

COMP 417 Assignment 1

Due: Sep 30, 2022 at 14:00pm

1 Objectives

The goal of this assignment is to implement the RRT algorithm for three different kinds of robots in a 2D plane. You will be given a starter code that implements some functionality of RRT. You will need to modify multiple functions which are annotated with "TODO" in the files `rrt planner point robot`, `rrt planner line robot` (Note they are the same file initially). Furthermore, you are allowed to make your own classes and keep variables around as you see fit. You do not need to make modifications to any other files. It is however recommended that you read the full code for your understanding. You can test the performance of your RRT implementation with different starting and target positions. Note you are given two example worlds, `shot.png`, and `simple.png` but are encouraged to test your algorithms on your self-made worlds. The code is in python3.

2 Simple Point Robot(60%)

This section uses a simple point robot with freedom to move in x,y and is worth 60 points. You will code in `rrt-planner-point-robot.py` and edit the TODO parts for part 1. For this section, the settings are as follows:

- `shot.png`, `simple.png` as the worlds.
- `startposition = (10, 270)`
- `targetposition = (900, 30)`

As part of this section, you will need to implement the following:

- (A) Implement RRT for omnidirectional point robot. You will expand the RRT with uniform sampling.
- (B) Implement RRT but instead of uniform sampling, sample with a Gaussian distribution centered on the destination. Does sampling the world

from different distributions when expanding RRT have any effect on the planner? Please Discuss. [Note you can use python random or NumPy libraries to sample]

- (C) Return to part B but this time with varying smallstep. E.g choose at least 10 different smallstep and for each run at least 10 trials. You will plot the following bar charts.
- (a) num-rrt-iterations vs step-size
 - (b) rrt-path-length vs step-size

3 Line Robot(40%)

The line robot is of unit length and moves in (x, y, θ) . You will edit rrt planner line robot.py and add code to any TODO parts. For this section, the settings are as follows:

- shot.png, simple.png as the worlds.
- *startposition* = (100, 630, 0)
- *targetposition* = (800, 150, 0)

For this section, note that the robot is a line robot and you must be careful to avoid collisions with obstacles. For instance, if you are too close to an obstacle, the line robot may not turn in that direction as it could be blocked.

- (A) Implement RRT algorithm for a line robot on 2D plane. Use uniform sampling for RRT. You can use a fixed robot length of 25(pixels).
- (B) Explore the effect of line robot size on the performance of the planner. You will need to rerun A with 10 different robot lengths ranging from 5(pixels) to 50(pixels). Run at least 10 trials for each robot length and plot the number of rrt iterations vs length of robot.

4 Things to submit

You will need to submit both your code and a report(in pdf format please) zipped together. The report will need to contain the following:

1. Discussions for 2A, 2B, 2C as well as 2 images(plots) for 2C. For 2C, Discuss the plots and propose a good step size according to your experiments.
2. Discussions for 3A, 3B as well as an image(plots) for 3B.
3. Try to keep the report to at most 2 pages excluding figures. Write the interesting things and difficulties you encountered.
4. If you tested with different worlds than the ones provided, please submit them as well.

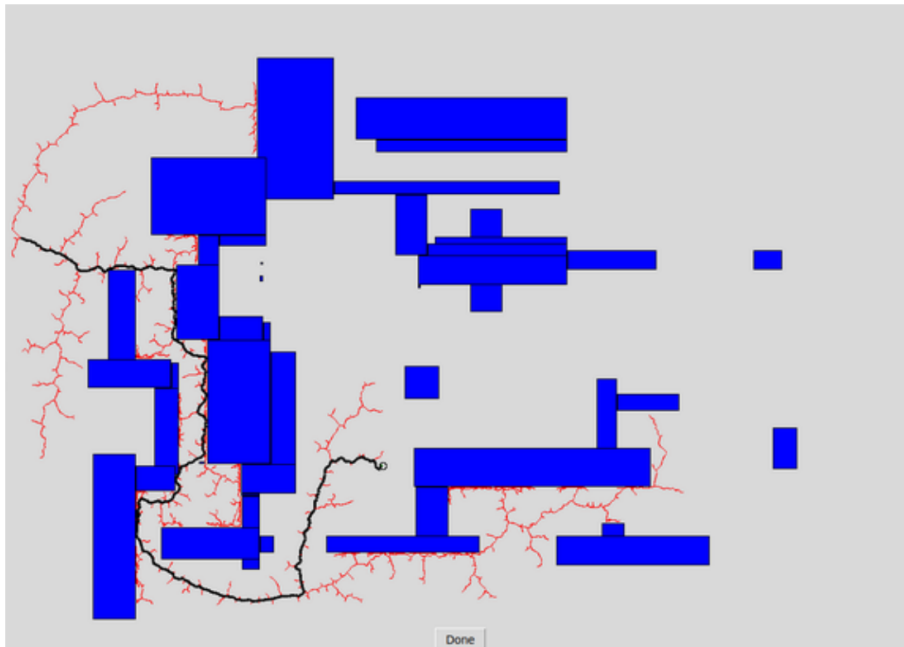


Figure 1: The final result of your algorithm should look somewhat like this for the shot world. Note that different seed will result in different paths.