Simulation-Based Engineering Lab
University of Wisconsin-Madison

Technical Report TR-2019-02

The Basics of Model Predictive Control in Multibody Dynamics

Shuping Chen and Dan Negrut

October 14, 2019

Abstract

A Model Predictive Control (MPC) approach is used in conjunction with two examples – mass-spring-damper and pendulum, to gauge the robustness of the MPC approach in the context of simple multibody dynamics problems. For the mass-spring-damper example, a sensitivity analysis with respect to the weight factor and the value m_2 of the second mass in the system was carried out to assess their influence on tracking performance. For the pendulum example, the solution used the same MPC controller used in the mass-spring-damper example.

Keywords: MPC, Mass-Spring-Damper, Pendulum, Tracking Control

Contents

1 Introduction

Model predictive control (MPC) has its roots in optimal control. The basic concept of MPC is to use a dynamic model to forecast system behavior, and optimize the input to produce the best decision - the control move at the current time. Models are therefore central to every form of MPC. Because the optimal control move depends on the initial state of the dynamic system, a second basic concept in MPC is to use the past record of measurements to determine the most likely initial state of the system. The state estimation problem is to examine the record of past data, and reconcile these measurement to determine the most likely initial state of the system. Both the regulation problem, in which a model forecast is used to produce the optimal control action, and the estimation problem, in which the past record of measurements is used to produce an optimal state estimate, involve dynamic models and optimization [1].

The goal of this paper is to implement the model predictive control method to track the desired motion for a mass-spring-damper example and a pendulum example in Matlab®. The control objective is to make the plant follow the given tracking objective with optimal inputs calculated from the MPC controller, that is, for the mass-spring-damper example, to make the x displacement of m_1 in the plant track the given displacement of the objective; for the pendulum example, to make the rotation angle of the pendulum track the given objective.

The remainder of this paper is organized as follows. In Section 2, we developed the mathematical equations of the plant and the predictive model for the mass-spring-damper example and the pendulum example. In Section 3, we followed the steps to design the MPC controllers for these two examples respectively. In Section 4, we presented the results of the model validation, tracking performance, and sensitivity analysis. In Section 5 we wrapped up the paper and presented potential future work for this study.

2 Modeling

Model predictive control systems are designed based on a mathematical model of the plant. The model to be used in the control system design is in the form of state-space. By using a state-space model, the information required for predicting ahead is represented by the state variable at the current time [2]. This section describes the mass-spring-damper model and pendulum model used for simulations.

2.1 Mass-Spring-Damper Example

The mass-spring-damper example is shown in Fig. 1. There are two masses in the system, where the mass of m_2 is much less than that of m_1. The input force is applied to the m_1.

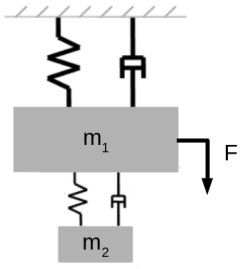


Figure 1: The mass-spring-damper with applied force

2.1.1 Plant

The dynamic equation of the system, governed by Newton's law, can be described as:

$$m_1\ddot{x}_1 = F(t) + m_1 g - K_1 x_1 - C_1\dot{x} + K_2\left(x_2 - x_1\right) + C_2\left(\dot{x}_2 - \dot{x}_1\right) \tag{1a}$$

$$m_2\ddot{x}_2 = m_2 g - K_2\left(x_2 - x_1\right) - C_2\left(\dot{x}_2 - \dot{x}_1\right), \tag{1b}$$

where K_1, K_2 and C_1, C_2 are the spring stiffness and damping coefficient of m_1, m_2 respectively and F is the external force.

Set $X_p = \begin{bmatrix} x_1 & \dot{x}_1 & x_2 & \dot{x}_2 \end{bmatrix}^T$ as the state variables, the state-space function of the plant is expressed as follows:

$$\dot{X}_p = \begin{bmatrix} \dot{x}_1 \\ \ddot{x}_1 \\ \dot{x}_2 \\ \ddot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\dfrac{K_1 + K_2}{m_1} & -\dfrac{C_1 + C_2}{m_1} & \dfrac{K_2}{m_1} & \dfrac{C_2}{m_1} \\ 0 & 0 & 0 & 1 \\ \dfrac{K_2}{m_2} & \dfrac{C_2}{m_2} & -\dfrac{K_2}{m_2} & -\dfrac{C_2}{m_2} \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{1}{m_1} \\ 0 \\ 0 \end{bmatrix} F(t) + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} g \tag{2a}$$

$$Y_p = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix}. \tag{2b}$$

2.1.2 Prediction Model

For the prediction model, its aim is to approximate the characteristics of the plant. Assuming that the mass of m_2 does not exist, the spring stiffness is K^* and the damping coefficient is C^*, the dynamic equation of the controller can be expressed as:

$$m_1\ddot{x} = F(t) + m_1 g - K^* x - C^*\dot{x}, \tag{3}$$

and the internal controller model is shown in Fig. 2,

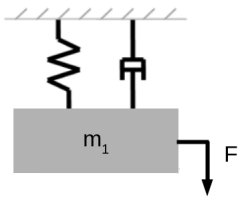


Figure 2: The prediction model

Set $X = \begin{bmatrix} x & \dot{x} \end{bmatrix}^T$ as the state variable, the model in the form of state-space function is shown as follows:

$$\dot{X} = \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\dfrac{K^*}{m_1} & -\dfrac{C^*}{m_1} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{1}{m_1} \end{bmatrix} F(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} g \tag{4a}$$

$$Y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \tag{4b}$$

2.2 Pendulum Example

Consider the pendulum example consisting of a mass point suspended at the end of a rigid link, as depicted in Fig. 3. Let r and θ denote the polar coordinate on the pendulum, and let $p = r\delta_r$ be the position vector of the pendulum, in which δ_r is the unit vectors in polar coordinates. Apply a torque T on the pendulum as the manipulated variable and neglect all frictions [1].

