Foundations of Robotics

Lab 7: Trajectory Generation

Overview

In this lab, we are going to implement the 3rd order polynomial time scaling and apply it for each segment of the trajectory and for both x and y coordinates.

Waypoints will be provided in the sample script, and should be included in the boundary/continuity constraints when generating trajectories.

Please control the Turtlebot robot to follow generated trajectories to visit all waypoints and plot the figures of the final/actual trajectories.

Submission

- Files to submit:
  - trajectory_generation.py (or .cpp)
- Grading rubric:
  - The robot can visit all waypoints by following 3rd-order polynomial trajectories. The absolute distance error for each waypoint should be within 0.1m.

Programming Tips

- We provide a sample script `trajectory_generation.py` for you. You need to complete `move_to_point` and `polynomial_time_scaling_3rd_order` functions in this code, and make sure the robot can pass all the waypoints. However, it is not required to use exactly the same code structure as that in this sample code. Alternatively, you can make changes to any part of this code, as long as you can have the robot visit all waypoints using 3rd-order polynomial trajectories.
- Please revisit the example in Waypoint Navigation section of the lecture slides, and pay attention to the tricks we discussed in the last slide, as they can be helpful for your design.
- In order to have a better numerical stability, we recommend using multiple relative time durations for polynomials in each segment of the trajectory.

Visualization

- You can reuse the visualization python script provided in Lab 3 to plot the trajectory. Remember to adjust the limits on x and y axes and include the plot in the submission.
- An example of the trajectory is provided as follows. It is a bit overshooting. You can do better :)