Minimum Deliverable

Tubular Retractor in Brain Surgery

System Evaluation

Caroline Hoerrner and Robby Waxman
choerrn1@jhu.edu, rwaxman5@jhu.edu

Table of Contents

1 Overview

A. System Description and Goals

The minimum deliverable goals set forth in the first stage of our project include:

- Hardware to allow for 2DOF movement of the tubular retractor
- Software to align retractor using motors, based on computer inputs.

Our design accomplishes these goals through the creation of a physical attachment outfitted with two motors. This attachment holds the retractor in place, and acts as an intermediary between the surgical arm secures the system to the greater surgical support framework and the retractor itself. A visual representation of the attachment is shown below in Figure 1.

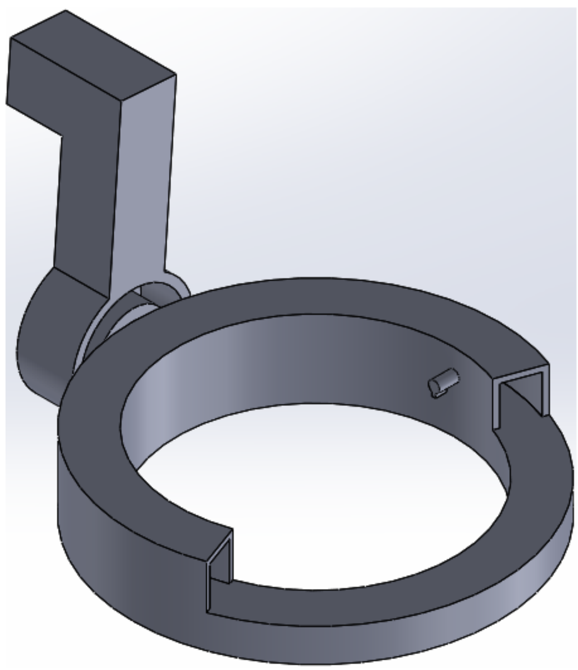

Figure 1: design prototype of physical attachment.

The design provides two distinct axes, each represented by one motor, that can be rotated independently of each other to achieve the desired retractor angle.

The motors fitted within the physical attachment interface with a computer via an arduino microcontroller. The software running the system is written in Arduino and uploaded to the microcontroller where the movements of the motors are controlled. Commands can be given to the motors by typing into the Arduino Serial Monitor.

B. Software Documentation

Class Description: input_angle_2D.ino

This file takes in two integer inputs from the user via the Serial Monitor and moves the two servo motors to the correct angles (in degrees). This class makes use of the Arduino Servo library (https://www.arduino.cc/reference/en/libraries/servo/).

Method Descriptions:

setup():

This method attaches two different servo motors to the specified arduino pins, and sets up the Serial Monitor data rate. Return is void.

loop():

This method prompts the user to enter two integer values and then parses those two integers into floats to be error corrected. After error correction, the two integers are written to the two servo motors. Return is void.

correct_angle(float input):

Given a float as input, this method alters the input slightly to adjust for minor errors in the servo motor accuracy and returns a corrected float. The method can return an uncorrected value, a value corrected by a linear adjustment, or a value corrected by quadratic adjustment. These corrections are based on equations derived in Testing Section 2A.

Variables:

myservoX - Servo object representing the motor that controls movement about the x plane.

myservoY - Servo object representing the motor that controls movement about the y plane.

x - float representing the desired angle of rotation about the x axis.

y - float representing the desired angle of rotation about the y axis.

x_corr - float representing the desired angle of rotation about the x axis corrected for errors in servo movement accuracy.

y_corr - float representing the desired angle of rotation about the y axis corrected for errors in servo movement accuracy.

C. Operating Environment

Given the code is written in Arduino, and the machine running the code needs to be connected directly to the microcontroller via USB connection during runtime, we found it most efficient to run the system in Arduino Studio.

Servo is a standard Arduino library, and needs no additional package imports to be used on any device. The Servo software is compatible with a number of different types of Arduino microcontroller. Given our budgetary restrictions, we chose the Arduino Uno.

D. Assumptions and Dependencies

The design of our system hinges on the use of a surgical arm similar to the Leyla Retractor (shown below) to hold the physical attachment in place. For the purposes of rudimentary testing, securing the attachment to any stable service will suffice, though this fails to accurately depict the usability of the system in neurosurgical procedures.

Figure 2. Leyla Surgical Arm

The hardware components of our system include basic servo motors, and an Arduino microcontroller. The use of software is dependent on the use of the Arduino Studio and Serial Monitor.

2 Testing

A. Servo Motor Alignment

Servo motors are controlled through the Servo.write() function which takes in a float data type and moves the motor to that given position in degrees. So for example, inputting 90 would move the servo to position 90 degrees, it would not move the motor for a total 90 degrees. Servos have a documented range of 0-180 degrees. The following analysis attempts to characterize the accuracy with which the servo motor moves to these desired angles, and filter the inputted data to optimize accuracy.

