Middle East Technical University   Department of Computer Engineering

CENG 460

Introduction to Robotics for Computer Engineering

Spring 2021-2022

Assignment 4 - RRT with Tricycle Model

Due date: July 6th, Wednesday, 23:55 (No Late Submission)

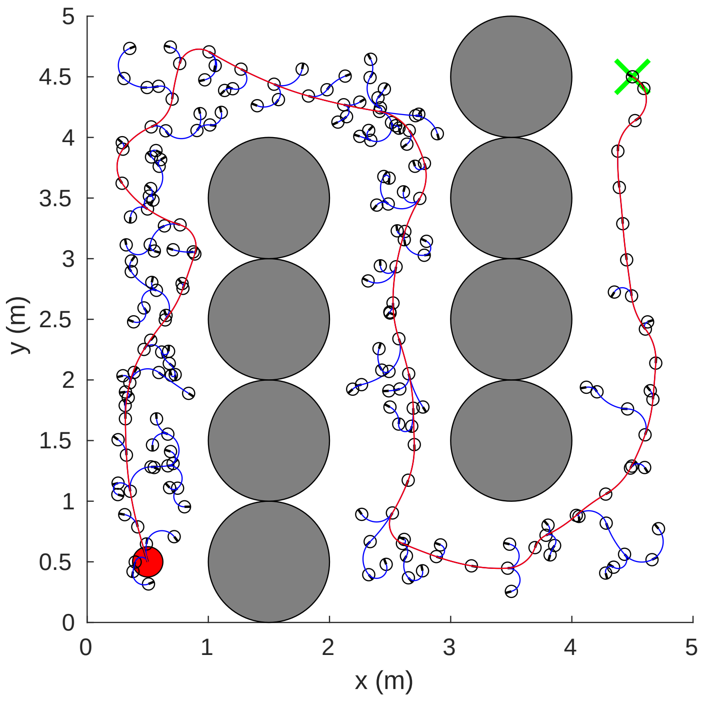

Figure 1: A Sneak peek of the final result.

1 Introduction

In this assignment, you will implement the RRT algorithm laid out in [1], p. 234 for the mobile robot in Assignment 3 with some modifications:

- **The Tricycle:** For simplicity, we will assume that the origin of the body frame, the center of the circular body of the robot and the center of the fixed wheel are coincident.

- **Edges are Circular Arcs:** Adhering to the wheel constraints, the local planner Δ is assumed to generate trajectories that draw circular arcs going forward around the ICR with a linear velocity of 1 m/s. These arcs are assumed to be no larger than semi-circles. The constant magnitude of the velocity ensures that edge lengths are mostly the same within the RRT (determined by the `step_size` parameter of the algorithm).
- **Sampling Positions Instead of Full Configurations:** Since the tricycle model is not very holonomic, the possibility of a uniformly sampled point being reachable by the local planner from another configuration is low. To that end, we will ignore the orientation of the robot when sampling a point. The orientation will depend on its parent in the RRT if it is added to the tree.
- **ϵ-Greedy Sampling:** With probabiliy ϵ, we sample a goal position p_{goal}. Otherwise, we sample a free point uniformly from the map. This ensures that the RRT will eventually hold a path towards p_{goal} from its root.

A significant part of the algorithm body is implemented for you in `three_wheel_RRT_one_step.m`, along with the code for sampling and collision checks.

