Assignment 03: Optimal Control

Andrea Del Prete*- andrea.delprete@unitn.it

May 24, 2020

1 Description

The goals of this assignment are:

- making practice using optimal control algorithms for generating reference trajectories;
- tracking the optimal reference trajectory on a system with disturbances and modeling errors;

2 Submission procedure

You are encouraged to work on the assignments in groups of 2 people. **If you have a good reason to work alone, then you can do it, but this has to be previously validated by one of the instructors.** Groups of more than 2 people are not allowed. The mark of the first two assignments contributes to 10% of you final mark for the class (i.e. 3 points out of 30). The final assignment (this one) contributes to 40% of your final mark (i.e. 12 points out of 30).

When you are done with the assignment, please submit a single compressed file (e.g., zip). **The file name should contain the surnames of the group members**, and it must contain:

- A pdf file with the answers to the questions, the **names and ID** of the group members; you are encouraged to include plots and/or numerical values obtained through simulations to support your answers. **This pdf does not need to be long. The strict page limit for the text is 4. You can add other pages for plots, tables and references.**
- The complete *arc* folder containing all the python code that you have developed.

If you are working in a group (i.e., 2 people) only one of you has to submit.

Submitting the pdf file without the code is not allowed and would result in zero points. Your code should be consistent with your answers (i.e. it should be possible to produce the results that motivated your answers using the code that you submitted). If your code does not even run, then your mark will be zero, so make sure to submit a correct code.

*Advanced Optimization-based Robot Control, Industrial Engineering Department, University of Trento.

3 Optimal Control

This assignment is flexible in order to allow students to choose the level of challenge that is appropriate to their understanding of the class material. The overall goal of this assignment is to let you play with optimal control algorithms so that you can gain experience with them. At the end of the assignment you should know which parameters are important when using an optimal control method, and how optimal control can be coupled with reactive control for trajectory tracking on a system subject to disturbances and modeling errors.

3.1 Optimal Control Algorithm

First of all, you need to choose a control algorithm. In class we have mainly focused on Single Shooting and Differential Dynamic Programming. It is recommended to choose one of them. You could also choose to implement Collocation or Multiple Shooting, but I expect that to be more challenging.

3.2 Robot and Task

As a second step, you have to choose a robot and a task that the robot should achieve. Here are some examples:

- UR5 robot that has to clean a surface with a sponge.
- UR5 robot that has to cut a specific shape with a laser.
- UR5 robot that has to drill a set of holes in a wall.
- UR5 robot that has to perform a pick and place task.
- Double pendulum that has to swing up.
- Quadcopter that has to fly through a window.

Feel free to choose a task that is not in the above list. When choosing a task, beware that:

- Tasks involving contact are harder than tasks without contact.
- Tasks involving complex constraints are harder.

3.3 Problem Formulation

As a third step you should transform the abstract description of the task into an optimal control problem. This boils down to formulating the cost function and the constraints. Describe both of them in details, trying to make it as realistic as possible. For instance, you can try to account for joint position bounds, motor torque bounds, motor velocity bounds, collision avoidance.

3.4 Implementation

Once you have your optimal control problem written down, you need to figure out how you can implement it in Python, starting from the code of the class. If you use Single Shooting, then you can easily extend it to account for inequality constraints, which you can use to model any bound. If you use DDP, then it's not easy to include inequality constraints, but you can approximate them using penalty functions with large weights.

For instance, you can have a cost term that is zero if the joint torques are within their bounds, and grows quadratically when they violate their bounds, such as:

$$
\begin{aligned}
c(u) &= 0 && \text{if} \quad u_{min} \leq u \leq u_{max} \\
&= ||u - u_{max}||^2 && \text{if} \quad u > u_{max} \\
&= ||u_{min} - u||^2 && \text{if} \quad u < u_{min}
\end{aligned}
\tag{1}
$$

When implementing new cost terms and constraints, start always by testing them relying on finite differencing for the computation of gradients/Jacobians. This is inefficient, but it allows you to debug one problem at a time. Once everything is working with finite differencing, then you can implement derivatives, checking that they give you roughly the same results as finite differencing.

3.5 Trajectory Tracking

Once you have a satisfying reference trajectory computed with the optimal control algorithm, you have to execute it in a realistic simulation environment. In this simulation you will introduce small perturbations and modeling errors, so applying the optimal control inputs computed offline will not result in the desired motion. To achieve the desired motion you will need to use a feedback controller.

If you have used DDP, then you could directly use the feedback gains computed by DDP for stabilizing the trajectory. Note however that DDP will give you reference trajectory and gains with a sampling time that is larger than the one you will use in the simulation environment. For instance, DDP could use a time step of 10 ms, while the simulation uses a time step of 1 ms. To account for this difference in sampling time, you must interpolate between the values given by DDP. For instance you could use either a zero-order hold, or a linear interpolation.

If instead you have used shooting (or if you don't want to use the DDP gains), you can rely on a reactive control method, such as Task Space Inverse Dynamics, Operational Space Control or Impedance Control. These methods indeed take a reference trajectory as input, and are then able to quickly compute joint torques to track this trajectory, rejecting perturbations through the use of state feedback. Choose a reactive control method and use it to track the (interpolated) reference trajectory computed via optimal control in a simulation with modeling errors and Coulomb friction.

4 Questions

Make sure that your report contains the answers to the following questions:

1. Why did you choose this optimal control method?
2. Why did you choose this task?
3. Which options did you consider when formulating the optimal control problem?
4. Which parameters of the optimal control method were critical to achieve a nice result?
5. Why did you choose to stabilize the system with this feedback controller?
6. How well did the controller track the reference trajectory? How could you improve that?
7. What would you do if you had to spend a few months working on this project?