

Practical Exercise

Modeling and Control of Hybrid Systems (sc4160)

2013 – Version 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 deliverable of this assignment is a written report of max. 50 pages (including cover pages and appendices) about the assignment that also contains the *appendix* with your m files. Make sure to mention your group number and student number on the cover page. The report is to be emailed as a **pdf** file to

b.deschutter@tudelft.nl

The *hard* deadline for emailing the report is Friday, June 21, 2013 at 17.00 p.m.

- You will be graded on the contents and the presentation of the report, on the originality¹ of your answers, on the correctness, the *efficiency*, and on the readability of the MATLAB files (i.e., do not forget to include explanatory comments in your MATLAB files).
- 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 lecturer (during the office hours or via email).

2 Set-up

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

¹I.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.

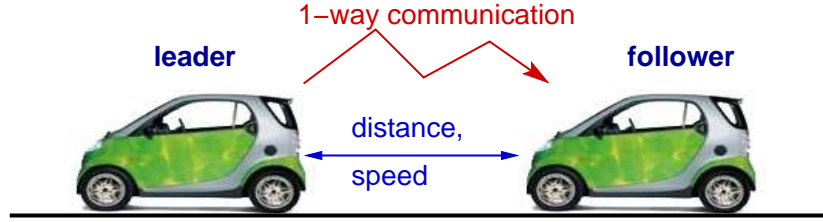


Figure 1: ACC set-up considered in the practical assignment.

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: $|a(t)| \leq a_{\text{comf,max}}$. The parameters of the vehicle are given in Table 1.

Parameter	Value	Units
m	850	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.

The continuous-time model for the position $x(t)$ and speed $v(t)$ of the following vehicle in the state-space form is formulated as:

$$\frac{d}{dt} \begin{bmatrix} x \\ v \end{bmatrix} = \begin{bmatrix} v(t) \\ \underbrace{\frac{b}{m}u(t)}_{F_{\text{drive}}} - \underbrace{\frac{c}{m}v^2}_{F_{\text{friction}}} \end{bmatrix}$$

Given 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 $V : [0, v_{\max}] \rightarrow \mathbb{R} : v \mapsto v^2$ 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

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

is minimized.

Note: Use analytic computations to determine the optimal α and β (and *not* numerical computations/optimization).

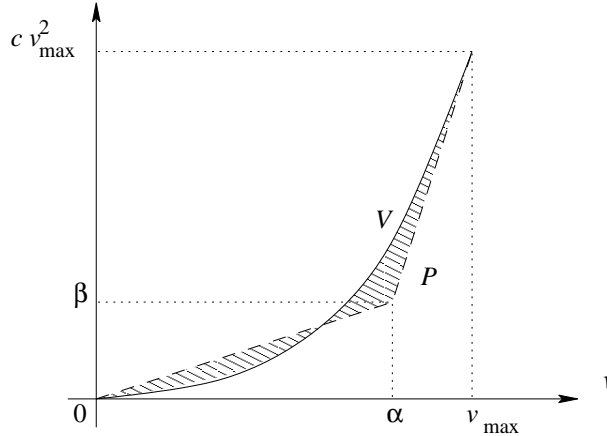


Figure 2: The quadratic function V and its PWA approximation P .

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 zero-mean white-noise input, and an arbitrary input of your choice. Can you explain where the differences — if any — come from?

Note: By properly selecting the initial speed you should be able to guarantee that the speed never becomes negative and that both regions of the PWA model are visited.

