CHEM-E7225 Advanced Process Control D — Winter 2026

Lecture 06 – Model Predictive Control (B)

Otacilio "Minho" Neto (otacilio.bneto@pm.me)

Previously, we have discussed optimal control problems and their numerical solution, focusing on the linear-quadratic regulator. In this lecture, we tackle the problem of designing a feedback controller using optimal control problems as their core component. The outcome is the popular class of controllers known as model-predictive controllers.

1 State-feedback via receding-horizon control

Recall the general optimal control problem (OCP) from last lecture,

$$\begin{aligned}
\underset{x[\cdot],\ u[\cdot]}{\text{minimize}} \quad & \sum_{k=0}^{N-1} L(x[k],u[k]) + V_f(x[N]) && \text{(1a)}\\
\text{subject to} \quad & x[k{+}1] = f_{\Delta t}(x[k],u[k]), \quad k=0,\dots,N{-}1, && \text{(1b)}\\
& h(x[k],u[k]) \preceq 0, \quad k=0,\dots,N{-}1, && \text{(1c)}\\
& x[0] = x_0, && \text{(1d)}
\end{aligned}$$

describing the problem of finding the *optimal control actions* $\mathbf{u}^\star = (u^\star[0],\dots,u^\star[N{-}1])$, and corresponding *optimal state responses* $\mathbf{x}^\star = (x^\star[0],\dots,x^\star[N])$, over some *control horizon* of length $N>0$. A controller design based on this OCP would be, in principle, very simple: we program our hardware (microcontroller, computer, etc.) to numerically solve the OCP, then apply the optimal control actions sequentially in time. Formally, the control law would be

$$u[n] = \underbrace{e_n \circ \texttt{OCP}(x_0)}_{K_{\text{OL}}(x_0,n)}, \qquad \text{where} \quad \texttt{OCP}(x_0) = \arg\min_{\mathbf{u}}\{V_N(x_0,\mathbf{u}) \mid \mathbf{u}\in\mathcal{U}_N(x_0)\} \tag{2}$$

where e_n is a nth-element selector and $\texttt{OCP}(x_0)$ returns a solution to the OCP (1), parametrized by x_0. Here, we are implicitly using the solution map $x[n] = f^n_{\Delta t}(x_0,\mathbf{u}_n)$ to eliminate the dependency on intermediate states.

The control law described above actually defines an *open-loop controller* (Figure 2). Note that the only external information provided to the controller is the initial state $x[0]=x_0$. The intermediate states $(x[1],\dots,x[N])$ are not measurements coming from the actual physical system, but are predictions obtained by using the solution map $f^n_{\Delta t}(\cdot)$. Since we are not monitoring the state, there is no *feedback*, and thus we are not protected against model uncertainty and the noise/disturbances that make our predictions deviate from the "true" system response. Finally, such a controller cannot operate a system *on-the-loop*, that is, continuously in time, due to having a fixed control horizon N. For this reason, an open-loop optimal controller is really *only practical for batch systems with very accurate models*.

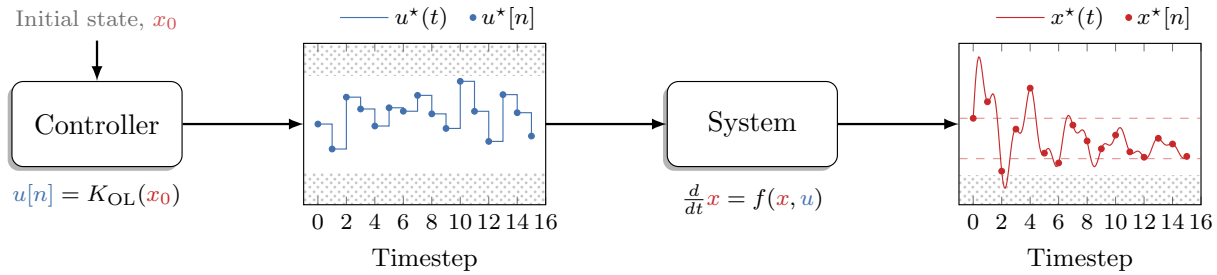

Figure 1. Illustration of the open-loop optimal controller. The controller solves an OCP with a fixed initial state x_0, without actually monitoring the response of the system. Since there is no *feedback*, the response caused by the inputs might deviate from the prediction–and even violate the path constraints. Conversely, note that the inputs always satisfy the path constraints.

