Simulation-Based Engineering Lab
University of Wisconsin-Madison

Technical Report TR-2019-03

Model Predictive Control for Multibody Dynamics via Cosimulation of Chrono and Matlab®/Simulink®

Shuping Chen and Dan Negrut

October 14, 2019

Abstract

This report focuses on implementing the model predictive control (MPC) and state estimation with cosimulation of Chrono and Matlab/Simulink® for a mass-spring-damper example and a pendulum example. The plants in the examples were generated using Chrono and the MPC controller with state estimator was designed in Matlab/Simulink®. With the plant measurement and optimal control input, the estimator used least squares estimation to estimate the plant state inputted to the MPC controller. With the estimated plant state and tracking objective, the regulator used model predictive control to calculate the optimal control signal inputted to the plant. In the simulation, firstly, the output of the model was compared with that of the plant to ensure the model provided a good approximation to the plant. Then without the disturbances in the control input and the output measurement, the estimated state and the plant state compared with the tracking objective using cosimulation of Chrono and Matlab/Simulink® were illustrated. After that, when the process and measurement noise were considered, the estimated state, the plant state and the output measurement were compared with the tracking objectives respectively, which illustrated that the displacement of the mass m_1 in the mass-spring-damper example and the rotation angle of the pendulum tracked the given objective using the proposed controller.

Keywords: MPC, Mass-Spring-Damper, Pendulum, Least Squares Estimation

Contents

1 Introduction

The models are central to every form of model predictive control (MPC) in the controller design. 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 [1]. In a different technical report [2], the model predictive control was implemented to follow the desired trajectory for a mass-spring-damper example and a pendulum example in Matlab®, in which we assumed the state of the plant inputted to the controller was known. However, when there are disturbances in the control process and the measurement, the state estimation should be considered in order to make the MPC method more robust. The goal of this report is to take the state estimation into account during MPC controller design using cosimulation of Chrono and Matlab/Simulink®.

In most applications, the variables that are conveniently or economically measureable (y) are a small subset of the variables required to model the system (x). Moreover, the measurement and the state evolution is corrupted by sensor noise and process noise, respectively. Determining a good state estimation for use in the regulator in the face of a noisy and incomplete output measurement is a challenging task. There are various approaches to optimize estimation problems according to different settings, such as least squares estimation, moving horizon estimation, extended kalman filtering, unscented kalman filtering and particle filtering, etc. Moreover, knowing the relationship between a least square problem and a statistically optimal estimation problem for the simple linear case, provides the engineer with valuable insight in choosing useful objective functions for nonlinear estimation [1].

The remainder of this paper is organized as follows: In Section 2 we developed the mathematical models for the mass-spring-damper example and the pendulum example. In Section 3 we introduced the design method of MPC controllers. In Section 4 we specified the linear state estimation method, i.e., the least squares estimation. In Section 5 we implemented the MPC controller and least squares estimator with cosimulation of Chrono and Matlab/Simulink® and the simulation results were then illustrated. In Section 6 we wrapped up the paper and listed the potential future work.

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 [3]. This section describes the mass-spring-damper example and pendulum example used for simulations.

2.1 Mass-Spring-Damper Example

2.1.1 Plant

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. The spring stiffness for m_1 and m_2 are K_1 and K_2 respectively. The damping of m_1 and m_2 are C_1 and C_2 respectively.

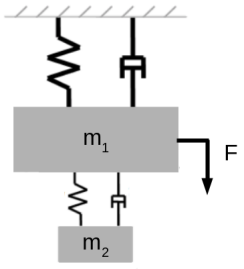

Figure 1: The mass-spring-damper with applied force

The dynamic equation of the system 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 ratio 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

The aim of the prediction model in MPC controller 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 ratio is C^*, the continuous time model for the plant can be given as:

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

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

2.2.1 Plant

Consider the pendulum suspended at the end of a rigid link depicted in Fig. 2. Let r and θ denote the polar coordinate of the center of 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. The pendulum has mass m, the only other external force acting on the pendulum is gravity, and we neglect friction [1].

Force analysis for the pendulum with applied torque is shown in Fig. 2

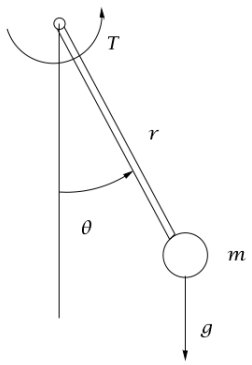

Figure 2: The pendulum with applied torque [1]

2.2.2 Prediction Model

We use a momentum balance on the pendulum mass (you may assume it is a point mass) to determine an equation for the acceleration of the pendulum due to gravity and the applied torque [1]

$$mR\ddot{\theta} - T/R + mg\sin\theta = 0. \tag{5}$$

Define a state vector $X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$, $x_1 = \theta$, $x_2 = \dot{\theta}$ and $u = T$. The state space function is given as follows [1]:

$$\frac{dx_1}{dt} = x_2 \tag{6a}$$

$$\frac{dx_2}{dt} = -\frac{g}{R}\sin x_1 + \frac{1}{mR^2}u \tag{6b}$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \tag{6c}$$

Assuming that the pendulum oscillates within a small angle, then the trigonometric functions can be simplified as: $\sin\theta \approx \theta$ and $\cos\theta \approx 1$, so the state space function of the system can be linearized as below:

$$\dot{X} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{R} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{mR^2} \end{bmatrix} u \tag{7a}$$

$$Y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \tag{7b}$$

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 [4].