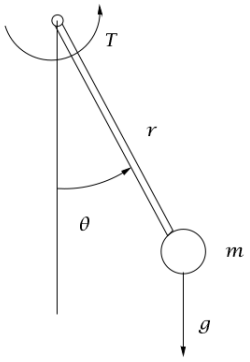


Figure 3: The pendulum with applied torque [1]

2.2.1 Plant

A momentum balance is implemented on the pendulum mass (assume it is a mass point) to determine both the force exerted by the link [1]

$$t = mR\dot{\theta}^2 + mg\cos\theta, \tag{5}$$

and the equation for the acceleration of the pendulum due to gravity and the applied torque [1]

$$mR\ddot{\theta} - T/R + mg\sin\theta = 0. \tag{6}$$

Choose $x = \begin{bmatrix} x_{p1} \\ x_{p2} \end{bmatrix}$ as the state variable, where $x_{p1} = \theta$, $x_{p2} = \dot{\theta}$, and applied torque u as the controlled input. Assuming that the force exerted at the link can be measured, the state space function is given as follows [1]:

$$\frac{dx_{p1}}{dt} = x_{p2} \tag{7a}$$

$$\frac{dx_{p2}}{dt} = -\frac{g}{R}\sin x_{p1} + \frac{1}{mR^2}u \tag{7b}$$

$$y_p = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \tag{7c}$$

the force exerted on the link is given as [1]:

$$t = mRx_{p2}^2 + mg\cos x_{p1}. \tag{8}$$

2.2.2 Prediction Model

Assuming that the pendulum oscillates within a small angle, then the trigonometric functions can be simplified as: $\sin\theta \approx \theta$ and $\cos\theta \approx 1$, and state vector can be defined as: $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$, where $x_1 = \theta$, $x_2 = \dot{\theta}$, so the state space function of the system can be linearized as below:

$$\dot{X} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\dfrac{g}{R} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{1}{mR^2} \end{bmatrix} u \tag{9a}$$

$$Y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \tag{9b}$$

3 MPC Controller Design

The essence of a MPC scheme is to apply a mathematical model of the system one desires to control to predict and optimize future system behavior online. Such a prediction is accomplished by employing an internal model over a fixed finite time horizon, called the prediction horizon, from the current system state. At each sampling time, the controller generates an optimal control sequence, called control horizon, by solving an optimization problem and the first element of this sequence is applied to the plant. The repetition of this process over time by using the updated measurements creates a feedback loop which continually controls the system, pushing it towards an optimal path. MPC differs, therefore, from conventional control in which the control law is precomputed offline [3].

After generating the prediction model, the general procedure of designing MPC controllers includes prediction of states, development of cost function and constraints, optimization and receding horizon control.

3.1 Mass-Spring-Damper Example

3.1.1 Prediction of the System States

The system to be controlled by MPC is modeled by a discrete state-space model. Herein, the time-continuous state-space functions cannot be used directly in designing the MPC controller, it is necessary to discretize the functions. The discretized state-space representation of the prediction model in the MPC controller is expressed as follows [2] [4]:

$$x(k+1) = A_d x(k) + B_d u(k) + H_d g, \tag{10}$$

where $A_d = I + TA, B_d = TB, H_d = TH$, and the matrix A, B and H correspond to equation (4a). T is the sampling time interval for the discrete state-space model. The output of the internal controller model is defined as

$$y_d(k) = C_d x(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix}. \tag{11}$$

The discretized state-space representation of the plant is expressed as:

$$x_p(k+1) = A_{pd} x_p(k) + B_{pd} u(k) + H_{pd} g, \tag{12}$$

where $A_{pd} = I + TA_p, B_{pd} = TB_p, H_{pd} = TH_p$, and the matrix A_p, B_p and H_p correspond to equation (2a). T is the sampling interval time for the discrete state-space model. The outputs of the plant is defined as

$$y_p(k) = C_p x_p(k) = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix}. \tag{13}$$

Define $\xi(k) = \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix}$ as the new state variable, $\eta(k)$ as the output state variable, and $\Delta u(k) = u(k) - u(k-1)$ as the control input increment, then the discrete state-space controller model can be translated into a new form as follows [2] [4]

$$\xi(k+1) = \tilde{A}_d \xi(k) + \tilde{B}_d \Delta u(k) + \tilde{H}_d g \tag{14a}$$

$$\eta(k) = \tilde{C}_d \xi(k), \tag{14b}$$

where $\tilde{A}_d = \begin{bmatrix} A_d & B_d \\ 0_{m \times n} & I_m \end{bmatrix}, \tilde{B}_d = \begin{bmatrix} B_d \\ I_m \end{bmatrix}, \tilde{H}_d = \begin{bmatrix} H_d \\ 0_m \end{bmatrix}, \tilde{C}_d = \begin{bmatrix} C_d & 0 \end{bmatrix}$ (m denotes the dimension of control input, n denotes the dimension of state variable) [4]. In this study, it is single input single output example, so $m = 1, n = 2, I_m = 1$.

Assuming $\tilde{A}_d$ and $\tilde{B}_d$ are the same through out N_p time steps (i.e., the prediction horizon), the predicted state of the controller model based on the current measurement (from the plant) at time k can be given as [2] [4]

$$\xi(k+1) = \tilde{A}_d\xi(k) + \tilde{B}_d\Delta u(k) + \tilde{H}_d g \tag{15a}$$

$$\xi(k+2) = \tilde{A}_d^2\xi(k) + \tilde{A}_d\tilde{B}_d\Delta u(k) + \tilde{B}_d\Delta u(k+1) + \tilde{A}_d\tilde{H}_d g + \tilde{H}_d g \tag{15b}$$

$$\begin{aligned}\xi(k+3) = {} & \tilde{A}_d^3\xi(k) + \tilde{A}_d^2\tilde{B}_d\Delta u(k) + \tilde{A}_d\tilde{B}_d\Delta u(k+1) + \tilde{B}_d\Delta u(k+2) + \tilde{A}_d^2\tilde{H}_d g \\ & + \tilde{A}_d\tilde{H}_d g + \tilde{H}_d g\end{aligned} \tag{15c}$$

$$\cdots$$

$$\begin{aligned}\xi(k+N_c) = {} & \tilde{A}_d^{N_c}\xi(k) + \tilde{A}_d^{N_c-1}\tilde{B}_d\Delta u(k) + \cdots + \tilde{B}_d\Delta u(k+N_c-1) \\ & + \left(\tilde{A}_d^{N_c-1} + \tilde{A}_d^{N_c-2} + \cdots + \tilde{A}_d + 1\right)\tilde{H}_d g\end{aligned} \tag{15d}$$

$$\cdots$$

$$\begin{aligned}\xi(k+N_p) = {} & \tilde{A}_d^{N_p}\xi(k) + \tilde{A}_d^{N_p-1}\tilde{B}_d\Delta u(k) + \cdots + \tilde{A}_d^{N_p-N_c}\tilde{B}_d\Delta u(k+N_c-1) \\ & + \left(\tilde{A}_d^{N_p-1} + \tilde{A}_d^{N_p-2} + \cdots + \tilde{A}_d + 1\right)\tilde{H}_d g.\end{aligned} \tag{15e}$$