1.1 Receding-horizon strategy

An ingenious idea to achieve *feedback* in optimal control settings is the *receding-horizon control* (RHC) strategy [1]. In this strategy, the controller (i) solves the OCP to obtain $\mathbf{u}^\star = (u^\star[0], \dots, u^\star[N-1])$ while fixing the *current state of the system* $x[n]$ as the initial state ($x_0 = x[n]$); then (iii) deploys only the first action $u^\star[0]$ to the physical system. After deploying the action, the controller *shifts* the control horizon such that the next measurement $x[n+1]$ becomes the *new initial state*, and then repeats this strategy after Δt [units-of-time]. Under this strategy, the control law is

$$u[n] = \underbrace{e_0 \circ \texttt{OCP}(x[n])}_{K_{\text{RHC}}(x[n])}, \qquad \text{where} \quad \texttt{OCP}(x[n]) = \arg\min_{\mathbf{u}} \{V_N(x[n], \mathbf{u}) \mid \mathbf{u} \in \mathcal{U}_N(x[n])\} \tag{3}$$

where $x[n] \in \mathbb{R}^{N_x}$ is the current state measurement from the system. Clearly, this is now a *feedback control law*. A controller that implements the RHC strategy is also known as a *model-predictive controller* (MPC).

Informally, the idea is to *replan* the optimal actions at each nth *control cycle*, accounting for deviations caused by noise/disturbances or incorrect predictions caused by model inaccuracies. Of course, this strategy is not without costs. In the open-loop control law (Eq. 2), the controller executes $\texttt{OCP}(\cdot)$ only once (and this can even be done *offline*). In the feedback control law (Eq. 3), the controller executes $\texttt{OCP}(\cdot)$ at each nth timestep, and it must be done *online* and efficiently enough to avoid delays between the measurement and the control action (known as *controller deadtime*).

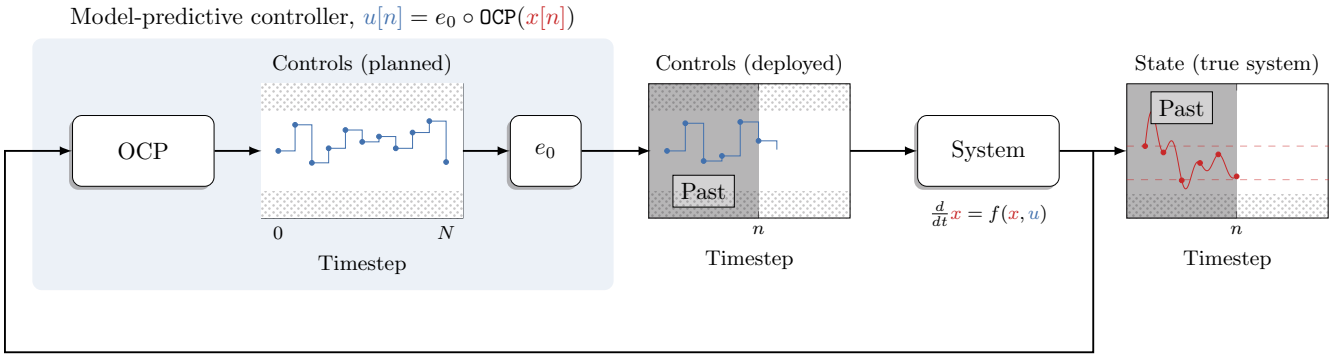

Figure 2. Illustration of the model-predictive controller. In this strategy, the OCP is solved for each nth timestep with the current state fixed as the initial state (i.e., $x_0 = x[n]$). The controller plans the whole action sequence $\mathbf{u} = (\mathbf{u}^\star[0], \dots, \mathbf{u}^\star[N])$, but only the first is actually deployed to the system; and then repeated until the next control cycle. Each control cycle has a duration of Δt [units-of-time].

2 Linear MPC – Design and implementation

A fundamental class of MPCs are those based on the linear-quadratic (LQ) optimal control problems

$$\underset{x[\cdot],\ u[\cdot]}{\text{minimize}} \quad \frac{1}{2}\sum_{k=0}^{N-1} \begin{bmatrix} x[k] \\ u[k] \end{bmatrix}^\mathsf{T} \begin{bmatrix} Q & S \\ S^\mathsf{T} & R \end{bmatrix} \begin{bmatrix} x[k] \\ u[k] \end{bmatrix} + \frac{1}{2} x[N]^\mathsf{T} Q_f x[N] \tag{4a}$$

$$\text{subject to} \quad x[k+1] = A_{\Delta t} x[k] + B_{\Delta t} u[k], \quad k = 0, \dots, N-1, \tag{4b}$$

$$x[0] = x_0, \tag{4c}$$

Linear-quadratic problems are convex, and they have widely-known analytical solutions (see Lecture Notes L05, §2.1). An LQ-based MPC is also known as a *linear model-predictive controller* (L-MPC), due to the model being linear.