The objective of MPC is to find the optimal control vector ΔU such that an error function between the set point and the predicted output is minimized. The procedures of designing MPC controller include prediction of system state, development of cost function, constraints, optimization and receding horizon control.

To design the MPC controller, it is necessary to discretize the functions. 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. Generally, the discrete state space controller model can be translated into a new form as follows [5]:

$$\xi(k+1) = \tilde{A}_d\xi(k) + \tilde{B}_d\Delta u(k) \tag{8a}$$

$$\eta(k) = \tilde{C}_d\xi(k), \tag{8b}$$

where $\tilde{A}_d = \begin{bmatrix} A_d & B_d \\ 0_{m\times n} & I_m \end{bmatrix}, \tilde{B}_d = \begin{bmatrix} B_d \\ I_m \end{bmatrix}, \tilde{H}_d = \begin{bmatrix} H_d \\ 0_m \end{bmatrix}, \tilde{C}_d = \begin{bmatrix} C_d & 0_m \end{bmatrix}$ (m denotes the dimension of control input, n denotes the dimension of state variable) [5].

With the prediction horizon N_p time steps and the control horizon N_c time steps, the prediction model of performance outputs over the prediction horizon N_p in a compact matrix form as [5]

$$\eta_m(k) = \Theta_m \xi(k) + \Gamma_m \Delta U_m, \tag{9}$$

where

$$\eta_m(k) = \begin{bmatrix} \eta(k+1) & \dots & \eta(k+N_p) \end{bmatrix}^T \tag{10}$$

$$\Theta_m = \begin{bmatrix} \tilde{C}_d\tilde{A}_d & \tilde{C}_d\tilde{A}_d^2 & \dots & \tilde{C}_d\tilde{A}_d^{N_c} & \dots & \tilde{C}_d\tilde{A}_d^{N_p} \end{bmatrix}^T \tag{11}$$

$$\Delta U_m = \begin{bmatrix} \Delta u(k) & \dots & \Delta u(k+m) & \dots & \Delta u(k+N_c-1) \end{bmatrix}^T \tag{12}$$

$$\Gamma_m = \begin{bmatrix} \tilde{C}_d\tilde{B}_d & 0 & \cdots & 0 \\ \tilde{C}_d\tilde{A}_d\tilde{B}_d & \tilde{C}_d\tilde{B}_d & \cdots & 0 \\ \vdots & \vdots & \cdots & 0 \\ \tilde{C}_d\tilde{A}_d^{N_c-1}\tilde{B}_d & \tilde{C}_d\tilde{A}_d^{N_c-2}\tilde{B}_d & \cdots & \tilde{C}_d\tilde{B}_d \\ \vdots & \vdots & \cdots & \vdots \\ \tilde{C}_d\tilde{A}_d^{N_p-1}\tilde{B}_d & \tilde{C}_d\tilde{A}_d^{N_p-2}\tilde{B}_d & \cdots & \tilde{C}_d\tilde{A}_d^{N_p-N_c}\tilde{B}_d \end{bmatrix}. \tag{13}$$

For development of cost function, we defined the cost function J that reflects the control objective as [3]

$$J = (G - \eta_m)^T Q (G - \eta_m) + \Delta U_m^T \bar{R} \Delta U_m, \tag{14}$$

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. Q used as the weight matrix for output reference tracking. $\bar{R}$ is a diagonal matrix in the form of $\bar{R} = r_w I_{N_c\times N_c}$ ($r_w \geq 0$) where r_w is used as tuning parameter [3].

In this study, we just considered the motion tracking performance without adding the control constraints. To find the optimal ΔU_m that will minimize the cost function J in (14), by using (9), J is expressed as [3]

$$J = (G - \Theta_m\xi(k))^T Q (G - \Theta_m\xi(k)) - 2\Delta U_m^T \Gamma^T Q (G - \Theta_m\xi(k)) + \Delta U_m^T \left(\Gamma^T Q \Gamma + \bar{R}\right) \Delta U_m. \tag{15}$$

From the first derivative of the cost function J, the necessary condition of the minimum J is obtained as [3]

$$\frac{\partial J}{\partial \Delta U_m} = 0, \tag{16}$$

from which we find the optimal solution for the control input signal as [3]

$$\Delta U_m = \left(\Gamma^T Q \Gamma + \bar{R}\right)^{-1} \Gamma^T Q (G - \Theta_m \xi(k)). \tag{17}$$

The actual control input with the first element of control sequences for the system can be calculated as [5]

$$u(k) = u(k-1) + \Delta u(k). \tag{18}$$

Although the optimal parameter vector ΔU_m contains the incremental control inputs: $\Delta u(k), \Delta u(k-1), \cdots, \Delta u(k+N_c-1)$, considering 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 [3].

4 State Estimation

In most applications, there are disturbances in the measurement and the state evolution, which will influence the current state of the system in the receding horizon control because the MPC obtains the optimal control action based on this initial state at each sampling instant. Hence, it is necessary to determining a good state estimate for use in the regulator under the condition of measurement noise and process noise.

