Programming Exercise MPC - Answers

Daniel Kiesewalter 16-949-752
Josip Pavlovic 16-939-597

14. May 2020

1 Modeling

1.

$$A^c = \begin{bmatrix} -\frac{1}{m_1}(\alpha_{1,2}+\alpha_{1,0}) & \frac{\alpha_{1,2}}{m_1} & 0 \\ \frac{\alpha_{1,2}}{m_2} & -\frac{1}{m_2}(\alpha_{1,2}+\alpha_{2,3}+\alpha_{2,0}) & \frac{\alpha_{2,3}}{m_2} \\ 0 & \frac{\alpha_{2,3}}{m_3} & -\frac{1}{m_3}(\alpha_{2,3}+\alpha_{3,0}) \end{bmatrix}$$

$$B^c = \begin{bmatrix} \frac{1}{m_1} & 0 \\ 0 & \frac{1}{m_2} \\ 0 & 0 \end{bmatrix}$$

$$d^c = \begin{bmatrix} \alpha_{1,0} * T_0 + w_1 \\ \alpha_{2,0} * T_0 + w_2 \\ \alpha_{3,0} * T_0 + w_3 \end{bmatrix}$$

2. Discretized using euler exact discretization. The corresponding Matlab-Code is:

```
A = expm(A_cont*Ts);
B = A_cont\(A-eye(size(A_cont)))*B_cont;
B_d = A_cont\(A-eye(size(A_cont)))*B_d_cont;
```

   where *A_cont* corresponds to A^c and *B_cont* to B^c.

3. After discretizing, the steady state was calculated using the Formula on Slide 30, Lec.7

```
tempor = [A-eye(size(A)) B;...
truck.C_ref zeros(2,2)];

xu = tempor\[-B_d*d; truck.b_ref];
```

$$T_{sp} = \begin{bmatrix} -21°C \\ 0.3°C \\ 7.32°C \end{bmatrix}$$

$$p_{sp} = \begin{bmatrix} -1927.5W \\ -976.5W \end{bmatrix}$$

4. The constraints are defined as follows:

$$T_{min} = \begin{bmatrix} -\infty \\ 0 \\ -\infty \end{bmatrix}$$

$$T_{max} = \begin{bmatrix} -15 \\ 4 \\ \infty \end{bmatrix}$$

$$p_{min} = \begin{bmatrix} -2500 \\ -2000 \end{bmatrix}$$

$$p_{max} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

whereas $\Delta x_{min} = -T_{sp} - T_{min}$ and $\Delta x_{max} = -T_{sp} + T_{max}$ are defined as follows:

$$\Delta x_{min} = \begin{bmatrix} 0 \\ -3 \\ 0 \end{bmatrix}$$

$$\Delta x_{max} = \begin{bmatrix} 6 \\ 3.7 \\ 0 \end{bmatrix}$$

and where $\Delta u_{min} = -p_{sp} - p_{min}$ and $\Delta u_{max} = -p_{sp} + p_{max}$

$$\Delta u_{min} = \begin{bmatrix} -573 \\ -1023 \end{bmatrix}$$

$$\Delta u_{max} = \begin{bmatrix} 1928 \\ 977 \end{bmatrix}$$

2 Unconstrained optimal control

5. The figure 1 corresponds to this exercise. Initially the controller aggressively cools the system because it is too warm, in order to decrease state cost. As we approach our desired temperature, cooling power decreases to maintain an equilibrium.

6. Using the infinite horizon solution P_∞ of the associated discrete-time Riccati equation,

$$P_\infty = A^T P_\infty A + Q - AT P_\infty B (B^T P_\infty B + R)^{-1} B^T P_\infty A \tag{1}$$

   one can compute the infinite horizon cost using the matlab function

```
    [X,L,G] = dare(A,B,Q,R)
```

