Practical Exercise

Modeling and Control of Hybrid Systems (sc4160)
2006 – Version 1.1

Delft Center for Systems and Control, Delft University of Technology

---

1 General remarks

- This practical exercise consists of several steps that are outlined in a road map. You should follow this road map and present the results in a clear and concise report. In this report you should clearly explain and motivate all the choices you have made while solving the practical exercise. In your report you should also add an evaluation and conclusions section of max. 1 page in which you briefly outline the main insights you have obtained while making this practical assignment. You should also add the MATLAB files you have written in an appendix to the report.

- The deliverables of this assignment are:

  - a written report about the assignment (to be emailed as a **pdf** file to `b.deschutter@dcsc.tudelft.nl`, or — in case you do not know how to make pdf files — to be delivered as a hardcopy to Bart De Schutter);
  - a zip file containing your MATLAB files (to be emailed to `b.deschutter@dcsc.tudelft.nl`).

- You will be graded on the contents and the presentation of the report, on the originality[1] of your answers, on the correctness, the *efficiency*, the readability of the MATLAB files (i.e., do not forget to include explanatory comments in your MATLAB files), and on your performance during the oral discussion about your report.
  The oral discussion will take place on Tuesday, March 28, 2006 (group 1: 9.00–9.30, group 2: 9.30–10.00, etc.). The deadline for emailing/delivering the reports on the assignment is Monday, March 27, 2006 at 10.00 a.m.

- We recommend you to keep the computations symbolic or analytic as long as possible and not to hardcode any of the parameters in your MATLAB programs (instead, write one separate MATLAB function or script that defines the parameters) so that you can easily take other parameter values, longer control horizons, other reference signals, etc. into account.
  Furthermore, since each step of this assignment depends on the preceding ones, we recommend that after Steps 2, 5 and 6 of the road map, you check your intermediate results with the teaching assistants Daniele Corona. He can be reached via email at `d.corona@dcsc.tudelft.nl` or during the office hours at room 8C-4-10.

[1] I.e., a correct answer that differs from the answers given by the other groups will be graded higher than a correct answer that is an almost literal copy of the answer of another group.

2 Set-up

We consider an adaptive cruise control (ACC) application in which 2 cars are driving one after the other (see Figure 1).

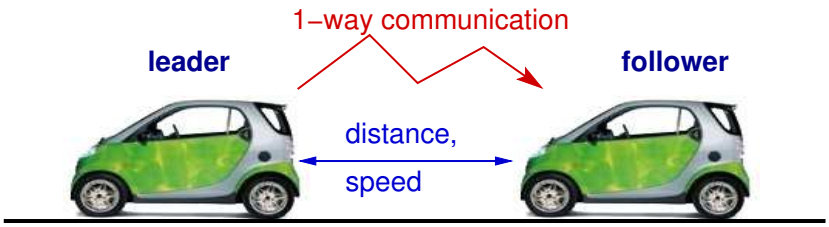

Figure 1: ACC set-up considered in the practical assignment.

In general, the aim of ACC is to ensure a minimal separation between the vehicles (i.e., distance keeping) and a speed adaptation (i.e., the speed differences between the vehicles should be kept as small as possible). In this exercise we will — for the sake of simplicity — only consider the speed adaptation control and we assume that the leading vehicle communicates its speed to the following vehicle, which then has to track this speed as well as possible.

For the vehicle dynamics we consider a simplified model in which the following forces act on the vehicle (which has mass m) at time t:

- the "driving" force $F_{\text{drive}}(t)$, which is proportional to the throttle input $u(t)$: $F_{\text{drive}}(t) = bu(t)$,
- a dynamic friction force $F_{\text{friction}}(t)$, which is proportional to the square of the speed $v(t)$ of the vehicle: $F_{\text{friction}}(t) = cv^2(t)$.

Braking will be simulated by applying a negative throttle. We will assume that the vehicles drive in the forward direction, so the speed will always be nonnegative. For passenger comfort during the ACC operation we also include a maximal acceleration/deceleration: $|a(t)| \leqslant a_{\text{comf,max}}$. The parameters of the vehicle are given in Table 1.

Parameter	Value	Units
m	800	kg
c	0.5	kg/m
b	3700	N
$u_{\max}$	0.9	—
$u_{\min}$	-1	—
$a_{\text{comf,max}}$	2.5	m/s^2

Table 1: Parameters of the vehicle.

