Machine Learning Capstone Project Proposal

Robot Motion Planning

By: Kambiz Mir
July 15, 2017

Project Background

This project is a simplified software simulation of Micromouse competitions began in 1970s, in which a robot mouse tries to find a path from a corner of the maze to its center. The robot mouse may make multiple runs in a given maze. In the first run, it tries to learn the maze map as much as possible and find paths to the maze center. In the subsequent runs, the robot mouse attempts to reach the center in the fastest time possible, using what it has learned in the first run.

Problem Statement

For this project, a simplified model of the world is provided by Udacity. There will be 2 runs. The goal is to write program to explore the world in first run and then find the fastest path to the center in second run.

The maze exists on an n x n grid of squares, n even. The minimum value of n is twelve, the maximum sixteen. The starting square will always have a wall on its right side (in addition to the outside walls on the left and bottom) and an opening on its top side. In the center of the grid is the goal room consisting of a 2 x 2 square; the robot must make it here from its starting square in order to register a successful run of the maze.

The robot has three obstacle sensors, mounted on the front of the robot, its right side, and its left side. Obstacle sensors detect the number of open squares in the direction of the sensor. On each time step of the simulation, the robot may choose to rotate clockwise or counterclockwise ninety degrees, then move forwards or backwards a distance of up to three units. It is assumed that the robot's turning and movement is perfect. If the robot tries to move into a wall, the robot stays where it is.

The robot's score for the maze is equal to the number of time steps required to execute the second run, plus one thirtieth the number of time steps required to execute the first run. A maximum of one thousand time steps are allotted to complete both runs for a single maze.

Solution Statement

In first run, the robot has no knowledge of the world other than its current location and coordinates of the center. The objective in first run is to traverse the maze and gather as much data as possible by moving and sensing. The more knowledge we collect from the environment, the higher will be the probability to find the shortest path in second run. Ideally if we can record the data for all squares in first run, we would be able to find the shortest path with guaranteed search algorithms like Dijkstra or A*. The strategy I'd like to try for first run is moving 1 cell at a

time in a Depth First Search (DFS) order for unknown squares and sense the walls and open edges for each square and record the observations.. When the robot gets to a dead end or a square which has exhausted all the possibilities for and needs to backtrack, it may move faster(2 or 3 squares) if no data will be lost on the way back. Also because robot can sense number of open squares in 3 directions, it is possible that we get the complete data for some of the squares without even stopping at them. If there are such squares, we can save steps and pass from them with steps longer than 1.

Assuming that we find data for all squares in first run, we can use Dijkstra or A* search if we have a good heuristic function to find the guaranteed shortest path. We need to generate data structure to represent the graph of squares considering graph nodes when the robot moves 1, 2 or 3 squares.

Evaluation Metrics

Evaluation is determined by definition of the problem as the number of time steps required to execute the second run, plus one thirtieth the number of time steps required to execute the first run.

Benchmark Model

The extreme simple case can be a maze without any walls in which the first run takes 2*(n-1) steps to discover the world and n-1 steps to get to the center in second run. The total steps will be 3*(n-1) and the total score will be (n-1)+(n-1)/15.

For a complex maze the first run can have many backtracks and can get close to $2n^2$. For the second run the worst shortest path can be close to n^2. The total steps will be $3n^2$ and The approximate worst case total score will be $n^2 + n^2/15$.

For n=16 the total steps is expected to be between 45 and 768 and the total score is expected to be between 16 and 273.

Project Design

For the first run, DFS should be implemented. For each cell we should keep track of squares in which robot stopped and sensed, also the squares which were discovered without stopping on them and also a stack of visited squares for backtracking.

For the second run, Dijkstra or A* should be implemented based on the data captured in the first run. Also, the robot should move 2 or 3 steps whenever possible.