Christopher Ho

500840595

[Christopher.j.ho@ryerson.ca](mailto:Christopher.j.ho@ryerson.ca)

Monday November 9, 2020

CPS 607 Lab 3 – Object Avoidance

The goal of the lab was for the robot to navigate a world surrounded by lines with high contrast. The Dark lines will represent by boundary where the robot cannot go. In addition, there will be objects to avoid at least as tall as the ultrasonic sensor. My approach to this lab builds off the previous lab which ensured that the robot stayed within its environment where black lines acted as boundaries. In summary, my previous approach was to always move forward until a sensor was activated. Depending on which sensor was activated, the car would turn in the opposite direction and continue moving forward.

Initially, my naïve approach was to combine the example Elegoo object avoidance code with my lab 2 code which kept the robot inside the test environment. What I found was that after passing the line sensor test, during object avoidance the robot would cross over the boundary lines. Upon further research I discovered that while using delay, it pauses the program for the desired milliseconds. That won’t work because I wanted to do ultrasonic sensor and line sensor checking simultaneously.

To solve this issue, I used the millis() function as a global timer combined with some delay() when appropriate. Using a global timing function, I could check the ultrasonic sensor at regular intervals while running my line sensor code. This was achieved using events. My event occurs at regular intervals of 100ms and during that time, the ultrasonic sensor evaluates the distance from my robot to the nearest object. When an object is detected, the robot scans the ultrasonic area to the left and right to determine the best direction to turn. Simultaneously during the event, the line sensors are being checked to determine if the robot can move forward or not.

During testing, I initially made the robot move forward after choosing a safe direction. However, this caused the same problem as before where the robot would drive over a boundary line. I resolved this problem by making the robot only move forward after passing the line sensor decision portion of my code. This is how I ended up making the ultrasonic sensor event only evaluate the best option and pick a direction but not move the robot forward.

One area that needs improvement is when there is an object on a fringe edge in the path of the robot but not directly in front of it. The ultrasonic sensor won’t detect it and deem it safe to move. But the robot will drive forward and graze the object causing a potential collision. A solution could be to create another event that checks a separate direction for the ultrasonic sensor. Then if any of the sensors detect an object close then evaluate the best direction and face that direction.