**Smart Exercise Assistant**

**ECE 445 Design Document**

Team 66 - William Gammon, Akhil Kandimalla, Platon Slynko

TA: Kyle Michal

1. **Introduction**
   1. **Objective**:

Today, almost everyone is exercising in some way or another but many are uninformed when it comes to the execution of some workout techniques. From 1990 to 2007 there have been more than 970,000 injuries stemming from sports, recreation, and exercise treated within hospital emergency rooms [2]. Without proper guidance from personal trainers on proper techniques, this puts people at a higher risk of injury while exercising.

We plan to implement a smart wearable device to improve exercising techniques. The person wearing this device would exercise as usual with data being gathered in the background utilizing Bluetooth sensors attached to the wrists, arm, and shoulder. These sensors will be collecting data from a built in gyroscope and accelerometer which would then be processed using a central hub located on the torso which will be connected via bluetooth to all the Bluetooth sensors. This data would be sent via bluetooth to an app where it would provide feedback to the user on ways to improve form or accuracy of the given exercise. The goal of the app is to select what workout the user wants to do, and our system will gather data on the user performing the exercise and once it is complete, the program will provide tips on how to improve based on the data gathered.

* 1. **Background**

The problem we are looking to solve is reducing the amount of injuries that occurs to individuals while exercising. Some of the many ways to help prevent injuries while exercising would be to stretch and warm up your body beforehand, to have correct form while doing the exercises, or to not lift heavier weights that cause you to break the correct form. We are looking to solve or improve the form that one would use while exercising to help reduce the amount of exercising injuries. FitBit has a similar concept as to what we are trying to achieve where they are monitoring a person’s physical activity and providing feedback on how one could become healthier as well as tips about eating a more well-rounded diet, and sleeping better. We differ from that in terms that we are looking to improve the quality of the workout that the individual is performing and help prevent injuries by providing feedback to the user. Additionally, our system is supposed to be attached to the body without complex attaching system and without being sewn into workout clothing, unlike many products in the market.

* 1. **High Level Requirements**
* Application must be able to accurately deduce relative limb angles to 3 degrees ( .05 radians).
* System must connect wirelessly to the main hub and be smaller than 6 x 6 x 3 cm
* System must be able to operate for 2 hours on a battery power

