MULTIVARIABLE CONTROL AND COORDINATION SYSTEMS (EE-477)

Case study:

Path tracking for an automated vehicle.

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

http://react.epfl.ch

Denis Gillet, Ezequiel Debada
Lausanne, EPFL, 2018

Exercise 1 Linearization, Discretization and Simulation

In this session we are going to go through the process of linearizing, discretizing, and simulating our system. Our aim is to make clear the practical similarities and differences between the continuous-time-nonlinear, continuous-time-linear, and discrete-time-linear models of the system.

In order to do that, we will implement the three mentioned models of the system in a single Simulink diagram, and will apply a certain sequence of control inputs in open loop. Different kinds of control sequences and linearization points will reveal differences between the studied versions of the model.

1.1 Provided files

- `open_loop_experiments.slx`. Simulink file with some of the blocks needed to simulate the system. In this file you will implement the continuous time non-linear, continuous time linear, and discrete time linear model, which are to be simulated in parallel given a certain sequence of control inputs to be applied in open loop.
- `ex1.m`. Class whose methods have to be completed. A description of every function to be completed is included within the file.
- `exercise1_Linearization.m`. Matlab script that sets up and runs the required simulations. Modify its content only if you want to run other experiments in addition to the ones proposed.
- `utilities.m`. Class that gathers auxiliary functions that will be used in the scripts provided during the case study sessions. There is no need in reviewing its content.
- `circle, path_1, path_2, path_3` mat files containing different paths to track that can be used in the experiments.

1.2 Exercises

Linearization

- Obtain the analytical expression of the nominal trajectories considering that the nominal state should describe a perfect path-tracking situation.
- Linearize the system and obtain the analytical expressions of the matrices A, B, C, D characterizing the continuous-time linear model of the system.
- Complete the method `getSystemParameters` which should return a column vector with the parameters of the system given the values $\kappa(s) = 1e^{-10}$, $L = 4$, $\sigma_a = 5$, $\sigma_s = 1$, $\alpha = 3$.
- Complete the method `getLinealModelArrays` which should return the matrices A, B, C, D given the column-vector with the parameters values.

Discretization

- Complete the method `getDiscreteLinearModel` which should return the matrices Φ, Γ describing the discrete-time linear model of the system, calculated following three different methods: Euler approximation, using the method that exploits the series representation of the matrix exponential (that is using the auxiliary array Ψ as shown in section 2.2.), and using the Matlab command `c2d`.

Simulation

- Complete the simulink diagram so that it applies the same sequence of control inputs to the three models of the system. Keep in mind that, the control inputs and states you receive/send from/to the "from/to workspace" blocks must be the absolute ones (i.e. u and x) and the linear models require $\tilde{u}$ and return $\tilde{x}$.
- Complete the method `getWorkingTrajectory` which should return the nominal trajectory of the control inputs and states in a format that can be used by the 'from-workspace' Simulink blocks.
- Complete the method `getInitialState` which should return the initial state $x(0)$ of the system and the initial state $\tilde{x}(0)$ of the linear models.
- Complete the method `getOpenLoopControlSignal` which should return the sequence of control inputs to be applied in open loop in a format that is compatible with the "from-workspace" simulink blocks. The sequence of control inputs should be designed so that you are able to observe the differences between models.
- Simulate and verify that everything works.

Experiments

- Does the discrete-time linear model emulate sufficiently well the continuous-time linear model of the system? What is then interesting about using the discrete-time linear model instead of the continuous-time linear model to design control strategies?
- Do you observe some differences between the linear and non-linear models? Why is that?
- Try keeping the same control sequence and changing the curvature/speed reference value used to linearize the model. Does it change your previous observation? why?