RBE-550 Motion Planning

Assignment 5
Wildfire

Abstract

With two major classes of motion planning algorithms at our disposal, it's time to put them to the test. Pick both a combinatorial and a sampling-based planning method, and compare their performance in navigating a firetruck across a deadly obstacle field in an attempt to extinguish as many fires as possible.

1 Introduction

Implement a planner that utilizes two forms of motion planning algorithms to navigate a firetruck through a cluttered, maze-like environment. Test two types of motion planning algorithms, first a combinatorial search algorithm, specifically A* or a variant, and then a Probabilistic RoadMap planner.

The firetruck must plan paths to the most important fire to fight, as fires spread, and new fires get started. There is no set algorithm to choose goal points, so use some creativity here. The simulation is run for a specified time, after which performance metrics are calculated. Use this data to generate plots and a brief report on your findings.

2 Environment

The environment consists of a flat square field, 250 meters on a side, filled with obstacles. The obstacles consist of large patches of un-navigable thick brush, trees, and weeds, suspiciously shaped like giant tetrominoes. While the environment is not specifically a grid, the base dimension for each obstacle square unit is 15 meters. Inside this field, a firetruck operates, attempting to extinguish fires that emerge.

Starting at time 0 and at 60 second intervals, an arsonist/wumpus sets a major conflagration at a random obstacle. This sets the obstacle state to *burning*. After 20 seconds in this state, the
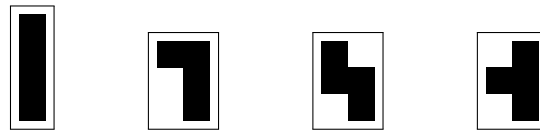
Figure 1: Example free tetromino obstacle shapes.

obstacle sets all obstacles within a 30 meter radius to the state of *burning*. Run the simulation for **3600 seconds**.

The truck starts at a random point in the map. If the truck stops within 10 meters of a burning obstacle, it sets the state to *extinguished*. Use a path planner to drive the truck to desired locations and attempt to extinguish as many obstacles as possible.

3 Implementation

Use any software to create a simulation of this firefighting world. Python, C++, MATLAB, ROS, or any other systems are acceptable. Third party libraries may be used for any component, including the planner implementations. Be clear in your report and delineate what you wrote versus external sources.

3.1 Obstacles

Create the obstacle field similar to the grid world assignment. That is, create obstacles from tetrominoes again (as in Figure 1). Use your magic Tetris generator to fill the environment to 20% obstacle coverage. Obstacles don't necessarily need to be orthogonal to the world, but it might be easier.

3.2 The Robot

The firetruck, being a Mercedes Unimog, is a car-like robot, with standard Ackerman steering (Explained). See Table 1 for the vehicle parameters. A full dynamic model is optional, but kinematics (including collision) must be considered. See Figure 2 for an example of the kinematics. Assume that the firetruck robot has air support from a small drone, in that an omniscient perception of the environment is provided to the planner. The maximum velocity and minimum turning radius must be enforced, but instantaneous acceleration is allowed.

3.3 Combinatorial Planner

For the first planner, implement A* or a variant. The search graph may be either a grid obtained through cellular decomposition, or a state lattice. Any graph size or density is allowed. See Chapter 2 of LaValle [2006] for examples.

3.4 Sampling-based Planner

The second type of planner utilized in this simulation is a sampling-based planner. Specifically, implement a Probabilistic RoadMap Planner, as discussed in Chapter 5 in LaValle [2006]. The RoadMap itself should be created a priori before simulation start. Note that a local planner must be used to generate kinematically correct paths.

4 Results

Run five 3600 second iterations of each firefighting scenario. Each iteration should have a different random obstacle field and arsonist pattern. But each at iteration an identical scenario must be run for each planner method.

At the end of each scenario, sum up the number of obstacles in each state (*intact, burning, extinguished*). Calculate two ratios: $\frac{N_{intact}}{N_{total}}$ and $\frac{N_{extinguished}}{N_{burned}}$, where N_{burned} is the total number of obstacles set to *burning* over the entire simulation. Sum these metrics over all iterations for each planner method, and create two bar charts showing the results.

Instrument each planner implementation to measure the execution time. (This includes roadmap generation). Create a bar chart comparing the CPU time required for each method, summed over all of the iterations.

Additionally, pick a particularly interesting scenario for each planner method, and create an animation of the simulation run (sped up to 10x realtime). Write a brief report discussing the results of this experiment. Which method is better suited for fighting fires? Does one method plan more efficient paths? Are there advantages to having a roadmap computed a priori? Which method requires more computational resources?

Submit full source code (not including third party code) as an appendix to your report.