1. **Design**
   1. **Principle of operation**

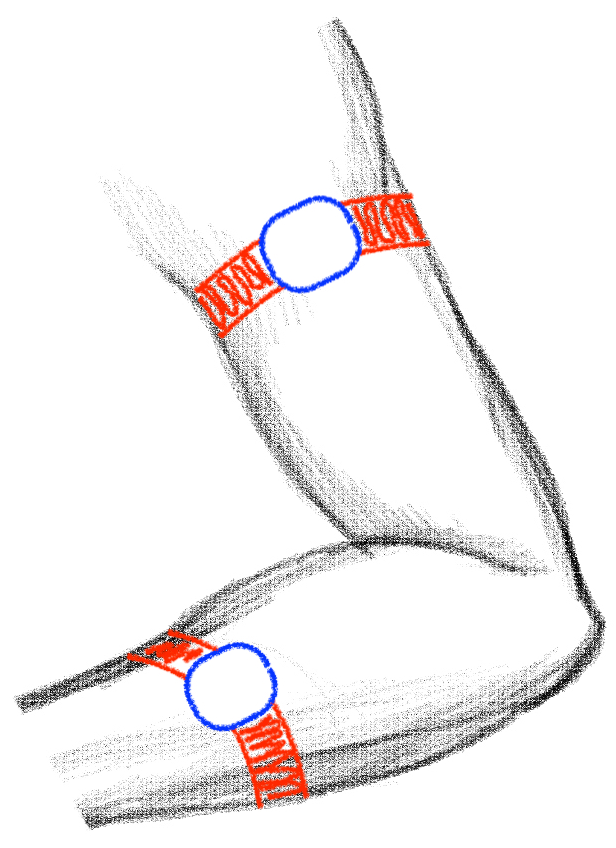
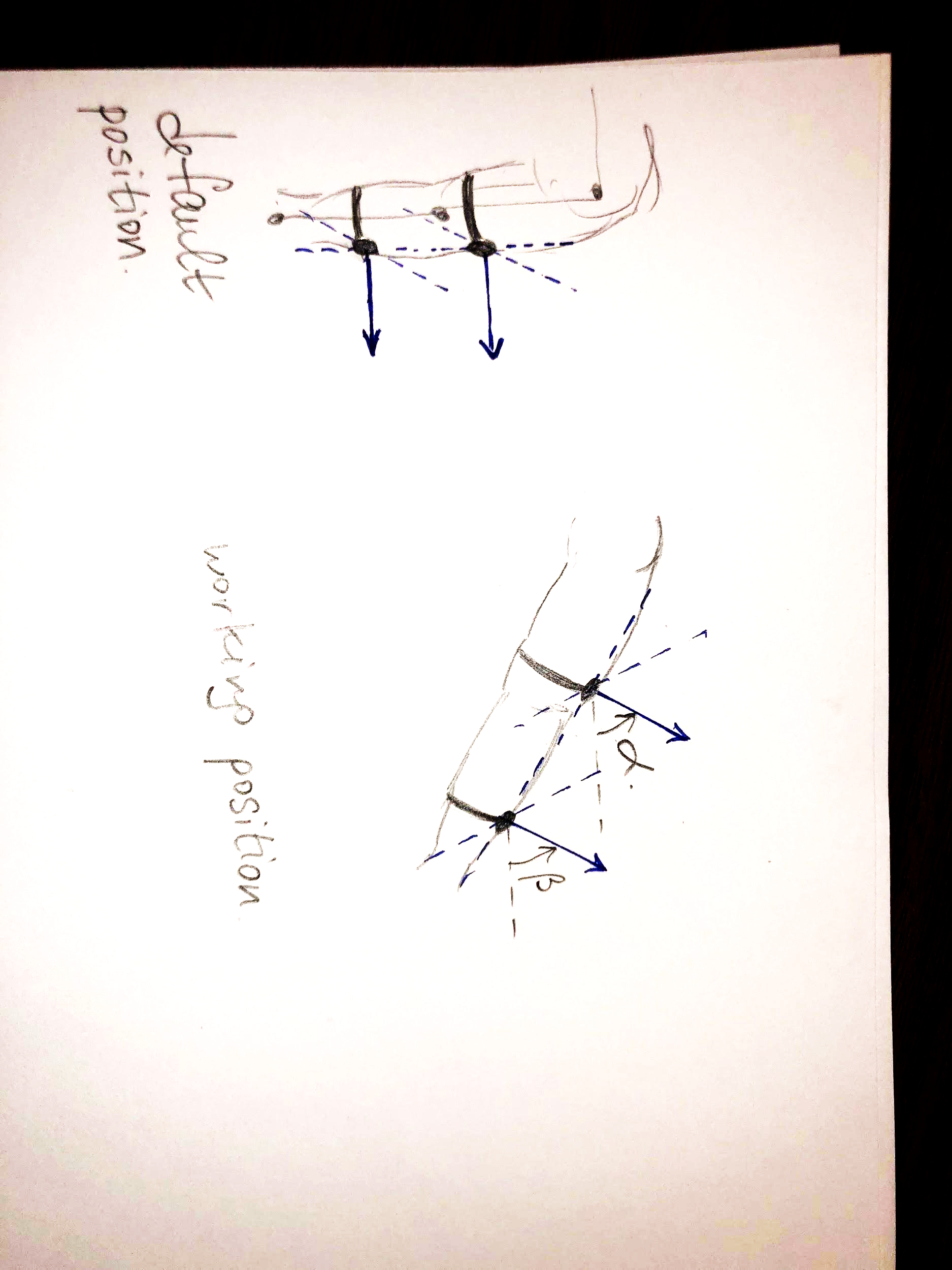
The core idea of this project is to deduce limb angles to generate an output of how the user is moving while exercising. Since limb dimensions are constant and assuming that the user wouldn’t change position of the bands that hold the sensors in place, there is no need to deduce relative position of the limbs or sensors, as they are going to be always constant. The sensor system would be placed according to the picture. As the only sensor data needed are the angles, the exact sensor positions would not matter. As a result, this project is using gyroscopes for its operation.