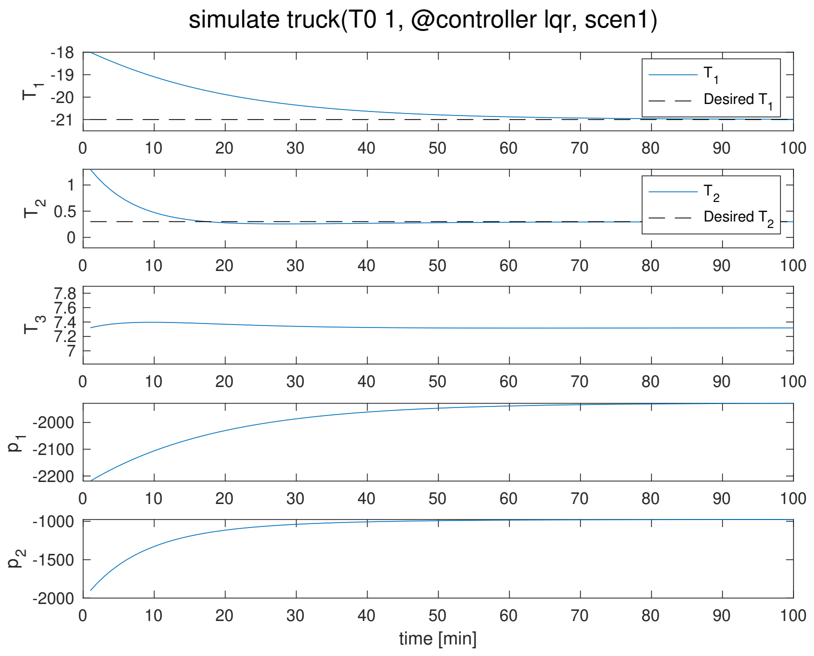

Figure 1: Closed loop simulation of LQR controller with T_{01}

3 A first model predictive controller

7. The cooling units initially operate at reduced cooling power, to allow the system, which is colder than our desired temperature to warm up. Whilst this is happening, the cooling power increases towards the power required to maintain a equilibrium at our desired temperature. Q and R values are tuned to make the state settle appropriately fast.

9. In comparison to an LQR controller, in which constraint satisfaction was explicitly achieved by adapting Q and R values, the MPC controller is algorithm is 'aware' of the system constraints and takes them into consideration, when minimizing the cost. The MPC controller however only minimized over the time horizon N, in comparison to the LQR which minimizes infinite time cost. As the initial condition lies in the feasible set the control for LQR is identical to the MPC control, as we do not run into any constraints imposed by actuator saturation.

10. Assuming that the initial condition is inside the set of feasible initial conditions, an optimal unconstrained controller will run with optimal performance in terms of cost to be minimized, whilst propagating along a feasible trajectory. Therefore it would incur the same cost as an MPC, which is exactly what is to prove in the exercise.

