EC4.403: Robotics: Planning and Navigation **Spring 2022**

Assignment-2B: Time-scaling non-holonomic trajectories

Released: April 14th, 2022 *Deadline: May 1st, 2022*

Instructions

- You are required to use the code from Assignment-2A for this assignment.
- Submit your code files and a report in the GitHub classroom repository. Submit the same individually on Moodle as well. Do not upload the simulation videos to the assignment repository.
- Provide brief explanations on the approach followed, approximations and assumptions used and screenshots of simulations generated.
- Provide the OneDrive/Google Drive links to the generated simulation videos in the report or in the repository README.
- Plagiarism check will be done on all submissions, so please do not copy. If found, you will be given a straight zero for the assignment.

Constant time-scaling

Given a holonomic dynamic obstacle with precisely known trajectory, you are required to time-scale the trajectory of a non-holonomic robot so that collisions are avoided. Assume a fixed trajectory for the dynamic obstacle with **constant velocity** (say, a straight line trajectory, or a polynomial/Bernstein trajectory between fixed start and goal positions). Make sure there are indeed collisions between the robot and the obstacle without time-scaling. You are expected to reuse the Bernstein non-holonomic trajectory generation code you wrote for Part A of this assignment for the trajectory of the robot.

Time-scaling by a constant may be written in the following form:

$$\dot{x}(\tau) = k\dot{x}(t) \tag{1}$$

Pick the constant such that the robot satisfies its **max and min velocity limits.**

For choosing the scaling constant, you are required to test two approaches:

- **Rule-based constant:** Fix a sensor radius of the robot, within which collisions with an obstacle may be detected. Scale the trajectory of the robot to avoid collisions with the obstacle once it reaches within the sensor limit by a pre-defined constant.

  What problems do you notice with this simple approach?

- **Collision-cone based:** Find a range of values for the appropriate scaling constant k based on the Velocity Obstacle approach discussed in class. As a hint, the constant should be the solution of a system of inequalities coming from the velocity limits and the collision cone. You may find the following useful: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.linprog.html. Choose any of the suitable constants and scale the robot trajectory to avoid collisions.

Linear time-scaling

Linear time-scaling may be written as:

$$\dot{x}(\tau) = s(t)\dot{x}(t) \tag{2}$$

where the scale s linearly depends on time as follows:

$$s(t) = a + bt \tag{3}$$

Note that for this question, the collision-cone based approach is not required, you may pick pre-defined constants a and b for time-scaling.

Based on your observations and experiments, answer the following questions **theoretically**. There is no need to submit codes for answering these questions, however, if you perform experiments to support your answer **BONUS marks** will be awarded.

- How can we ensure smoother robot trajectory using time-scaling to avoid discontinuities?
- You have been taught Model Predictive Control for robot trajectory planning in class. Suppose you start with an initial guess for the robot trajectory. How would the trajectories given by MPC vary from that given by simple time-scaling with respect to the initial guess, given both are able to avoid collisions successfully?
- How would you extend time-scaling to a system of two robots trying to avoid collisions with each other?

An example simulation:

shorturl.at/akvA3

Note that this simulation is for a multiagent case, you are not required to generate an identical simulation for multiple agents. You can assume a pre-determined trajectory for the obstacle.

Deliverables:

- Codes for the algorithm and all experiments. You can use Python or MATLAB. Please ensure the code is well written, and we can ask you to explain certain snippets of it during the evaluations.
- Links to **at least three** simulation videos with varying start, goal positions, waypoints and velocity/orientation constraints, and 1 video avoiding a static obstacle.
- A LaTeX report showing the derivation, answering the questions and containing screenshots of results with explanations.

Feel free to reach out to the TA for any queries. All the best!