The L-MPC will be the main controller design considered in this course. In the following, we will demonstrate how this design can cover different types of problems, include the control of *nonlinear systems*. The extension of the L-MPC to constrained control problems will be the topic of next week's lectures.

The matrices (A, B) refer to a continuous-time linear system, whereas $(A_{\Delta t}, B_{\Delta t})$ refers to its discretization with $\Delta t > 0$. Careful not to confuse them!

2.1 Regulation and tracking

The LQ Problem (4) represents a *regulation* problem, that is, the problem of stabilizing a system to the origin. However, the most common objective in control applications is *setpoint tracking*, that is, using feedback control to

drive a system towards some reference setpoint $(x_{\text{ref}}, u_{\text{ref}})$. Formally, such problems can be written as the convex QP

$$\underset{x[\cdot],\ u[\cdot]}{\text{minimize}} \quad \frac{1}{2}\sum_{k=0}^{N-1}\begin{bmatrix}x[k]-x_{\text{ref}}\\ u[k]-u_{\text{ref}}\end{bmatrix}^{\mathsf{T}}\begin{bmatrix}Q & S\\ S^{\mathsf{T}} & R\end{bmatrix}\begin{bmatrix}x[k]-x_{\text{ref}}\\ u[k]-u_{\text{ref}}\end{bmatrix}+\frac{1}{2}(x[N]-x_{\text{ref}})^{\mathsf{T}}Q_f(x[N]-x_{\text{ref}}) \tag{5a}$$

$$\text{subject to} \quad x[k{+}1] = A_{\Delta t}x[k] + B_{\Delta t}u[k], \quad k=0,\dots,N{-}1, \tag{5b}$$

$$x[0] = x_0, \tag{5c}$$

We can show that such setpoint tracking problems are actually equivalent to regulation problems. Let $(x_{\text{ref}}, u_{\text{ref}})$ be a chosen *equilibrium point*, that is, $0 = Ax_{\text{ref}} + Bu_{\text{ref}}$. Now, define the deviation signals $x^\delta[n] = x[n] - x_{\text{ref}}$ and $u^\delta[n] = u[n] - u_{\text{ref}}$. It is a (easy to show) fact that

$$x[n{+}1] = A_{\Delta t}x[n] + B_{\Delta t}u[n] \quad \iff \quad x^\delta[n{+}1] = A_{\Delta t}x^\delta[n] + B_{\Delta t}u^\delta[n],$$

that is, the system in deviation variables follow the same dynamics as the original linear system. Therefore, by a simple *change of variable*, the setpoint tracking Problem (5) can be written as the regulation problem

$$\underset{x^\delta[\cdot],\ u^\delta[\cdot]}{\text{minimize}} \quad \frac{1}{2}\sum_{k=0}^{N-1}\begin{bmatrix}x^\delta[k]\\ u^\delta[k]\end{bmatrix}^{\mathsf{T}}\begin{bmatrix}Q & S\\ S^{\mathsf{T}} & R\end{bmatrix}\begin{bmatrix}x^\delta[k]\\ u^\delta[k]\end{bmatrix}+\frac{1}{2}x^\delta[N]^{\mathsf{T}}Q_f x^\delta[N] \tag{6a}$$

$$\text{subject to} \quad x^\delta[k{+}1] = A_{\Delta t}x^\delta[k] + B_{\Delta t}u^\delta[k], \quad k=0,\dots,N{-}1, \tag{6b}$$

$$x^\delta[0] = x_0^\delta \tag{6c}$$

and the corresponding L-MPC control law takes the form $u^\delta[n] = K_{\text{RHC}}(x^\delta[n])$, or, in terms of the original signals,

$$u[n] = u_{\text{ref}} + K_{\text{RHC}}(x[n]{-}x_{\text{ref}}). \tag{7}$$

Therefore, for setpoint tracking, we simply *shift* the initial state as $x[n]{-}x_{\text{ref}}$, solve the LQ Regulation Problem (6), which is structurally identical to the original problem, then *shift* the solution by u_{ref}. The strategy remains valid even if the references $(x_{\text{ref}}, u_{\text{ref}})$ change in time. This highlights another interesting fact: setpoint tracking in linear systems is equivalent to regulating the system to the origin, then *offsetting* it by an affine term (also called the *bias term*).