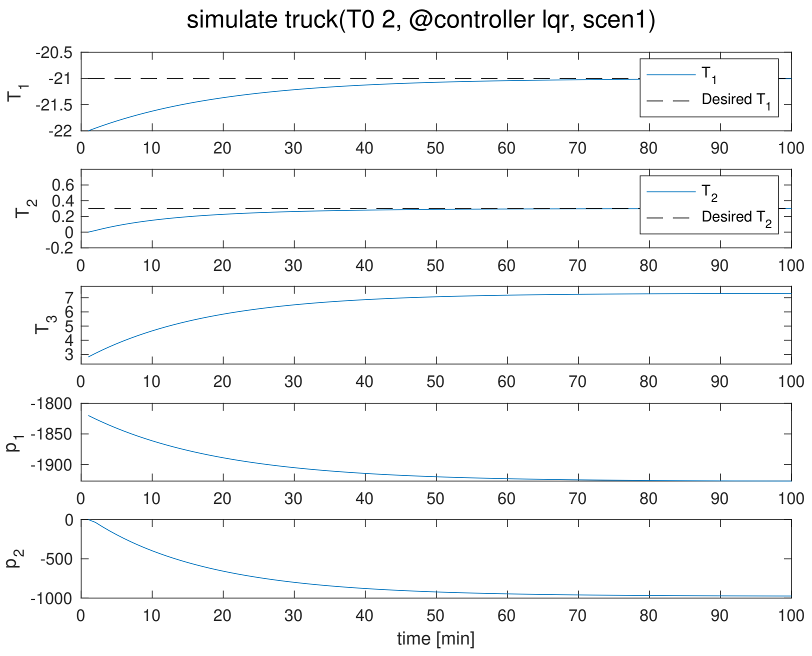

Figure 2: Closed loop simulation of LQR controller with T_{02}

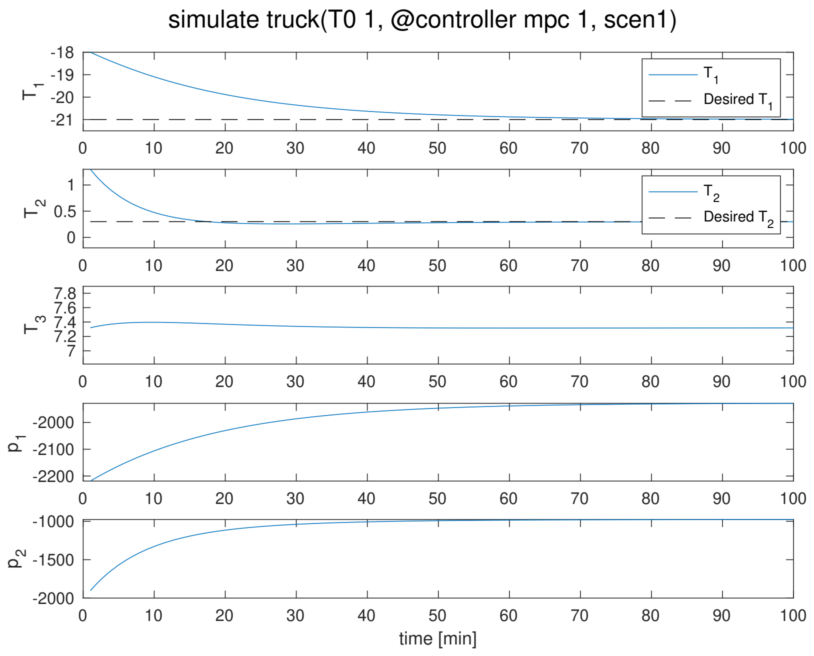

Figure 3: Closed loop simulation of MPC_1 controller with T_{01}

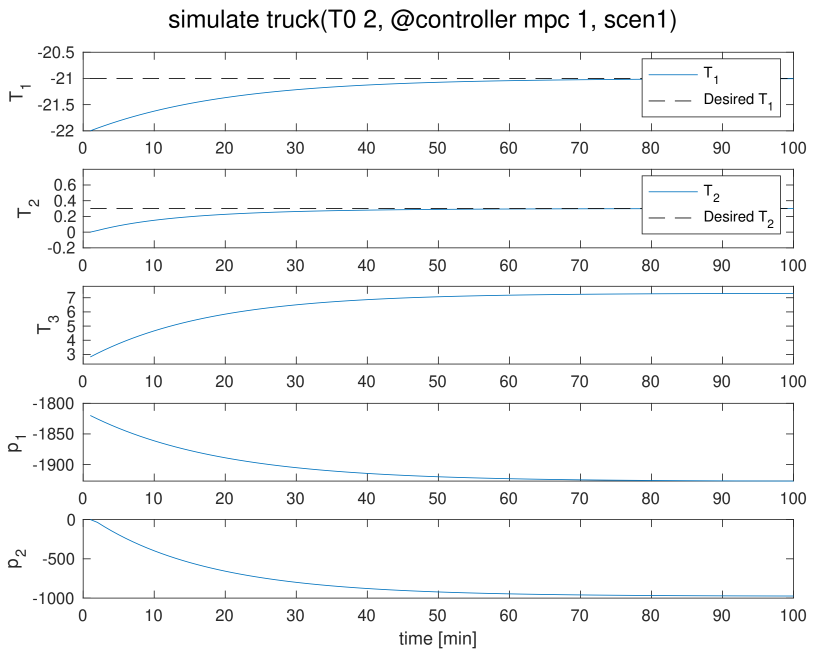

Figure 4: Closed loop simulation of MPC_1 controller with T_{02}

4 MPC with theoretical closed-loop guarantees

11. The proof follows from the lecture 6 slide 35. We prove stability by showing that the optimal cost function is a Lyapunov function. We need to show that

$$J^*(x(k+1)) < J^*(x(k)) \quad \forall x(k) \neq 0 \tag{2}$$

We start using:

$$J^*(x(k)) = \underbrace{l_f(x_N^*)}_{=0} + \sum_{i=0}^{N-1} l(x_i^*, u_i^*) \tag{3}$$

which can be rewritten as follows:

$$\begin{aligned} J^*(x(k+1)) \leq \widetilde{J}(x(k+1)) &= \sum_{i=1}^{N-1} l(x_i^*, u_i^*) + l(x_N, 0) \\ &= \sum_{i=0}^{N-1} l(x_i^*, u_i^*) - l(x_0^*, u_0^*) + l(x_N, 0) \end{aligned} \tag{4}$$

$$= J^*(x(k)) - \underbrace{l(x(k), u_0^*)}_{Subtract\ cost\ at\ stage\ k} + \underbrace{l(0,0)}_{Add\ cost\ for\ staying\ at\ 0=0}$$

$\rightarrow$J$^*(x)$ is a Lyapunov function $\rightarrow$ *Lyapunov* Stability fulfilled

