Project Deliverable F

GNG 2101

Team E1.1

Jacob Hepburn

Isaac Loge

Jordan DeRose

Evan Trainor

Griffin Richard

Presented To:

TA: Aradhya Agnihotri

Professor: Jason Foster

Table of Contents

Design Constraints

Non-Functional Design Constraints

Two non-functional design requirements are budget and aesthetics.

Budget:

Since we have a limited budget of $100 we cannot buy new and perfect components such as motors, gears, chains, electrical components, etc. Instead, we will need to salvage some components that may not be ideal for the job. For example, instead of purchasing new gears that will be the perfect size we will be using old gears taken from the freewheel of a bike.

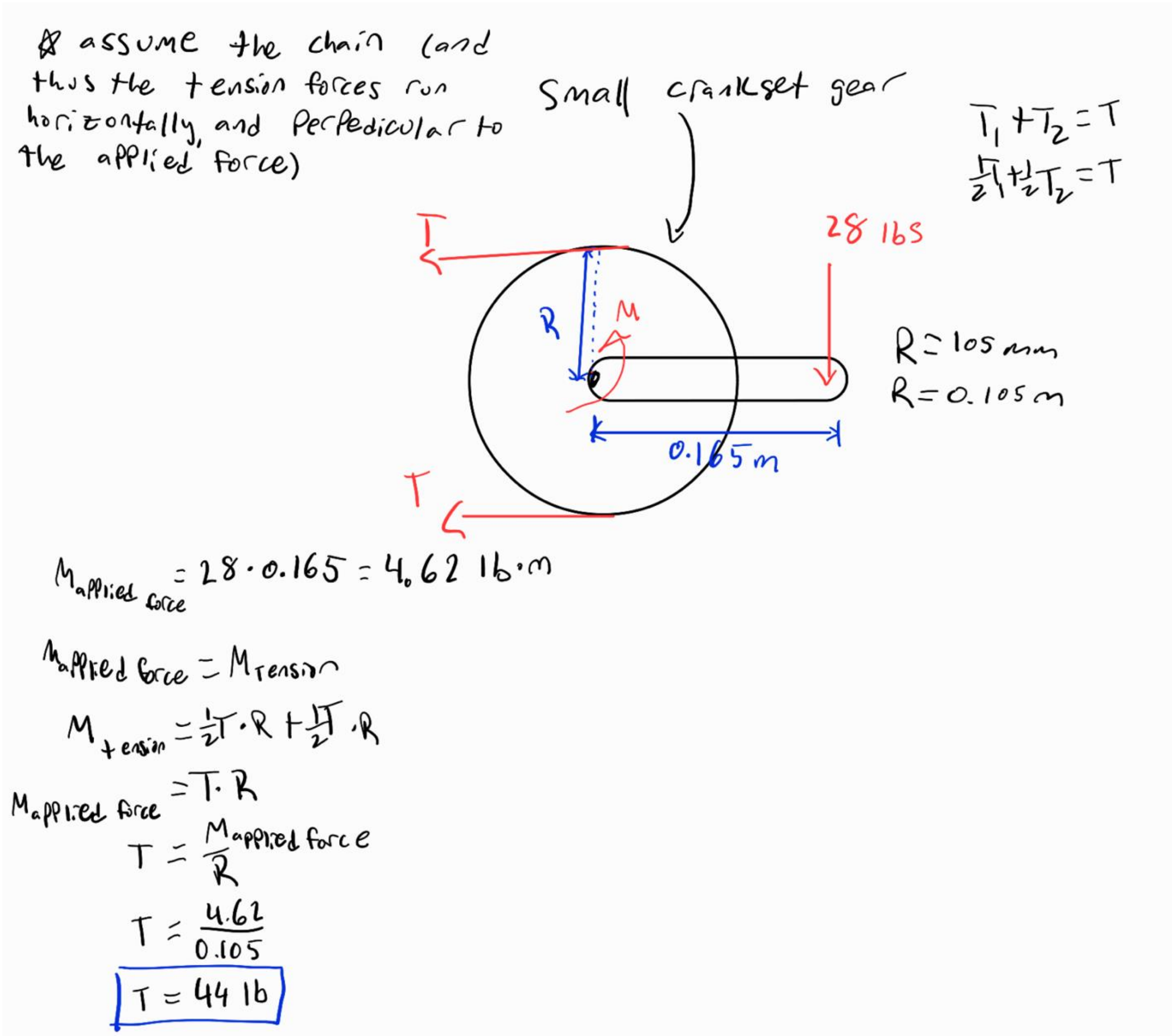
Aesthetics:

It was made abundantly clear to us in the second client meeting that the window opener must be aesthetically pleasing. Our original idea was to have an arm that would extend and bend in order to open and close the window. We felt that this would be the most robust and simple design design however the client didn't like how it looked leading to our current chain driven design. However, another aesthetic change will be made to the design which will be to extent the box that the mechanism sits in so that it spans the whole window.

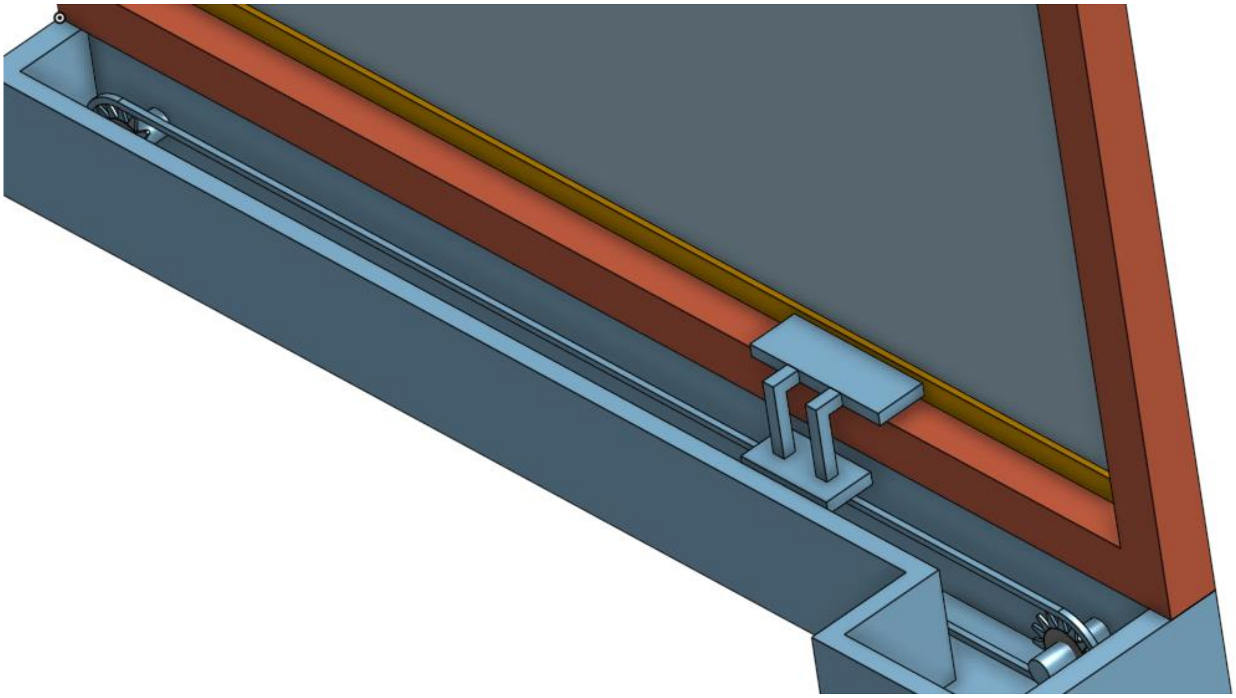
Proof of Concept and Updated Design

Using secondhand bike parts will be a viable solution to reduce the cost based of the following information. A cyclist on Quora named Branko Dodig shared his cycling data. With his data he calculated his force output during an endurance ride to be 28lbs[1]. This, however, is not equivalent it the maximum force a bike can tolerate. He produced this force resulting in an average power output of 171 watts, but professional cyclists are capable of averaging over 400 watts of power[2].