2.2 Linearization-based control of nonlinear systems

Until now, we discussed L-MPC designs with the underlying assumption that the system being controlled follows a linear dynamical model. However, most relevant applications concern nonlinear dynamical systems. Thankfully, the L-MPC can still be applied to the control of nonlinear systems via linearization.

The task is driving a *nonlinear* system to a reference setpoint $(x_{\text{ref}}, u_{\text{ref}})$ which is an equilibrium (i.e., $f(x_{\text{ref}}, u_{\text{ref}}) = 0$). Due to the *Lyapunov's linearization method* (see Lecture Notes L02, §3), we know that stabilizing the *linearization* of this system implies the stabilization of the nonlinear system. Thus, we design the tracking problem

$$\underset{x^\delta[\cdot],\ u^\delta[\cdot]}{\text{minimize}} \quad \frac{1}{2}\sum_{k=0}^{N-1}\begin{bmatrix}x^\delta[k]\\ u^\delta[k]\end{bmatrix}^{\mathsf{T}}\begin{bmatrix}Q & S\\ S^{\mathsf{T}} & R\end{bmatrix}\begin{bmatrix}x^\delta[k]\\ u^\delta[k]\end{bmatrix}+\frac{1}{2}x^\delta[N]^{\mathsf{T}}Q_f x^\delta[N] \tag{8a}$$

$$\text{subject to} \quad x^\delta[k{+}1] = A_{\Delta t}x^\delta[k] + B_{\Delta t}u^\delta[k], \quad k=0,\dots,N{-}1, \tag{8b}$$

$$x^\delta[0] = x_n^\delta \tag{8c}$$

where $(A_{\Delta t},\ B_{\Delta t})$ are the time-discretization of the Jacobian matrices

$$A = \tfrac{\partial}{\partial x}f(x_{\text{ref}}, u_{\text{ref}}) \qquad \text{and} \qquad B = \tfrac{\partial}{\partial u}f(x_{\text{ref}}, u_{\text{ref}}), \tag{9}$$

and the corresponding L-MPC control law is still $u[n] = u_{\text{ref}} + K_{\text{RHC}}(x[n]{-}x_{\text{ref}})$. Again, this problem is structurally identical to the original LQR Problem (convinced yet about its importance?), with the slight difference that a setpoint change also implies a change in the dynamic matrices $(A_{\Delta t},\ B_{\Delta t})$. The need to compute Jacobians at every setpoint change also shows the importance of computational tools such as *automatic differentiation* (AD) in optimal control.

> *Remember that the above strategy works* as long as the system does not escape the region of attraction of $(x_{\text{ref}}, u_{\text{ref}})$. *If that's not the case, then a more advanced control method is required.*

Example 2.1 ***Optimal control of nonlinear second-order system***

Consider the 2nd-order damped harmonic oscillator from Lecture 02, whose dynamics are given as

$$\begin{bmatrix} \frac{\mathrm{d}}{\mathrm{d}t}x_1 \\ \frac{\mathrm{d}}{\mathrm{d}t}x_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\sin(x_1) - 0.5x_2 + u \end{bmatrix}. \tag{10}$$

The linearization of $f(\cdot)$ around any $(x_{\mathrm{ref}}, u_{\mathrm{ref}})$ is defined by the matrices

$$A = \frac{\partial}{\partial x} f(x_{\mathrm{ref}}, u_{\mathrm{ref}}) = \begin{bmatrix} 0 & 1 \\ -\cos(x_{\mathrm{ref},1}) & -0.5 \end{bmatrix} \quad \text{and} \quad B = \frac{\partial}{\partial u} f(x_{\mathrm{ref}}, u_{\mathrm{ref}}) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{11}$$

Under the control law $u[n] = u_{\mathrm{ref}} + K_{\mathrm{RHC}}(x[n] - x_{\mathrm{ref}})$, the phase portrait of the nonlinear system is shown in Figure 3 for two different equilibrium points. When considering $x_{\mathrm{ref}} = [\pi \;\; 0]^\mathsf{T}$ and $u_{\mathrm{ref}} = 0$, the closed-loop system displays a single equilibrium point, which *is asymptotically stable and has a large region of attraction*. When considering $x_{\mathrm{ref}} = [2\pi \;\; 0]^\mathsf{T}$ and $u_{\mathrm{ref}} = 0$, the phase portrait is similar to that of the open-loop system (Lecture 02), but the only attracting equilibrium is the point x_{ref}.

