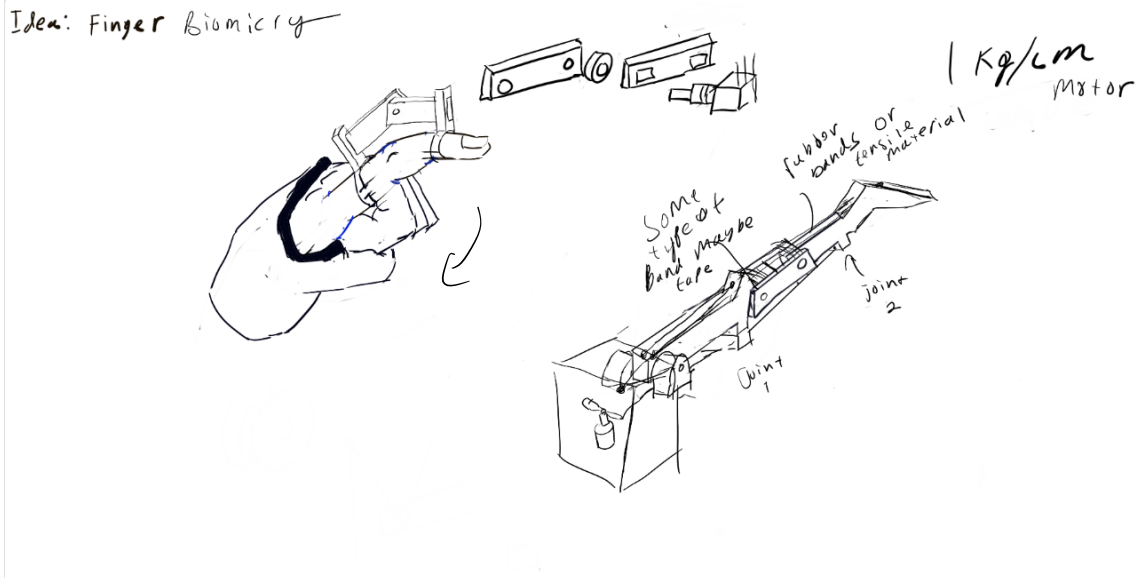
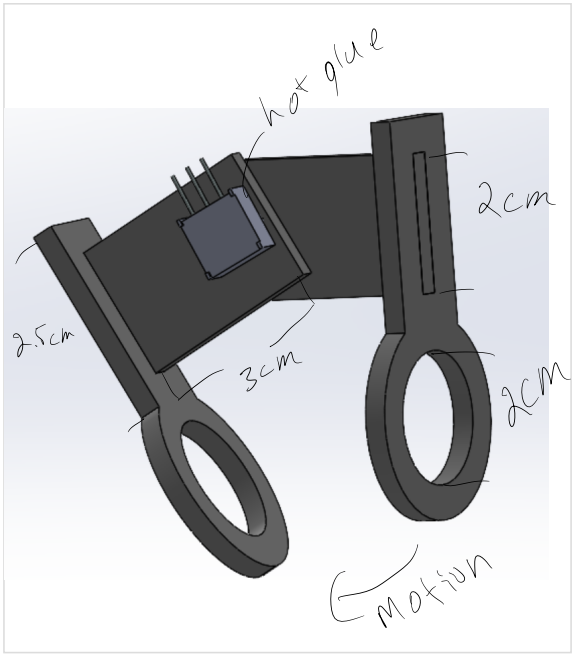
Part 1

I'm using 3 of my late days for Part 1.

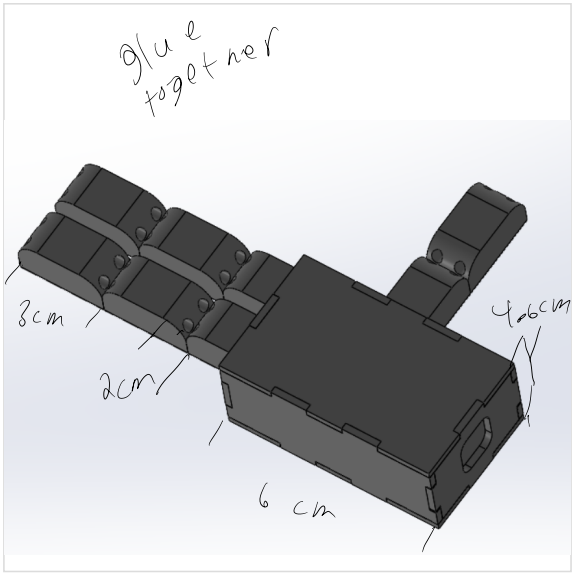
3.1.1a Plan out what you are going to build. Think about how it is going to move and how to attach the sensors and servos. It may be a good idea to see how the servo in your kit can mount to the inside of your structure and the limits of the motion. **Create and submit a dimensioned drawing of both input and output sides of your waldo system. A CAD drawing will work well here, but is not required. Pay attention to mounting details. Bring to a meeting with TA during recitation to be sure that your design is feasible and get recommendations.**