Figure 3: Sensor placement on an arm

Since sensor placement on the user’s arm would be quite different every new session, the need for calibration arises. At the beginning of each session, the user would be prompted to calibrate the sensors. After pressing the button in the application, the user would be asked to lower their arms for 3 seconds. Since the user would have to lower their arms after holding their smartphone, the computation system would have to wait until a steady state is reached. This means that the system would ignore sensor data for calibration until the change in all of the angles would not exceed 5 degrees or 0.09 radians for at least 0.5 of a second. After that, the system would record all angles for a second and would find an average angles, which would be labeled “default” and would be used as a reference for any further angle change.

If this process is interrupted by angle values changing for more than 3 degrees, the error message would be displayed and the user would be promoted to try calibration again. In case the user adjusts the bands, he/she can recalibrate again to keep system measurements correct.

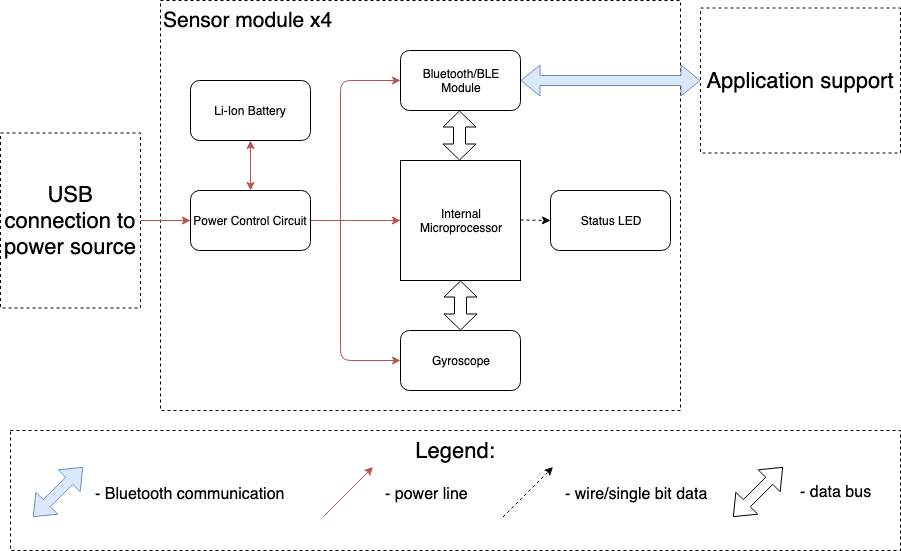
Afterwards, to get the actual angles, default corresponding angles would be subtracted from all of the measured angles.