As for the mass-spring-damper example, the estimated state variables include the displacement and velocity of mass m_1, the displacement and velocity of mass m_2, which are given as $Y = \begin{bmatrix} x_1 & \dot{x}_1 & x_2 & \dot{x}_2 \end{bmatrix}^T$. So, the state-space function of the model is expressed as follows:

$$\dot{X} = \begin{bmatrix} \dot{x}_1 \\ \ddot{x}_1 \\ \dot{x}_2 \\ \ddot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\dfrac{K_1+K_2}{m_1} & -\dfrac{C_1+C_2}{m_1} & \dfrac{K_2}{m_1} & \dfrac{C_2}{m_1} \\ 0 & 0 & 0 & 1 \\ \dfrac{K_2}{m_2} & \dfrac{C_2}{m_2} & -\dfrac{K_2}{m_2} & -\dfrac{C_2}{m_2} \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{1}{m_1} \\ 0 \\ 0 \end{bmatrix} F(t) + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} g \tag{19a}$$

$$Y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix}. \tag{19b}$$

As for the pendulum example, the estimated state variables include the rotation angle and angular velocity of the pendulum which are given as $Y = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$, where $x_1 = \theta$, $x_2 = \dot{\theta}$.

So, the state-space function of the model is express as follows:

$$\dot{X} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{R} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{mR^2} \end{bmatrix} u \tag{20a}$$

$$Y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \tag{20b}$$

The method of least squares state estimation is specified as follows [1].
Consider a time horizon with measurements $y(k), k = 0, 1, \ldots, T$. We consider the prior information to be our best initial guess of the initial state $x(0)$, denoted $\bar{x}(0)$, and weighting matrices $P^-(0)$, Q, and R for the initial state, process disturbance, and measurement disturbance. A reasonably flexible choice for objective function is [1]

$$V_T(x(T)) = \frac{1}{2}\bigg(|x(0) - \bar{x}(0)|^2_{(P^-(0))^{-1}} + \sum_{k=0}^{T-1} |x(k+1) - Ax(k)|^2_{Q^{-1}} + \sum_{k=0}^{T} |y(k) - Cx(k)|^2_{R^{-1}} \bigg), \tag{21}$$

in which $x(T) = (x(0), x(1), \ldots, x(T))$. Using forward dynamic programming(DP), we can decompose and solve recursively the least squares state estimation problem. To see clearly how the procedure works, first we write out the terms in the state estimation least squares problem. [1]

$$\min_{x(0),\ldots,x(T)} \frac{1}{2}\bigg(|x(0) - \bar{x}(0)|^2_{(P^-(0))^{-1}} + |y(0) - Cx(0)|^2_{R^{-1}} + |x(1) - Ax(0)|^2_{Q^{-1}} + |y(1) - Cx(1)|^2_{R^{-1}} + |x(2) - Ax(1)|^2_{Q^{-1}} + \ldots + |x(T) - Ax(T-1)|^2_{Q^{-1}} + |y(T) - Cx(T)|^2_{R^{-1}} \bigg). \tag{22}$$

Combine the prior and measurement to define V_0 [1]

$$V_0(x(0)) = \frac{1}{2}\bigg(\underbrace{|x(0) - \bar{x}(0)|^2_{(P^-(0))^{-1}}}_{prior} + \underbrace{|y(0) - Cx(0)|^2_{R^{-1}}}_{measurement} \bigg), \tag{23}$$

which can be expressed also as

$$V_0\left(x(0)\right) = \frac{1}{2}\left(|x(0) - \bar{x}(0)|^2_{(P^-(0))^{-1}} + |(y(0) - C\bar{x}(0)) - C(x(0) - \bar{x}(0))|^2_{R^{-1}} \right), \tag{24}$$

we can combine these two terms into a single quadratic function [1]

$$V_0(x(0)) = (1/2)\left(x(0) - \bar{x}(0) - v\right)' \tilde{H}^{-1} \left(x(0) - \bar{x}(0) - v\right) + constant, \tag{25}$$

in which

$$v = P^-(0)C'(CP^-(0)C' + R)^{-1}\left(y(0) - C\bar{x}(0)\right) \tag{26}$$

$$\tilde{H} = P^-(0) - P^-(0)C'(CP^-(0)C' + R)^{-1}CP^-(0), \tag{27}$$

and we set the constant term to zero because it does not depend on $x(1)$. If we define [1]

$$P(0) = P^-(0) - P^-(0)C'(CP^-(0)C' + R)^{-1}CP^-(0) \tag{28}$$
$$L(0) = P^-(0)C'(CP^-(0)C' + R)^{-1}, \tag{29}$$

and the state estimate $\hat{x}(0)$ is given as follows [1]

$$\hat{x}(0) = \bar{x}(0) + v = \bar{x}(0) + L(0)\left(y(0) - C\bar{x}(0)\right), \tag{30}$$

and we have derived the following compact expression for the function V_0 [1]

$$V_0\left(x(0)\right) = \frac{1}{2}|x(0) - \hat{x}(0)|^2_{P(0)^{-1}}. \tag{31}$$

