Data-Driven Control Systems Design **(Due: 22/12/2023)**

Forward and Inverse Data-Driven Control for Rovers Navigation

Author: Prof. Giovanni Russo **Project**

Instructions: Read carefully the instructions below before you start working on the project and before you make your submission. The description of the project is given on the next page.

- Groups will find on elearning a partially-filled jupyter notebook corresponding to their project. The file, read jointly with the project description, can drive you throughout the development of your project;
- Each group (2 or 3 students) will need to return a fully working jupyter notebook that tackles the project. The *skeleton* jupyter notebook on elearning can be used freely by each group;
- Each group will have to produce, together with the code, a report (max. 3 pages plus a free number of pages for the appendix - this can contain supporting figures and implementation details). You will need to submit a report that also serves as a documentation of your code. The code needs to be sent to the instructors together and has to be fully working;
- While the structure of the report is free and needs to be determined by each group, it needs to outline at least the following elements: (i) what is the formulation of the control problem; (ii) the design choices you decided to follow and why (e.g. why you used a specific algorithm); (iii) relate your code to the methodology; (iv) numerical results (e.g. plots from simulations) and discussion on the performance; (v) discussion on limitations, possible future developments and improvements.
- **You need to submit the code by the date on top of the page. The report can be submitted one week before the exam date. Each student needs to state to which parts he/she contributed**;
- At the exam, each group will present the work. Groups will have to prepare a presentation (approx. 20 minutes) and the time between the students needs to be split equally;
- While the structure of the presentation is free and needs to be determined by each group, it needs to contain at least the following elements: (i) problem formulation, illustrating how the problem was formalized starting from the high-level description (and the code); (ii) which algorithms you decided to use and why; (iii) numerical results and discussion on the performance; (iv) possible improvements;
- The assessment will include an interview, individual for each student. During this step, each student will be asked questions aimed at checking the individual understanding of the design choices, the methodology behind these choices and how he/she would further develop over the solution found (if applicable). The questions will be related to all the parts of the project. Moreover, questions can be related to any of the aspects seen during the lectures that are related with the developed solution;
- The project tasks have been organized in steps and these steps reflect the grading system. The grading system is given below. Further, projects will be benchmarked against each other;
- The report, the presentation and the interviews will be in Italian;
- All relevant files will need to be sent to `giovarusso@unisa.it`.

Grading system. The final grade takes into account: (i) the quality of the code-base produced by the students (weight: 20%); (ii) how each WP has been solved (30%); (iii) how the results were presented (20%); (iv) how each student answered to the individual interviews (30%). Each student can obtain the *cum laude* grade only if: (i) the project was correctly completed (mandatory WPs); (ii) the individual student successfully completes at least two WPs; (iii) the interview is outstanding (demonstrating full knowledge and critical assessment of the methods). Projects from different groups will be benchmarked against each other and this *competitive* element will be factored into the final grade. The code/report from each group of students must be of their own.

Project description

Context. Data-driven algorithms will be developed allowing unicycle robots to navigate in uncertain environments so that they: (i) reach a desired destination; (ii) avoid obstacles. In this context, students will design two algorithms that are crucial to enable autonomous navigation: (i) a forward data-driven control algorithm to compute the optimal policy; (ii) an inverse data-driven control algorithm to estimate, from observations, the robot navigation cost. Moreover, students will validate their results by leveraging a state-of-the-art robotics platform: the *Robotarium* by *Georgia Tech*. The platform offers both a high fidelity simulator and a remotely accessible experimental hardware facility. In their mandatory WPs, students will use the simulator to validate their results. Additionally, by using the Robotarium APIs, students will also have the opportunity to deploy their algorithm on real robots and obtain a hardware validation of their algorithms (this last WP is optional).

Functions description. The problem of properly controlling robots so that these can autonomously navigate in an uncertain environment has attracted much attention. This is largely due to the many engineering applications, spanning from automated warehouses to spatial exploration, for which autonomous navigation is crucial. Students will therefore design a full data-driven control pipeline that:

- handles robots with noisy sensors that need to navigate in environments characterized by large uncertainty;
- returns the optimal policy for autonomous navigation. By leveraging data, i.e. without knowing the robot's model, the control policy will allow robots to reach the desired destination while avoiding obstacles;
- allows to reconstruct the navigation cost used by the robots from observations. Once the cost is reconstructed, it can be exported to other robots, thus allowing them to also fulfill the control task. By using the estimated cost, these new robots can sinthesize a policy from demonstrations going beyond imitation;
- simulate, validate and visualize the results;
- critically discuss the performance and identify potential improvements.

Specific tasks. In order to successfully complete the project, this has been broken down into the following work-packages (WPs). Within a group, each student will be responsible of one or more WPs (WPs need to be equally split among students) and this has to be clearly indicated in both the report and the presentation.

WP0: formalize the control problem, analyze and understand the supporting code on elearning. Define the key elements of the problem. Relate the tools you plan to use to the methodology seen in class. The group should provide a formal problem statement of the control problem (direct and inverse);

WP1: compute the policy to tackle the forward control problem. Discuss how the solution compares to the other methods seen in class (without coding);

WP2: simulate and visualize the robots trajectories in the environment provided. Critically discuss the results, also using initial conditions different from the ones given in the notebook;

WP3: reverse-engineer the features that have been provided and critically discuss them. Are these good features? Why?

WP4: solve the inverse problem and obtain an estimated cost. Plot the estimated cost and use this cost in the forward problem. Does it allow the robot to complete the task? Critically discuss the results, also using initial conditions different from the ones given in the notebook;

WP5: determine what is the weight of each feature (consider a plot) and propose (without necessarily coding): (i) one different set of features; (i) one different cost that you think will work for the application;

Optional: deploy the policy (with estimated cost) on the Robotarium hardware and validate it on the real robots. Record a video of the experiment and discuss the results.

Note: the Jupyter notebook includes a number of *tasks*. These need to be solved in order to complete the WPs. By *implementing, developing,...* it is meant that the corresponding task needs to be fully executed and the code (if any) fully working.