Step 4: Now discretize the PWA model of the vehicle using a sample step T with $T = 0.125$ 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 2), the smoothness of the throttle signal (for groups 3 and 4), and the smoothness of the derivative of the throttle signal (for groups 5 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 (modulo 6)	$J_{\text{input}}(k)$
1	$\ \tilde{u}(k)\ _1$
2	$\ \tilde{u}(k)\ _\infty$
3	$\ \Delta\tilde{u}(k)\ _1$
4	$\ \Delta\tilde{u}(k)\ _\infty$
5	$\ \Delta^2\tilde{u}(k)\ _1$
6	$\ \Delta^2\tilde{u}(k)\ _\infty$

where $\tilde{v}(k) = [v(k+1) \dots v(k+N_p)]^T$, $\tilde{v}_{\text{ref}}(k) = [v_{\text{ref}}(k+1) \dots v_{\text{ref}}(k+N_p)]^T$, and $\tilde{u}(k) = [u(k) \dots u(k+N_p-1)]^T$ with N_p the prediction horizon. Note that for a signal $s(\cdot)$ we have $\Delta s(k) = s(k) - s(k-1)$ and $\Delta^2 s(k) = \Delta s(k) - \Delta s(k-1) = s(k) - 2s(k-1) + s(k-2)$.

In order to get a well-defined objective function groups 3, 4, 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 $[0, T_{\text{end}}]$ (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 (i.e., the control horizon) up to 6, and for arbitrary values of λ . Note that a discretized version of the comfort constraint $-a_{\text{comf,max}} \leq a(t) \leq a_{\text{comf,max}}$ 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 $v(t) \geq 0$.

Hints & notes:

- Note that by introducing one or more dummy variables optimization problems of the form $\min_{\theta \in \mathbb{R}^n} \|\theta\|_1$ subject to $A\theta \leq b$ or $\min_{\theta \in \mathbb{R}^n} \|\theta\|_\infty$ subject to $A\theta \leq b$ can be transformed into a linear programming (LP) problem:

- Recall that for $\theta \in \mathbb{R}^n$, we have $\|\theta\|_1 = \sum_{i=1}^n |\theta_i|$. Now it is easy to verify that any optimal solution (ρ^*, θ^*) of the problem

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

is also an optimal solution of $\min_{\theta \in \mathbb{R}^n} \|\theta\|_1$ subject to $A\theta \leq b$ (and vice versa if we set $\rho^* = \|\theta^*\|_1$).

- Similarly, recall that for $\theta \in \mathbb{R}^n$, we have $\|\theta\|_\infty = \max_{i=1, \dots, n} |\theta_i|$. Now it is easy to verify that any optimal solution (τ^*, θ^*) of the problem

$$\min_{\tau \in \mathbb{R}, \theta \in \mathbb{R}^n} \tau \quad \text{subject to} \quad -\tau \leq \theta_i \leq \tau \quad \text{for } i = 1, \dots, n \quad \text{and} \quad A\theta \leq b$$

is also an optimal solution of $\min_{\theta \in \mathbb{R}^n} \|\theta\|_\infty$ subject to $A\theta \leq b$ (and vice versa if we set $\tau^* = \|\theta^*\|_\infty$).

- 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 which you could use one of the following options:

- use the `cplex` command of the TOMLAB cplex toolbox (see <http://tomopt.com/tomlab/>). With this command you can solve MILP problems (see also the `mip_solve_tomlab_cplex` script on the course website).

- use the `mpt_solveMILP` command of the MPT toolbox (see <http://control.ee.ethz.ch/~mpt/>).

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 you 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`

Note that there is an on-line reference guide for the Multi-Parametric Toolbox at

<http://control.ee.ethz.ch/~mpt/docs/>

The man page for `mpt_solveMILP` is

http://control.ee.ethz.ch/~mpt/docs/refguide/mpt/solvers/mpt_solveMILP.html

- As we want you to get some insight into the hybrid MPC method and the relation with integer programming, you are *not* allowed to solve the entire exercise using the `mpt_control` or `mpt_optControl` commands (or related commands) of the MPT toolbox or the Hybrid Toolbox (see <http://cse.lab.imtlucca.it/~bemporad/hybrid/toolbox/>). However, feel free to compare the results obtained with your own programs to those obtained with the MPT toolbox or the Hybrid Toolbox, and to discuss the differences, if any.