5 Grading and Submission

This assignment is due 2021-11-15 @ 12:00 UTC. *Late submissions are not accepted.* Upload completed assignment components (as individual files, not a single ZIP or TAR file) to the course

Table 1: Truck parameters

width	2.2 meters
length	4.9 meters
wheelbase	3 meters
minimum turning radius	13 meters
maximum velocity	10 meters per second

site on Canvas.

Weight	Type	Component
15%	source	combinatorial planner implementation
15%	source	sampling-based planner implementation
15%	PDF	intact ratio figure
15%	PDF	extinguished to burned ratio figure
10%	PDF	computational resources figure
10%	PDF	brief report
10%	video	combinatorial planner animation
10%	video	sampling-based planner animation

6 References

Engineering Explained. Ackerman steering - explained. URL `https://www.youtube.com/watch?v=oYMMdjbmQXc`.

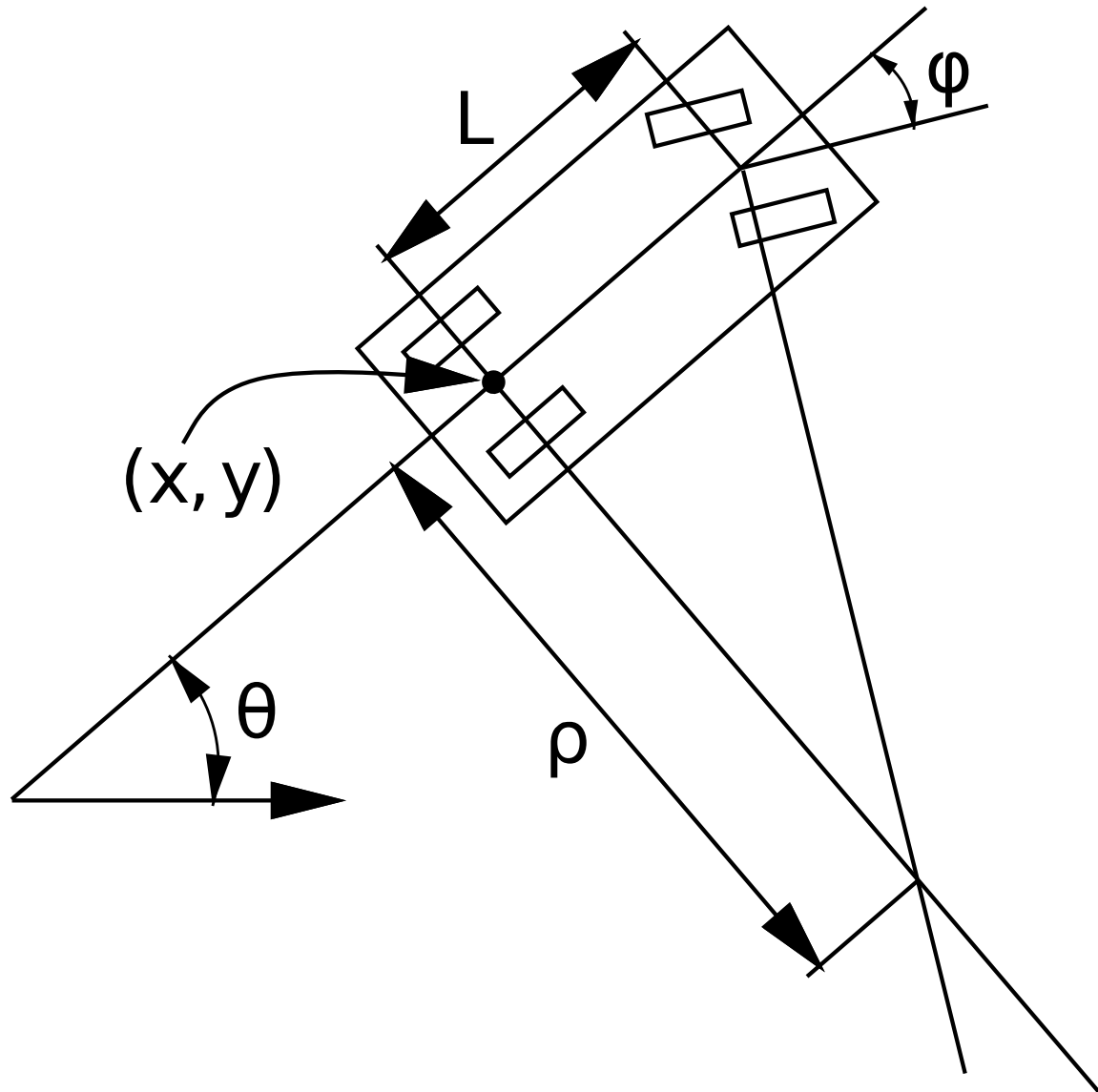Steven M. LaValle. *Planning Algorithms*. Cambridge University Press, May 2006. ISBN 9780521862059. URL `http://lavalle.pl/planning/`.

Last update: November 8, 2021

Figure 2: Trailer kinematics