Introduction

This is the final piece of coursework, which will count for 67% of the overall course mark. This coursework is composed of two parts, each with an equal weighting:

- Part A (straightforward): Tracking a square.
- Part B (challenging): You choose one from a list of options.

This assignment is open-ended in order to assess whether you (i) have a deep understanding of the material, (ii) have read beyond the lectures and (iii) are able to solve problems that you have not seen in previous assignments or that were not covered in lectures. You will also be assessed on your ability to clearly and rigorously define and solve mathematical problems. There is not a single, correct solution and trade-offs are inevitable. You will therefore have to demonstrate a critical understanding of the advantages, disadvantages and trade-offs of a solution.

Framework

You should use the MATLAB/SIMULINK files you have been given in previous assignments as a starting point. Please modify these as appropriate in order to complete the tasks below. Though you have already been provided with `pcode` that are answers of previous assignments, you are highly encouraged to use your own `m`-files.

You will be given the opportunity to implement your controller designed in Part A on the hardware during the hands-on session, which will be timetabled during the last two weeks of term. You are therefore recommended to implement and test your controller on the Simulink model given to you in Assignment 3 or 4. Because of time constraints and other unexpected issues, it may not be possible to test your controller designed in Part B during the hardware hands-on session. Note you should not include any results from the hardware session in the report.

Tasks

Part A: Tracking a square

In previous assignments you were asked to design a regulator that would take the system from an initial state to a different target state, while satisfying some constraints. You now have to modify the code to track a time-varying reference signal. In particular, you have to ensure that the mass of the *pendulum* can track a square in the XY-plane with a given amount of error. It is up to you to define the four corners of the square, the maximum error and the time it takes to complete the square, etc.

You should pay particular attention to the following:

- There is no point in implementing constrained MPC if the closed-loop response is the same as for an unconstrained linear controller. You have to compare your results against an unconstrained RHC law with the same cost function on the states and inputs as the constrained controller.

- You should compare results for both the linear and nonlinear simulation.
- Your sample time should be larger than the time it takes to solve a single QP.
- The inputs are constrained between -1 and 1. These are hard, physical constraints.
- The minimum and maximum values that the cart can take in the XY-plane are defined in the variables `xRange` and `yRange` in the file **Params_Simscape.mat**. These cannot be changed and represent physical limits, which cannot be violated. Please bear these values in mind and allow for some safety margin when defining constraints on the states.
- The starting point of the cart is defined by the variables `xZero` and `yZero`. You are allowed to change these, as long as they are within the above-mentioned range.

Part B: Choose one

You have to choose from one of the following options — please choose only one:

1. Discuss the practical challenges that arise with MPC in a continuous process industry plant, both (i) during the implementation phase and (ii) during the entire life cycle.
2. Implement a controller that ensures the pendulum tracks something other than a square, e.g. a triangle, circle and/or any other interesting path in the XY-plane.
3. Implement a tracking controller that uses a preview of a time-varying reference and demonstrate that this does better than a non-preview controller.
4. Implement a predictive controller with a non-quadratic cost on the states and inputs.
5. Implement a predictive controller that can guarantee zero tracking error in the presence of a constant disturbance and/or plant-model mismatch.
6. Implement a predictive controller that requires less computational resources, compared to the controllers designed in previous assignments, without changing the solution to the QPs.
7. Implement a predictive controller that requires less computational resources, compared to the controllers designed in previous assignments, without significantly changing the closed-loop behaviour.
8. Implement a predictive controller that uses 'blocking' for the input sequence.
9. Implement a predictive controller that has different control and prediction horizons.
10. Implement a predictive controller that uses the so-called delta-u formulation, i.e. the change in input $\Delta u_k := u_k - u_{k-1}$ is included in the cost and/or constraints.
11. Implement an explicit predictive control law. See the documentation for the MATLAB Model Predictive Control Toolbox or the MPT Toolbox (`http://people.ee.ethz.ch/~mpt/3/`), as well as the paper `http://dx.doi.org/10.1016/S0005-1098(01)00174-1`
12. Implement a nonlinear predictive controller, e.g. using ACADO (`https://acado.github.io`), ICLOCS (`http://www.ee.ic.ac.uk/ICLOCS`), YALMIP (`https://yalmip.github.io`) or any other suitable software package that you can find.

13. Implement a predictive controller with a guarantee of closed-loop stability and/or invariance. You might find the MPT Toolbox (`http://people.ee.ethz.ch/~mpt/3/`) useful for computing an invariant set.
14. Implement a predictive controller that minimizes the time to reach a target state.
15. Implement an economic predictive controller and compare the performance to one based on a standard tracking/regulator formulation, as implemented in previous assignments.
16. Implement a robust predictive controller that guarantees constraint satisfaction and/or a certain amount of closed-loop tracking error despite the presence of uncertainties.
17. Implement any predictive control scheme that you have come across and have found interesting, but is not in the list above.

As with Part A, there is no point in implementing a predictive controller if the same response can be achieved with an unconstrained linear controller. You therefore have to demonstrate the advantage of your method compared to a similar, unconstrained controller, if appropriate and possible.

For all the options above, marks will be given for elegant discussions that demonstrate a deep understanding, compared to trivial solutions that demonstrate a superficial understanding. Note that some of the options above might wrongly be considered to be 'easier' than others.

Submission details

Your should submit your report, together with all the MATLAB/SIMULINK files needed to generate the results, as a single ZIP file on Blackboard. Do not include any external software packages, if used.

Report

Your report should be submitted as a PDF and adhere to the following:

- Be written with the IEEE Control Systems Society conference paper A4 template.

  A template for LaTeX can be downloaded from
  `http://css.paperplaza.net/conferences/support/tex.php`
  A template for Word can be downloaded from
  `http://css.paperplaza.net/conferences/support/word.php`
- Be in two-column format.
- The title should be your full name and CID.
- Part A and Part B together should not be longer than 6 columns (3 pages), subject to the following additional requirements:
  - Part A should not be longer than four columns (2 pages).
  - Part B should not be longer than four columns (2 pages).

  In other words, one of the parts cannot be longer than three columns (1.5 pages). It is up to you whether you want to assign an equal amount of space to both parts.

Please make sure you clearly define the optimal control problem as well as the optimization problem and list the numerical values that parameterize the controller, e.g. the sample period, horizon length, cost weights, constraints, etc. You do not need to define symbols that have already been defined in previous assignments. However, all new symbols have to be clearly defined. A reader should ideally be able to replicate your results without needing to refer to your code.

Code

You should also write and include the following two MATLAB scripts in the ZIP file:

PartA.m Runs all MATLAB/SIMULINK simulations to generate the data and figures included in Part A of the report.

PartB.m Runs all MATLAB/SIMULINK simulations to generate the data and figures included in Part B of the report.

Marking allocation

- Clarity of expression, grammar, legibility of figures, typesetting and presentation. [25%]
- Are the optimal control problems and algorithms clearly defined, if necessary, with no repeat of material already used in previous assignments and lectures? Are all new variables defined? Has the literature been properly referenced? [25%]
- Are the mathematics correct and the numerical results replicable? Are the numerical values for all control parameters given? Have trade-offs, advantages and disadvantages been explored and discussed? [20%]
- Does the report discuss concepts beyond that covered in lectures? Does the discussion demonstrate a deep understanding of the presented material? [30%]