12. The figure 5 corresponds to this exercise.

13. The cost for T_{init}^1 is:

$$\begin{aligned} MPC_1 &: 1.3946 * 10^6 \\ MPC_2 &: 1.6002 * 10^6 \end{aligned} \tag{5}$$

and the cost for T_{init}^2 is:

$$\begin{aligned} MPC_1 &: 2.1783 * 10^6 \\ MPC_2 &: 8.9116 * 10^6 \end{aligned} \tag{6}$$

Both controllers have a similar setup, apart from the key difference that MPC_2 has a terminal state condition, but no infinite time cost. With T_{init}^1 the costs are similar as the control actions are similar. With T_{init}^2 the cost of MPC_2 are higher as the controller has to control more abruptly in order to be feasible and reach the terminal state $x_{30} = 0$, resulting in increased cost for the input.

14. We choose terminal cost $x * P * x$, where P is the solution of the discrete time algebraic Ricatti equation. The proof follows from the lecture 6 slide 41, requiring the terminal cost to be a continuous Lyapunov function.

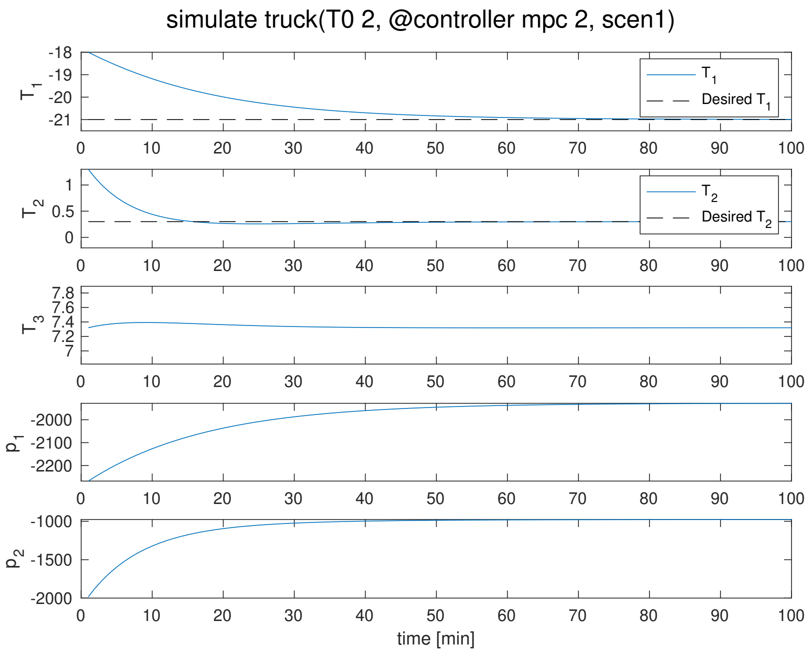

Figure 5: Closed loop simulation of MPC_2 controller at T_{01}

15. The closed-loop trajectories for both initial conditions are identical, as can be seen from the figures 6 & 7 for MPC_3 and figures 3 & 4 for MPC_1. This is because these initial conditions are inside the set of the feasible initial conditions for both systems.

16. For (4) with initial cond. T^2_{init} it is not possible to reach the terminal state $x_{30} = 0$ within the time horizon $N = 30$ i.e. T^2_{init} is not in the set of initial feasible states. For (5) the set of allowed terminal states has a higher cardinality, allowing for a greater set of initial feasible states .