Input system



output system



3.1.1b Create files that can be used to fabricate the components using the laser cutter and or 3D printers available in the RPL. **Submit the required files to be fabricated on canvas as separate files. Be sure to have your parts have your name etched in them so we know who they go to. On canvas, also submit a list (with one sentence description) of the parts you have requested to be made.**

done

3.1.2 Now you need a way of reading ADC channels continuously. Create two subroutines: one for setting up an ADC port (so the user can specify what port they would like to setup), and one for reading the setup ADC port. Try to make your subroutine as clean and efficient (short and fast) as possible. We will post the routine we like best as an example for the rest of the class. Optional: write the routines (or add more) that will allow the efficient reading of multiple ADC ports. **Submit your commented code (we will have a separate submission for code this time).**

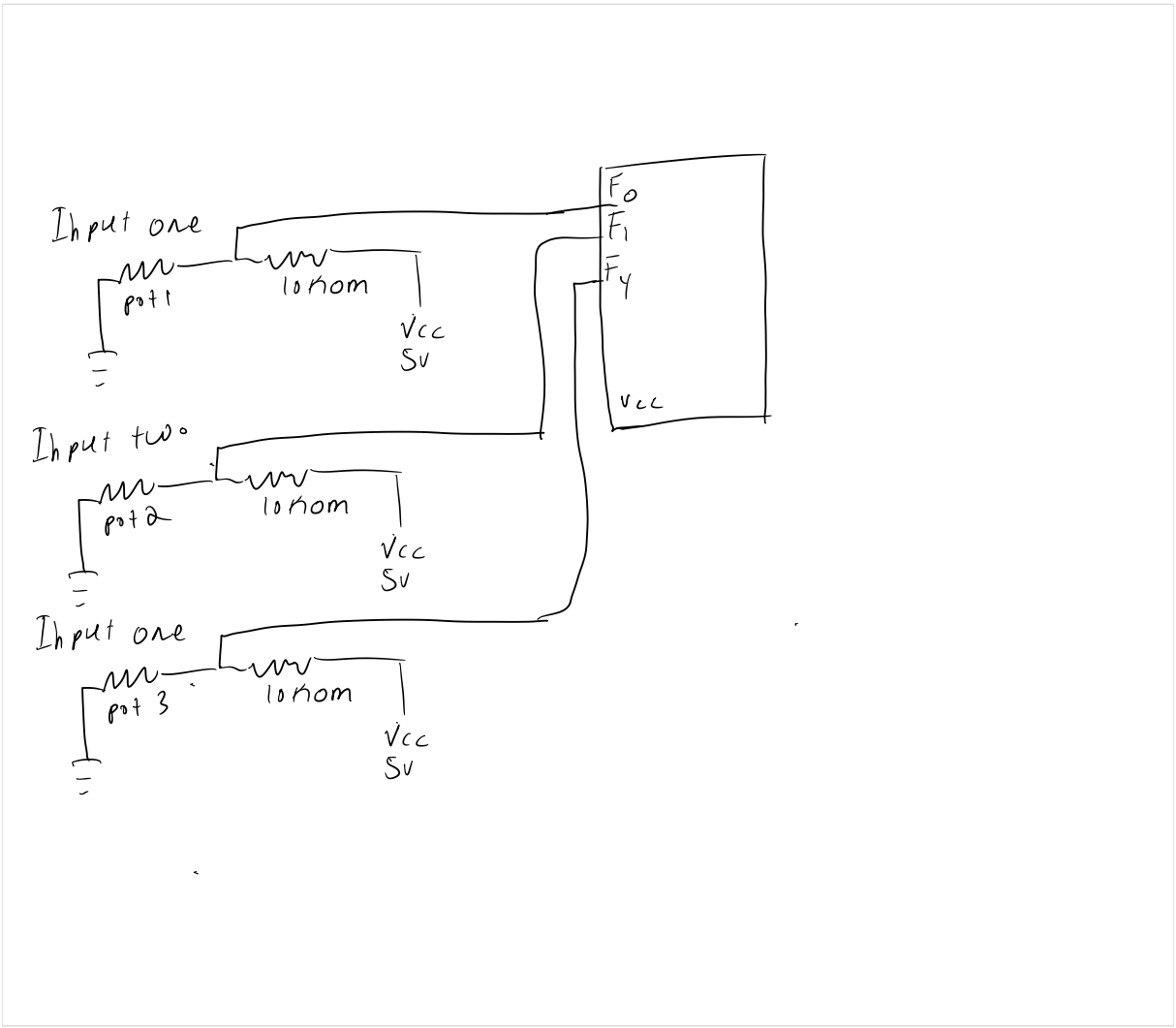
done

3.1.3 Interface the Teensy with two analog sensors (such as potentiometers) so their positions can be displayed to the USB serial port. Build the sensing side of your Waldo and submit a video of it moving with the position of the joints