Project 1 Report
Navigating a known Obstacle Course with the Lego Robot

Group 9

Ashwitha Kassetty
Kierra Thompson
Andy Sustaita

CSE 4360-001: Autonomous Robots
Dr. Manfred Huber
November 7, 2020

Objective:

To navigate a mobile robot through an obstacle course to a goal location. The start position, the goal position, and the obstacle locations of the goal are given before the robot is started.

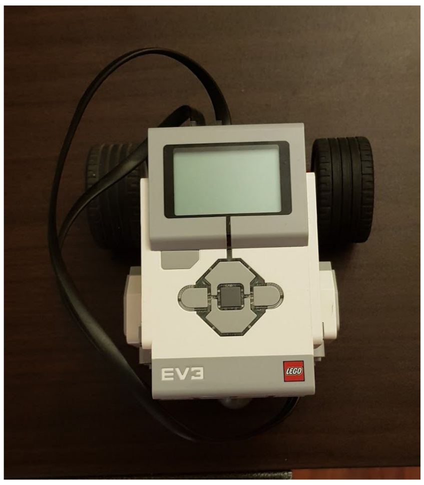
Workspace:

The workspace for the robot is a rectangular area of 4.88m x 3.05m in size. The obstacles are squares of 0.305m x 0.305m in size, which will be randomly placed within the workspace. The goal is a black circle with a radius of 0.305m.

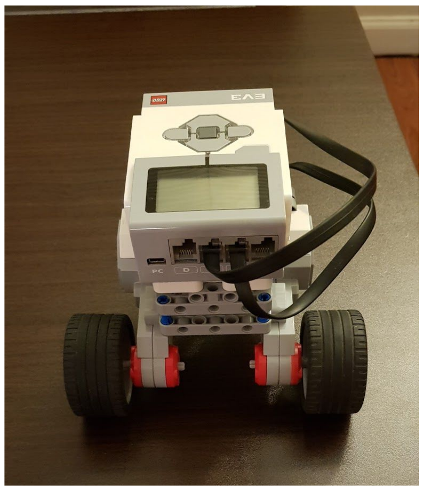
Robot Design:

We designed a three-wheeled robot. The front of our robot has two front wheels attached to the motors and in the back, we have a smaller wheel that allows our robot to make turns. We made a base to be able to mount the EV3 high enough off the ground and away from the wheels.

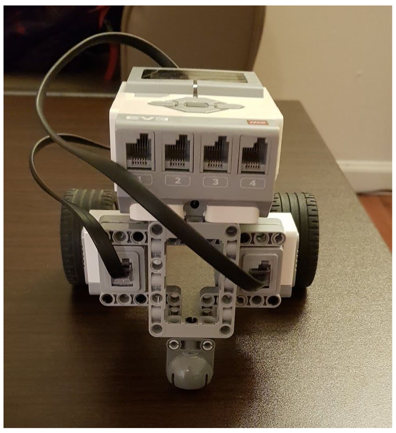
It was fun to try and test different parts and ways of building the robot. It was like being a kid again deciding which lego parts we wanted to use, and how exactly we wanted it to look and function.



Top View



Front View



Back View

Navigation Strategy:

Since the robot does not contain sensors to locate or navigate through obstacles, it will be provided with the positions of the start, goal, and the obstacles in x-y coordinates. This entire workspace is mapped into a 2-D world with 16 columns and 10 rows (each corresponding to one floor tile in the lab).

We compute the Manhattan distance for each tile within the workspace. This is accomplished by creating a grid that mimics the space, then calculates the distance that each square is from the goal. The goal position is given a value of 0 and the obstacles are given a position of -1. After the grid is filled we pass the data to the Breadth-First Search function to determine if there exists a path from the start position to the goal position.

Using Breadth-First Search, the algorithm begins at the start position and compares the neighboring tiles with a lesser value than the tile the robot is positioned at. The algorithm then provides the next direction the robot should traverse. The directions are divided into top, bottom, left, and right. The robot can perform three directional movements - forward, left turn, right turn. The robot always prioritizes moving forward on the path over the turns. This is done to reduce errors, such as a sudden change in direction.

If the goal is reachable, the robot will traverse through the workspace from start to goal. Once within the goal, the statement “Goal Reached.” will be printed onto the screen. If the goal is unreachable, the robot will print “Goal Unreachable.” onto the screen. Both outcomes will require for you to press enter on the robot to exit the program.

Problems Faced:

We encountered many bugs due to memory leaks. There was no clear indication of a memory leak in the program from the robot, so it took a long time to find and fix the errors. The errors due to the memory leaks would cause the program to not upload to the EV3 robot. For some reason, the switch statements we used did not work, so we had to change the format of our code where we used switch statements to an if-else block.

The Code:

Attached is the code we are working with. The code is not complete as our demonstration of the robot is on Monday, and we are still fixing some last-minute errors in the code up until then.