The sequence of future incremental control input computed at time k is denoted by ΔU_m, also represented as $\Delta U_m = \left[\Delta u(k), \ldots, \Delta u(k+m), \ldots, \Delta u(k+N_c-1)\right]^T$. The control input varies for N_c time steps (i.e., the control horizon) and then is held constant up to the preview horizon. The predicted output for the prediction state-space model is denoted by $\eta_m(k) = \left[\eta(k+1), \ldots, \eta(k+N_p)\right]^T$ in this situation, it is straightforward to derive the prediction model of performance outputs over the prediction horizon N_p in a compact matrix form as [2] [4]:

$$\eta_m(k) = \Theta_m\xi(k) + \Gamma_m\Delta U_m + \Psi_m\tilde{H}_d g, \tag{16}$$

where

$$\Theta_m = \begin{bmatrix}\tilde{C}_d\tilde{A}_d & \tilde{C}_d\tilde{A}_d^2 & \ldots & \tilde{C}_d\tilde{A}_d^{N_c} & \ldots & \tilde{C}_d\tilde{A}_d^{N_p}\end{bmatrix}^T \tag{17}$$

$$\eta_m(k) = \begin{bmatrix}\eta(k+1) & \ldots & \eta(k+N_p)\end{bmatrix}^T \tag{18}$$

$$\Delta U_m = \begin{bmatrix}\Delta u(k) & \ldots & \Delta u(k+m) & \ldots & \Delta u(k+N_c-1)\end{bmatrix}^T \tag{19}$$

$$\Gamma_m = \begin{bmatrix}\tilde{C}_d\tilde{B}_d & 0 & \cdots & 0 \\ \tilde{C}_d\tilde{A}_d\tilde{B}_d & \tilde{C}_d\tilde{B}_d & \cdots & 0 \\ \vdots & \vdots & \cdots & 0 \\ \tilde{C}_d\tilde{A}_d^{N_c-1}\tilde{B}_d & \tilde{C}_d\tilde{A}_d^{N_c-2}\tilde{B}_d & \cdots & \tilde{C}_d\tilde{B}_d \\ \vdots & \vdots & \cdots & \vdots \\ \tilde{C}_d\tilde{A}_d^{N_p-1}\tilde{B}_d & \tilde{C}_d\tilde{A}_d^{N_p-2}\tilde{B}_d & \cdots & \tilde{C}_d\tilde{A}_d^{N_p-N_c}\tilde{B}_d\end{bmatrix} \tag{20}$$

$$\Psi_m = \begin{bmatrix} \tilde{C}_d \\ \tilde{C}_d\tilde{A}_d + \tilde{C}_d \\ \tilde{C}_d\tilde{A}_d^2 + \tilde{C}_d\tilde{A}_d + \tilde{C}_d \\ \vdots \\ \tilde{C}_d\tilde{A}_d^{N_p-1} + \tilde{C}_d\tilde{A}_d^{N_p-2} + \cdots + \tilde{C}_d\tilde{A}_d + \tilde{C}_d \end{bmatrix}. \tag{21}$$

3.1.2 Development of Cost Function

Model predictive control solves an optimization problem, specifically, a quadratic program (QP), at each control interval. The solution determines the manipulated variables (MVs) to be used in the plant until the next control interval. The standard cost function is the sum of four terms including output reference tracking, manipulated variable tracking, manipulated variable move suppression and constraint violation, each focusing on a particular aspect of controller performance. The cost function is given as follows:

$$J(z_k) = J_y(z_k) + J_u(z_k) + J_{\Delta u(z_k)} + J_e(z_k). \tag{22}$$

Here, z_k is the QP decision. The standard cost function can also use the following alternative,

$$J(z_k) = \sum_{i=0}^{p-1} \left\{ \left[e_y^T(k+i)Qe_y(k+i)\right] + \left[e_u^T(k+i)R_u e_u(k+i)\right] + \left[\Delta u^T(k+i)R_{\Delta u}\Delta u(k+i)\right] \right\} + \rho_e \varepsilon_k^2, \tag{23}$$

where $Q_{(N_p \times N_p)}$, R_u and $R_{\Delta u(N_c \times N_c)}$ are positive-semi-definite weight matrices.

For a given set-point of x movement at sample time k_i, within a prediction horizon the objective of the predictive control system is to bring the predicted output as close as possible to the given function, where we assume that the set-point signal remains constant in the optimization window. This objective is then translated to find the 'best' control parameter vector ΔU such that an error function between the set-point and the predicted output is minimized. By using incremental sequence of control input signal directly, we define the cost function J that reflects the control objective as [2]:

$$J = (G - \eta_m)^T Q (G - \eta_m) + \Delta U_m^T \bar{R} \Delta U_m, \tag{24}$$

where the first term is linked to the objective of minimizing the errors between the predicted model output and the set-point signal while the second term reflects the consideration of manipulated variable move suppression, preferring small manipulated variable adjustment moves ΔU_m. Q used as the weight matrix for output reference tracking. $\bar{R}$ is a diagonal matrix in the form that $\bar{R} = r_w I_{N_c \times N_c}$ $(r_w \geq 0)$ where r_w is used as tuning parameter [2].

