**Friday, February 16th, 2024**

I continued working on the Jetson 🡪 Arduino communication and tested out my code from Wednesday. I was successfully able to control the steering servo over the web interface I was previously using. However, the throttle motor did not spin no matter what input was applied.

To debug this, I figured out how to send back the received data to the host computer so that I could debug why the throttle was seemingly always 0. However, when I got the data back, it was exactly the same as I sent it, so I was unsure why exactly the throttle motor were not working.

**Wednesday, February 21st, 2024**

I continued debugging the serial communication problem. I realized that I was actually just sending back the receive buffer and copying it into the transmit buffer. Obviously, that would not tell me anything useful since it’s the same buffer. The problem lied in getting the data out of the buffer into a usable format. I fixed my debugging code so that it actually read out the values like it was supposed to and then attempted to send those values back to the host. This resulted in only the steering value being sent back to the host, confirming that the Arduino was not even reading the throttle value to begin with.

As a debugging measure, I ran one of last week’s scripts which simply runs the motors forward and backwards with the Arduino, without any input from the host computer. This code also did not work, which is concerning, because it worked last week and I did not change anything. I spent some time debugging it with the diagnostic tool before realizing the motor controller was not plugged into the motor since the car was taken apart last Friday to figure out how its steering could be fixed. I did find out that the VESC was receiving the appropriate input signals, however, which is good to know.

I thus tried to figure out if I could send two values over one number, since that one number was being transmitted perfectly fine. I looked at some overly complex solutions before simply stuffing the two numbers in a 32-bit integer, by keeping the first number, and adding 65536 \* the second number, essentially putting these two 16-bit numbers together. The two inputs could be trivially recovered by simply doing the division and modulus of the new number by 65536, to get the first component and second component.

However, this did not work either. I think I will bring in a battery compatible with the backup motor controller I took from one of my club’s other cars and try to use that one instead of the VESC. I’ve used that type of motor controller successfully with a Raspberry Pi in the past so I think using it with a Jetson will be similar. The difference between these motor controllers and the VESC is open-loop versus closed-loop control: the VESC has a built-in PID controller which maintains the motor’s speed at a constant RPM. Thus, the actual velocity of the car can be calculated, and I can be certain that the car is moving at a certain speed when I command it to do so. On the other hand, this motor controller will just be a simple 0-100% power input with no guarantees of consistency.