Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institute for Dynamic Systems and Control
IDSC
Institut für Dynamische Systeme und Regelungstechnik

Dr. T. Albin

151-0310-00 **Model Predictive Engine Control** **Group Work 2**

Topic: **NMPC in Simulink**

Review: 15. 05. 20

May 14, 2020

In the first group work, you implemented a single step of the NMPC algorithm in both MATLAB and Simulink. If all went well, the optimal control trajectories should be the same, and with that you could verify your Simulink implementation. In this second group work, you will continue to use Simulink and implement a complete NMPC algorithm. Once more, a MATLAB template script `main_GW02.m` is provided to facilitate your work. It includes a new reference trajectory for $p_{\mathrm{im,ref}}(\cdot)$ and $x_{\mathrm{bg,ref}}(\cdot)$ — other than the ones introduced previously — that you can use to assess the performance of your controller.

Steps

Use the following list of steps as a guideline to complete this group work.

a) Extend your Simulink implementation of a single NMPC step such that the optimization problem is solved every `parNMPC.Ts` seconds. As in the first group work, simulate the output of your NMPC using the ROM. Use the output of the ROM states as initial conditions for the next NMPC step. In order to validate the correctness of your implementation, you can compare the optimal state and/or output trajectories from the ROM to the first value of each NMPC optimization. For this, the SQP method should be allowed to fully converge, so you need to (temporarily) set the maximum number of SQP iterations sufficiently high (say, a thousand).

b) Add feedback and offset-free control to your NMPC. For the initial development and testing, use the ROM as the simulation model. By adding an artificial offset to the output, you can check if your algorithm works correctly. After that, use the MVM (without delays) as the simulation model. Any plant-model mismatch should be compensated for by the offset-free control. For this, you need to run the ROM in parallel, compare its outputs to the "measured" ones, and apply a suitable correction to the reference that is fed to the NMPC. Moreover, the ROM simulation serves as a simple state observer, which you can use for the initial conditions of your optimization. This is necessary because the MVM, like the real engine, does not provide access to the state variables.

c) Add an output-delay compensation and use the MVM with delays to tune the compensation to obtain satisfactory performance (most importantly, oscillations should be avoided). A good starting point are values of 100 and 300 ms for the delays on p_{im} and x_{bg}, respectively.

d) Tune your NMPC for good reference tracking performance and disturbance rejection.

e) Submit one version of your NMPC for review by the TAs. We will give you feedback in the third group work session. You need to hand in all the files of your solution, consisting of:

- `cSourceFiles` folder and contents
- `sFunctions` folder and contents
- `createCasadiFunctions.m`

- `main_GW02.m`
- `NMPC.slx`

Do not include the `slprj` folder that is generated by Simulink. Create a single ZIP file of the above files/folders and send it to the TAs until midnight of 20.05.2020.