3.1.3 Constraints

The manipulated inputs to most physical systems are bounded subject to physical limitations. These constraints are presented by linear inequalities. Constraints are usually imposed

on states or outputs for reasons of safety, operability, product quality, etc. We should bear in mind one general distinction between input constraints and output or state constraints. The input constraints often represent physical limit. In these cases, if the controller does not respect the input constraints, the physical system enforces limitations on them. In contrast, the output or state constraints are usually desired ones. Whether they are achievable depends on disturbances affecting the system. It is often the function of an MPC controller to determine in real time whether the output or state constraints are achievable or not, and relax them in some satisfactory manner. These considerations lead implementers of MPC often to set up the optimization problem using hard constraints for the input constraints and some form of soft constraints for the output or state constraints [1].

When doing the optimization, for the cost function, the increment of the control input signal in the time domain will be solved. In this study, we just consider the tracking performance without adding the control constraints.

3.1.4 Optimization

To find the optimal ΔU_m that will minimize the cost function J in (24), by using (16), J is expressed as

$$\begin{aligned} J = &\left(G - \Theta_m \xi(k)\right)^T Q \left(G - \Theta_m \xi(k)\right) - 2\left(\Delta U_m^T \Gamma^T + \left(\Psi_m \tilde{H}_d g\right)^T\right) Q \left(G - \Theta_m \xi(k)\right) \\ &+ \Delta U_m^T \left(\Gamma^T Q \Gamma + \bar{R}\right) \Delta U_m + 2\Delta U_m^T \Gamma^T Q \left(\Psi_m \tilde{H}_d g\right) + \left(\Psi_m \tilde{H}_d g\right)^T Q \left(\Psi_m \tilde{H}_d g\right). \end{aligned} \tag{25}$$

From the first derivative of the cost function J [2]:

$$\frac{\partial J}{\partial \Delta U_m} = -2\Gamma^T Q \left(G - \Theta_m \xi(k)\right) + 2\left(\Gamma^T Q \Gamma + \bar{R}\right) \Delta U_m + 2\Gamma^T Q \left(\Psi_m \tilde{H}_d g\right). \tag{26}$$

The necessary condition of the minimum J is obtained as [2]

$$\frac{\partial J}{\partial \Delta U_m} = 0, \tag{27}$$

from which we find the optimal solution for the control input signal as

$$\Delta U_m = \left(\Gamma^T Q \Gamma + \bar{R}\right)^{-1} \Gamma^T Q \left(G - \Theta_m \xi(k)\right) - \left(\Gamma^T Q \Gamma + \bar{R}\right)^{-1} \Gamma^T Q \left(\Psi_m \tilde{H}_d g\right), \tag{28}$$

with the assumption that $\left(\Gamma^T Q \Gamma + \bar{R}\right)^{-1}$ exists. The optimal incremental control input sequence will be obtained after solving the cost function in every control cycle, which is [4]

$$\Delta U_m(k) = \begin{bmatrix} \Delta u(k) & \Delta u(k-1) & \cdots & \Delta u(k+N_c-1) \end{bmatrix}^T. \tag{29}$$

Then, the actual control input with the first element of control sequences for the system can be calculated as [4]:

$$u(k) = u(k-1) + \Delta u(k). \tag{30}$$

3.1.5 Receding Horizon Control

Although the optimal parameter vector ΔU_m contains the control sequence $\Delta u(k), \Delta u(k-1), \cdots, \Delta u(k+N_c-1)$, due to the receding horizon control principle, we only implement the first sample of this sequence, i.e., $\Delta u(k_i)$, and the input signal to the plant is calculated as $u(k_i) = u(k_i-1) + \Delta u(k_i)$, while ignoring the rest of the sequence. When the next sample period arrives, the latest measurement is taken to form the state vector $x(k_i+1)$ for calculation of the new sequence of the control signal. This procedure is repeated in real time to give the receding horizon control law [2].

3.2 Pendulum Example

The MPC controller designed for the pendulum example is almost the same as the mass-spring-damper example.

3.2.1 Prediction of States and Optimal Control Outputs

The state space form of the predictive model for pendulum example is given as

$$\dot{X} = AX + Bu(t); \tag{31a}$$

$$Y = CX; \tag{31b}$$

where $A = \begin{bmatrix} 0 & 1 \\ -\frac{g}{R} & 0 \end{bmatrix}$,$B = \begin{bmatrix} 0 \\ \frac{1}{mR^2} \end{bmatrix}$, $C = \begin{bmatrix} 1 & 0 \end{bmatrix}$. Then, the discretized state-space representation of the predictive model is given as:

$$x(k+1) = A_d x(k) + B_d u(k) \tag{32a}$$

$$y_d(k) = C_d x(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix}, \tag{32b}$$

where $A_d = I + TA, B_d = TB$. And T is the sampling time interval for the discrete state-space model.

Define $\xi(k) = \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix}$ as the new state variable, $\eta(k)$ as the output state variable, and $\Delta u(k) = u(k) - u(k-1)$ as the control input increment, then the discrete state-space controller model can be translated into a new form as follows [2] [4]:

$$\xi(k+1) = \tilde{A}_d \xi(k) + \tilde{B}_d \Delta u(k) \tag{33a}$$

$$\eta(k) = \tilde{C}_d \xi(k), \tag{33b}$$

where $\tilde{A}_d = \begin{bmatrix} A_d & B_d \\ 0_{m\times n} & I_m \end{bmatrix}, \tilde{B}_d = \begin{bmatrix} B_d \\ I_m \end{bmatrix}, \tilde{C}_d = \begin{bmatrix} C_d & 0 \end{bmatrix}$ (where m denotes the dimension of control input, n denotes the dimension of state variable). This example is a single input single output problem, so $m = 1, n = 2, I_m = 1$ [4].

Denote the sequence of future incremental control input computed at time k as ΔU_m, that is $\Delta U_m = \left[\Delta u(k), \ldots, \Delta u(k+m), \ldots, \Delta u(k+N_c-1)\right]^T$. We assume that the control input varies for only the N_c time steps and then is held constant up to the preview horizon. The output of the predictive model over the prediction horizon N_p in a compact matrix form can be derived as [2] [4]:

$$\eta_m(k) = \Theta_m \xi(k) + \Gamma_m \Delta U_m, \tag{34}$$

where Θ_m, $\eta_m(k)$ and Γ_m are the same as (17), (18) and (20).