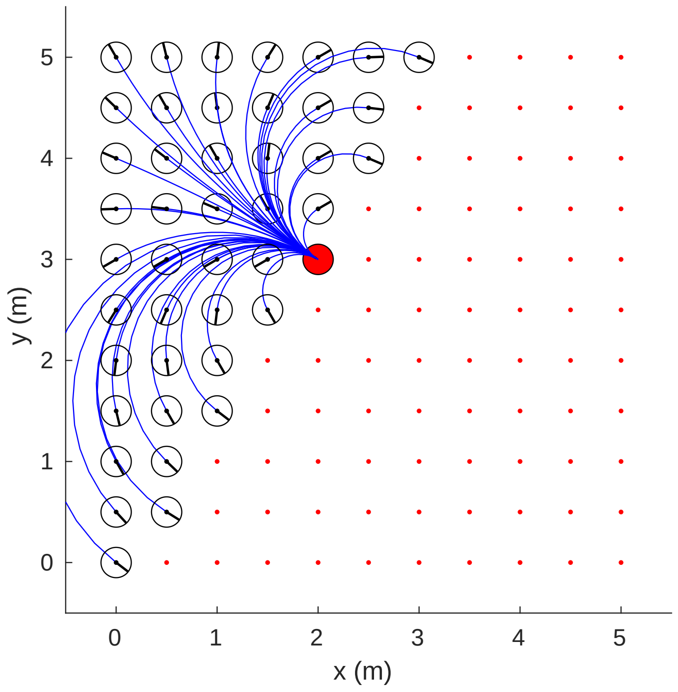

Figure 2: Testing the local planner.

2 Implementation

1. **The Local Planner Δ (20pts):** Implement the function within `local_planner.m`. Test your implementation with `arc_test.m` (Figure 2).
2. **Closest Vertex (30pts):** This step corresponds to line 1, Algorithm 11, p. 234 in [1]. Following the specifications in the comments, implement it within `three_wheel_RRT_one_step.m`.
3. **Steer (20pts):** This step corresponds to line 2, Algorithm 11, p. 234 in [1]. Following the specifications in the comments, implement it within `three_wheel_RRT_one_step.m` using Δ.
4. **Experiments with RRT (15pts):** The main RRT algorithm should be ready if you have completed the above steps. Test your implementation with `three_wheel_RRT_example.m` by animating/plotting the RRT (Figure 1). Report the effects of low/moderate/high values of `step_size` and ϵ with plots.

5. **Goal Path (15pts):** Implement the algorithm for finding the path from the root towards p_{goal} in `three_wheel_RRT_find_goal_path.m`.

3 Tips

- There are no easy-to-access reference/pointer types in MATLAB and it is call by value (everything is sort-of copied, even non-primitive types). This makes tree operations a little bit more complicated than other programming languages. To make things easier, a simple tree structure and some example algorithms implemented for it are provided for you with the assignment text. Feel free to use/obtain tricks from them in your implementations.
- You are highly encouraged to go over and tinker with plotting functions. Similar to the example tree algorithms, there are valuable hints hidden within them.

4 Other Specifications

- All of the rotation matrices are to be assumed right handed.
- The code supplied with the assignment text uses Corke's Robotics Toolbox (see Assignment I text for installation instructions). You are allowed to use the functions this code utilizes (`trot2` and `transl2`). For other functions/utilities, please publicly ask in ODTUCLASS. We may or may not allow it.
- Your programming assignment will be reviewed in a glass-box fashion, therefore please be diligent with your comments in your implementation (you do not have to put comments in every line, just put comments to places that you think it is complicated for the reader to understand. Your comments should provide information about *why* rather than *what*. Avoid "`a = a.*b % element-wise multiply a and b`" sort of comments).
- Please ask your questions regarding the assignment publicly in ODTUCLASS, unless your question reveals part of your solution. In that case, send a mail to `onem@ceng.metu.edu.tr`.
- **This is an individual assignment. Using any piece of code that is not your own is strictly forbidden and constitutes as cheating. This includes friends, previous homeworks, or the Internet. The violators will be punished according to the department regulations.**

5 Submission

Submit your solution as a compressed zip file name `hw4_eXXXXXXX.zip` to ODTUCLASS, containing:

- The report containing your experiments with RRT parameters and plots.
- All of your matlab code regarding the plots within your report. The plots should be reproducable.
- Filled-in stubs: `three_wheel_RRT_one_step.m`, `local_planner.m`, `three_wheel_RRT_find_goal_path.m`.

References

[1] Choset et al., Principles of Robot Motion: Theory, Algorithms, and Implementation, MIT Press, 2005.