A free body diagram was created in order to calculate the tension on the chain when someone is applying 28lbs of force to the pedal. The radius of the smallest gear of a crankset is about 150 mm (using the smallest gear results in the greatest tension force on the chain). From there, the moment about the center of the freewheel was calculated. With the moment an estimation for the chain tension can be made. With 28lbs of force applied it is estimated that 44lbs of tension is in the chain and it is highly probable that the bike components can withstand much higher forces. Also, from manually opening and closing the window it is safe to say that it does not take 44lbs of force to do so. This is not a perfect calculation, however it does tell us that the bike components will be sufficient for opening and closing the window.



We created a simple CAD model of our design and showed it to the client. She thought the design looked good and gave us the okay in terms of aesthetics.



Prototype 2

Client Feedback

From the Client meeting 2, we got feedback about both the software and hardware design. For the software design the client agreed with us that our best presented option was the Bluetooth app option. After learning this, we made the decision to develop the User Interface for Prototype 1. Our feedback from the client was that the preferred design we chose was not desirable due to aesthetics. Upon learning this we changed the design to a design that is under the window and in a concealed form factor.

Untested Product Assumptions

We decided that the largest untested feature was the connectivity between our app and the device. Additionally, we wanted to test whether our user interface would be simple enough for an average user to use without training. Another untested assumption is that the ability of the system to hold onto the window.

Prototype 2 design plan

For this deliverable, our plan is to build a prototype that consists of two components. The first component is an app that enables a connection to the second prototype, which is an Arduino that will operate as the controller for the device. The goal was to be able to communicate a manual control, to either open or close the window; communicate a schedule, to open and/or close the window at specific times; and communicate a temperature limit, to set a temperature limit where the window will automatically close. The goal of this prototype is to test the assumption that we will be able to communicate using Bluetooth Low Energy between an iPhone and an Arduino, and to test the assumption that the user interface would be simple enough for an average user to use without training.