3.2.2 Development of Cost Function

Here, we defined the cost function J that reflects the control objective as [2]

$$J = (G - \eta_m)^T Q (G - \eta_m) + \Delta U_m^T \bar{R} \Delta U_m, \tag{35}$$

where the first term is linked to the objective of minimizing the errors between the predicted model output and the set-point signal while the second term reflects the consideration of manipulated variable move suppression, preferring small manipulated variable adjustment moves ΔU_m. Q used as the weight matrix for output reference tracking. $\bar{R}$ is a diagonal matrix in the form that $\bar{R} = r_w I_{N_c \times N_c}$ $(r_w \geq 0)$ where r_w is used as tuning parameter [2].

3.2.3 Constraints

In this example, we assume the angular movement of the pendulum is small so as to linearized the model and just consider the motion tracking performance without adding the control constraints.

3.2.4 Optimization

To find the optimal ΔU_m that will minimize the cost function J in (35), by using (34), J is expressed as

$$J = (G - \Theta_m \xi(k))^T Q (G - \Theta_m \xi(k)) - 2\Delta U_m^T \Gamma^T Q (G - \Theta_m \xi(k)) + \Delta U_m^T \left(\Gamma^T Q \Gamma + \bar{R}\right) \Delta U_m. \tag{36}$$

From the first derivative of the cost function J [2]:

$$\frac{\partial J}{\partial \Delta U_m} = -2\Gamma^T Q (G - \Theta_m \xi(k)) + 2\left(\Gamma^T Q \Gamma + \bar{R}\right) \Delta U_m. \tag{37}$$

The necessary condition to minimize J is [2]:

$$\frac{\partial J}{\partial \Delta U_m} = 0, \tag{38}$$

from which we find the optimal solution for the control input signal as [2]

$$\Delta U_m = \left(\Gamma^T Q \Gamma + \bar{R}\right)^{-1} \Gamma^T Q (G - \Theta_m \xi(k)). \tag{39}$$

The optimal incremental control input sequence is obtained by solving the cost function in every control cycle, which is [4]

$$\Delta U_m(k) = \begin{bmatrix} \Delta u(k) & \Delta u(k-1) & \cdots & \Delta u(k+N_c-1) \end{bmatrix}^T. \tag{40}$$

So, the actual control input with the first element of control sequences for the system can be calculated as [4]

$$u(k) = u(k-1) + \Delta u(k). \tag{41}$$

3.2.5 Receding Horizon Control

Although the optimal parameter vector ΔU_m contains the controls $\Delta u(k), \Delta u(k-1), \cdots, \Delta u(k+N_c-1)$ with the receding horizon control principle, we only implement the first sample of this sequence, i.e., $\Delta u(k_i)$, and the input signal to the plant is calculated as $u(k_i) = u(k_i-1) + \Delta u(k_i)$, while ignoring the rest of the sequence. When the next sample period arrives, the more recent measurement is taken to form the state vector $x(k_i+1)$ for calculation of the new sequence of the control signal. This procedure is repeated in real time to give the receding horizon control law [2].

4 Simulation Results

4.1 Definition of Objectives and Parameters

Before implementing the MPC control method, we defined the given tracking objective as $G(t)$. The function of tracking objective for the mass-spring-damper is given as:

$$G(t) = \begin{cases} 0.02 & 0 \le t < 2 \\ 0.02 + 0.1 \times \sin(t-2) & 2 \le t \le 5.14 \\ 0.02 & 5.14 < t \le 10. \end{cases}$$

The function of tracking objective for the pendulum is:

$$G(t) = \begin{cases} 0.03 & 0 \le t < 2 \\ 0.03 + 0.1 \times \sin(t-2) & 2 \le t \le 5.14 \\ 0.03 & 5.14 < t \le 10. \end{cases}$$

The tracking objectives for the mass-spring-damper example and the pendulum example are presented in Fig.4 and Fig.5 respectively. For the MPC implementation, firstly, we compared the output of the prediction model with that of the plant to make sure that the internal prediction model is approximate to the plant. Then we simulated the control system and compared the tracking results with the objective. The model parameters used in the controller were given in Table.1

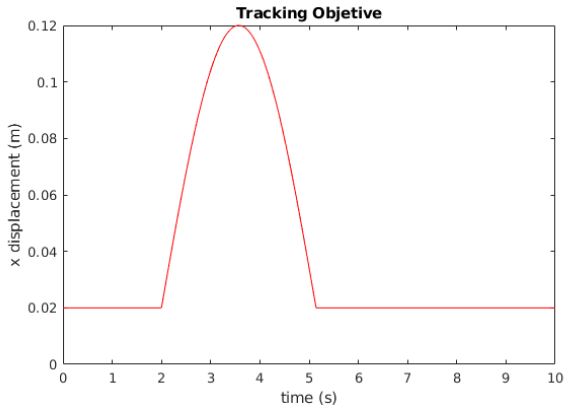


Figure 4: Tracking objective of the mass-spring-damper example

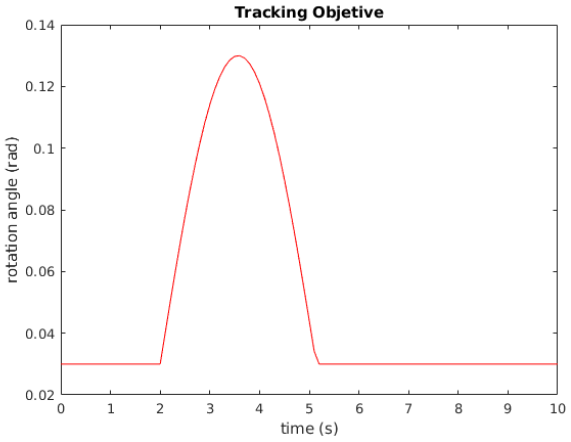


Figure 5: Tracking objective of the pendulum example

Table 1: Modeling parameters

Symbol	Parameter description	Value
m_1	Mass of m_1	$1(kg)$
m_2	Mass of m_2	$0.01(kg)$
K_1	Spring stiffness for m_1 in the plant	$20(N/m)$
K_2	Spring stiffness for m_2 in the plant	$1(N/m)$
C_1	Damping for m_1 in the plant	$0.02(Ns/m)$
C_2	Damping for m_2 in the plant	$0.02(Ns/m)$
K^*	Spring stiffness for m_1 in predictive model	$20(kg)$
C^*	Damping for m_1 in predictive model	$0.02(N/m)$
m	Mass of pendulum	$0.01(kg)$
R	Link distance of pendulum	$0.2(m)$
g	Gravity acceleration	$9.8(m/s^2)$

4.2 Analysis for the Mass-Spring-Damper Example

For the mass-spring-damper example, to implement the proposed MPC control, the plant and the prediction model is generated in Matlab®. The optimal control outputs (i.e., the input to the plant) were calculated from cost function based on the prediction model. During the optimization, the measurement obtained from the plant was input to the MPC controller and the cost function was set to minimize the error between the predicted outputs and the tracking objective in order to calculate the optimal control signal. With the continuous control loop in the receding horizon control, the x displacement of the plant tracked the given objective.