**Resting Position Working Position**

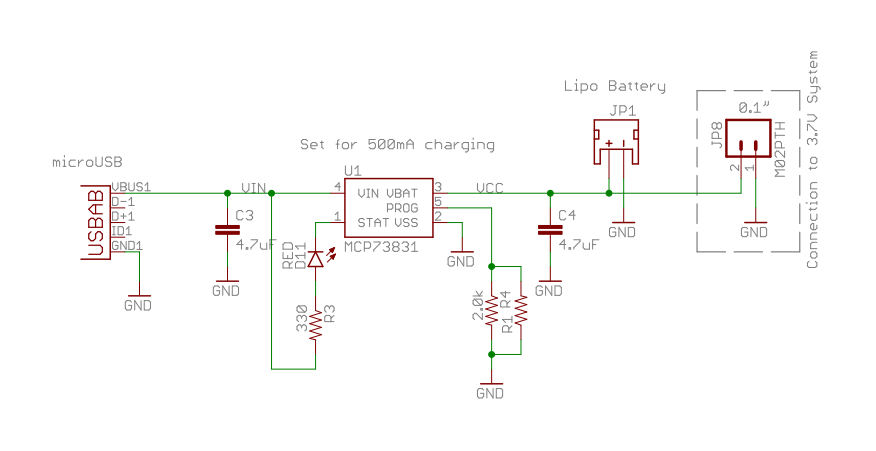
**Figure 5: Diagram of angle measurements**

* 1. **Block Diagram**

****

**Figure 1: System Diagram**

* 1. **Sensor Module**
  2. Each sensor module consists of an internal microprocessor, gyroscope that collects data, bluetooth module that communicates with an application and a power control circuit that powers the module. Sensor module is responsible for collecting data from the gyroscope and sending it via Bluetooth to an application.
     1. **Power Control Circuit**

**Figure 2. Power Control Schematic [3]**

Power control circuit controls the state of the sensor modules, whether they are “on” or “off”. Additionally, it allows for the connected Li-Ion battery to be charged while the circuit is in “off” mode. Be sure not to overcharge Li-Ion batteries to reduce battery failure.

|  |  |
| --- | --- |
| Requirement | Verification |
| 1. Must allow battery to be charged while circuit is “off” without affecting the rest of the circuit. 2. Must properly charge Li-Ion batteries supplying power to all Bluetooth sensors and Main Module and not exceed 50**°**C threshold rating. | 1.Test voltage at nodes of inertial sensor while charging battery to ensure 0V +/- 0.001V power drainage.  2. While charging, verify temperature of batteries do not exceed threshold rating (50**°**C). |

* + 1. **Li-Ion Battery**

These small batteries must power all of the components within our Bluetooth Gyroscope PCB.

|  |  |
| --- | --- |
| Requirement | Verification |
| Must supply between 3.3-5V and store 300mAh | Can verify from spec sheet. |

* + 1. **Internal Microprocessor**

The microcontroller, chosen ESP32, is responsible for sending data to the app through Bluetooth module and processing of the received sensor data from the gyroscope. Since our project is going to be data-heavy for microcontroller, we chose this device for its sizable memory of 1MB flash, which would provide about 60 seconds of data recording.

|  |  |
| --- | --- |
| Requirement | Verification |
| The microcontroller must be able to receive, process and send data of at least 20 kilo-bytes per second. | Assemble on PCB as specified in the datasheet. Verify on datasheet for microcontroller. |

* + 1. **Bluetooth Module**

Allows communication with the application.

|  |  |
| --- | --- |
| Requirement | Verification |
| Must support BLE and Bluetooth 4.0 | Assemble on PCB as specified in the datasheet. Verify on datasheet for Bluetooth module. |

* + 1. **Status LED**

The green LED light would provide information about main module state. It would start blinking when main module is turned on. It would turn solid color when connection is successful between the Main Module and all Bluetooth Inertial sensors.

|  |  |
| --- | --- |
| Requirement | Verification |
| Status LED must stay solid green when successfully connected to module, and be visible from 2m away. | Connect to Inertial sensors and view status LED from 2m away. |

