Felix Seckinger

Udacity Machine Learning Nanodegree

19.12.2017

Domain Background

In the micromouse project a robot mouse is attempting to find the fastest way from a corner to the center of the maze. The maze is unfamiliar 16x16 grid maze. The micromouse is starting in a certain corner of the maze and needs to navigate to a goal that is located near the center. In a first run the mouse maps out the maze in order to find the center. In the following runs the mouse will figure out the best paths to the center and look for the fastest route to the goal area. In this project the micromouse will navigate through a virtual maze. The goal is to obtain the fastest time possible in a series of test runs.

Problem Statement

The main goal of the project is to teach the robot mouse how to get to the center of the maze as fast as possible. The robot is starting at the left bottom corner of the maze. From this starting point the robot uses as less moves as possible to reach the goal in the shortest time possible. The maze consists of an equal number of squares. The height and width of the maze has a certain number of rows and columns. This equal number is 12, 14 or 16. The starting point is defined by walls on the left, right and back sides. Therefor the robot can only move forward in a first step. The goal is defined by a 2x2 square.

In a first run the robot is exploring the maze. Within this trial the robot explores the structure of the maze and finds possible routes to the goal area. A trial is successful if the robot mouse reaches the goal. But after reaching the goal the robot is allowed to continue exploring the maze.

On the second run the robot mouse is using all the collected information and attempts to travel to the goal in the shortest time possible. The performance of the robot mouse is measured by two things. The total number of steps in the first trial is counted and divided by 30. The total number of steps to reach the goal in the second trial. The mouse is only allowed to make 90 degree turns. In a single movement the mouse is allowed to take three steps. After 1000 steps the trial is abandoned.

Datasets and Inputs

The information about the layout of the maze is provided in a text file. Here is the number of squares for each side of the maze described and the structure of the maze defined (including walls and openings).

Because the mouse needs to know about its environment (location, number of walls, intended action, prior information and routes) the location of the mouse is defined by a 2-digit pair. Every action is described by a number from -3 to 3 (steps taken) and a number for the rotation -90, 0 or 90.

Solution statement

Every trial is done to solve a different problem. In the first trial the layout of the maze is identified. The solution would be that the mouse gets knowledge about the maze layout and all paths leading to the goal.

During the second trial the robot mouse is trying to reach the goal in an optimal route.

This setting is easy to quantify. Each step taken by the mouse is counted and the time the mouse is navigating recorded. After the first trial the mouse should be able to choose between different routes and use the shortest path available. The solution can be validated by repeating trials in additional rounds. If the mouse found the optimal route before the additional rounds should not lead to a new solution.

Benchmark model

The benchmark model will be scored by the performance score (problem statement). The score is the sum of number of steps in trial 2 taken plus number of steps in trial 1 divided by 30. By limiting the possible steps t 1000 the performance of the method is going to be very basic. During this 1000 steps the robot is making random decisions where to turn and make the next step. If he is able to find a solution in less than 1000 steps a new benchmark is set to the number of steps in trial 1. The robot should then find an optimized solution in trial 2.

Evaluation Metrics

The evaluation metric to quantify the solution and performance is like mentioned above: score = [Number of total steps in trial 2] + [Number of total steps in trial 1 / 30]. The optimal solution will have a low score. The second trial has a larger impact on the final score than the first trial has.

E.g.: The robot took 500 steps during trial 1 and 10 during the second trial. The score will be 10 + 16,667 = 26,667

Project Design

Here I am defining the project workflow. In order to define the workflow I will analyze the starter code that is given (robot.py, maze.py, showmaze.py, test_maze_##.txt.). The content of the maze is handled in the maze file. Its content is decoded into a virtual maze. After this I will work on the robot.py. Here I will implement algorithms that are relevant for the exploration and optimization trial. Then the benchmark model will be run and the results recorded. By using tester.py I will test the optimized model. The single results will be visualized graphically. In case the model is not optimal the process is repeated until an optimal solution is found. The whole process of the project is documented and reported.

1. Project Parameters
2. Preprocess Maze Data
3. Editing robot.py
4. Running model and recording performance
5. Testing and visualizing results (test.py)
6. Repeat if not optimal
7. Report and document results

For this project I am considering some common algorithms.

- Wall Follower: The mouse is asked to take every turn right or left possible. This should eventually lead to the goal. But the assumption is that the mouse is not caught in a circular loop.
- Dijkstra's: Is an algorithm to find a shortest path between nodes in a graph.
- Dead Reckoning: This algorithm makes random decisions at each step.

- A*: Is recognized as best first search algorithm. It is detecting all possible solutions and chooses the shortest path to the goal.
- Breadth First Search: Is an algorithm for searching tree or graph data structures. The closest neighbours are considered instead of looking for next level neighbours.
- Depth First Search: Is an algorithm for searching tree or graph data structures. It starts at the root and explores as far as possible along each branch.
- Flood Fill: The connected area to a given node is presented in a multidimensional array

Sources:

- Udacity Capstone Proposal template

https://github.com/udacity/machinelearning/blob/master/projects/capstone/capstone_proposal_template.md

- Robot Motion Planning Capstone Project

https://docs.google.com/document/d/1ZFCH6jS3A5At7_v5IUM5OpAXJYiutFuSIjTzV_E-vdE/pub

- Micromouse Competition

http://micromouseusa.com/ and

https://en.wikipedia.org/wiki/Micromouse

- A* algorithm, Breadth-first-search, Width-first-search

http://bryukh.com/labyrinth-algorithms/

- Dijkstra's algorithm

https://www.youtube.com/watch?v=gdmfOwyQlcI

-