Ankle Injury Prevention

ECE 445 Individual Progress Report

Group 14

Skyler Shi (jingtao2)

Teammates: Matt Miller (mjm12) & Erin Sarver (esarver2)

TA: Sowji Akshintala

Fall 2020

1. Introduction

1.1 Team Project Overview

The purpose of our project is to build a basketball shoe that can successfully monitor a user's ankle usage. The shoe should inform the user after each game played how much stress they put their ankles under so that they can adjust their physical activity amount accordingly to reduce injury risk.

Ankle ligament sprain or tear is the most common injury for basketball players across all levels of play[1][2]. Hence, we want to build a product that reduces player risk for this injury.

1.2 Individual Responsibilities

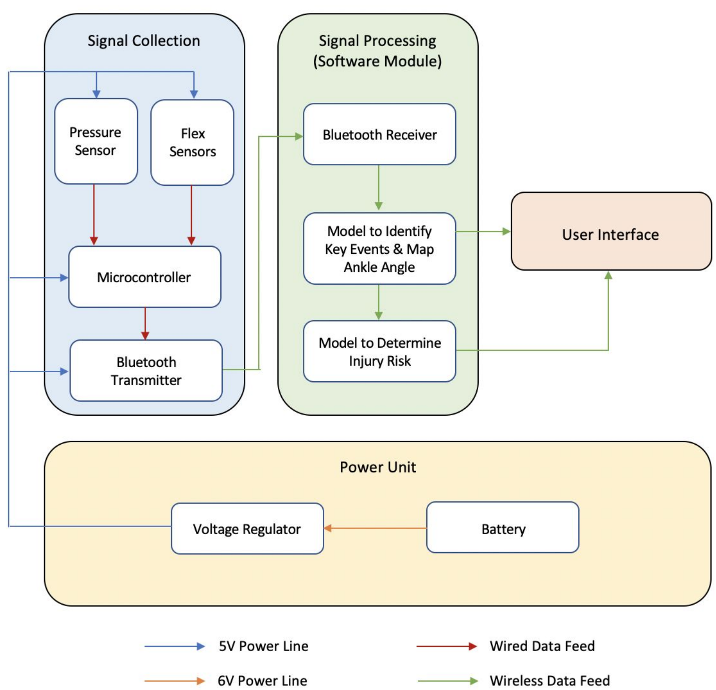


Figure 1: Functional Block Diagram of Ankle Injury Prevention Device

Our team designed our product so that all sensor data is sent to the user's laptop and processed and displayed there. We designed it this way because the user needs a sizable screen to view and understand the data collected. Adding a LCD display to the hardware product on our user’s shoe would add unnecessary weight to their shoe.

This is denoted in the block diagram of our product where the blue “Signal Collection” unit sends sensor data to the green “Signal Processing” unit.

I am personally responsible for coding out the entire green “Signal Processing” unit and for creating the red “User Interface” block. My teammates are responsible for the blue “Signal Collection” and yellow “Power Unit” blocks.

I have helped work on creating the physical design of the shoe, discussing with teammates whether it is better to mount the flex sensors directly onto the shoe or onto a soft sleeve.

2. Individual Design Work

2.1 Signal Processing Data Flow Diagram

I created a Data Flow diagram to help myself better understand what data outputs each software module will produce and how they will interact with each other.

The flex sensor data is preprocessed accordingly. The pressure sensor data is preprocessed accordingly. I then use preprocessed flex sensor data to determine 3 core metrics: Total Ankle Stress, Average Ankle Stress, and Ankle Angle Variance.

After determining timestamps of key events such as Jump, Land, Change-of-Direction, I calculate the 3 core-metrics again during these high-impact events to help the user understand their movement profile better during high-risk events.