Adding the next term to the function $V_0(\cdot)$ and denote the sum as $V(\cdot)$ [1]:

$$V\left(x(0), x(1)\right) = V_0(x(0)) + (1/2)|x(1) - Ax(0)|^2_{Q^{-1}} \tag{32}$$
$$V\left(x(0), x(1)\right) = \frac{1}{2}\left(|x(0) - \hat{x}(0)|^2_{P(0)^{-1}} + |x(1) - Ax(0)|^2_{Q^{-1}}\right), \tag{33}$$

add the two quadratics above to obtain

$$V\left(x(0), x(1)\right) = (1/2)|x(0) - v|^2_{\tilde{H}^{-1}} + d, \tag{34}$$

in which

$$v = \hat{x}(0) + P(0)A'(AP(0)A' + Q)^{-1}(x(1) - A\hat{x}(0)) \tag{35}$$
$$d = (1/2)\Big(|v - \hat{x}(0)|^2_{P(0)^{-1}} + |x(1) - Av|^2_{Q^{-1}}\Big). \tag{36}$$

This form is convenient for optimization over the first decision variable $x(0)$; by inspection the solution is $x(0) = v$ and the cost is d. We define the arrival cost to be the result of this optimization [1]

$$V_1^-(x(1)) = \min_{x(0)} V(x(0), x(1)), \tag{37}$$

substituting v into the expression for d and simplifying gives [1]

$$V_1^-(x(1)) = (1/2)|x(1) - A\hat{x}(0)|^2_{(P^-(1))^{-1}}, \tag{38}$$

in which

$$P^-(1) = AP(0)A' + Q. \tag{39}$$

We define $\hat{x}^-(1) = A\hat{x}(0)$ and express the arrival cost compactly as [1]

$$V_1^-(x(1)) = (1/2)|x(1) - \hat{x}^-(1)|^2_{(P^{-1}(1))^{-1}}. \tag{40}$$

Then, for the next stage of the optimization, we combine the arrival cost and measurement to obtain [1]:

$$V_1(x(1)) = \underbrace{V_1^-(x(1))}_{prior} + \underbrace{(1/2)|y(1) - Cx(1)|^2_{R^{-1}}}_{measurement} \tag{41}$$

$$V_1(x(1)) = \frac{1}{2}\left(|x(1) - \hat{x}^-(1)|^2_{(P^-(1))^{-1}} + |y(1) - Cx(1)|^2_{R^{-1}}\right), \tag{42}$$

we can see that this equation is of exactly the same form as (23) in the previous step, by simply changing the variable names, we have [1]

$$P(1) = P^-(1) - P^-(1)C'(CP^-(1)C' + R)^{-1}CP^-(1) \tag{43}$$

$$L(1) = P^-(1)C'(CP^-(1)C' + R)^{-1} \tag{44}$$

$$\hat{x}(1) = \hat{x}^-(1) + L(1)\left(y(1) - C\hat{x}^-(1)\right), \tag{45}$$

and the cost function V_1 is defined as [1]

$$V_1(x(1)) = (1/2)(x(1) - \hat{x}(1))'P(1)^{-1}(x(1) - \hat{x}(1)), \tag{46}$$

in which

$$\hat{x}^-(1) = A\hat{x}(0) \tag{47}$$

$$P^-(1) = AP(0)A' + Q. \tag{48}$$

The final step is recursion, which can be summarized by two steps. First, adds the measurement at time k which can be expressed as [1]:

$$P(k) = P^-(k) - P^-(k)C'\left(CP^-(k)C' + R\right)CP^-(k) \tag{49}$$

$$L(k) = P^-(k)C'\left(CP^-(k)C' + R\right)^{-1} \tag{50}$$

$$\hat{x}(k) = \hat{x}^-(k) + L(k)\left(y(k) - C\hat{x}^-(k)\right). \tag{51}$$

Second, propagates the model to time $k+1$ and produces [1]:

$$\hat{x}^-(k+1) = A\hat{x}(k) \tag{52}$$

$$P^-(k+1) = AP(k)A' + Q, \tag{53}$$

and the recursion starts with the prior information $\hat{x}^-(0) = \bar{x}(0)$ and $P^-(0)$. The process terminates with the final measurement $y(T)$, at which point we have recursively solved the original problem (22) [1].

To figure out how the theory of least squares estimation works, we take a simple analytical example to implement this method, for instance, the pendulum example. The state-space function of the model and the estimated state are given in (20a) and (20b). The plant for this example is given in (5) and simulated in Matlab®. The initial state for the example

is $\begin{bmatrix}0.2 & 0\end{bmatrix}^T$ and the input signal is zero. We generate the normally distributed random signal and add this noise to the plant output state and the control input respectively. Using least squares estimation method, the estimator uses known inputs and the measurements to generate the output and state estimates. The following figures showed the comparison between the measurement of the pendulum rotation angular and the plant output and the comparison between the estimated rotation angular and the plant output. In Fig. 4, the noise level has been significantly reduced by the estimator compared with that in Fig. 3 .

