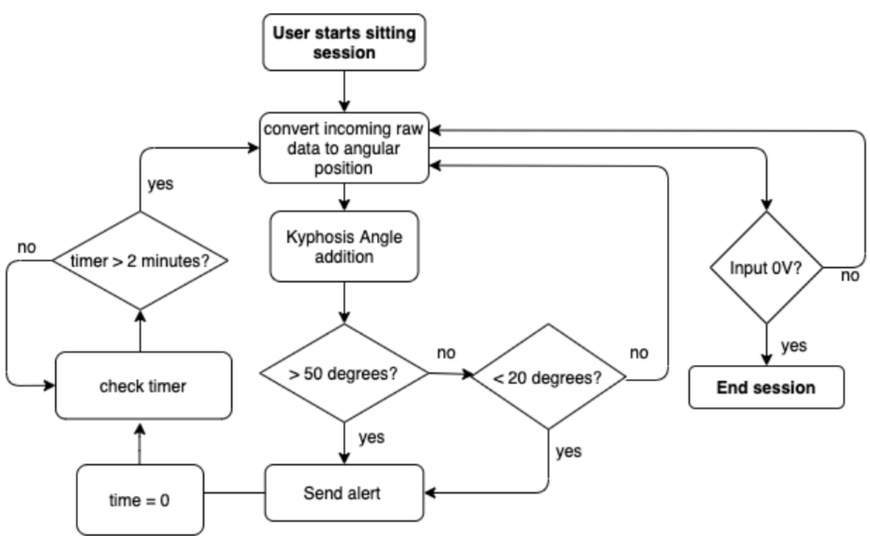


Fig 11. Flowchart of Sensor code

Arduino Code

sketch_apr20a

```
#include<Wire.h>

const int MPU_addr=0x68;
const int MPU_addr2 = 0x69;

int16_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZ;
int16_t AcX2, AcY2, AcZ2, GyX2, GyY2, GyZ2;

int minVal=265; int maxVal=402;


double x; double y; double z;
/*double x2; double y2; double z2;*/

const int motorpin = 6;

void setup(){
  Wire.begin();
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);

/*  Wire.beginTransmission(MPU_addr2);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);*/
  Serial.begin(9600);

  pinMode(motorpin, OUTPUT);

  }

void loop(){
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x3B);
  Wire.endTransmission(false);
  Wire.requestFrom(MPU_addr,14,true);
  AcX=Wire.read()<<8|Wire.read();
  AcY=Wire.read()<<8|Wire.read();
  AcZ=Wire.read()<<8|Wire.read();
  int xAng = map(AcX,minVal,maxVal,-90,90);
  int yAng = map(AcY,minVal,maxVal,-90,90);
  int zAng = map(AcZ,minVal,maxVal,-90,90);

x= RAD_TO_DEG * (atan2(-yAng, -zAng)+PI);
y= RAD_TO_DEG * (atan2(-xAng, -zAng)+PI);
z= RAD_TO_DEG * (atan2(-yAng, -xAng)+PI);

/*  Wire.beginTransmission(MPU_addr2);
  Wire.write(0x3B);
  Wire.endTransmission(false);
  Wire.requestFrom(MPU_addr2,14,true);
  AcX2 = Wire.read()<<8|Wire.read();
  AcY2 = Wire.read()<<8|Wire.read();
  AcZ2 = Wire.read()<<8|Wire.read();
  int xAng2 = map(AcX2,minVal,maxVal, -90,90);
  int yAng2 = map(AcY2,minVal,maxVal, -90,90);
  int zAng2 = map(AcZ2,minVal,maxVal, -90,90);

x2 = RAD_TO_DEG * (atan2(-yAng2, -zAng2)+PI);
y2 = RAD_TO_DEG * (atan2(-xAng2, -zAng2)+PI);
z2 = RAD_TO_DEG * (atan2(-yAng2, -xAng2)+PI);*/

int total_angle;

/*KYPHOSIS ANGLE CALCULATION*/

total_angle = x - 45;


/*Serial.println("----------------------------------------");

Serial.print("AngleX= "); Serial.println(x);

Serial.print("AngleY= "); Serial.println(y);

Serial.print("AngleZ= "); Serial.println(z); */

/*Serial.println("----------------------------------------");
Serial.print("AngleX2 = "); Serial.println(x2);
Serial.print("AngleY2 = "); Serial.println(y2);
Serial.print("AngleZ2 = "); Serial.println(z2);*/
Serial.print("Total Angle= ");
Serial.println(total_angle);
Serial.print("AngleX= "); Serial.println(x);

delay(500);

if (total_angle < 35 ||total_angle > 55) {
  analogWrite(motorpin, 255);
  delay(1000);
  analogWrite(motorpin, 0);
  delay(120000);
}

}
```

