East West University

Department of CSE

Assignment

Enhanced Dynamic Robot Movement Simulation

Course Code: CSE366

Course Title: Artificial Intelligence

Submitted By:

Md. Arfan Ahmed

ID: 2020-1-60-139

Submitted to:

Dr. Mohammad Rifat Ahmmad Rashid

Assistant Professor, Department of Computer Science and Engineering

Date of Submission: 05 March, 2024

Introduction:

This report is about the design, implementation of the Enhanced Dynamic Robot Movement Simulation. We have evaluated the performance of two pathfinding algorithms, Uniform Cost Search and A* to determine the optimal path for a robot traveling through a dynamic environment with randomly placed obstacles. The simulation insights into the efficiency of these algorithms, shedding light on how each algorithm influences robot’s movement within the specified grid.

Environment Setup:

The initial phase involved creating a simulation environment represented by a 10x10 grid. The Environment class was implemented to place obstacles, a start position, and an end position within the grid. The obstacles are dynamically placed. So that the robot can dynamically navigate to the goal each time.

Robot Implementation:

The Robot was represented by the Agent Class, equipped with movement capabilities and position tracking. It has methods to manage the robot's energy levels and battery status. Battery management was a crucial aspect, with the robot starting at 100% energy and decreasing the battery level by 10% for each move. It is required to recharge the battery level to 100% when the battery level reached to 0%.

Pathfinding Algorithms:

I have used two pathfinding algorithms, Uniform Cost Search (UCS) and A* (A Star) to find optimal paths in the given environment. The evaluation criteria focused on the number of times the robot needed to recharge its battery while traversing the path to the goal. By the analysis it was made possible to understand each algorithms energy efficiency performance.

Task Optimization and Safety:

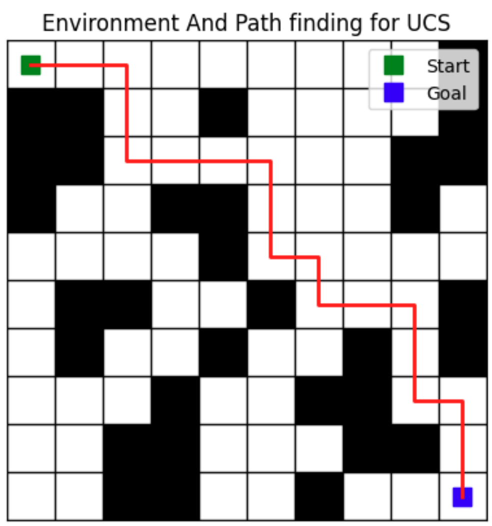
Techniques to reduce the travel time and energy consumption were implemented, and safety was guaranteed by giving the robot the ability to recognize and steer clear of any collisions. The movement of the robot is constrained when encountering obstacles in order to avoid potential collisions. Robot adjust its path to navigate around obstacles, ensuring a safe travel.

Visualization:

I have used libraries like matplotlib to enhanced the project's visualization. The grid, obstacles, paths were visualized, providing a clear representation of the simulation. Visualization played a crucial role in understanding the robot's behavior and the impact of different algorithms on its movement. In the simulated environment, the start position is represented by the color Green, while the goal position is represented by the color Blue. The path taken by the robot is visually represented as a Red line. Grid cells shaded in black are considered as obstacles. adding complexity to the robot's navigation challenges. This color-coded representation enhances the clarity of the simulation, making it easier to interpret the start, goal, obstacles, and the traversed path.

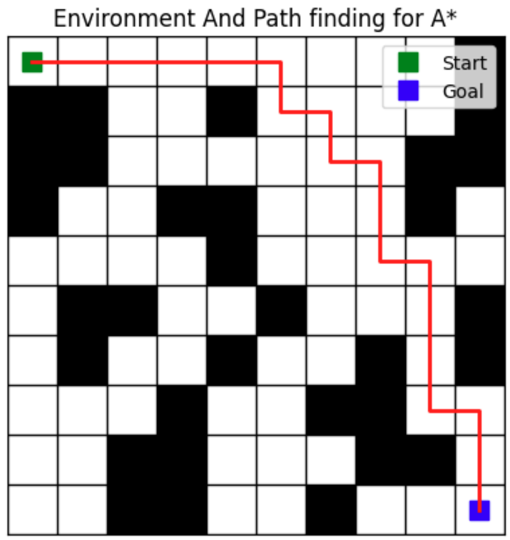
Output:

```
Solution Path for UCS: [(0, 0), (0, 0), (0, 1), (0, 2), (1, 2), (2, 2), (2, 3), (2, 4), (2, 5), (3, 5), (4, 5), (4, 6), (5, 6), (5, 7), (5, 8), (6, 8), (7, 8), (7, 9), (8, 9), (9, 9)]
Final Percentage for UCS: 50
Recharge Count for UCS: 6
Recharge Indexes for UCS: [(3, 2), (1, 6), (5, 4), (7, 0), (6, 5), (9, 5)]
```



On the above simulation output we can observe the Robot path using the Uniform cost search algorithm. It shows us the solution path. The final battery percentage for the robot after reaching the goal stands at 50%. The robot had to be recharged a total of 6 times during its travel through the environment. The recharged indexes indicate the specific grids when the robot autonomously recharged itself, restoring its battery levels from 0% to 100%.

```
Solution Path for A*: [(0, 0), (0, 0), (0, 1), (0, 2), (0, 3), (0, 4), (0, 5), (1, 5), (1, 6), (2, 6), (2, 7), (3, 7), (4, 7), (4, 8), (5, 8), (6, 8), (7, 8), (7, 9), (8, 9), (9, 9)]
Final Percentage for A*: 70
Recharge Count for A*: 4
Recharge Indexes for A*: [(3, 2), (5, 3), (3, 6), (6, 8)]
```



On the above simulation output we can observe the Robot path using the A* search algorithm. It shows us the solution path. The final battery percentage for the robot after reaching the goal stands at 70%. The robot had to be recharged a total of 4 times during its travel through the environment. The recharged indexes indicate the specific grids when the robot autonomously recharged itself, restoring its battery levels from 0% to 100%.

Conclusion:

In conclusion it is clear from the simulation and observation of the robot’s path and battery use that the A* algorithm works more energy efficiently that the Uniform cost search algorithm. When compared to UCS, the A* algorithm showed a more effective pathfinding technique, which led to fewer battery recharges. So, it can be said that A* is better option for guaranteeing energy efficient navigation.