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

Assignment-2A: Fitting polynomials to non-holonomic trajectories

Released: March 15th, 2022 *Deadline: March 31st, 2022*

Instructions

- **This is a 2-part assignment. The current deadline is for only this half of the assignment.** The second part will be released after the submission of the current half, and you will be required to use the code and observations from this part to complete it.
- **Deadline extensions will not be provided for the current half of the assignment to allow adequate time for the second half.**
- Submit your code files and a report in the GitHub classroom repository. 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.
- Starter code in Python has been provided in the assignment repository, but if you are not comfortable with Python, feel free to use MATLAB.
- 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.

Fitting Bernstein polynomials to Non-holonomic robot trajectories

The goal of this task is to fit Bernstein polynomials (`https://en.wikipedia.org/wiki/Bernstein_polynomial`) to model the trajectory of a non-holonomic robot. The robot is to navigate a two dimensional space, travelling from its initial location to a goal location. The robot should also be able to navigate through a waypoint along the trajectory and satisfy boundary constraints (i.e, initial and final velocity, acceleration, orientation etc).

The starter code provided contains the Bernstein basis polynomials of order-5, along with their product polynomials. You are required to answer the following questions in the report, and support your answers with simulations/observations from your experiments. **Note that flat theoretical answers may not fetch as much credit as answers backed with experimental observations**.

- How would you use Bernstein polynomials in the case of multi-rotor UAVs? (code not required, a brief explanation is sufficient)
- What changes/constraints are to be introduced to accommodate more waypoints?
- How would you avoid collisions of a non-holonomic robot with trajectory fitted using Bernstein polynomials, with a static obstacle? (no time for time-scaling, simulate a video showing avoidance of a single static obstacle)
- How can we ensure multiple non-holonomic robots with trajectories approximated using Bernstein polynomials do not collide? (intuitive understanding sufficient, no need for code)
- What are the advantages of using Bernstein polynomials over other approximation techniques?
- BONUS: Try with higher/lower order of Bernstein polynomials and report your observations.

For the BONUS, you may find it helpful to use Mathematica for computing the product polynomials(https://www.wolfram.com/mathematica/).

Generate simulation videos for the robot trajectories and plots of the evolution of the x and y-coordinates with time. In case of velocity, acceleration or orientation constraints, show their necessary plots vs time to ensure the correctness of your algorithm.

Follow the instructions in the starter code to save the snapshots of the robot trajectory at every step, and then stitch the snapshots together using the script **mkmovie.sh** to generate the simulation videos as in the previous assignment. Refer to the repository README on GitHub classroom for instructions to use the video-generating script.

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.
- x vs time plots, y vs time plots and velocity/acceleration/orientation plots for all the experiments, with clearly marked start, goal points and waypoint.
- 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!