Authors: Michael Bonnet, Allison Gardiner, Noah Walker
Class: CSE 4360-001 FA2021
Date: 11/3/2021
Assignment: Project 1

Project 1

Robot Design

We decided to follow the Medium Motor Driving Base assembly instructions found on the ev3 website. We chose to do this for a few reasons:

- Copying a known configuration reduces time spent on robot design and construction
- Copying a manufacturer-recommended configuration potentially reduces time spent adjusting robot design
- Design did not include sensors, reducing time spent coding the robot
- Two wheels simplifies kinematics of the robot
- Three points of contact with the ground nigh-guarantee stability

Navigation Strategy

We chose to use the A* search with a Manhattan Distance heuristic as our navigation strategy. While this method does yield the "best" path, this isn't why we chose it.

We chose it because:

- This method handles a uniform cell decomposition workspace well
- This method will yield a path that does not have diagonal transitions between cells, but instead goes at right angles
- Purely right angle turns means less coding and less testing

Of course, A* does not itself actually give the robot directions. It simply generates an ordered list of states, in this case in the form of ordered pairs. To translate from a list of ordered pairs to a set of instructions, we:

1) Calculate the change in X and change in Y for each transition between states, giving us in ordered list form, n-1 transitions on n states in the path list. In a way, this is getting us the rate of change between states, almost like a first derivative.
2) Using the transition list, detect whether or not there is a heading change. We detect a heading change by iterating through the transition list and checking to see if the "current" transition's X and Y changes are different from the previous transitions X and Y changes, in which case, depending on the change between changes detected, is translated as a specific heading change (from one cardinal direction to another, with the robot always starting facing east). When that heading change is detected, the appropriate "turn"

command is appended to an instructions list followed by a “move forward” or “move backward” command. If no heading change is detected, a “move forward” or “move backward” command alone is appended to the instructions list. In a way, this is getting us the rate of change between transitions, almost like a second derivative.

3) The main.py file runs a function that translates the instructions list to actual motor movements.

To create the environment for this pathfinding/navigation, we created a 32x20 grid of 6-inch-on-a-side cells, and marked any cell containing or adjacent to an obstacle as an obstacle, so that we’d have 2 feet of guaranteed clearance between obstacles. This helps us steer clear of obstacles when running along gridlines despite our navigation method assuming the robot is in the middle of a floor square.

We named our robot Solomon.