being moved and the angle being displayed in a serial window on a computer. **Submit a video (as a link in your document: youtube, google drive, etc.) of your device and the serial window showing the motion, include drawings, and any code used (we will have a separate submission for code this time). Provide a short discussion of the sensitivity of your device (the number of ADC counts over the full range of motion, linearity and apparent noise sources.)**

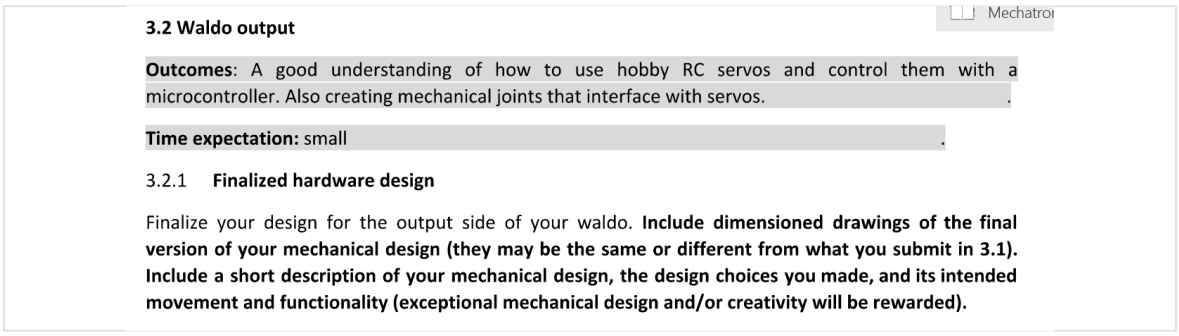
Video link: https://youtu.be/54gxGYMEYFs

Because of the nature of my design I do not use the full range of motion of my potentiometer. Instead I bend the potiometer by about 60 degrees because that is the range of motion of my finger. As a result my resitance varies from about 5 kohms to 7 kohms. This effectively lowered my resolution because my voltage is only varying from 1.5 to 2 volts. The adc count varied slightly

from input 1, 2 ,3 but they were roughly all a 100 adc counts.
A circuit diagram of the input system:



I'm using 1 late day for part 3.2



3.2 Waldo output

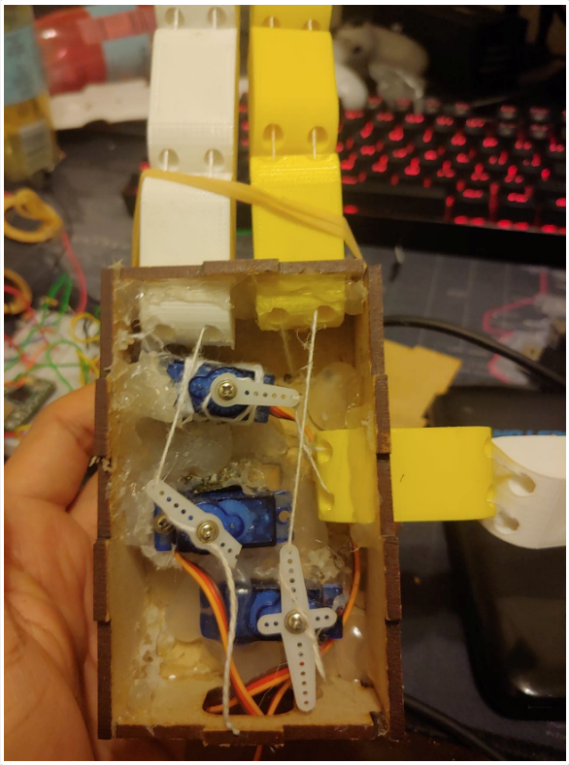
Outcomes: A good understanding of how to use hobby RC servos and control them with a microcontroller. Also creating mechanical joints that interface with servos.