4.2.1 Validation of the Prediction Model

Assuming that the initial condition of the plant is $X_p = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^T$, the output of the prediction model and that of the plant with sinusoidal input signal applied on m_1 are compared in order to make sure that the prediction model was approximate to the plant.

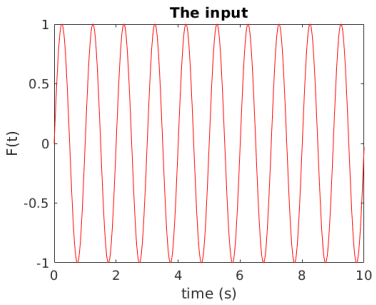


Figure 6: the input

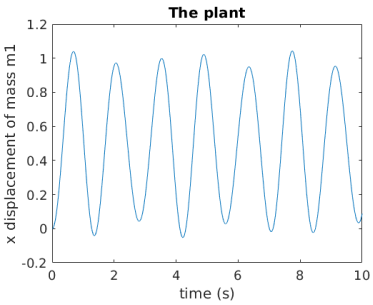


Figure 7: the plant output

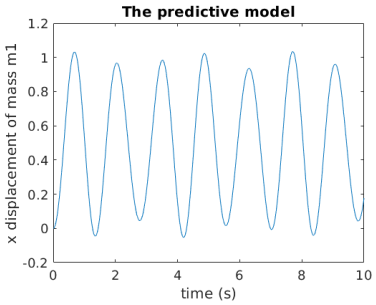


Figure 8: the predicted output

4.2.2 Tracking Performance

We followed the steps to build up the model predictive controller including prediction of the states and controller input, definition of cost function and calculation of the optimal control signal, and the receding horizon control. Then, we simulated the x displacement of the mass m_1 in the plant and compared it with the tracking objective to see whether it's followed. The comparison between outputs of the plant and the tracking objective and that between the prediction model and the tracking objective were shown in Fig. 9 and Fig. 10 respectively.

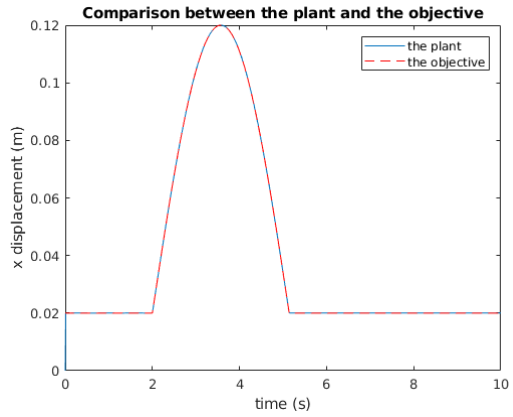


Figure 9: Comparison between the plant output and the objective

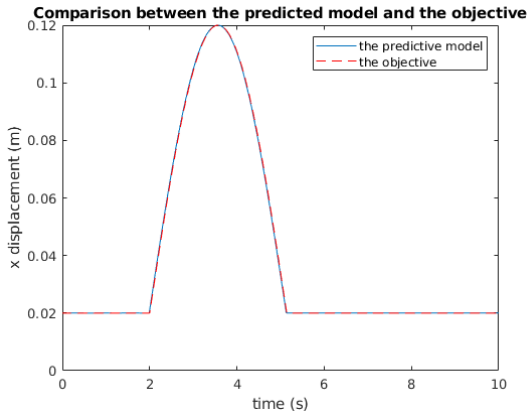


Figure 10: Comparison between the predicted output and the objective

The above results showed that the x displacement of the plant and the prediction model followed the tracking objective with the optimal input signal generated from the MPC controller. The error between the output of the plant and the objective was marginally small.

4.2.3 Sensitivity Analysis

Firstly, to find out the influence of the weight factor $\bar{R}$ in the cost function, we compared the control input signal, control input increment and the tracking results with different $\bar{R}$ while keeping the weight factor in the first term constant. The comparison was shown in Fig. 11 and Fig. 12 respectively.