Setup:

The Servo motor was set to the 90 degree position, and a protractor was attached to the front face of the physical attachment. A weighted string was anchored above the 90 degree mark. The servo was then tested at 5 degree increments from 1 to 180 degrees, and the protractor reading was photographed and analyzed using the matlab image analysis protractor function (Find Documentation Here). An image of the setup is found below. The data was analyzed in python, and both quadratic and linear fits were applied using the scipy optimize library (https://docs.scipy.org/doc/scipy/reference/tutorial/optimize.html).

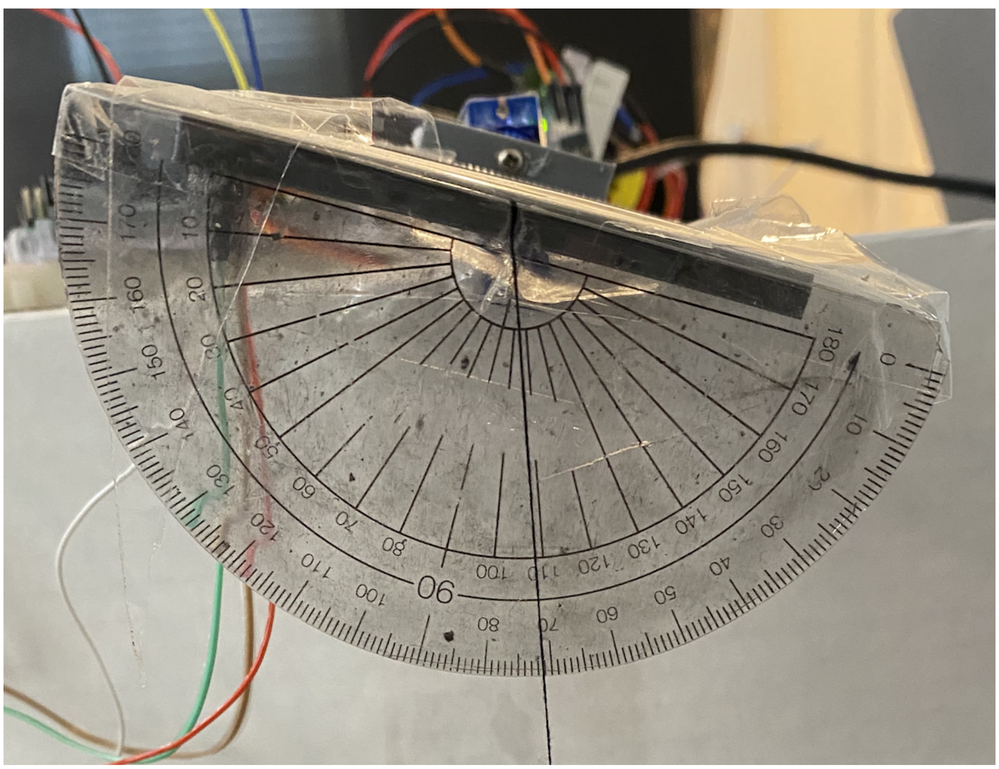

Figure 3. Angle Actuation Setup

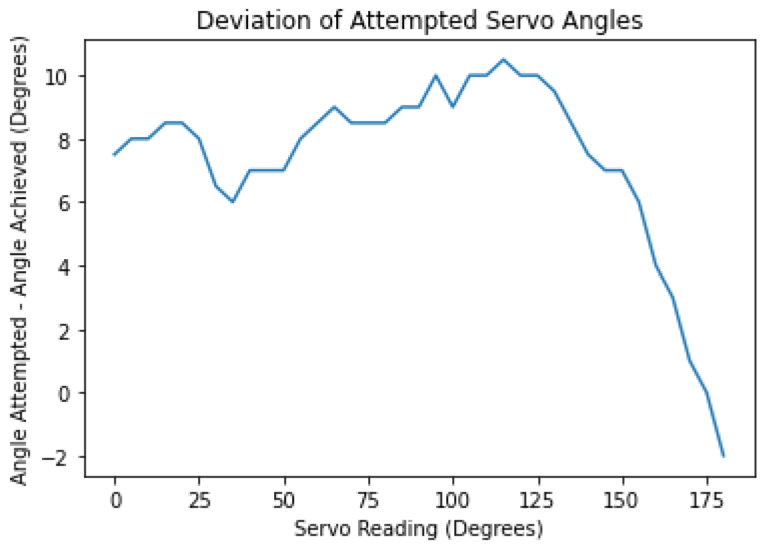

Figure 4. Deviation From Attempted Angle

```
Linear fit:
y = 1.02154 * x + -9.44824
Quadratic fit:
y = 0.86836 * x + 0.00081 * x^2 + -4.38479
```