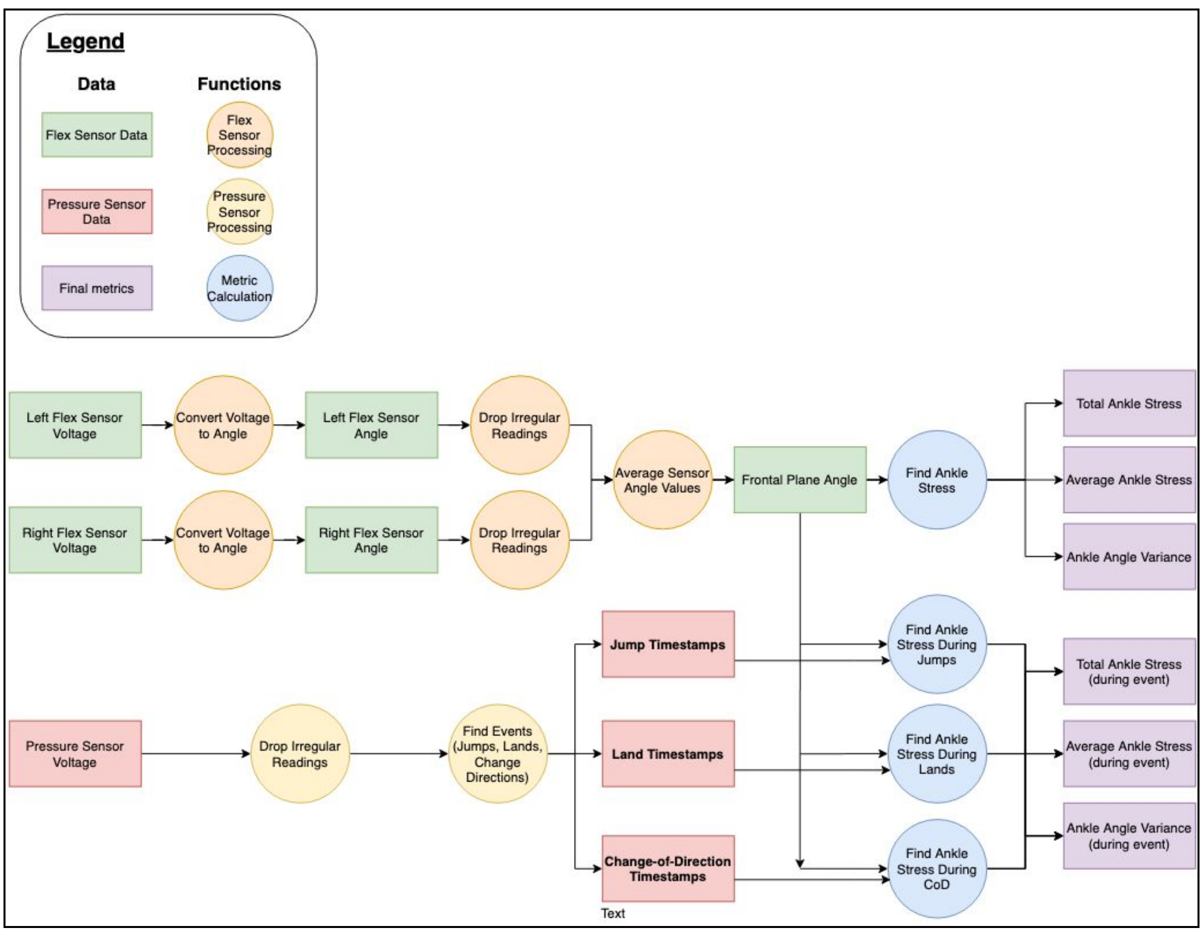


Figure 2: Data Flow Diagram of Signal Processing Module

2.2 Signal Processing Design

Here are the design considerations I made for finishing the software work:

2.2.1 Pressure Sensor Processing

I created a software module to process pressure sensor data.

First, the software module drops all faulty readings. I consider any reading outside of our voltage sampling range to be faulty. In the following weeks, I will communicate with my teammates to determine the correct bounds for this range.

When faulty values are dropped, there are then missing values in our pressure sensor readings. I will interpolate these missing values by either using a window-averaging approach or by re-sampling the signal. Which approach is selected depends on which approach works better on our real-world data. This will be determined when we can read real-world data.

Now that the pressure sensor data is preprocessed, the software module works to extract timestamps of key events. The key-events we care about are all events that occur when the user applies a larger-than-usual amount of pressure to their foot. Hence, I extracted peaks in pressure sensor data. Below is a diagram of my test results on randomly generated data. Real-world data should be noisier than the data I generated below, so I will have to do a lot more testing and tuning when I receive real-world data.

