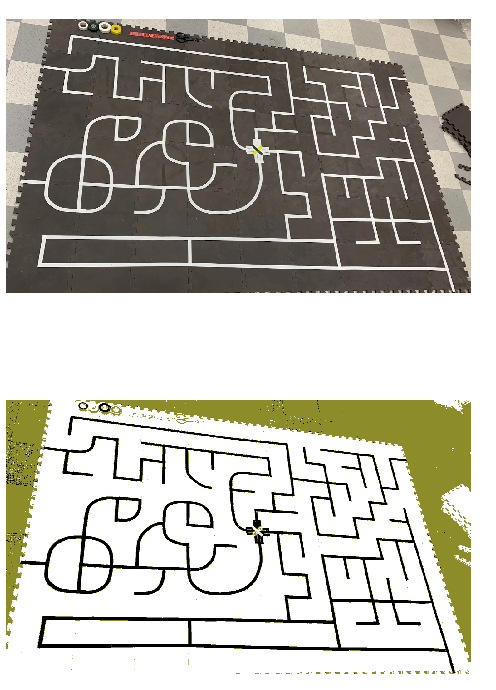
**TEAM 2 - WEEK 1 REPORT - JUNE 31 - 408i SPRING 2022**

**Michael Delatte**

During lab this week, Michael developed a photo-to-maze tool. His first attempt was to convert the photos to black, white, and another color to represent the track, maze mat, and non-maze area. Next, he tried fixing the photo, by stretching each line of pixels and removing the non-maze area pixels. This produced a poor result, but the plan is to project the maze to be an upright photo, by identifying edges, finding the angle with respect to normal, and implementing a simple rotation/projection. After, the program will create sections of the fixed image, and put it into a binary matrix, representing if there is a track or not. Then using a simple recursive algorithm, the program will almost instantly solve the problem. It will record the fastest path, and make a list of instructions to finish the maze.0

**Wesley Catbagan:**

During the lab this week, Wesley worked on looking at the class examples on the class GitHub. Working through the examples was important for understanding how the mouse will work and gave us an idea of how we want to implement our code moving forward. A large part of the lab was spent testing the motor function and trying to figure out what power settings will be needed to have controlled movement. After testing it is clear that the motors need different amounts of power for the same output which means our code will have to account for that. We plan on using the encoders. The rest of the lab was spent discussing steps moving forward to create a plan for our project.

**Erik Bryson:**

My primary focus is the controls aspect right now, specifically smooth motor control, and therefore my first go at experimenting I set to attacking the encoders. The idea is that all compensation of voltage fluctuation can be ignored by looking at the end result of what the wheels are doing, or in other words, the encoders tell all. And so I spent a few hours troubleshooting the EncoderTest.ino code and in the end was able to see read out of both encoders. I did several quick tests: The first being if I could exceed the Tik-Rate of the nano by moving the wheel a little faster than the highest speed that wheel will be operating at, and then slowly return the wheel to its starting point to see if the nano’s encoder value missed any ticks and it did not, it returned to the same magnitude. The second test, which was more of an observation, was how many tiks per complete rotation of the wheel, and found it to be 360. ( Rotated the wheel 10 times and divided the encoder’s readout value by 10. )

All of this focus on the encoders is again in the sight of smooth motor control, with the idea being that observation of the end result of the wheels behaviors, velocity and acceleration, verses what the wheels are being instructed to do, can now be compensated through a PID control loop to produce the desired torque amidst varying voltage of the battery.

I plan to next implement this PID loop function, for the rest of the team to use, this week.