* 1. **Android Application**

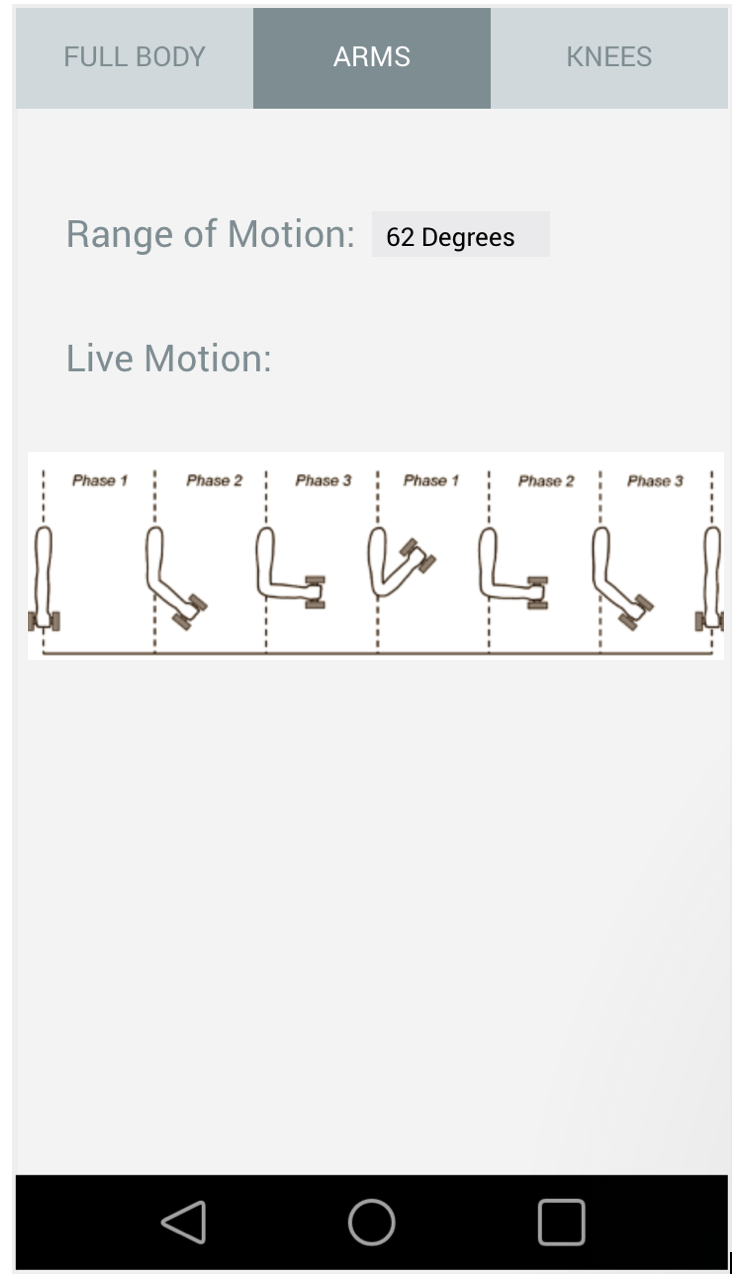
This application would be the main interface for the user. The application would allow the user to choose the exercise that the user wishes to perform. This would help the application in understanding the orientation of the sensors that are placed on the body.

The sensor modules on the body would send data to the application at a rate of

About 100 Hz. The application would then take the average of the incoming data

Every 500 milliseconds and display the position of the selected limbs in a timeline

As shown in the application prototype below.

[7]

|  |  |
| --- | --- |
| Requirement | Verification |
| Must successfully display the limb angles to within 3 degrees ( .05 radians) | Verify accuracy of calculations by having actual angle measurements and testing from results on the application at angles 45, 90 and 180 degrees. |

* 1. **Tolerance Analysis**

A critical requirement within our design that would be the accuracy of the data gathered from our gyroscope sensors. According to the datasheet for our ITG-1010 gyro, the data has a ±0.1 error due to noise [4]. Over time we see that this may cause some issues in calculating the correct angle between both gyroscopes. In order to find the correct angle of the gyroscope, we must first initialize it to some base reference point From here we must integrate with respect to time and add this to our original angle The equation as follows is:

This equation would be excellent for a single calculation, but we are continuously sending data and in order to keep up with the user we would need to take a different approach [5]. Since we cannot take a continuous integral, we will use the approximation of:

Where is a finite number of samples taken at a given interval. We know that over time we will have a ±0.1°/s drift. So our new equation to account for the maximum drift would more closely resemble:

Where From here we can see that our gyroscopes will begin to drift over time, and if we want to keep our accuracy to within 3 degrees, then we would account for the maximum error for both of the sensors in either arm due to this value [6]. If our is set to 0° for both sensors, then we can gather the respective values. In order to find the difference in angles we are simply going to subtract, since we have a straight point of reference. From there we can find the difference between the errors that are calculated from the ideal angle location:

;

;

* 1. **Risk Analysis**

Some of the factors that could hamper with the successful presentation of our project include:

* The sensor module being oversized to be on a person’s body while exercising.

This could be caused by the battery being too big.

* Bluetooth connectivity being unstable, that would prevent the main module from receiving data on a frequent basis. This problem could potentially prevent the microprocessor from being able to differentiate among good posture, incorrect posture and injurious posture due to inaccuracies in data.
* Since this is a wearable device, it would mean that the battery life of the sensor modules and main module is long enough to be used over a period of at least one month, for about one hour per day.

1. **Schedule**

|  |  |  |  |
| --- | --- | --- | --- |
| **Week** | **William** | **Platon** | **Akhil** |
| 2/18 | Begin designing overall schematic of main module | Begin programming of internal microprocessor | Begin programming User Interface |
| 2/25 | Implement main component design for schematic | Continue program. Successfully collect data from BLE sensors via Bluetooth | Continue programming User Interface, successfully collect bluetooth data |
| 3/4 | Finalize schematic of main component | Begin collecting samples of testing data used in implementing position tracking algorithm | Begin programming position tracking algorithm |
| 3/11 | Revise and design PCB on Eagle | Continue collecting data and order other inertial sensors | Continue programming position tracking algorithm with test samples collected |
| 3/18 | Finish and order PCB design. | Find case and band to secure battery and sensors. | Debug position tracking algorithm |
| 3/25 | Implement sensors and module attachment to body | Implement Bluetooth connectivity between sensors and application | Continue programming position tracking algorithm consult with implementation from main module |
| 4/1 | Begin Soldering PCB sensors together | Debug Bluetooth connectivity and main module | Improve on User Interface on app debug for main module connection |
| 4/8 | Continue soldering PCB sensors together | Solder PCB sensors | Begin testing Bluetooth data between main module and app |
| 4/15 | Collect data, debug issues with any connectivity | Continue bug fixes on Bluetooth connection and main module | Debug and improve position tracking algorithm with main module connection |
| 4/22 | Conduct testing for demo | Conduct testing for demo | Finalize User Interface and Bluetooth connection to main module |
| 4/29 | Prepare for final presentation | Prepare for final presentation | Prepare for final presentation |

**3.1 Cost Analysis**

Our fixed development costs are estimated at $120/hour, 10 hours/week, for three people. Assuming that data processing and UX/UI design parts would be 20% of the total final product:

Since the system has significant amount of parts of the same type, this table will list individual part retail price, required number of parts and then prototype system price with retail pricing. The last column will show commercial system price with wholesale pricing.

|  |  |  |  |
| --- | --- | --- | --- |
| **Part** | **Cost (prototype)** | **Number of parts** | **Total prototype cost** |
| MCP73831-2 Battery management unit | $0.60 | 4 | $2.40 |
| AP2112 Voltage Regulator | $0.36 | 4 | $1.44 |
| CP2104 USB to UART | $1.65 | 4 | $6.60 |
| ESP 32 Module | $8.95 | 4 | $22.00 |
| 3.7V Lithium-Ion Battery Rechargeable (Secondary) 400Amh | $6.95 | 4 | $27.8 |
| PCB | $3.00 | 4 | $12.00 |
| Assorted resistors, capacitors, crystals | $5 | 4 | $20 |
| TOTAL | $92.24 | | |

Since some of the parts used in this project are combined modules, there is a significant price increase on these; notably, ESP32 integrated module. Scaling up the production for this product would mean that we need to buy Bluetooth modules, antennas, and smaller parts like resistors, capacitors and such in bulk. Additionally, batteries might need to be assembled in local facilities instead of buying batteries with protective circuits together. Afterwards, we would need to develop custom PCBs and assembly full product in local facilities.