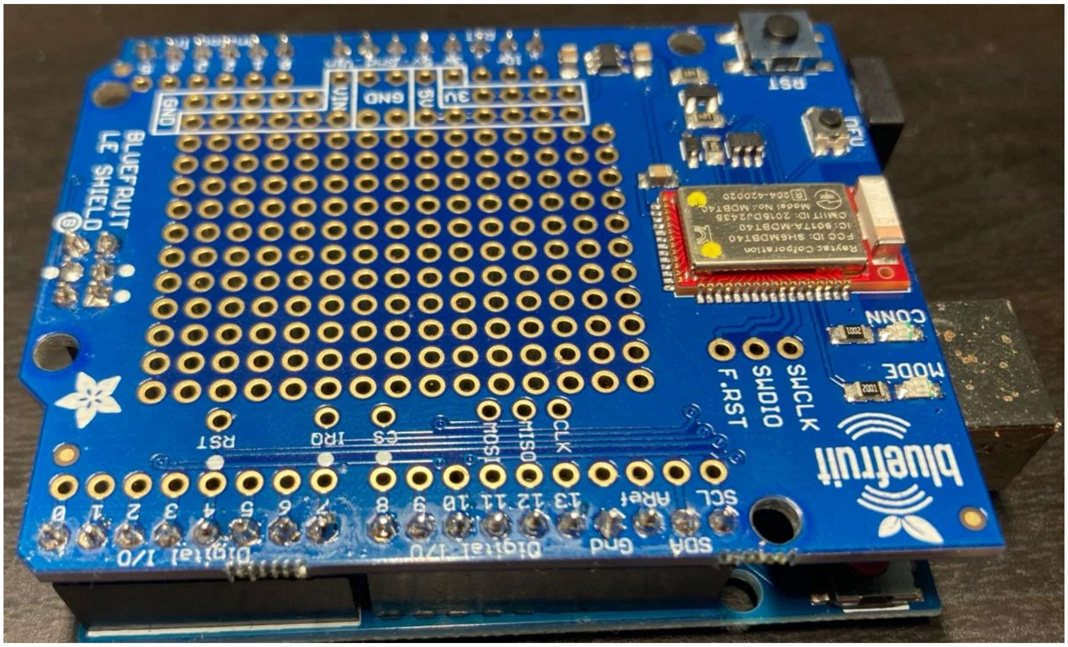
Prototype 2 construction

For the Arduino portion of the prototype, the decision was made to use the Adafruit BlueFruit Bluetooth LE Shield (henceforth referred to as the Bluefruit). The Bluefruit was chosen since it satisfied the requirements to allow for Bluetooth LE connection with a phone, it was available through Makerstore, and the shield format allows us to easily mount components for future prototypes.

The connection of the Bluefruit to the Arduino was done through soldering the pin headers to the Bluefruit, and the pins can then be connected into the Arduino. This setup is shown in Figure 1.

Figure 1

Connection of the Bluefruit to the Arduino



For the software section we began by taking advantage of the open-source example provided by Adafruit (the creators of the Bluetooth module). The example [1] provides a tutorial on how to create a simple chat app and Arduino program that allows for communication back and forth. While the UI was completely different and the Arduino program needed to incorporate many additional features, this tutorial provided an example on how to create a connection between the devices. Working from here the first task was to code the UI for the app which was based on the prototype from Deliverable E. The backend was the based on the Adafruit tutorial [3], which was modified to only send specific commands to the device and interpret the commands from the Arduino. The Arduino code was then written to send and interpret communications from the app.

Testing of prototype 2

The testing of connectivity was very simple as it either worked or it didn't. The setup for connectivity was uploading the code onto my iPhone directly from XCode, and testing connectivity with the Arduino. Software has the advantage of rapid iteration, which allows for frequent attempts at connectivity. This was fortunate, as the first several hundred iterations failed. The largest issue faced was that the code that tried sending signals was calling on a different version of the Bluetooth Central Manager variable, which made the Arduino believe it was connected but all signals weren't received. The results of the test were eventually that it worked and satisfied all goals.

In order to test the connectivity distance, I got my roommate to stand on a street corner and I walked along the street until the device disconnected. I then walked back towards him until it was able to regain connectivity. The distance, 74.88m, was then measured on google maps using the measure feature. While this distance is less than the 100m maximum Bluetooth Low Energy offers, it is more than adequate for the application.

Client meeting 2 presentation

For the client meeting 2 presentations we are planning to show the client the finished product of the app, and the concept for the physical model of the mechanical component that will open the window. For things that we would like to come from this meeting is the client's thoughts on the app and to see what things to be changed to optimize the user experience within the app, as well as to see what the client thoughts where on the new design of the mechanical mechanism and see if they had any feedback on the concept.

References:

[1] Dodig, B. (n.d.). Re: *How much force do you apply on a bicycle pedal?* [Comment on the forum question *How much force do you apply on a bicycle pedal?*]. Quora. https://www.quora.com/How-much-maximum-pedal-force-would-the-leg-of-an-average-person-generate-while-cycling

[2] De Vroet, M. (2023, October 11). *Just how good are male pro road cyclists?*. Velo. https://velo.outsideonline.com/road/road-training/just-how-good-are-male-pro-road-cyclists/

[3] Beaton, T. (n.d.). *Build a bluetooth app using swift 5*. Adafruit Learning System. https://learn.adafruit.com/build-a-bluetooth-app-using-swift-5/overview