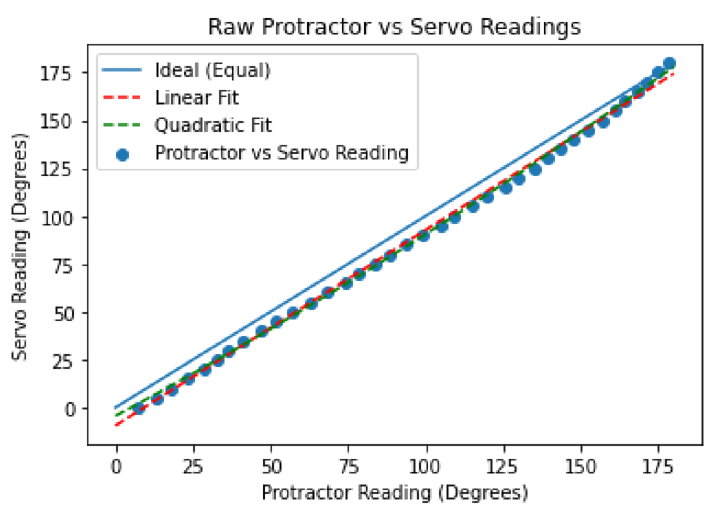

Figure 5. Data fitting

Once fit lines were derived, tests were run again, altering the servo input based on the fit line equations. The true angle was determined again based on the protractor reading and overall error was of each fit was determined.

B. Overall System Function

System usability and potential application in microneurosurgery was assessed using the following rubric. Our group created this rubric to assess all iterations of our system, and will be assigning our minimum deliverable system grades based on its function.

Intuitive Control: 4 - The system relies on a simple typing of desired input. Once input is provided by user, movement is smooth and controlled.

Precision of Selection: 2 - The goal of the system is to allow for ease of retractor angling. The method of inputting angles does not create an intuitive environment for achieving a desired orientation.

Minimizing Interruption: 1 - The system takes surgeon focus completely away from the procedure and involves interaction with another interface all together, making it very disruptive.

Ease in Mastering: 2 - While the action of typing two numbers into a computer does not take much effort to master, it is somewhat difficult to quantify a desired angle, especially in two dimensions.

Category	4	3	2	1
Intuitive Control	The system is incredibly intuitive to control, and takes nearly no time for new users to navigate successfully. Movements can be made smoothly and effectively.	The system is overall intuitive and takes little time for a new user to navigate. Smooth movements come after quick learning.	The system is not intuitive but can be learned after a period of adjustment. After some time, movements are somewhat smooth but not perfect.	The use of the system is not intuitive, and is difficult to control for even the intermediate user. Movements are sharp and unpredictable.
Precision of Position Selection	The achieved positioning can be controlled down to the millimeter. Small and precise movement can be achieved.	Overall precise movement can be achieved but with small flaws in accuracy.	The desired positioning can be achieved but movement is often imprecise.	The system shows little effectiveness in achieving a desired position, and can only go within inches of the desired orientation.

Minimizing Procedure Interruption	The system creates no interruptions in surgical workflow and can be used without shifting view from surgical microscope.	System takes slight attention away from surgical workflow but can be used with minimal distraction.	System takes the surgeon's view away from the surgical microscope, but can quickly be reestablished with short interruption to procedure.	The system is disruptive to procedures and requires complete focus, drawing the surgeon away fully from the surgical field.
Ease in Mastering Technique	Mastering system controls can be done within minutes and needs little training before use.	Mastering the system takes several test runs before use, but is overall achievable.	Mastering the system takes multiple extensive training sessions, but is overall achievable.	Mastering system controls is nearly impossible. Even experienced users struggle to get desired output.

3 Evaluation

A. Results and Discussion

In the x axis servo, the average absolute error of the uncorrected angle values is 7.46 degrees. Once the linear correction is applied, this error drops to 1.95 degrees. The quadratic correction results in an average 1.46 degree error, making it our preferred correction for further use.

In the y axis servo, the average absolute error of the uncorrected angle values is 15.71. Once linear correction is applied, this error drops to 2.41 degrees.

We believe that error under 3 degrees is acceptable for the purposes of our device, and does not detract from the usability.

The overall control of the system can be vastly improved by introducing a more interactive control element. The major flaws of our current control system include the time and attention it takes away from the procedure at hand, and that it does not make for intuitive angle actuation.

B. Next Steps

We are pleased at the results of our basic angle entry system. However, we know that this does not make for intuitive surgeon use. Our next steps are to fulfill the expected deliverables of our project, including more intuitive angle input strategies that involve IMU sensors.