Time expectation: small

3.2.1 **Finalized hardware design**

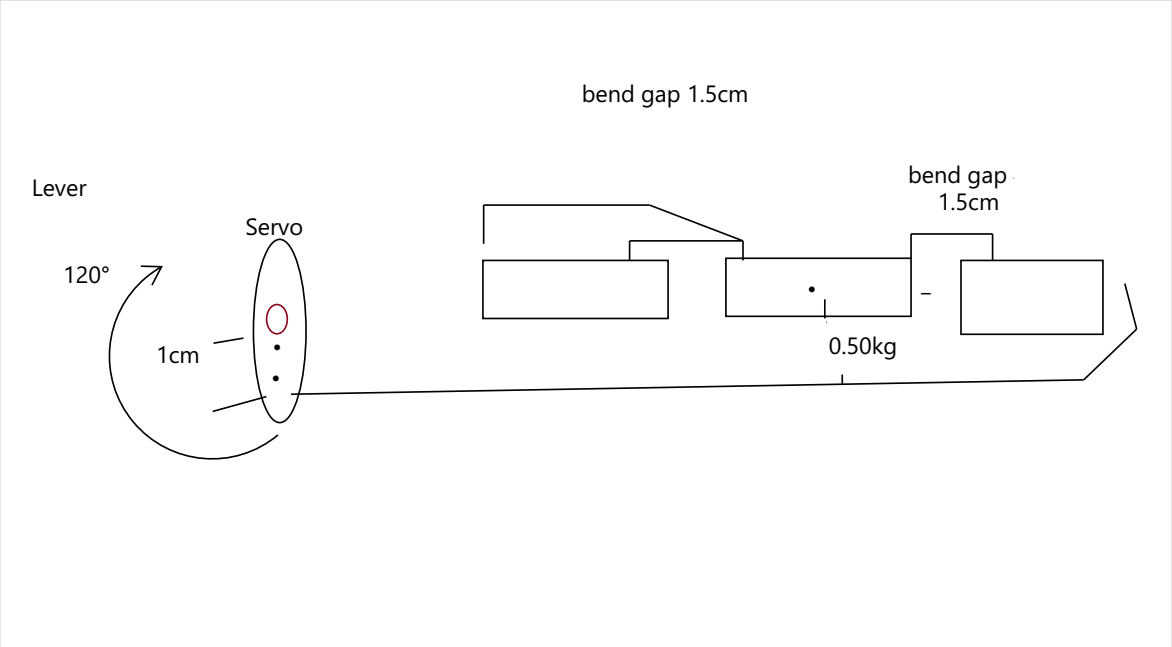
Finalize your design for the output side of your waldo. **Include dimensioned drawings of the final version of your mechanical design (they may be the same or different from what you submit in 3.1). Include a short description of your mechanical design, the design choices you made, and its intended movement and functionality (exceptional mechanical design and/or creativity will be rewarded).**

Same design as in part 1 above. This system works by mimicing the motion of a hand using the input system. When I bend my finger a servor put in a tension system with the 3d printed finger will bend the finger as my finger bends. Arranging the servors in a neat way in the case took a lot of consideration with the final arangement being:



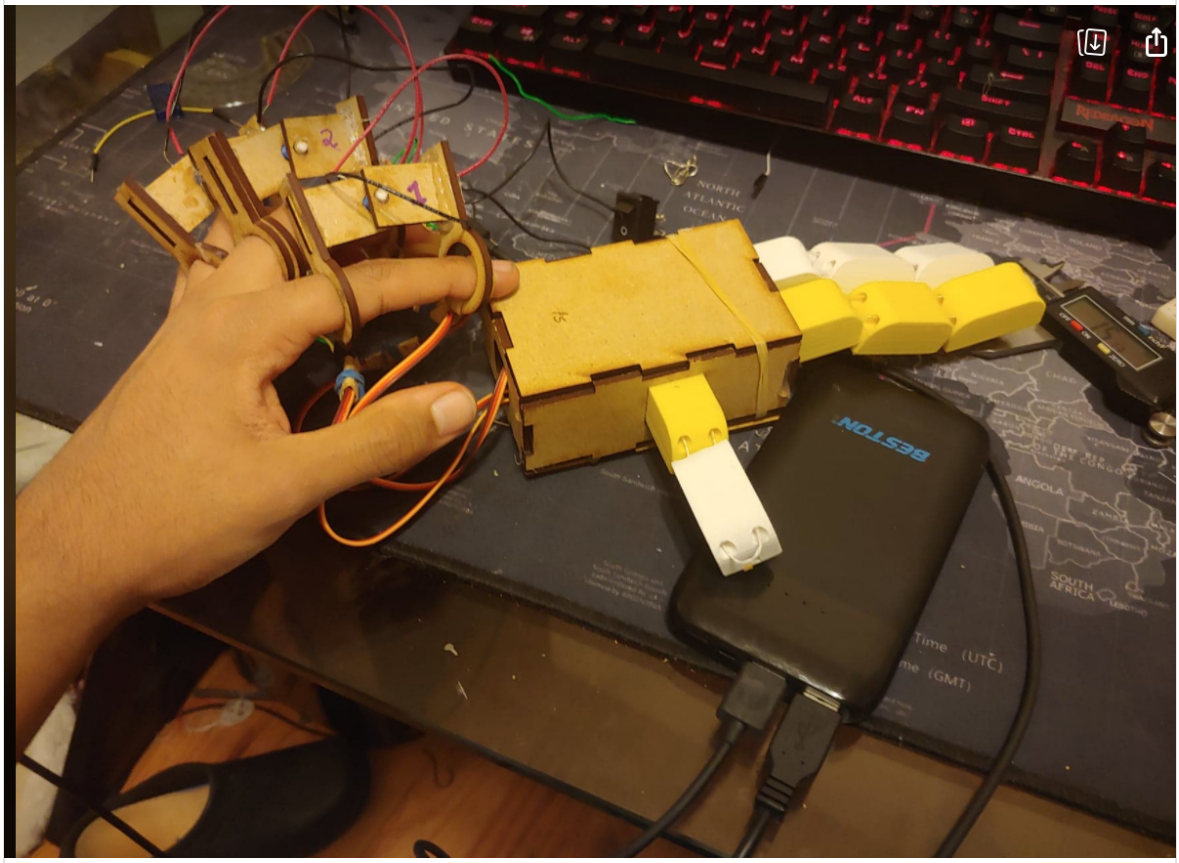
This took a lot of rearrangement and fiddling but I was finally able to get the servors to not get tangled with one another. When I designed this I provided just enough space to accomadate the 3 servors.

I also had to make sure that the servos could handle the torque. I did this by approximating the weight using the density of pla and modeling the sytstem as a simple lever . I estimated the mass to be about 0.33 kg but I multiplied that up to by a factor of 1.5 so I had 0.5 kg.



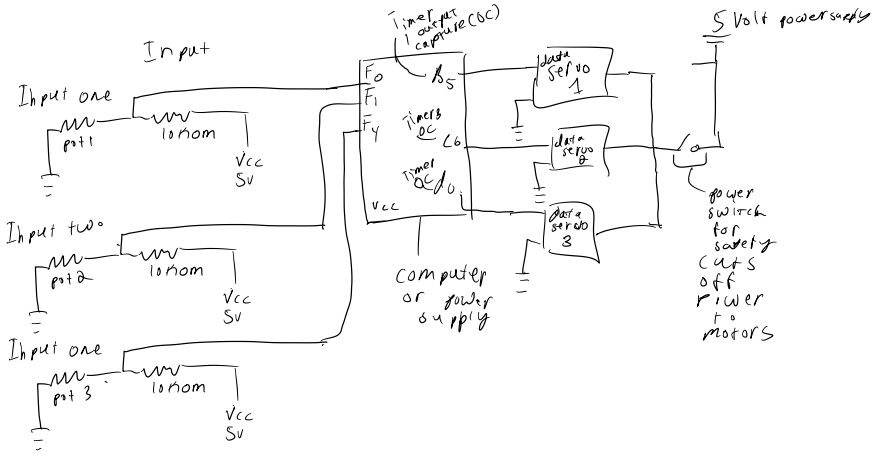
The motor is specced for 1cm/ Kg so this was feasible. I also had to make sure the arc the servo traveled was roughly equivalent to the bend gaps on the finger. I need the servor to stretch the string by about 3cm. The servor traveled an arc length of about 2 cm so I deemed it good enough worst case I figured I could use a slightly larger servor wing to increase the arc length because I had overestimated the weight by 1.5x. a servor lever arm of 1.5 cm worked fine and at 1.5 cm the sevor would be able to handle 0.66 kg.

