Summer Soft Robotics Internship Conclusion

This summer, I interned with Dr. Hanyu Zhu and Lissa Blackert. I assisted in their soft robotics project by working on creating a neural network model to control the robot's movement. The goal of the model was to be able to input end effector coordinates and receive the appropriate motor positionings to move the robot into place.

In order to learn how to build a neural network model, I started the internship off by taking courses through Codecademy and Coursera on the Keras library for Python. This library is the more accessible API for using the TensorFlow library. On Codecademy, I went down their "Building Deep Learning Models with TensorFlow" skill path. On Coursera, I took the Keras courses offered by IBM.

The first step towards actually building the model for the soft robot was to build a practice model. The robot that I chose to practice on was the Mitsubishi PA-10 6-DOF robotic arm. This robot was suitable for my purposes because the necessary motor and position data was available online for this robot through Sebastian Starke's DeepIK Github repository. His repository contained over 50,000 unique data points, which is enough to train a reliable neural network on. Building the forward kinematics model was a straightforward process. I created a network with three hidden layers, each with a dropout rate of 0.05. I also introduced an early stopping point to stop training the model once it is no longer improving its performance. I was able to get the mean squared error of the model to 0.01. Building the inverse kinematics model was a lot more difficult. This is due to the nature of the robotic arm. There are multiple ways for the robot to reach different end effector positions. Although I used the same structure that had proven effective for the forward kinematics model, the one-to-many nature of the inverse kinematics problem meant that this model could not be trained as easily. I ended up with a mean squared error of 0.06.

I then conducted manual testing of my models. I input data for a half rotation of motor two into the forward model, which should output an arched movement. I encountered a strange issue where the predictions for any positive motor rotation value would output an incorrect x-coordinate. In order to figure out why this occurs, I searched the provided data for answers. It turned out that there were no similar data points to my manual testing points with a positive motor two rotation. This explained why the model was unable to accurately predict end effector position at those points: it had no prior data to base its output on. When I compared the model's predictions for my manually tested points to the given data points, the predictions were pretty accurate. Looking through the given data also revealed that the end effector had a smaller range of motion than I had previously assumed. Accounting for this made my model's predicted output's error even smaller. I then used the forward model's output to manually test my inverse kinematics model. The results did not appear to be very accurate at first. However, it is important to remember that there are multiple possible solutions for the model to choose from. When I put the inverse model's predictions back through the forward model for interpretation, the results were more accurate than the pure motor output would have us believe. Putting the original

manual testing data through the models three times also contributed to the error, so it is not a pur reflection of what the inverse kinematics model can do.

Creating the inverse kinematics model revealed the one-to-many function problem. I looked through published literature for solutions to this problem. Although many solutions have been posed over the past few decades, the most practical one for my purposes implements a recurrent network. It included the current state of the system to choose the best possible output for the input, which was the output that had the lowest possible calculated error when factoring in movement costs.

This semester, I learned about neural networks and how to build them using Keras. I also learned how to effectively find information in published literature in order to solve problems I encounter. I will be continuing my work with Dr. Zhu and Lissa next semester in order to implement the solutions I found in the literature.