3 Tasks & Road map

Step 1: Note that as we are only considering the speed adaptation and as the leading vehicle communicates its speed to the follower, we only have to consider the following vehicle.
Write down the continuous-time model for the position $x(t)$ and speed $v(t)$ of the following vehicle.
Give the maximal throttle input $u_{\max}$ and the maximal braking input $u_{\min}$, determine the maximal speed $v_{\max}$ and the maximal acceleration $a_{\text{acc,max}}$ and deceleration $a_{\text{dec,max}}$ of the vehicle.

Step 2: Construct a piecewise affine (PWA) approximation P with 2 regions of the friction force curve $V : [0, v_{\max}] \to \mathbb{R} : v \mapsto v^2$ as follows. We want a perfect match for $v = 0$ and $v = v_{\max}$. This implies that we still have two degrees of freedom, i.e, the coordinates (α, β) of the middle edge point of the PWA curve (see Figure 2). Now determine α and β such that the *squared* area between V and P (i.e., the squared area corresponding the hashed region in Figure 2) is minimized, or equivalently, such that

$$\int_0^{v_{\max}} \big(V(v) - P(v)\big)^2 \, dv$$

is minimized.

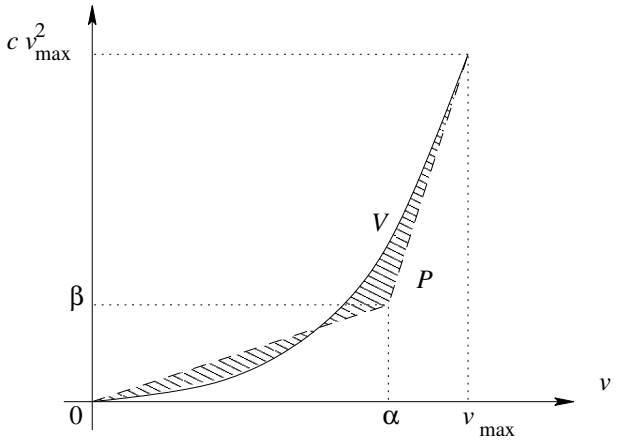

Figure 2: The quadratic function V and its PWA approximation P.

Step 3: Now approximate the friction force using the PWA function P instead of the quadratic function V. Compare the output of the resulting continuous-time PWA model with that of the original model for a sinusoidal throttle input, a white-noise input, and an arbitrary input of your choice. Can you explain where the differences — if any — come from?

Step 4: Now discretize the PWA model of the vehicle using a sample step T with $T = 0.2\,\text{s}$ and a forward Euler rule for the discretization. Compare the discrete-time model with the two continuous-time models for the three input signals selected in Step 3.

Step 5: Transform the discrete-time PWA model of Step 4 into an MLD model.
As we are only considering speed tracking in this assignment and not distance keeping, the position of the car will not influence the performance (i.e., the control objective) in any way. Hence, for the transformation of the discrete-time PWA model into an MLD model, the only state variable that should be considered is the speed.

Step 6: Now we design an MPC controller for the MLD model using the implicit MPC approach. The performance J should be a trade-off between the tracking J_{track} (i.e., the difference between the speed and the reference speed communicated by the leading vehicle) and the input energy J_{input} (for groups 1 and 4), the smoothness of the throttle signal (for groups 2 and 5), and the smoothness of the derivative of the throttle signal (for groups 3 and 6). More specifically, we have

$$J(k) = J_{\text{track}}(k) + \lambda J_{\text{input}}(k)$$

with $J_{\text{track}}(k) = \|\tilde{v}(k) - \tilde{v}_{\text{ref}}(k)\|_1$ and $J_{\text{input}}(k)$ as given in the following table:

Group number	$J_{\text{input}}(k)$
1	$\|\tilde{u}(k)\|_1$
2	$\|\Delta\tilde{u}(k)\|_1$
3	$\|\Delta^2\tilde{u}(k)\|_1$
4	$\|\tilde{u}(k)\|_\infty$
5	$\|\Delta\tilde{u}(k)\|_\infty$
6	$\|\Delta^2\tilde{u}(k)\|_\infty$

where $\tilde{v}(k) = [v(k+1)\dots v(k+N_{\text{p}})]^{\text{T}}$, $\tilde{v}_{\text{ref}}(k) = [\tilde{v}_{\text{ref}}(k+1)\dots\tilde{v}_{\text{ref}}(k+N_{\text{p}})]^{\text{T}}$, and $\tilde{u}(k) = [u(k)\dots u(k+N_{\text{p}}-1)]^{\text{T}}$. In order to get a well-defined objective function groups 2, 3, 5 and 6 may assume that $u(k_0-2) = u(k_0-1) = 0$ where k_0 corresponds to the first sample step of the total simulation period $[0,T]$ (cf. Step 8).
Write a MATLAB file that computes the optimal MPC input sequence for a given sample step k for values of N_{p} and N_{c} up to 6, and for arbitrary values of λ. Note that a discretized version of the comfort constraint $-a_{\text{comf,max}} \leqslant a(t) \leqslant a_{\text{comf,max}}$ should also be taken into account!
Also note that due to the approximation made in Step 2, which is only valid for nonnegative speeds, we should also *explicitly* add the constraint $v(t) \geqslant 0$.

Hints:

- Note that by introducing one or more dummy variables optimization problems of the form $\min_{\theta\in\mathbb{R}^n} \|\theta\|_1$ subject to $A\theta \leqslant b$ or $\min_{\theta\in\mathbb{R}^n} \|\theta\|_\infty$ subject to $A\theta \leqslant b$ can be transformed into a linear programming (LP) problem.
  E.g., it is easy to verify that any optimal solution (ρ^*, θ^*) of the problem

  $$\min_{\rho,\theta\in\mathbb{R}^n} \rho_1 + \dots + \rho_n \quad \text{subject to } -\rho \leqslant \theta \leqslant \rho \text{ and } A\theta \leqslant b$$

  is also an optimal solution of $\min_{\theta\in\mathbb{R}^n} \|\theta\|_1$ subject to $A\theta \leqslant b$ (and vice versa if we set $\rho^* = \|\theta^*\|_1$).
- Using the hint above the MPC optimization problem at step k can be transformed into a mixed-integer linear programming problem (MILP). In order to solve this problem you will need an MILP solver. For this you can use the Multi-Parametric Toolbox, which can be downloaded from
  `http://control.ee.ethz.ch/~mpt/`
  The Multi-Parametric Toolbox is suited for both Windows and Linux. To install and activate this toolbox, see the instructions at

`http://control.ee.ethz.ch/~mpt/docs/install.php`
For step 2 of the installation procedure we recommend to use the `addpath(genpath(...))` approach. Note that this command should be typed every time you (re)start MATLAB and want to use the Multi-Parametric Toolbox.
The command to solve MILP problems is `mpt_solveMILP` (type `help mpt_solveMILP` inside MATLAB for more information; we recommend to use the default solver, i.e., do not specify any solver).
Note that there is an on-line reference guide for the Multi-Parametric Toolbox at
`http://control.ee.ethz.ch/~mpt/docs/`

Step 7: Write a MATLAB file to simulate the closed-loop behavior of the system (i.e., apply the receding horizon approach in which at each step the optimal MPC control input is recomputed and applied to the system) using

a) the discrete-time PWA model,

b) the original continuous-time model.

The MATLAB file should allow the discrete-time PWA model or the original continuous-time model to be used as the simulation model.

Step 8: Select an appropriate value λ based on the nominal values of J_{track} and J_{input} (this might require some tuning and iteration).
Consider two combinations of N_{p} and N_{c}: first the combination $(N_{\text{p},1}, N_{\text{c},1}) = (3,2)$ which is the same for all groups, and another combination $(N_{\text{p},2}, N_{\text{c},2})$ that you may select yourself with $N_{\text{p},i} \in \{4,5,6\}$ and $1 < N_{\text{c},i} < N_{\text{p},i}$. For each combination run your program for the discrete-time PWA model and the original continuous-time model for the time interval $[0, T_{\text{end}}]$ with $T_{\text{end}} = 30$, for $v(0) = 0.9\alpha$ where α is the value found in Step 2, and for the speed reference signal v_{ref} which defined as follows (see also Figure 3):

$$v_{\text{ref}}(t) = \begin{cases} 0.85\alpha & \text{for } 0 \leqslant t \leqslant 3 \\ 1.2\alpha & \text{for } 3 < t \leqslant 9 \\ 1.2\alpha - \dfrac{1}{12}\alpha(t-9) & \text{for } 9 < t \leqslant 15 \\ 0.7\alpha & \text{for } 15 < t \leqslant 18 \\ 0.7\alpha + \dfrac{4}{15}\alpha(t-18) & \text{for } 18 < t \leqslant 21 \\ 0.9\alpha & \text{for } 21 < t \leqslant 30 \end{cases} .$$

Make a plot of the evolution of the controlled closed-loop system in the (x,v) phase plane and of the evolution of x, v, the acceleration a, v_{ref}, $v - v_{\text{ref}}$, u and Δu over time. Compare the obtained trajectories and discuss the differences, if any.

