Andrew Gates & Regan Stovall
Professor Matthew Tolentino
TCES 460
19 December 2017

Remote Robotic Hand

- Challenges Faced -

There were many challenges faced when trying to overcome this project. The two main challenges were dealing with the hardware portion of the robotic hand as well as the software portion. With the physical construction of the robotic hand, the main challenge was to create a hand that mimicked the movement of a human hand, but was also sturdy enough to pick up objects. As for the software portion, the challenge was getting 2 Pis to communicate via TCP to stream data back and forth reliably for the various sensors and motors.

- Overcoming Challenges Faced -

For the hardware portion - this initially seemed like the easiest part, we wanted it to be nice and semi-professional, and since neither of us had enough experience designing 3D parts, we decided to laser cut a design. The first attempt was pathetic, the design was weak, the fingers were single axis and would get stuck in the downward direction. The actual cutting of the board was sloppy and not always consistent depthwise. The second attempt was a bit better, experience with design and print setup greatly improved the results, but the actual design just didn't work as well as hoped. We made some final modifications, cut the new board, and spent a painfully tedious 5 hours assembling the hand. Surprisingly it went off without a hitch. A few of our wires were switched accidentally, and one of the motors came unsoldered in the glove, but for the most part we were just holding our breath for the bug in the design that didn't really happen.

For the software portion - we initially tried getting the two Pis to simply read from GPIO using WiringPi. Once we had that working we made the Pis work with analog readings of the flex and pressure sensors through the use of a MCP3008 A/D converter chip. After we had our Pis working with all of our sensors and motors we connected the two Pis via serial to stream data between them. This was however wasted time since we weren't able to actually translate this into TCP as efficiently as we would have hoped. Once we had everything working with serial we switched to TCP to at first just send a message back and forth, but then added to our code to allow for constant streaming. Once we had data streaming between the Pis we added in connecting the sensor data that was received to the various motors that had to be controlled.

- Communication Protocol Used -

We used TCP (Transmission Control Protocol) to use one Pi as a server and the other as a client. One would open up a socket using a given value, and the other would connect to the same socket using the IP of the server Pi. Once the two were connected they could send data back and forth.

- What We Learned -

There were many things that we learned during this project. The main thing we learned was how to design and implement a project of this scale. There were many checkpoints that needed to be accomplished in order to get to our final product, but this project gave us a great way to work through checkpoints to achieve our goal. We also got great experience in creating wireless communication between two devices, experience in prototyping and designing a deliverable product, and how to organize and polish our final product.

- What We Would Build Next -

An interesting idea we had to expand upon this project would be to add a wheel to the back of the hand and make the hand able to move and navigate using the wheel as well as the fingers to crawl.

- Group Members -

Regan Stovall & Andrew Gates

- Software Environment/Language Used -

We used C as well as command line and a simple text editor for editing and compiling our code. We used C due to the many libraries for WiringPi that were written in C, which we used for GPIO, PWM, and the MCP3008 chip.