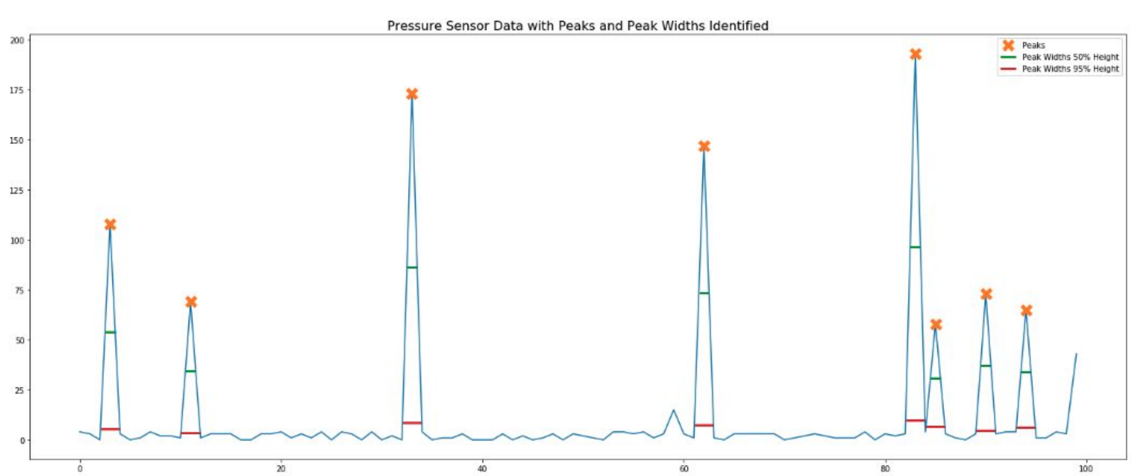


Figure 3: Current Test Results of Finding Events using Randomly Generated Data

Right now, the module finds peaks and uses their widths to determine the start and end time-stamps of events. When we have real-data, I can use the height of the peaks, the relative distance between peaks, the width of the peaks and other information to discern between jumps, lands and change-of-directions. Attending office hours, a TA told me that the best way to discern between these events is to observe the difference in pressure sensor patterns using real data. I cannot make a confident guess on how these events will make the pressure sensor data behave differently as of now.

2.2.2 Flex Sensor Processing

Whereas I can directly work with Pressure Sensor voltage data since we do not care about specific PSI readings, I need to convert Flex Sensor voltage readings into angle measurements. We do care about the specific angle readings.

To do so, I use an equation my teammate Erin derived in our Design Document:

$$\theta^\circ = \frac{\frac{V_O R_1}{V_{CC} - V_O} - R_{min}}{R_{max} - R_{min}} * 90^\circ$$

I have created variables for each of the constants shown in the equation above. Once my teammates ascertain the specific values of each constant, I just have to assign the values to my variables and I can get the angle.

Likewise to pressure sensor processing, I also drop faulty values determined by the bounds of the flex sensor [0° , 90°]. I also drop values when the two flex sensors disagree in value by more than 40%. This threshold can be tuned later when I have real-world data.

Again, to interpolate the missing values, I will use either a window-averaging approach or a signal-resampling approach. Which approach exactly is used depends on their effectiveness on real-world data.

2.2.3 Metric Calculation

I created a module/function for calculating all three metrics we care about: total ankle stress, average ankle stress, ankle angle variance. As can be seen from the Data Flow Diagram, this function is used multiple times - once for calculating the three metrics across all time, and multiple times for calculating the three metrics across all jump durations, land durations and change-of-direction durations.

The function needs two inputs: a sequence of flex sensor angle data, a list of timestamps of durations we care about. The function will use the timestamps to select the sub-sequences of flex sensor angle data and perform calculations on those durations.

To calculate total ankle stress, we find the total time during which a user's ankle exceeds their range of motion threshold. This can be seen as finding the area between two curves - the area between the blue curve and the red curve and the area between the blue curve and yellow curve.