Step 9: Now we examine the robustness of the MPC controller. Assume that there is a measurement error in the speed of the following vehicle such that the measured speed $v_{\text{meas}}(k)$ equals the actual speed $v(k)$ plus a zero-mean white-noise term $e(k)$ with some standard deviation σ_e. Note that if $\sigma_e \neq 0$ the MPC controller will use a "wrong" initial state for determining the optimal MPC input. Consider three different noise levels: $\sigma_{e,1} = 0.5$, $\sigma_{e,2} = 1$, and $\sigma_{e,3} = 2$ and examine the effect of the measurement error on the performance of the MPC controller

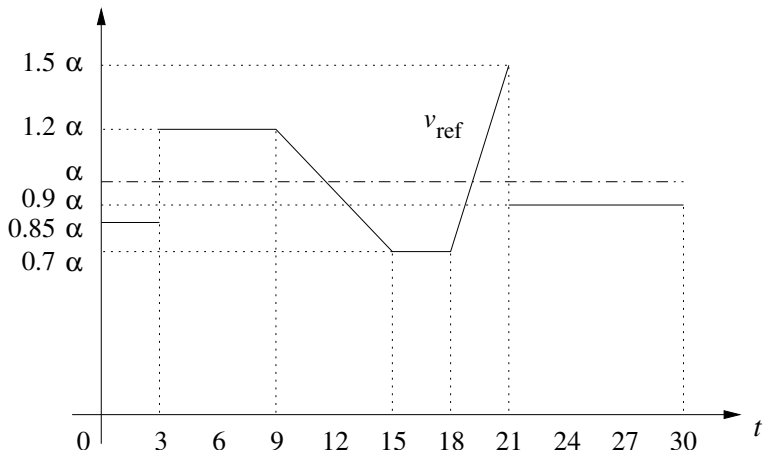

Figure 3: The reference speed signal to be used.

for a closed-loop simulation over the time interval $[0, T_{\text{end},2}]$ with $T_{\text{end},2} = 40$, for $v(0) = 0.9\alpha$ where α is the value found in Step 2, and for the speed reference signal $v_{\text{ref},2}$ which is defined as follows:

$$v_{\text{ref},2}(t) = \begin{cases} 0.85\alpha & \text{for } 0 \leqslant t \leqslant 15 \\ 1.1\alpha & \text{for } 15 < t \leqslant 40 \end{cases} .$$

For each of the three noise levels, make a plot of the evolution of v, the acceleration a, v_{ref}, $v - v_{\text{ref}}$, u and Δu over time. Compare the evolutions for the three noise levels, and discuss the differences, if any.

Depending on the progress of the students, the following step should also be performed (see the Announcements section of the SC4160 web site or check your email for more information):

Step 10: You will have noticed that computing the optimal MPC input using the `mpt_solveMILP` function requires quite some computation time, especially for large N_{p} and N_{c}. If this time is larger than the sampling time T_{c} of the controller (in our case $T_{\text{c}} = T = 0.2$ s), then the (basic) on-line MPC optimization approach is not feasible. One of the possible solutions is then to use the explicit MPC approach in which for each possible current state $x(k)$ the optimal MPC input $u^*(k)$ is computed off-line using multi-parametric mixed-integer linear programming, and stored in a look-up table (cf. page 104 of the lecture notes and the references [19, 20, 23, 24, 32] of the lecture notes).
Now you should apply the explicit MPC approach to the ACC example and repeat Steps 6–8 but now with explicit MPC instead of implicit MPC. In order to compute the explicit MPC solution you can use the `mpt_mpmilp` function of the Multi-Parametric Toolbox.
Compare the computation times required for each approach with $N_{\text{p}} = 3$ and $N_{\text{c}} = 2$, and explain the results.