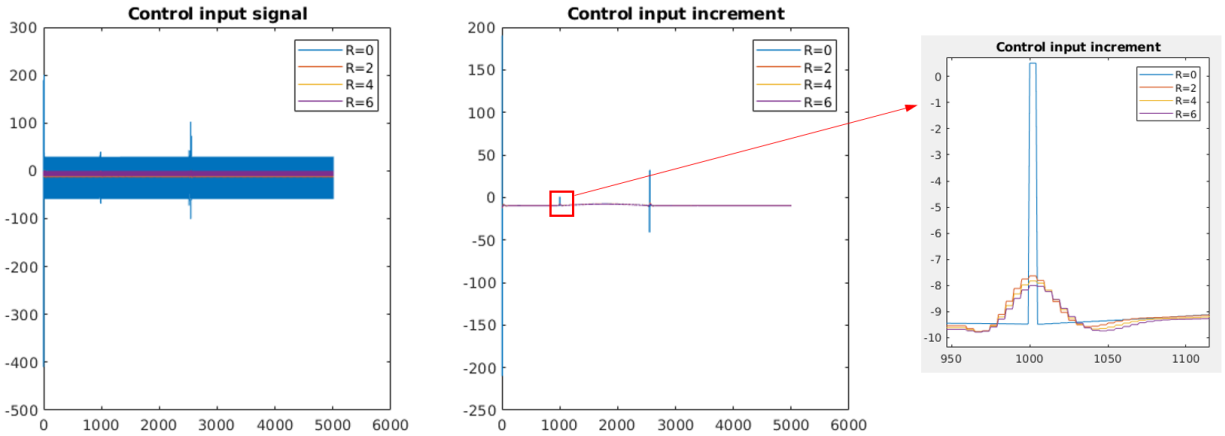


Figure 11: Comparison of control inputs and increments

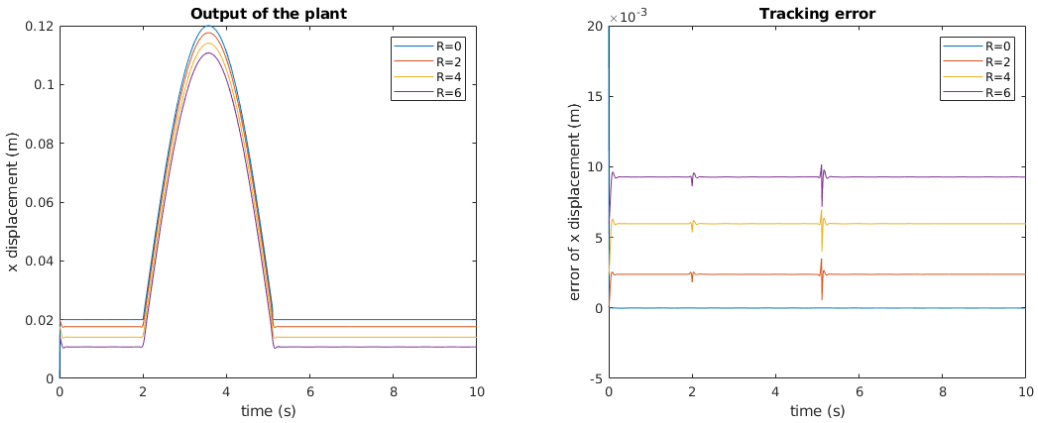
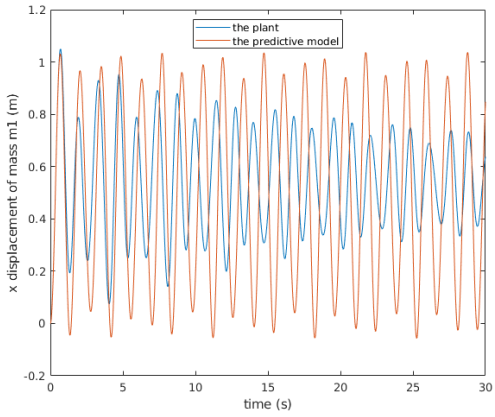


Figure 12: Comparison of tracking results

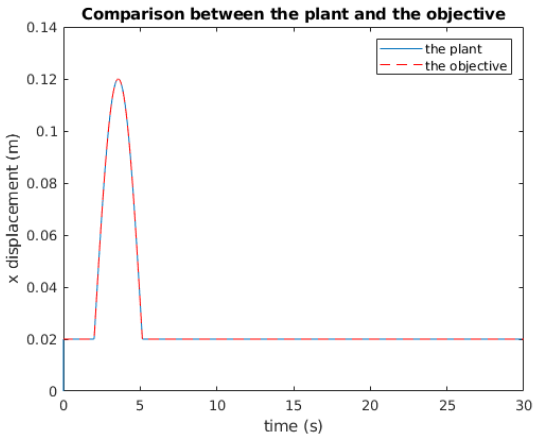
In the comparison, the weight factor $\bar{R}$ in the second term of the cost function (24) was set with value 0, 2, 4 and 6 in order to figure out its influence on the control input and input increment. When the value was 0, the control input signal was large and the change of control input was not smooth. With the value of $\bar{R}$ increased, the control input signal decreased and the change became smoother. The similar results could be found in the increment of control input signal. But this change had worsen the tracking performance, that was, with the value of $\bar{R}$ increased, the error between the tracking objective and the output of the plant became larger.

Secondly, when the plant and the prediction model became increasingly more different, we should take a look at how the MPC tracking performance will change. To consider this situation, we changed the parameters for m_2 in the plant. Furthermore, we lengthened the simulation time from 10 seconds to 30 seconds to see whether the tracking error would increase with the proposed controller.

Case 1, we assumed the mass of m_2 in the plant is 10 times greater than before, i.e., $m_2 = 0.1(kg)$ and other parameters were kept unchanged, the results of the tracking performance were given in Fig. 13.



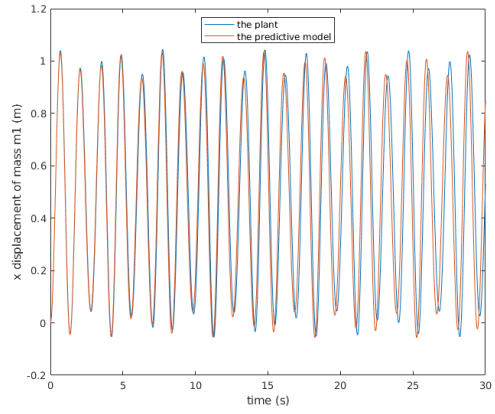
(a) Comparison of the plant and the prediction model



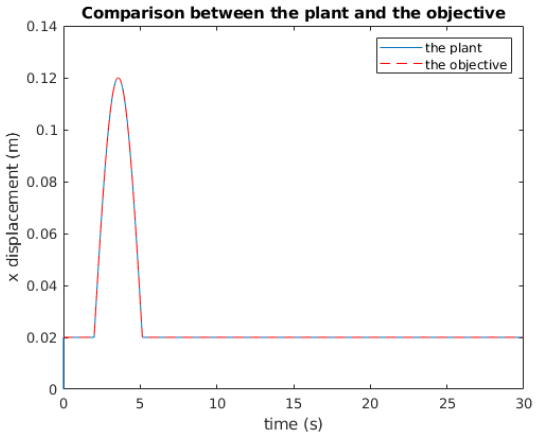
(b) Comparison between the plant and the objective

Figure 13: Case 1

Case 2, we assumed the spring stiffness of m_2 in the plant is bigger than before, i.e., $K_2 = 2.43$ and other parameters were kept unchanged, the results of the tracking performance were given in Fig. 14.