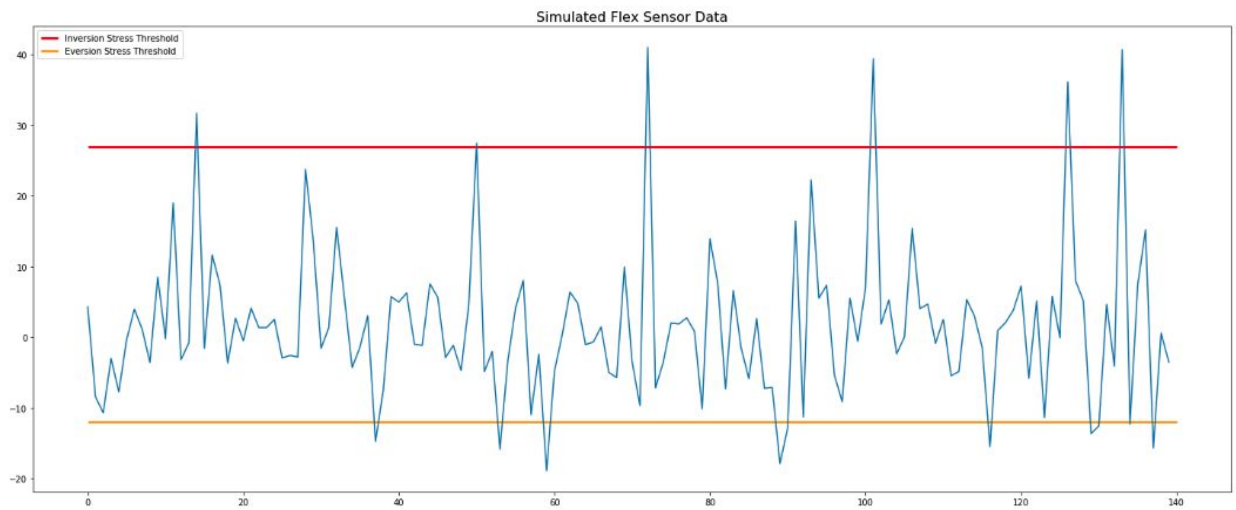


Figure 4: Current Test Results of Calculating Ankle Stress using Randomly Generated Data

Average ankle stress is just total ankle stress divided by the total number of minutes a player plays. The total number of minutes a player plays can be derived from the length of the sequence and the sampling rate.

Ankle Angle variance is just the variance of all flex sensor angle values throughout a defined duration.

2.3 Risk of Injury Model

I want to help users determine their risk of injury by running a machine learning classification model on their collected data.

After one session of using the device, one user will generate one row of data that looks like this:

User_id	User Weight	User Height	Average Ankle Stress	Ankle Angle Variance	Average Ankle Stress (Jump)	Ankle Angle Variance (Jump)	Average Ankle Stress (Land)	Ankle Angle Variance (Land)	Average Ankle Stress (Change Direction)	Ankle Angle Variance (Change Direction)
12	78 kg	185 cm	0.56	3	0.89	2	0.99	7	0.34	5

Users can also self-declare what they think their injury risk is by recalling their injury history. Injury risk can perhaps be more quantitatively defined as "Number of injuries per 1000 minutes played". Hence, we can have one row of data like this:

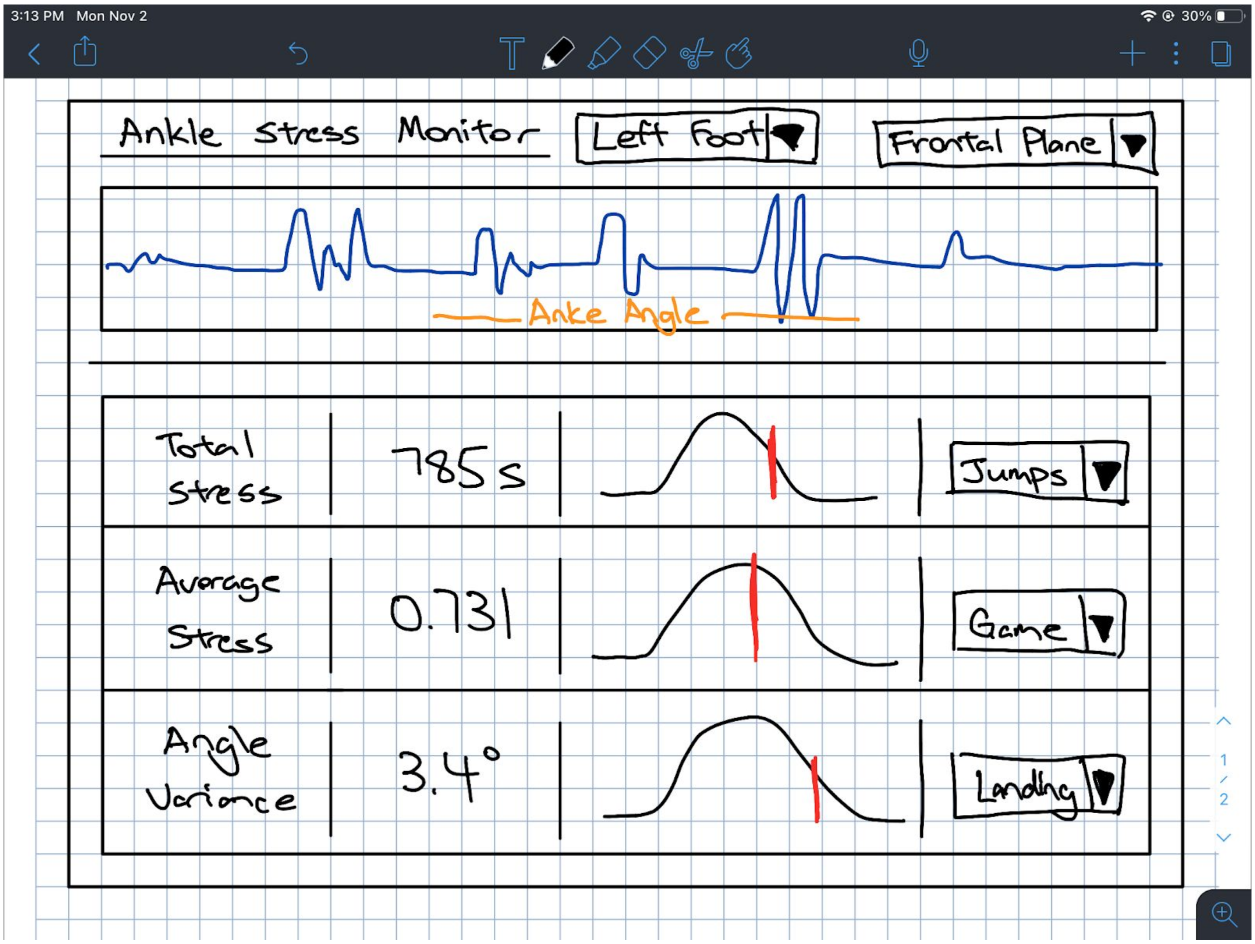
User_id	User Weight	User Height	**User Injury Risk**	Average Ankle Stress	Ankle Angle Variance	Average Ankle Stress (Jump)	Ankle Angle Variance (Jump)	Average Ankle Stress (Land)	Ankle Angle Variance (Land)	Average Ankle Stress (Change Direction)	Ankle Angle Variance (Change Direction)
12	78 kg	185 cm	**0.78**	0.56	3	0.89	2	0.99	7	0.34	5

