Alex Medellin and Nick Rummel

CSC463

Dr. Girard

September 27, 2017

**Maze Navigation Lab**

1. Additional pieces were placed on the front of the robot to make space for the touch sensors, which were placed on the right and left sides of the front of the robot. In order to hide the sensor wires, pieces were placed along the sides of Handy-Board cradle that extended out and above the sides. The inside of these pieces was kept hollow and the ends were kept open to allow us to run the wires through and to the back of the board. The wheel base, cradle for the Handy-Board and motor positions were not altered in any way. The motor wires were also concealed in the same way the as the previous assignment.
2. We hard-coded time values for turning so the turns would be right angles. However, if the robot ran on another type of surface, the angles may not be 90 degree turns due to the friction between the tires and the surface. We have the robot to try a left turn by default for the first try. Then, it starts logging a history of the turns. In the case of the maze we used, it worked fine. If the maze would have been to the right, it would have taken our robot longer to navigate. The robot also does not turn until it “hits” the wall (the touch sensors are pressed) and backs up.
3. Yes. With the addition of range sensors or some type of sight, the robot would be able to detect obstacles from a distance. Instead of hitting a wall and readjusting, the robot would avoid the obstacle all together because it would be able to detect an object or wall before encountering it. By avoiding obstacles, it would be able to navigate and map out the maze much faster and easier.
4. Using a random number for turning could allow the robot to finish the maze more quickly if each random number is correct. However, it could potentially slow down navigation time if it keeps turning the wrong way. To an extreme, the robot may never finish the maze or keep turning in circles because the robot’s behavior is unpredictable. Turning in the same direction could be just as disastrous. For example, if the robot keeps turning right to follow the maze but the maze suddenly turns to the left, the robot may end up navigating back to the initial starting position. With a consistent turning direction, the programmer is at an advantage. This is because the programmer can predict the behavior of the robot.
5. Depending on how the loop is constructed, the robot may have problems due to turning at 90-degree angles. However, the code was written to keep a history of successful turns so that the robot has a guess of which way to turn next. If it fails to turn the correct way, it will try turning the opposite direction. Eventually, the robot would navigate through the maze but potentially at a slower rate.