1. **Ethics and Safety**

Some of the safety concerns we had about this project first started with the Lithium-Ion batteries with the potential exploding [1], considering we would have so many placed on the user. To fix this we will be implementing a separate charging feature where the batteries will not be charging while the user is wearing them. This still brings into the question of them overheating while being on a person who is exercising. To improve this we will separate the battery from the user and circuitry. There is also the issue that the person performing exercises with our devices will be sweating. On top of this, the user would be performing physical exercises and may cause damage to themselves or to the circuits while working out. In order to fix this we will encase our sensors to provide protection from potentially circuit damaging moisture and damage. We are responsible for warning the public of our potential dangers following IEEE Code of Ethics #1 - To hold the public safety first and to disclose factors of our project that might endanger the public. This would also be an example of where we tried to align with the IEEE Code of Ethics #9 -To avoid injuring others through malicious intent. In that there may be a risk and to notify the user and provide protective separation of the battery and main circuit.

We plan to have our design work with anyone to can place the sensors on their arms. This is one implementation of the IEEE Code of Ethics #8 -To treat everyone fairly and not to discriminate. In allowing our device to be used almost universally we see this as something that would help people in their pursuit to exercise. We will also be processing data based on the provided data gathered from the BLE sensors, and in doing so aligning with the IEEE Code of Ethics #3 - To be honest and using the data to provide claims. We will strictly be gathering data from the user, and not generate answers based on false reasoning.

**References**

[1] Consumer Reports, ‘Why Lithium-Ion Batteries Still Explode, and What’s Being Done to Fix the Problem ‘ 2016. [Online]. Available:

‘<https://www.consumerreports.org/safety-recalls/why-lithium-ion-batteries-still-explode-and-whats-being-done-to-fix-the-problem/> [Accessed: 7-Feb-2019].

[2] Nationwide Children’s, ‘New National Study Examines Weight Training-Related Injuries’ 2010. [Online]. Available:

<https://www.nationwidechildrens.org/newsroom/news-releases/2010/03/new-national-study-examines-weight-training-related-injuries> [Accessed: 21-Feb-2019].

[3] SparkFun, ‘LiPo Charger Basic - Micro USB’ 2017. [Online]. Available: <https://www.sparkfun.com/products/10217> [Accessed 21-Feb-2019].

[4] InvenSense, ‘ITG-1010 Product Specifications’ 2015. [Online]. Available: <https://www.invensense.com/wp-content/uploads/2015/07/PS-ITG-1010A-00-v1.1.pdf> [Accessed 7-Mar-2019].

[5] Pieter-Jan, ‘Getting the angular position from Gyroscope Data’ 2012. [Online]. Available: <http://www.pieter-jan.com/node/7> [Accessed 7-Mar-2019].

[6] AnalogDevices, ‘Gyro Mechanical Performance’ 2019. [Online] Available:

<https://www.analog.com/en/technical-articles/gyro-mechanical-performance.html>

[Accessed 7-Mar-2019].

[7] fluid, ‘Create Web and Mobile Prototypes in Minutes’ 2011. [Online] Available: <https://www.fluidui.com/> [Accessed 7-Mar-2019].

[8] Espressif, ‘ESP32 Series Datasheet’ 2019. [Online] Available: <https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>

[Accessed 7-Mar-2019].

[9] Silicon Labs, ‘Single Chip USB to UART Bridge’ 2017. [Online] Available: <https://www.silabs.com/documents/public/data-sheets/cp2104.pdf> [Accessed Mar-7-2019]