Final design:



3.2.2 Hardware fabrication and analysis

Make the output side of your waldo. Use the supplied RC servos, or you may optionally purchase your own servos. Do an analysis of the potential current usage by your circuit, including the current sourcing capabilities of your power supply. You may need to search online for more precise SG90 servo current specifications. **Submit a document that includes your analysis of the total current draw of your circuit in the worst case (teensy, potentiometers, servos etc.) as well as the current sourcing capability of your power supply(s). If you use something other than the SG90 servos, include a spec sheet for the servo you used. Include a schematic circuit diagram of both the input and output sides of your waldo that includes the power source(s).**



From https://protosupplies.com/product/servo-motor-micro-sg90/ I found these specs

TECHNICAL SPECIFICATIONS

Motor Model		SG90
Drive Type		Analog
Degree Rotation		180° (±15°)
Operating Ratings		
	Voltage	4.8-6VDC (5V Typical)
	Current (idle)	10mA (typical)
	Current (typical during movement)	100-250mA
	Current (stall)	360mA (measured)
	Stall Torque	1.7 kg-cm (measured)
	Speed	0.12s / 60 degree (varies with VDC)
Dimensions		
	Cable Length	24cm (9.5")
	Motor Housing L x W x H	23 x 12 x 26mm (0.9 x0 .5 x 1")
	Motor Height (w/ shaft)	32mm (1.26")
	Motor Housing Width with Mounting Ears	32mm (1.26")

Teensy max current is about 50 ma

Max current draw

$$(3 \cdot 360) + 50 = 1130$$

1.1 Amps is the worst case scenario and my power supply can provide 2.1 amps so this is fine

video links:

Adelle rumour: https://youtu.be/mxR53A1ZLsA
Spanish flea: https://youtu.be/J28-HisG8Fc

3.3.0
This lab took me a lot of time partly because of the complicated design I chose.Regardless I really enjoyed this lab. There were various things that took a great deal of time however. I was only able to get my parts a last wednesday even though I had submitted the entire design early. This delay happened for two reasons. One, the 3d printers wer painfully slow and two, I didn't know that the laser cutter couldn't handle units in millimeters so all my parts were oversized the first time I got them. I didn't recieve my parts until last wednesday. This meant that I didn't have time to do a second recut so I spent a lot of time coming up with at home solutions to design challenges. For example, one problem I had was that I didn't realize that the upper ring on your finger would have to slide up your finger in order to use the input. I fixed this by taking two pieces that were designated for the base the finger and using them on the tip. This meant that the tip could slide freely because it was over sized and I wrapped a rubber band around it to be able to pull the tip back up. There were countless little design challenges that I had to spend a large ammount of time fixing. And because I got my designs so late I had to rely on inefficient home solutions instead of simply recutting or redesigning.

The second big hurdle I faced was one my servors died on me. The day this was due. That was an awful setback because I pretty much had everything working, but I had to spend a whole two days debuging the problem and fixing it. The reason the sevor broke was bacause the gear can wear out from repeated use, and because they were cheap plastic they broke really easily even if you were within the spec constraints. This wear causes jamming in the servo that can be fixed by dissambelling the servo and manually turning the gear to fix the jam. It took me a long time to figure this out by myself. I was also lucky because I had a few spare servos 9g from personal, but I was able to scrap together the components from a few broken servos to get a working one. you can see below for an example of gear failure. A total of three of my servos failed. I found that running the pwm at a lower frequency reduced the tendency of failure. I used 30 hertz instead of the recomended 50 herts.

Gear Failure:

The final huge time sink was the code. Most of the code was fine but I had one bug in implementing output capture on timer0. I kept getting the wrong frequency. I realized the problem was that timer0 was an 8 bit timer so I had to reduce the prescaler to /1024 to be able to get the desired frequency I wanted. It took me a really long time to figure this out like a days worth of debugging and searching. I feel like being in a lab setting would aleviate this problem substantially.