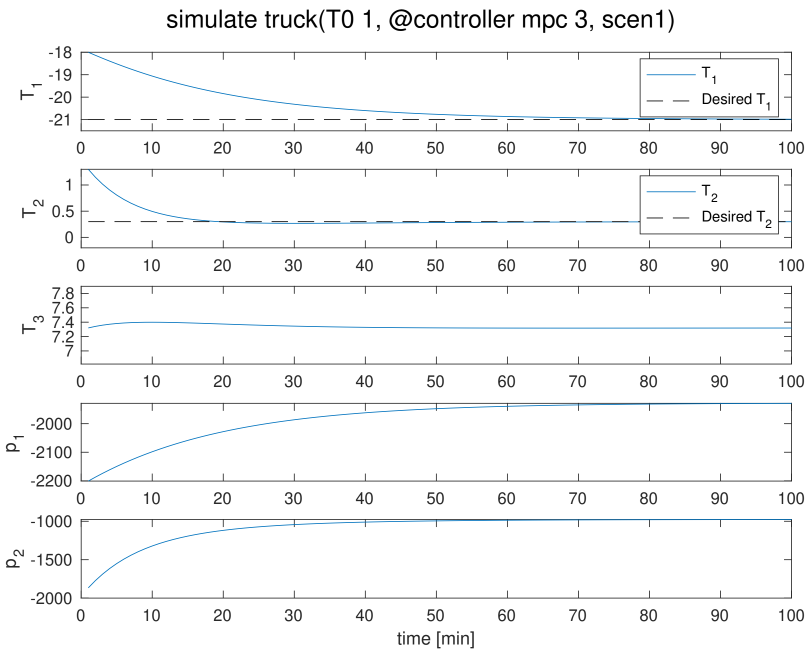

Figure 6: Closed loop simulation of MPC_3 controller at T_{01}

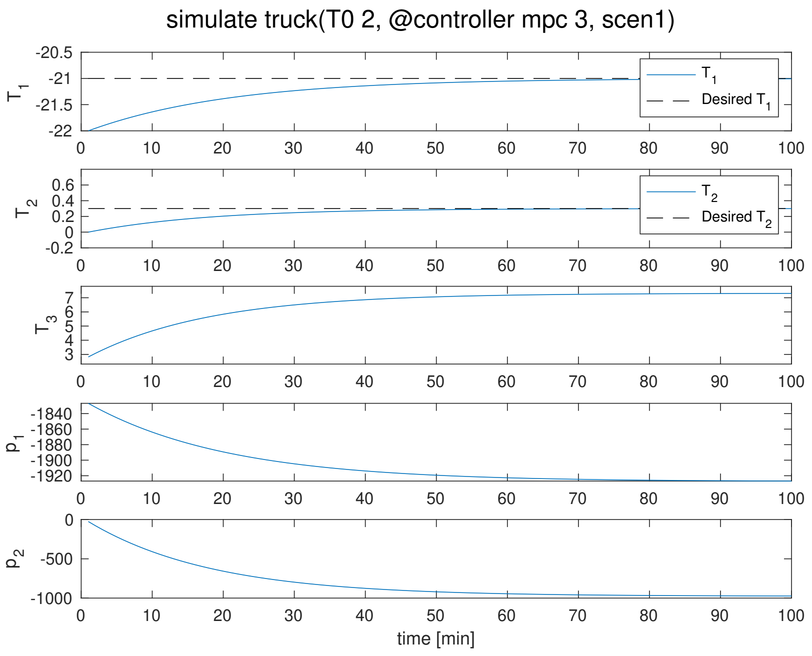

Figure 7: Closed loop simulation of MPC_3 controller at T_{02}

5 Soft constraints

17. In figure 8 it can be noticed that the cooling power is maximal as long as the constraints are violated, in order to minimize the cost associated with the soft-constraint violation. After achieving the boundary's the cooling power decreases step by step in order to minimize the upcoming cost of input and state.

18. The figure 9 corresponds to this exercise.

19. The figure 10 corresponds to the scenario of the MPC_3 controller simulated at $T(0) = T_{init}^{(2)}$, whereas the figure 11 corresponds to the MPC_4 controller.

