Controllable Robotic Arm

Revan Murton

Robotics at Falmouth University

Objectives

My project is a remote controlled 2-dimensional arm that allows the user to draw/write remotely, the main objectives I had for this project was:

- To use Fusion 360 to design and print the base, motor and pen pieces of the arm.
- To use inverse kinematics to calculate the required angles for both servos to have the pen reach the desired coordinates.
- To use Python packages to acquire the cursor position and send the result via serial to the Arduino

Introduction

My goal was to design a 2 dimensional robotic arm that allowed a user to control its movements with the movement of the cursor on the screen, with the robot and the user working together to draw and image/ write text. I designed the robotic arm off of the similar designs within the robotics lab, this allowed me to focus on modifying my similar design for use with the much smaller SG90 servo.The arm is broken up into three separate pieces: the base section provides a stable footing for the arm and acts as a counter balance, the servo section houses both servos and controls the rotation for each joints of the robotic arm, the pen section allows the physical drawing to be done. I chose this project in order to fulfill the requirements of the Human vs Robot contract, in particular the collaboration aspect mentioned.

Method

For this project I focused on the physical components first, this was because I knew in order for the code to function correctly the length of both arms needed to be known. This allowed me to narrow down what I needed to work at any given point increasing my efficiency. To reinforce this point I began working on the python code last, as this was the least important part of the project and could be removed if I had to without significant loss.

Materials

The following physical components were used to build this project:

- Base, motor and pen sections made from PLA:
- 2 x SG90 servos
- 6 x female to female jumper wires
- Arduino UNO

The following programs were used to build this project:

Fusion 360

PyCharm

Arduino IDE I printed multiple versions of the Base,Motor and pen sections as the project progressed as I made adjustments to improve and hopefully perfect the design.

Inverse Kinematics

(1) is the calculations that represents the inverse kinematics required to get the angle for the second servo, due to the way I calculated this I needed to use the second angle in order to calculate the first.
(2) is the calculations to get the angle for the first joint.
(3) and (4) are the calculations tp convert the angles for the first and second servos respectively.

$$Angle2Rad = \frac{x^2 + y^2 - a^2 - b^2}{2ab} \tag{1}$$

$$Angle1Rad = \arctan 2\frac{y}{x} - \arctan 2\frac{a + bAngle2Rad}{bAngle2Rad} \tag{2}$$

$$Angle1Degree = Angle1Rad \times \frac{180}{\Pi} \tag{3}$$

$$Angle2Degree = Angle2Rad \times \frac{180}{\Pi} \tag{4}$$

Arduino Code

I've used 3 cases for the states the Arduino can be in, the first waits till there is serial input from python and assigns the serial.read to a variable. The next splits the input into two integers, these integers represent the x and y coordinates respectively. The last state does the calculations for the angles using these two integers which allows both servos to rotate to the correct angle for the end effector to end up above the desired position.