Fig 12. Actual Arduino code for sensors

Error Calculation Code

```
void calculate_IMU_error() {
  while (c < 200) {
    Wire.beginTransmission(MPU);
    Wire.write(0x3B);
    Wire.endTransmission(false);
    Wire.requestFrom(MPU, 6, true);
    AccX = (Wire.read() << 8 | Wire.read()) / 16384.0 ;
    AccY = (Wire.read() << 8 | Wire.read()) / 16384.0 ;
    AccZ = (Wire.read() << 8 | Wire.read()) / 16384.0 ;
    AccErrorX = AccErrorX + ((atan((AccY) / sqrt(pow((AccX), 2) + pow((AccZ), 2))) * 180 / PI));
    AccErrorY = AccErrorY + ((atan(-1 * (AccX) / sqrt(pow((AccY), 2) + pow((AccZ), 2))) * 180 / PI));
  }
  AccErrorX = AccErrorX / 200;
  AccErrorY = AccErrorY / 200;
  c = 0;
  while (c < 200) {
    Wire.beginTransmission(MPU);
    Wire.write(0x43);
    Wire.endTransmission(false);
    Wire.requestFrom(MPU, 6, true);
    GyroX = Wire.read() << 8 | Wire.read();
    GyroY = Wire.read() << 8 | Wire.read();
    GyroZ = Wire.read() << 8 | Wire.read();
    GyroErrorX = GyroErrorX + (GyroX / 131.0);
    GyroErrorY = GyroErrorY + (GyroY / 131.0);
    GyroErrorZ = GyroErrorZ + (GyroZ / 131.0);
    c++;
  }
  GyroErrorX = GyroErrorX / 200;
  GyroErrorY = GyroErrorY / 200;
  GyroErrorZ = GyroErrorZ / 200;
```

Fig 13. Actual Arduino Code for sensors (error filter)

Appendix C Circuit Schematic of PCB

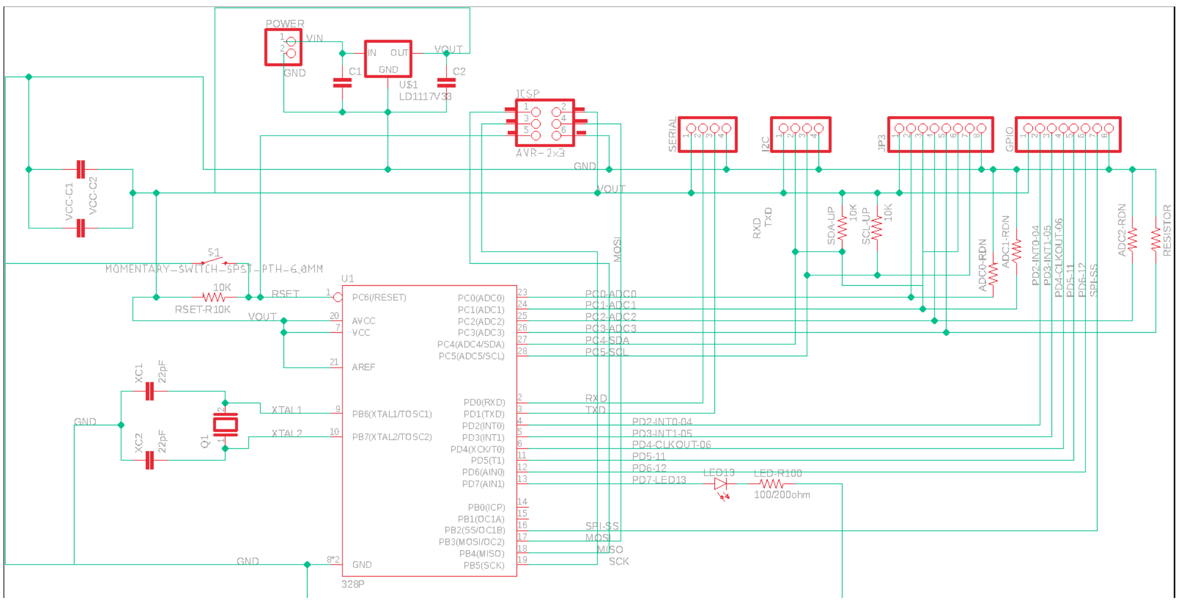


Fig 14. PCB Schematic

Appendix D Final Cost of Parts

Part	**Quantity**	**Cost per Unit**	**Total Cost**
GY-521 MPU-6050 Accelerometer/Gyroscope Sensors	3	$3	$8.99
Seeed Technology Co., Ltd 316040001 Vibrational Motor	1	$1.2	$1.2
Sparkfun Li-Po charger	1	$10.50	$10.50
3.7V 2500mAh 105151 Lipo Battery Rechargeable Lithium Polymer ion Battery Pack with JST Connector	1	$4.95	$4.95
Linear Voltage Regulators 1.2-37V Adj Positive 1.5 Amp Output	3	$1.95	$1.95
ATmega328p Microcontroller	3	$2.95	$8.85

RN-42 Arduino Wireless Bluetooth Receiver RF Transceiver Module Serial Port Transmitter Module	1	$27.95	$27.95
Double Sided Medical Grade Adhesive Tape Roll, 1 Inches x 108 Inches-Clear	1	$7.99	$7.99
1 Yard material of Duck Canvas and Felt Canvas	2	$10	$20
Styrofoam board material for boxes	1	$5	$5
Total			**$101.28**

Table 9: Cost Table