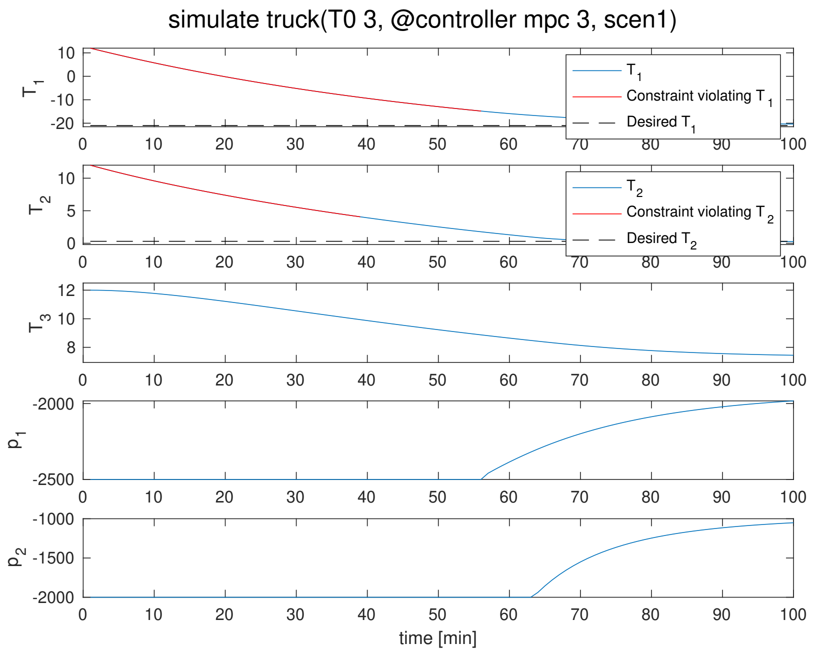

Figure 8: Closed loop simulation of MPC_3 controller at T_{03}

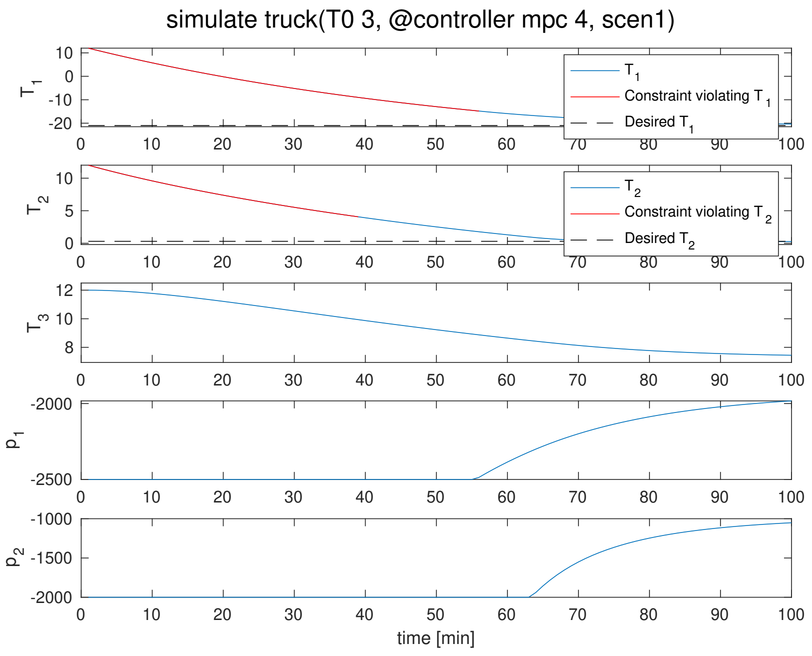

Figure 9: Closed loop simulation of MPC_4 controller at T_{03}

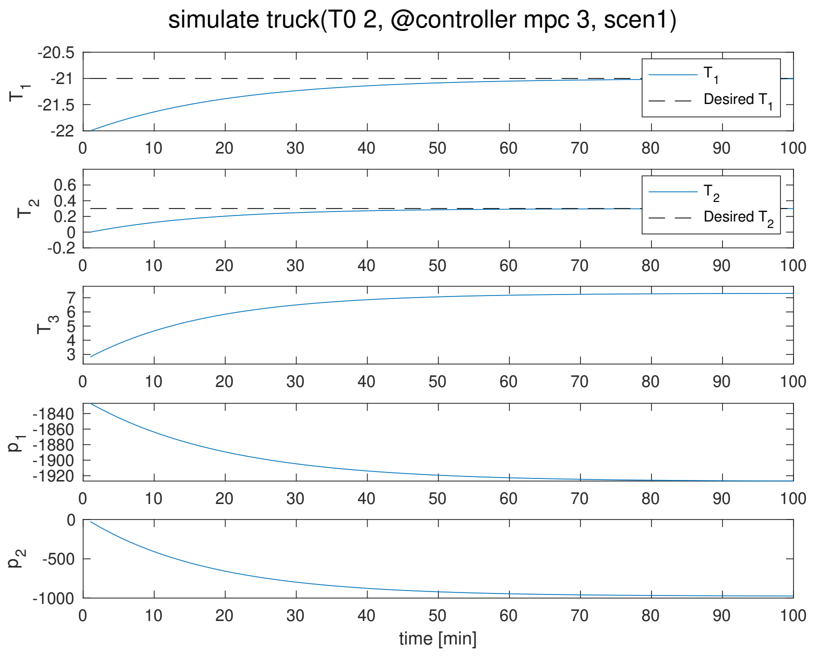

Figure 10: Closed loop simulation of MPC_3 controller at T_{02}

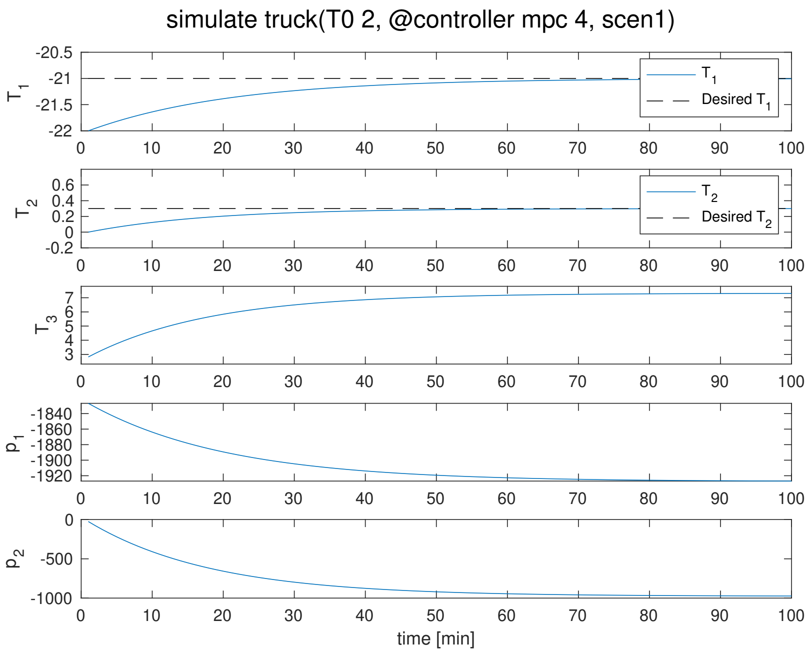

Figure 11: Closed loop simulation of MPC_4 controller at T_{02}

6 Offset-free MPC

20. $$A_{aug} = \begin{bmatrix} A & B_d \\ zeros(size(A)) & eye(size(A)) \end{bmatrix}$$

where

$$B_d = \begin{bmatrix} \frac{1}{m_1} & 0 & 0 \\ 0 & \frac{1}{m_2} & 0 \\ 0 & 0 & \frac{1}{m_2} \end{bmatrix}$$

$$B_{aug} = \begin{bmatrix} B \\ zeros(size(A,1), size(B,2)) \end{bmatrix}$$

$$C_{aug} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$D_{aug} = \begin{bmatrix} 0 \end{bmatrix}$$

21. We solve following system of equations to get our steady states, with the estimated disturbance.