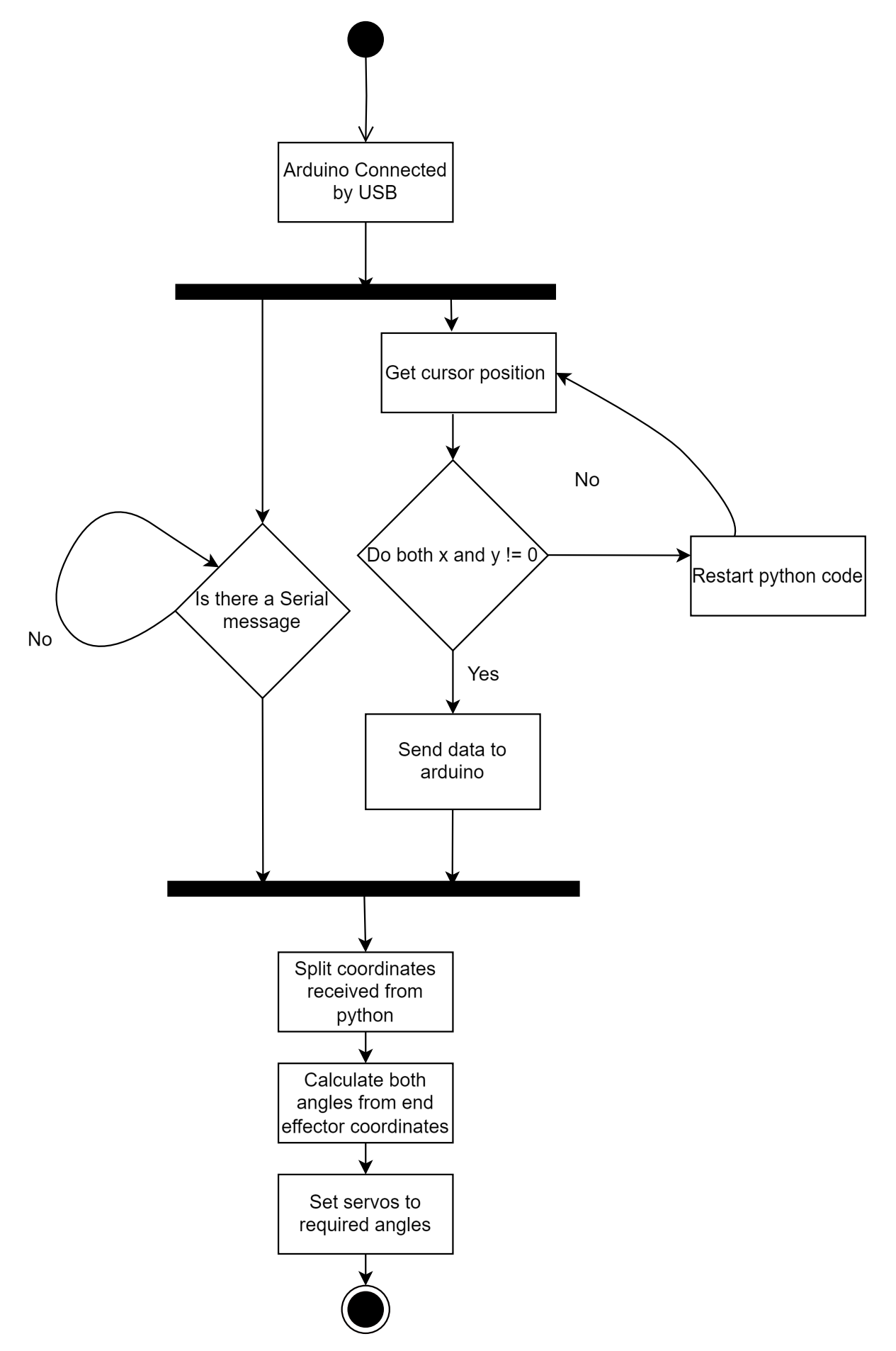
Arduino Connected by USB
Get cursor position
No
Is there a Serial message
Do both x and y != 0
Restart python code
No
Yes
Send data to arduino
Split coordinates received from python
Calculate both angles from end effector coordinates
Set servos to required angles

Figure 1:State Diagram (SVG)