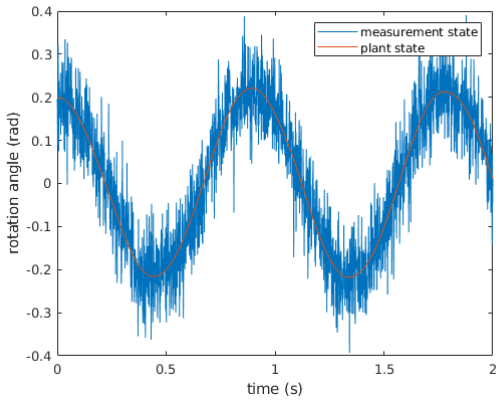

Figure 3: Comparison between estimated state with the plant state

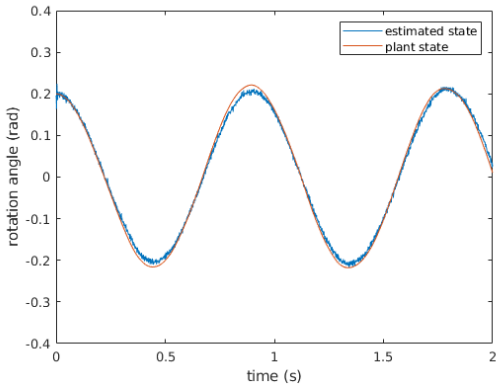

Figure 4: Comparison between measurement with the plant state

5 Simulation Results

5.1 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:

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

The function of tracking objective for the pendulum is:

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

The tracking objectives for the mass-spring-damper system and the pendulum are given in Fig.5 and Fig.6 respectively.

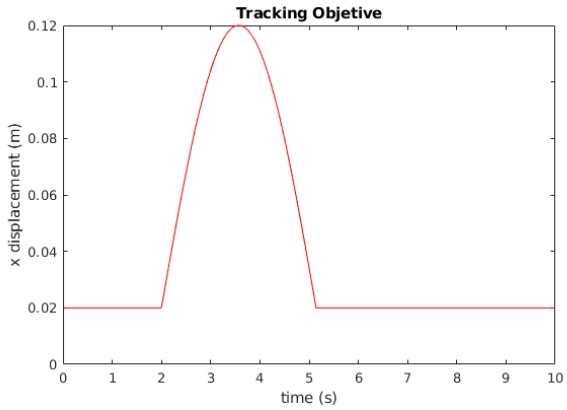

Figure 5: Objective for the mass-spring-damper system

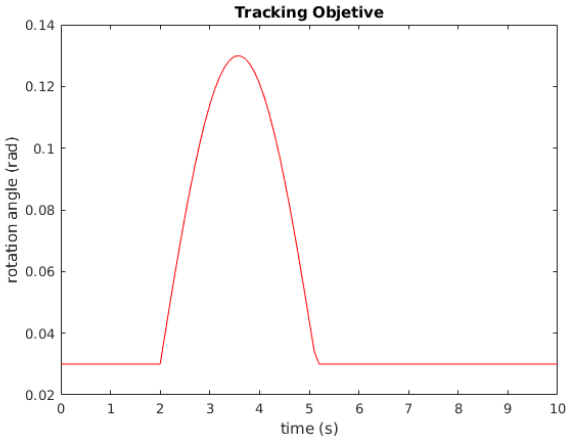

Figure 6: Objective for the pendulum system

With respect to 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 controllers were given in Table.1

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	$0.02(Ns/m)$
m	Mass of pendulum	$0.01(kg)$
R	Link distance	$0.2(m)$
g	Gravity acceleration	$9.8(m/s^2)$

5.2 Cosimulation for the Mass-Spring-Damper Example

For cosimulation, we generated the plant in Chrono and designed the MPC controller with estimator in Matlab/Simulink®. During each time step, with the optimal control signal, the plant produced the output measurement to the estimator. The estimator used the information of optimal control input, the current state of the model and the plant output measurement to calculate the estimated state recursively with least squares estimation. With the estimated state from the estimator, the controller calculated the optimal control input signal with model predictive control method.

As for the mass-spring-damper example, the structure of cosimulation for the MPC controller with state estimator can be designed as below:

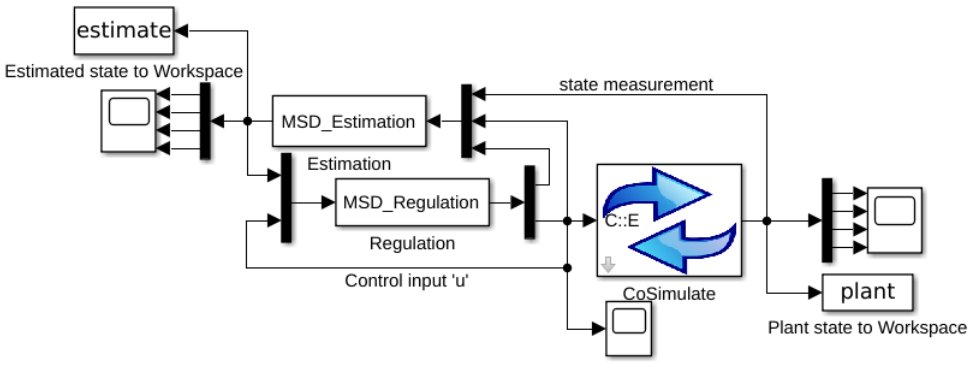

Figure 7: Cosimulation of MPC controller with state estimator

5.2.1 Model Validation

The mass-spring-damper plant generated in Chrono is given as below

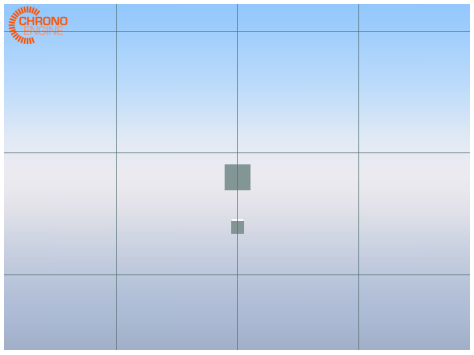

Figure 8: The plant for a mass-spring-damper example generated in Chrono

The model in MPC controller is the base for the state prediction, it is necessary to compare the outputs of the prediction model and the plant to confirm that the model approximates to the plant. The initial state is $\begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^T$ and the control input signal is zero. The simulation results of the model and the plant are listed in Fig. 9 and Fig. 10