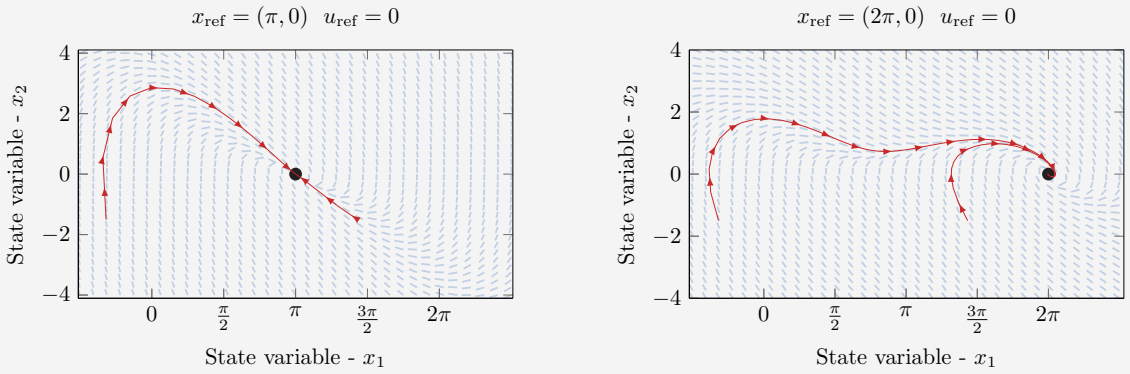

Figure 3. Phase portrait of the 2nd-order damped harmonic oscillator under L-MPC operation.

2.3 Extensions based on state-augmentation

The L-MPC can still cover many other *penalty-like* objective functions, such as minimizing the rate of change in the input $u(t)$ and state $x(t)$ signals. Moreover, it can also be extended to include *extra* features, such as integral action and affine dynamical models. We discuss some of these setups in the following.

Minimizing the rate of change

A common objective in control applications is to minimize the rate of change in the controls (for instance, to avoid breaking the equipment). Formally, this implies extending the objective function $L(\cdot)$ with a quadratic term,

$$\widetilde{L}(x[k], u[k], u[k-1]) = L(x[k], u[k]) + (u[k]-u[k-1])^\mathsf{T} R_\Delta (u[k]-u[k-1]).$$

Guess what? The standard LQ Problem (4) can still cover these objectives. In this direction, let us augment the state signal as $\tilde{x} = \mathtt{col}(x, x_z)$ such that $x_z[k] = u[k-1]$, that is, we add a "buffer" state to remember the last controls. The extended linear system becomes

$$\begin{bmatrix} x[k+1] \\ x_z[k+1] \end{bmatrix} = \begin{bmatrix} A_{\Delta t} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x[k] \\ x_z[k] \end{bmatrix} + \begin{bmatrix} B_{\Delta t} \\ I \end{bmatrix} u[k], \qquad \begin{bmatrix} x[0] \\ x_z[0] \end{bmatrix} = \begin{bmatrix} x_0 \\ 0 \end{bmatrix}, \tag{12}$$

The extended cost function thus becomes

$$\widetilde{L}(\tilde{x}[k], u[k]) = \frac{1}{2} \begin{bmatrix} \tilde{x}[k] \\ u[k] \end{bmatrix}^\mathsf{T} \begin{bmatrix} \widetilde{Q} & \widetilde{S} \\ \widetilde{S}^\mathsf{T} & \widetilde{R} \end{bmatrix} \begin{bmatrix} \tilde{x}[k] \\ u[k] \end{bmatrix}, \quad V_f(x[N]) = \frac{1}{2} \tilde{x}[N]^\mathsf{T} \widetilde{Q}_f \tilde{x}[N]$$

with augmented tuning matrices $\widetilde{Q} = \mathrm{blkdiag}(Q, R_\Delta)$, $\widetilde{S} = \mathrm{blkdiag}(S, -R_\Delta)$, $\widetilde{R} = R + R_\Delta$, and $\widetilde{Q}_f = \mathrm{blkdiag}(Q_f, R_\Delta)$. Finally, we remark that a similar procedure can be done if it's desired to penalize the rate of change of states as well.