Results

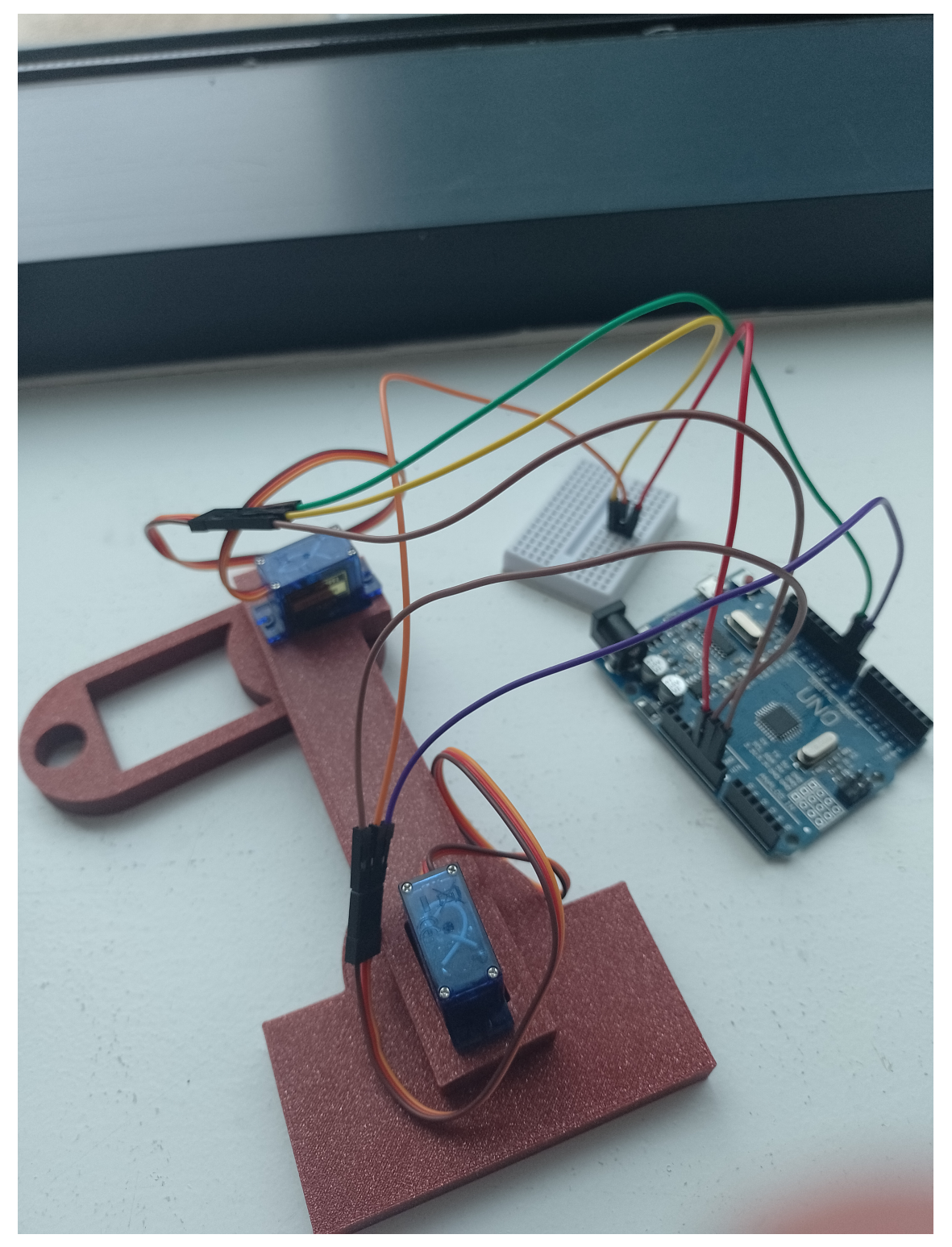

Figure 2:Image of Robot

Conclusion

Overall I gained significant knowledge in the use of Fusion 360 for designing and 3D printing this was because of how many different iterations of my project I designed, this helped me practise and focus on the areas I was uncomfortable while providing me with a record of the progress I made. In terms of coding I figured out how to effectively use states in my project, with them being a great way to break up large chunks of code and by using Serial.println() they are also a great way to have visual feedback of what state the robot is in while it is running. I also got to practise both the Cosine rule and SohCahToa when developing the inverse kinematics code. Although I'd done these calculations before it was awesome to relearn and refresh my memory when it came to these topics.

Acknowledgements

Contact Information

- Github link: `https://github.falmouth.ac.uk/Games-Academy-Student-Work-23-24/COMP102-2309236`
- Email: rm305181@Falmouth.ac.uk
- Phone: +1 (000) 111 1111

Games Academy