$$\begin{bmatrix} A - I & B \\ HC & 0 \end{bmatrix} \begin{bmatrix} x_s \\ u_s \end{bmatrix} = \begin{bmatrix} -B_d \hat{d} \\ r \end{bmatrix}$$

In order to have stable error, we need eigenvalues of less than 1 in our system of error dynamics i.e. eigenvalues less than 1 the following

$$\begin{bmatrix} A & B_d \\ 0 & I \end{bmatrix} + \begin{bmatrix} L_x \\ L_d \end{bmatrix} \begin{bmatrix} C & 0 \end{bmatrix}$$

This can be achieved with the Matlab function *place*(), which designs a controller according to you desired eigenvalues. This lead us to:

$$L = 100 * \begin{bmatrix} 0.002 & 0 & 0 \\ 0 & 0.0019 & 0.002 \\ 3.3757 & -0.0381 & 0.0004 \\ -0.0380 & 6.87 & -0.1442 \\ 0.0002 & -0.1591 & 8.7366 \end{bmatrix}$$

The resulting error dynamics have poles at $0.02,\ -0.03,\ 0.01,\ -0.02,\ 0.01$ and -0.01.

22. In figure 13 there is no peak at $t \approx 2$ in cooling power, as there is in figure 12. The peak is caused due to the varying estimate of our disturbance. Furthermore, there is a constant offset in the plot of 13 .