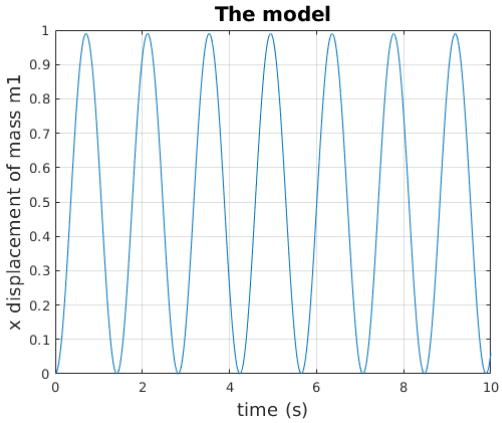

Figure 9: the displacement of m_1 in the mass-spring-damper model

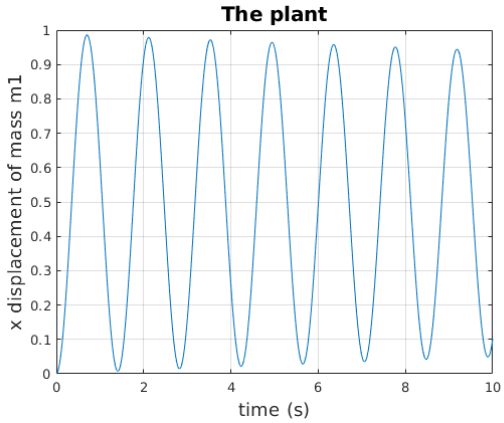

Figure 10: the displacement of m_1 in the mass-spring-damper plant

5.2.2 Tracking Control Performance

In the estimator, the input signals include the measurement from the plant and optimal control input calculated from the MPC controller. Assuming there were no disturbances in the control input and the plant output, the estimated state of the plant was calculated with least squares estimation and the tracking performance of the proposed controller was illustrated as follows:

A. Comparison between the displacement of mass m_1 in the plant outputs and the tracking objective was given in Fig. 11:

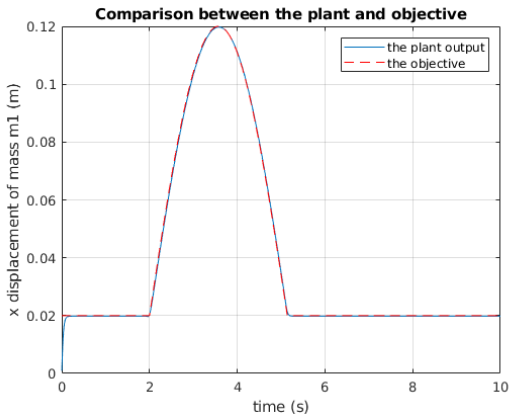

Figure 11: Comparison between the plant output and the objective

B. Comparison between the estimated state of the displacement of mass m_1 and the plant was given in Fig. 12:

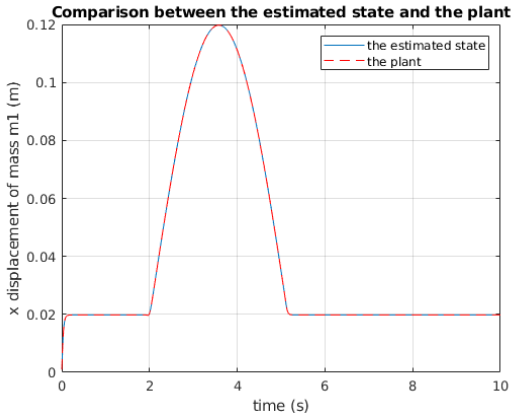

Figure 12: Comparison between the estimated state and the plant

When there were disturbances in the output measurement and the control input, the tracking performance of the proposed controller was also presented. Here, we assume the adding noise is normally distributed random signal, for the measurement noise, the mean value is zero, the variance is 1e-7; for the process noise, the mean value is zero, the variance is 1e-5. After adding the measurement noise to the plant output and adding process noise to the control input, the cosimulation for MPC control was designed as in Fig. 13

