Intelligent Search with Dynamic Obstacles

Venkata Vasudha Kirthi Chodimella, Akash Joshi, Zijie Wang
Project Group 12

Problem Statement

In a perfect world, there are few obstacles when compared to a real-world environment. In the simulation environment many times the problem is simplified. However, in a real-world environment, things are constantly changing. When considering an intelligent path planning algorithm, many times the fundamental search algorithms do not take into account a dynamic environment. The environment remains static, as the agent plans and navigates the world. While this is good to prove that the algorithm works, it may be beneficial to push these fundamental algorithms to the limit to see how they perform under less-than-perfect conditions. For example, when considering a simple path planning algorithm in an environment as dynamic as downtown Tempe, these algorithm may not perform as expected. The sporadic child, the meandering tech-addicted texting college student, or a street that is under construction may all contribute to an infinitely dynamic environment. How will the intelligent agent react? Can the agent adapt to dynamic obstacles and still navigate successfully to the target state? The purpose of this project is to push the limits of the simple search algorithms by introducing a level of uncertainty. In doing so, the fundamental search algorithms shall be improved upon to show that with a little help they can still perform successfully when faced with an ever-changing world.

Motivation

The path planning algorithms of homework 1 used a simplified environment to prove and analyze the performance of each algorithm. While this approach served its purpose, it did not represent a real-world environment. The position of the obstacles were randomly assigned during the creation of the environment. As a result, this enabled the path planning algorithm used by the robot to determine a path, and execute the necessary steps without worrying about any changes that may occur in the environment while navigating. If, however, the environment were to change while the robot was navigating it predetermined path, it is possible that the robot may have collided with an obstacle. For example, if the the robot calculated a path to the goal state, but a new obstacle appeared directly in it path, then the algorithm would have no way of charting a new course. Towards that end, it may be beneficial to build on each of the fundamental search algorithms such that when they are presented with a dynamic environment, there are tools that can be used to safely re-route to the goal state without colliding with a newly discovered obstacle.

Approach

To implement a more robust searching algorithm, a dynamic environment in which obstacle locations are not static will be used. For homework 1, environment obstacles were static. This made the job of the search algorithm used by the robot much easier. The simulation environment used for homework 1 will be built upon to allow obstacles to be placed in the environment at any time after the initial path has been planned with the obstacles that are present at the start of the simulation. Obstacles may be placed directly in the path of the robot while the robot is in motion. As a result, the robot will have to recalculate its path based on the new location of obstacles. Additionally, to allow the robot more freedom when choosing a path to the goal state, the number of available actions that the robot may take will be increase. Specifically, diagonal motion will be added to the robots set of possible actions.

Task Assignment

There are many new tasks that are needed in order to implement the improved searching algorithms. Among them, modifying the simulation environment such that obstacles are dynamically created throughout the simulation. Additionally, an intelligent controller that will allow the robot to dynamically create new paths when faced with an unexpected change in the dynamically changing environment. Last, the controller for the robot needs to have additional actions added to its set of possible actions. Specifically, the robot will have 4 new possible actions added. Northwest, northeast, southwest, and southeast will all be added to the robots set of actions. After an initial design has been agreed upon, each of the four members will contribute to a feature that is best suited for project completion.