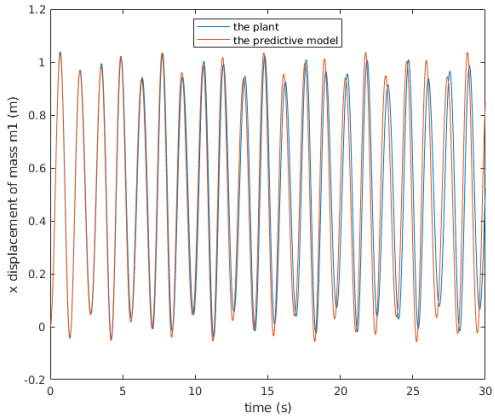
(a) Comparison of the plant and the prediction model



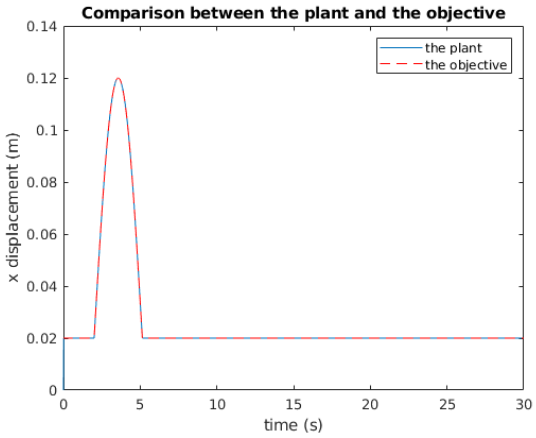
(b) Comparison between the plant and the objective

Figure 14: Case 2

Case 3, we assumed the damping coefficient of m_2 in the plant was bigger than before, i.e., $C_2 = 0.2$ and other parameters were kept unchanged, the results of the tracking performance were given in Fig. 15.



(a) Comparison of the plant and the prediction model



(b) Comparison between the plant and the objective

Figure 15: Case 3

In the above comparisons, the plant and the prediction model had followed the tracking objective under three conditions based on MPC control.

4.3 Analysis for the Pendulum Example

Similar to the mass-spring-damper example, to implement the proposed MPC control, the plant and the prediction model of pendulum example were generated in Matlab®. Firstly, we compared the performance of the plant and the predictive model with initial condition $x_0 = \begin{bmatrix} 0.2 & 0 \end{bmatrix}^T$ and input $u = 0$. Then, we defined the cost function to minimize the error between the output of the plant and the tracking objective. The optimal control signal could be calculated by optimizing the current plant states. After that, we simulated the control system and compared the tracking results with the objective. The comparison between the predictive model and the plant and that between the output of the plant and the tracking objective were shown in Fig. 16 and Fig. 17 respectively.

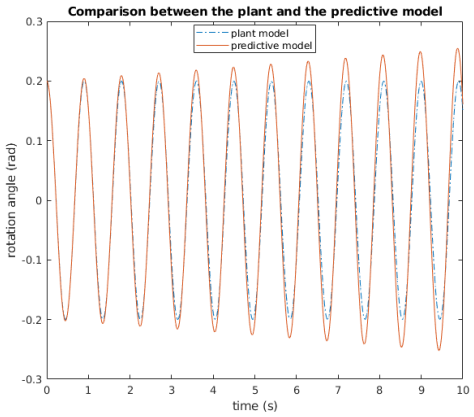


Figure 16: Comparison between the prediction model and the plant

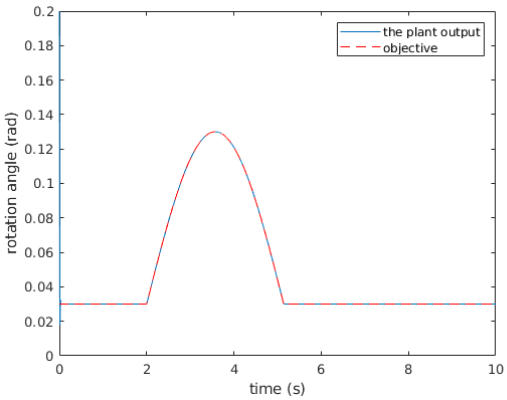


Figure 17: Comparison of outputs between the plant and the objective

In the comparison, the predictive model was approximate to the plant when the rotation angle is small. The rotation angle of the plant has followed the objective with minor error.

5 Conclusions and Future Work

In this study, we implemented the model predictive control for a mass-spring-damper example and a pendulum example to track the given objective in Matlab®. In the mass-spring-damper example, we compared the outputs of the plant and the prediction model to ensure the model is approximate to the plant, then we gave the results of the comparison between the plant and the tracking objective and the comparison between the prediction model and the objective, which demonstrated that the tracking performance is good using MPC control. Moreover, we analysed the sensitivity of weight factor in the cost function to find out the influence on the control input and increment. Besides, considering the situation that the model become increasingly different from the plant, we compared the tracking results under three cases in which the MPC method can be implemented as well. In the pendulum example, the rotation angle of the plant tracked the objective as well with the proposed control method used in the mass-spring-damper example.

In the future study, we will generate the plant in Chrono and the controller in Matlab/Simulink® and implement the model predictive control with cosimulation of Chrono and Matlab/Simulink®. Herein, we assumed the state is known, if there is uncertainty in the model, the state estimation will be needed in the controller in order to make the MPC method more robust.

Acknowledgments

The authors would like to thank Heran Shen of Columbia University for his comments on an early version of this document.

References

[1] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: theory, computation, and design*. Nob Hill Publishing, Madison, Wisconsin, 2nd ed., 2018.

[2] L. Wang, *Model predictive control system design and implementation using MATLAB®*. Springer Science & Business Media, 2009.

[3] F. Kuhne, W. F. Lages, and J. M. G. da Silva, "Model predictive control of a mobile robot using linearization," *Proceedings of mechatronics and robotics*, pp. 525–530, 2004.

[4] J. Gong, Y. Jiang, W. Xu, K. Liu, H. Guo, and Y. Sun, "Multi-constrained model predictive control for autonomous ground vehicle trajectory tracking," *Journal of Beijing Institute of Technology*, vol. 24, no. 4, pp. 441–448, 2015.