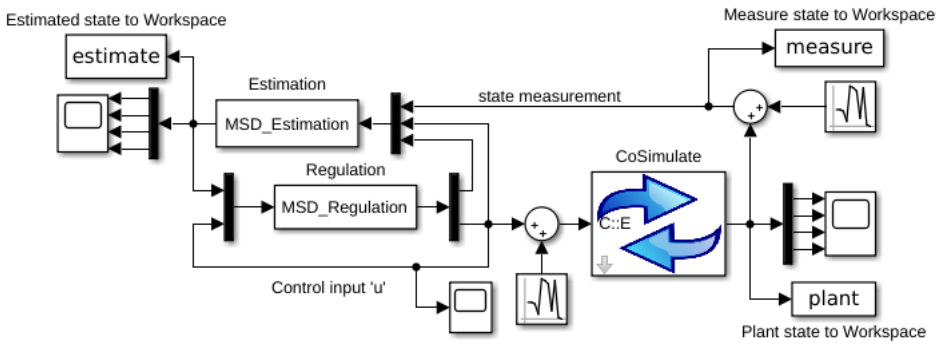

Figure 13: Cosimulation of MPC control adding noise for the mass-spring-damper

A. The estimated state of the displacement of m_1 compared to the plant was illustrated in Fig. 14.

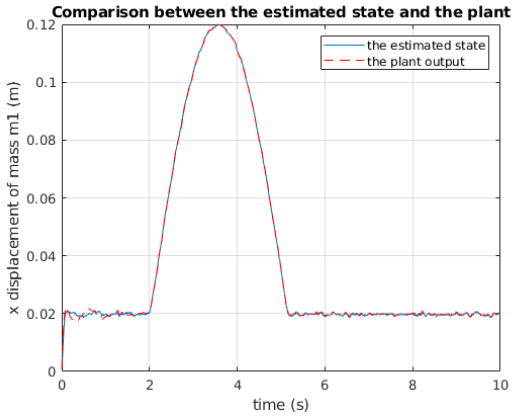

Figure 14: Comparison between the estimated state and the plant

B. To see the tracking performance, the displacement of m_1 of the plant output compared to the tracking objective was illustrated in Fig. 15.

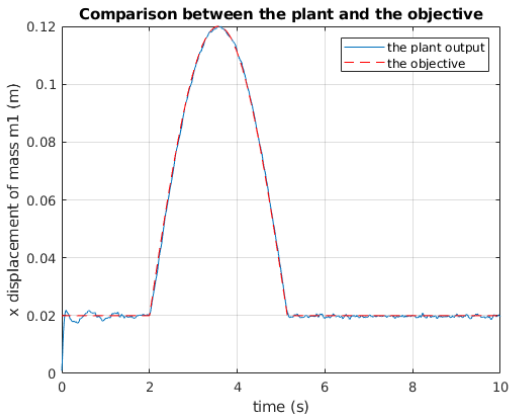

Figure 15: Comparison between the plant and the objective

5.3 Cosimulation for the Pendulum Example

As for the pendulum example, the state estimation also used the least squares estimation method. The structure of cosimulation for MPC controller with state estimator was designed as follows:

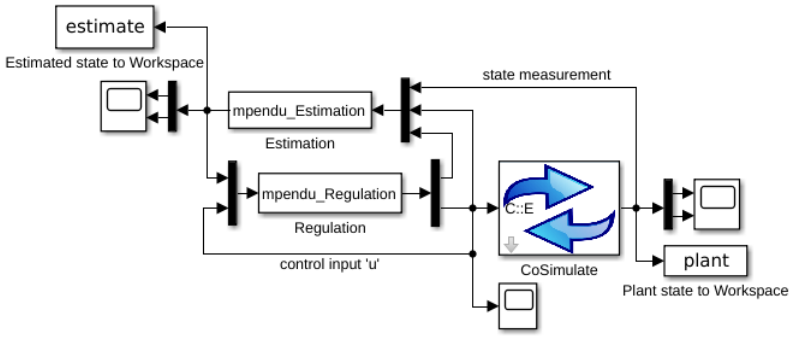

Figure 16: MPC controller with state estimator

5.3.1 Model Validation

The pendulum plant generated in Chrono is given as below:

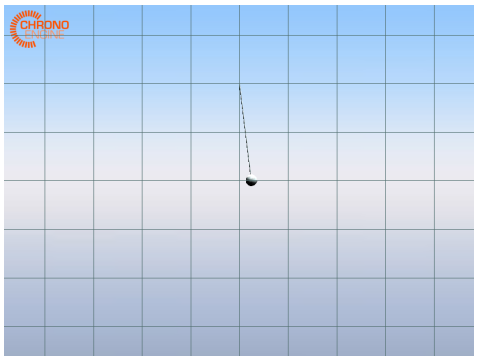

Figure 17: The plant of a pendulum example generated in Chrono

Similarly,it is necessary to compare the outputs of the model and the plant to confirm that the model approximates to the plant, since the accuracy of the model in MPC controller is important for the plant dynamics prediction. The initial state for the pendulum is $\begin{bmatrix}0 & 1\end{bmatrix}^T$ and the control input is zero. The simulation results are listed in Fig. 18 and Fig. 19