Affine systems

A control problem with an affine dynamical model $x[k+1] = A_{\Delta t}x[k] + B_{\Delta t}u[k] + c$ can be reformulated into the standard L-MPC. Again, this is achieved by augmenting the state with $\tilde{x} = \texttt{col}(x, x_z)$ such that $x_z[k] = c$, that is, we add a constant state variable representing the affine term. The extended linear system becomes

$$\begin{bmatrix} x[k+1] \\ x_z[k+1] \end{bmatrix} = \begin{bmatrix} A_{\Delta t} & I \\ 0 & I \end{bmatrix} \begin{bmatrix} x[k] \\ x_z[k] \end{bmatrix} + \begin{bmatrix} B_{\Delta t} \\ 0 \end{bmatrix} u[k] \qquad \begin{bmatrix} x[0] \\ x_z[0] \end{bmatrix} = \begin{bmatrix} x_0 \\ c \end{bmatrix}, \tag{13}$$

The extended cost function thus becomes

$$\widetilde{L}(\tilde{x}[k], u[k]) = \frac{1}{2} \begin{bmatrix} \tilde{x}[k] \\ u[k] \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \widetilde{Q} & \widetilde{S} \\ \widetilde{S}^{\mathsf{T}} & R \end{bmatrix} \begin{bmatrix} \tilde{x}[k] \\ u[k] \end{bmatrix}, \quad V_f(x[N]) = \frac{1}{2}\tilde{x}[N]^{\mathsf{T}}\widetilde{Q}_f\tilde{x}[N]$$

with augmented tuning matrices $\widetilde{Q} = \text{blkdiag}(Q, 0)$, $\widetilde{S} = \text{blkdiag}(S, 0)$, and $\widetilde{Q}_f = \text{blkdiag}(Q_f, 0)$.

The augmentation above is not actually useful, as most solvers would prefer us to simply write the equality constraints in affine form. However, it highlights a difference between affine and linear systems: the system can never be regulated to the origin, because the affine term itself is nonzero and it incurs into a nonzero control action.

Integral action

The standard L-MPC formulation only accounts for the current (and future) setpoint tracking error. A quite powerful idea in control theory is to simultaneously minimize the *past values* of the tracking error, since this can be shown to improve the robustness to noise/disturbance; specially if these perturbations are slowly varying [2]. This property is known as *integral action*, since it is formalized by an integrating term in the cost function:

$$\widetilde{L}(x[k], u[k], e[k]) = L(x[k], u[k]) + e[k]^{\mathsf{T}}Q_e e[k].$$

where $e[k] = \sum_{i=0}^{k} x[k]$ (assuming, without loss of generality, that $x_{\text{ref}} = 0$). We incorporate integral action to the L-MPC by augmenting the state signal as $\tilde{x} = \texttt{col}(x, x_z)$, with $x_z[k] = e[k]$, leading to the extended linear system

$$\begin{bmatrix} x[k+1] \\ x_z[k+1] \end{bmatrix} = \begin{bmatrix} A_{\Delta t} & 0 \\ I & I \end{bmatrix} \begin{bmatrix} x[k] \\ x_z[k] \end{bmatrix} + \begin{bmatrix} B_{\Delta t} \\ 0 \end{bmatrix} u[k], \qquad \begin{bmatrix} x[0] \\ x_z[0] \end{bmatrix} = \begin{bmatrix} x_0 \\ x_{zn} \end{bmatrix}, \tag{14}$$

with x_{zn} being the accumulated setpoint tracking error until the nth control cycle. The extended cost function is

$$\widetilde{L}(\tilde{x}[k], u[k]) = \frac{1}{2} \begin{bmatrix} \tilde{x}[k] \\ u[k] \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \widetilde{Q} & \widetilde{S} \\ \widetilde{S}^{\mathsf{T}} & R \end{bmatrix} \begin{bmatrix} \tilde{x}[k] \\ u[k] \end{bmatrix}, \quad V_f(x[N]) = \frac{1}{2}\tilde{x}[N]^{\mathsf{T}}\widetilde{Q}_f\tilde{x}[N]$$

with augmented tuning matrices $\widetilde{Q} = \text{blkdiag}(Q, Q_e)$, $\widetilde{S} = \text{blkdiag}(S, 0)$, and $\widetilde{Q}_f = \text{blkdiag}(Q_f, Q_e)$.

References

[1] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl. *Model Predictive Control: Theory, Computation and Design*. 2nd ed. Nob Hill Publ., LLC., 2020. ISBN: 978-0-9759377-5-4.

[2] D. E. Seborg et al. *Process dynamics and control*. John Wiley & Sons, 2016.