When I have a lot of rows of this kind of data, I can run a machine learning regression model to determine the injury risk of new users.

However, I realize this machine learning model cannot be implemented for the demo since I would have to collect a lot of data from a lot of different users. This is simply not possible. Hence, for the purposes of the demo, this module will not be implemented due to infeasibility. I cannot make a sound recommendation of injury risk to the user without collecting a lot of data first. It would be unethical to do so.

2.4 User Interface

The user interface is a very straightforward software module. After some sketching, I determined I want the user interface to look like this to reflect all the important metrics to the user:



I can build the entire user-interface in python with dash by plotly.

2.5 Testing & Verification

Most of our project's testing and verification requirements happen on the hardware side. For software, the most important thing that I need to test for is whether the metrics calculated align with our understanding of the physical world.

To test for this, I will use the R&V specified in our design document. The basic idea is to have a user bend their ankles to different thresholds and observe how the metric changes in software.

I also need to test if all data sent by the microcontroller is received by the laptop for its bluetooth capabilities. To test for this, I will send data for a controlled amount of seconds. I will calculate the theoretical number of samples I should receive and verify it with the actual number of samples I received.

Both of these verifications can be done when we collect real data.

3. Conclusion

3.1 Self-Assessment

I think I have taken responsibility for a good amount of work on our project. My teammates are more proficient in hardware and I am more proficient in software. Hence, we divided the workload between us using hardware and software.

I am slightly behind schedule. I have:

1) Completed software logic to drop irregular data for flex/pressure sensors
2) Completed software module to find events - jump, land, change-of-direction.
3) Implemented mini SQLite database to store metrics for demo purposes

I still need to:

1) Implement method to interpolate missing data based on real-world data
2) Finish function to calculate metrics
3) Implement user interface
4) Tune pressure-sensor processing module to discern between events
5) Implement bluetooth module to collect data from microcontroller

My signal processing tasks do depend on the successful completion of hardware and signal collection. I cannot really process the data if I do not know what it looks like. To finish within the next two weeks, we need to finish the hardware R&V and receive the shoe from the machine shop as soon as possible so we can begin collecting real data.

3.2 Remaining Schedule

Week	Objectives
11/1 - 11/7	● Completely finish User Interface (this can be done using dummy data) ● Weekend: work with team to collect

	data from sensors and receive via bluetooth
11/8 - 11/15	• Implement method to interpolate missing data based on real-world data • Tune pressure-sensor processing module to discern between events • Finish function to calculate metrics
11/15 - 11/18	• Do whatever is necessary for the demo to work well!

Citations

[1] Drakos, M., MD, “Injury in the National Basketball Association”, *National Center for Biotechnology Information*, 2010. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3445097/. [Accessed: 01- Sep- 2020].

[2] Hamilton, D., MD, “Ankle, Foot Injuries Most Common for Basketball”, *West Tennessee Bone & Joint*, N.D. [Online]. Available: https://wtbjc.com/our-practice/newsletter-stories/ankle/ankle-foot-injuries-bball.php. [Accessed: 01- Sep- 2020].