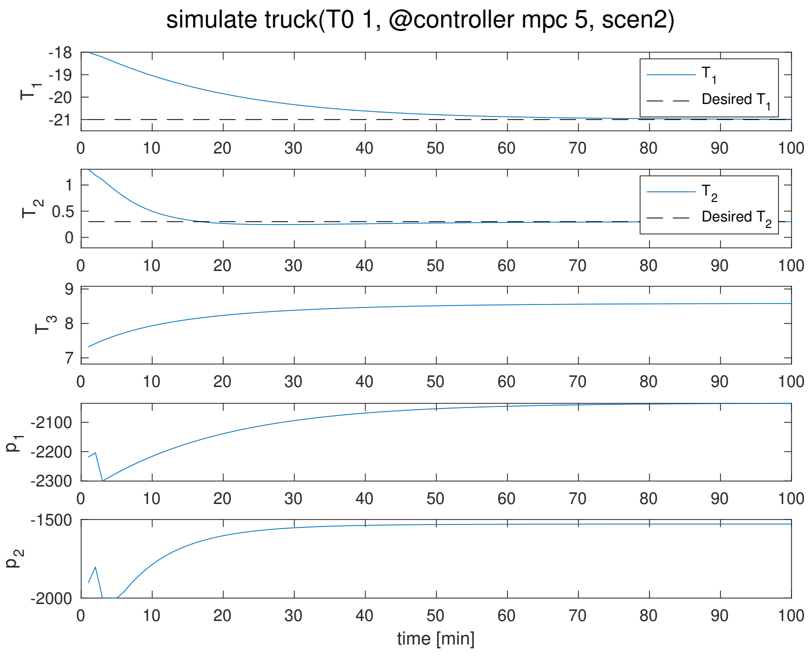

Figure 12: Closed loop simulation of MPC_5 controller at T_{01} and scene 2

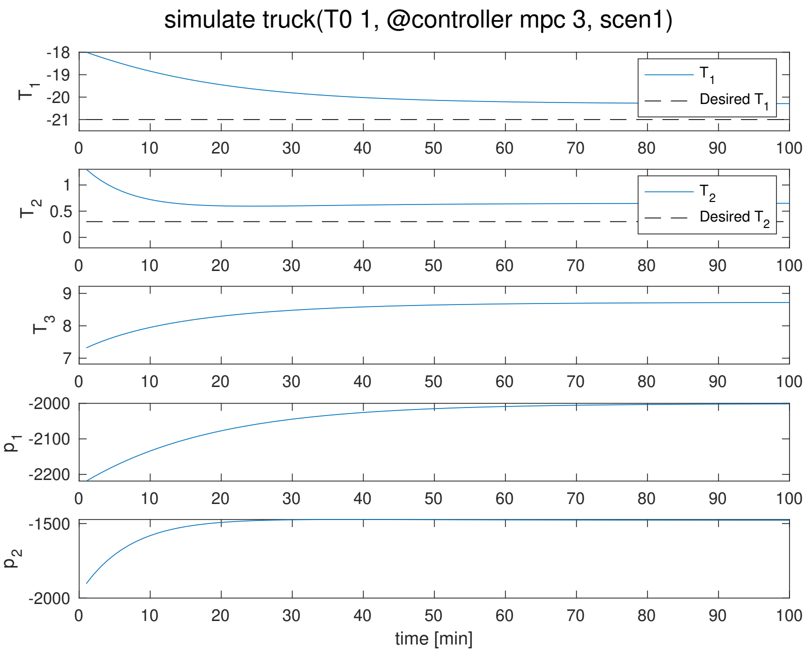

Figure 13: Closed loop simulation of MPC_3 controller at T_{02} and scene 2

7 FORCES pro

23.

$$\begin{aligned} t\_sim\_forces &= 0.5403 \\ t\_sim &= 2.3831 \end{aligned} \tag{7}$$

As expected, Forces is faster.

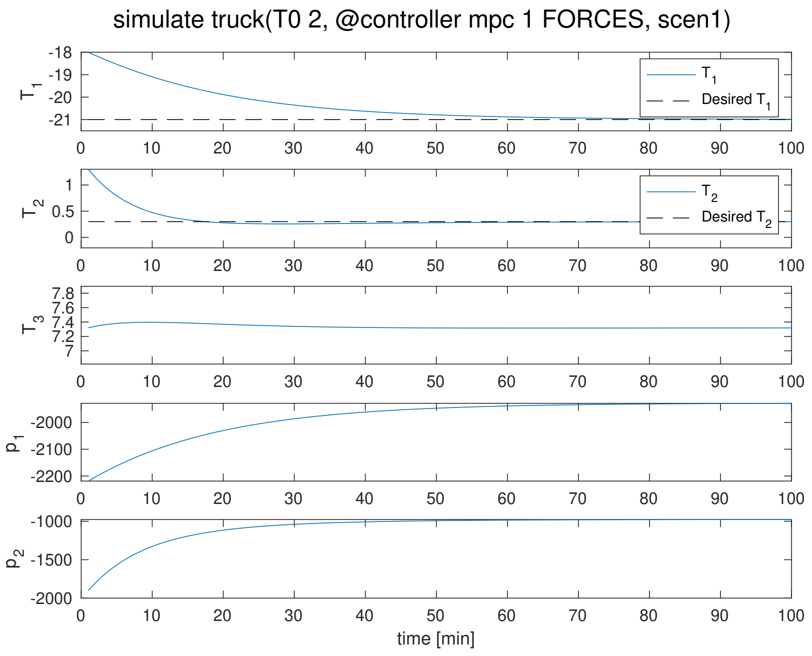

Figure 14: Closed loop simulation of MPC_1 using Forces pro at T_{02}