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: Assume $\lambda = 0.1$ (in general, an appropriate value of λ is determined by taking into account the nominal values of J_{track} and J_{input} and by some tuning and iteration). Consider two combinations of N_p and N_c : first the combination $(N_{p,1}, N_{c,1}) = (5, 4)$ which is the same for all groups, and another combination $(N_{p,2}, N_{c,2})$ that you may select yourself with $N_{p,i} \in \{7, 8, 9, 10\}$ and $1 < N_{c,i} < N_{p,i}$. 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 $T_{\text{end}} = 25$, 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):

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

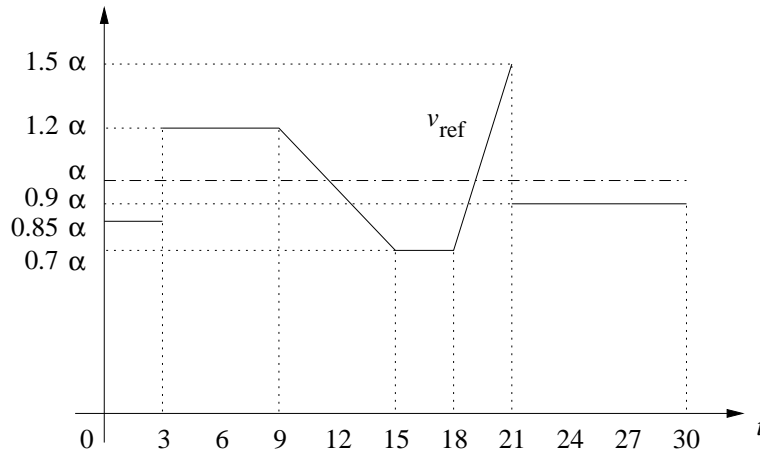


Figure 3: The reference speed signal to be used.

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 consider the continuous-time PWA model (obtained in step 3) as a switched system. In this case, what will be the type of switching? state dependent or time dependent? autonomous or controlled?

In this step, we want to solve the speed tracking problem as a stabilization problem for PWA systems. In other words, we aim at finding a state feedback controller under which the closed-loop PWA system will be stable and moreover, the speed tracking error will be as less as possible.

The PWA model has the following form:

$$\dot{x} = A_i x + a_i + B_i u \quad \text{when } E_i x \geq e_i, i \in I$$

with x the state of the following car (for simplicity we only consider the speed). The reference speed signal is the same as in Fig. 3. You can use the results of Theorem 4.5.4 in the Lecture notes (pages 76 and 88). But it should be noted that here a_i is nonzero in some regions and also we have to integrate the reference tracking problem. (Please see the additional note on the website useful for formulating this problem).

1. Find the state feedback $u = K_i \cdot x$ that guarantees stability of the system and forces the speed to follow the reference speed (Fig. 3). Please ignore the constraints on the throttle input and also the limit on the acceleration. In order to achieve better transient behavior, we define positive definite matrices Q_i and solve the corresponding feasibility problem in order to find the feedback gains K_i . First try to solve the problem with $Q_i = 0$ and next choose Q_i as $\gamma_i \cdot I_i$, with I_i the identity matrix and γ_i a positive scalar. Increase $\gamma_i > 0$ until no feasible solution can be found for K_i . Report the obtained γ_i and K_i .
2. Repeat the previous step but this time with taking into account the constraint $-1 \leq u \leq 0.9$.
3. Write a MATLAB file to simulate the closed-loop behavior of the system (i.e., applying the state feedback controller) using a) the continuous-time PWA model, b) the original continuous-time model. Compare the results with the ones obtained from the MPC approach.

Depending on your progress, the following step could also be performed (if you complete the bonus step, your report is allowed to count up to 60 pages):

Bonus step: 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 $T_c = T = 0.125$ 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 $v(k)$ and future reference vector $\tilde{v}_{\text{ref}}(k)$ 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 112 of the lecture notes and the references [23, 24, 28, 29, 36] 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 off-line and on-line computation times required for each approach with $N_p = 4$ and $N_c = 4$, and explain the results.