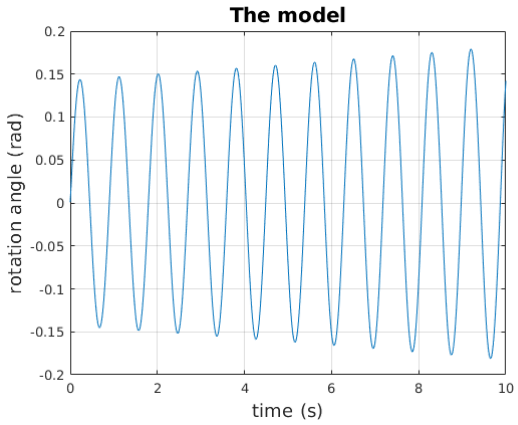

Figure 18: the rotation angle of the pendulum model

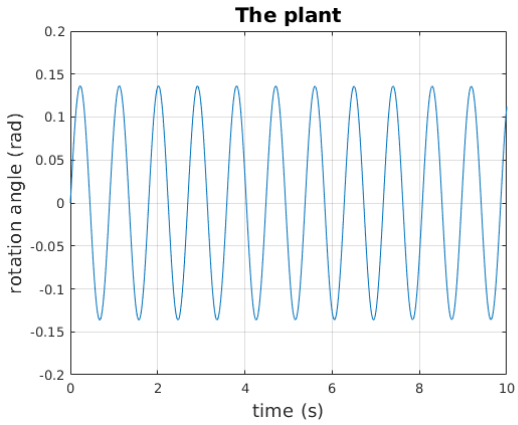

Figure 19: the rotation angle of the pendulum plant

5.3.2 Tracking Control Performance

In the estimator, the input signals include the measurement from the plant and the optimal control input calculated from the MPC controller. When there were no disturbances in the control input and the plant output state, the tracking performance of the proposed controller for pendulum was illustrated as follows:

A. Comparison between the plant state and the tracking objective is given in Fig. 20:

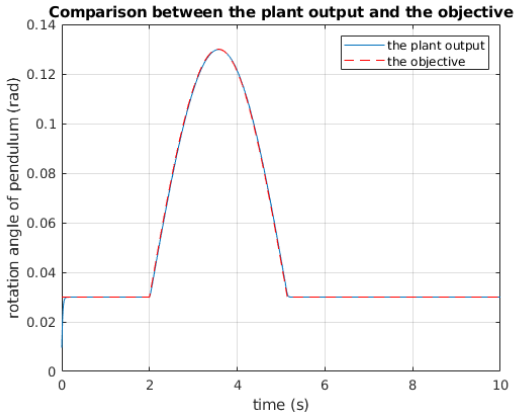

Figure 20: Comparison between the plant output state and the tracking objective

B. Comparison between the estimated state and the plant output is given in Fig. 21:

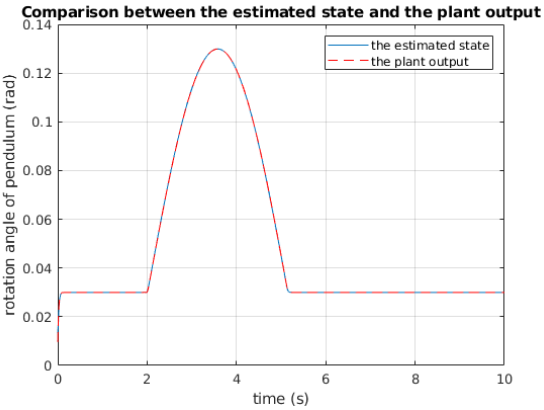

Figure 21: Comparison between the estimated state and the plant

When there were disturbances in the plant output and the control input, the cosimulation of the proposed controller was designed in Fig. 22. Here, we assume the adding noise is normally distributed random signal, for the measurement noise, the mean value is zero, the variance is 1e-7; for the process noise, the mean value is zero, the variance is 1e-5.

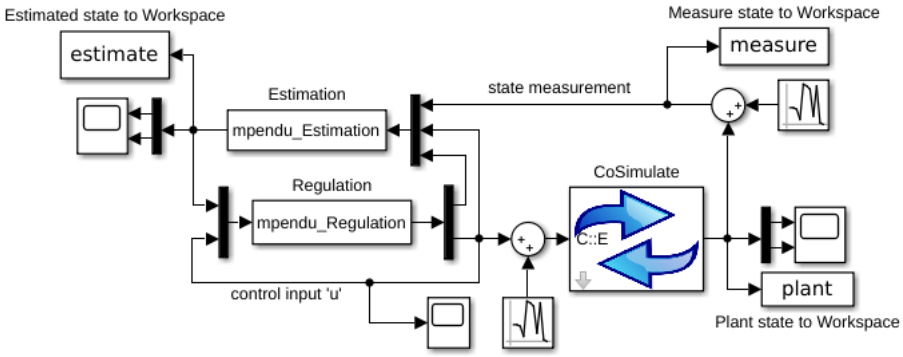

Figure 22: Cosimulation of MPC controller with state estimator for the pendulum

A. The estimated state of the rotation angle of the pendulum compared to the plant output was illustrated in Fig. 23.

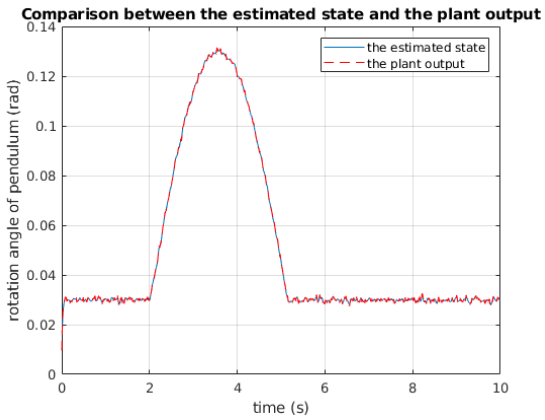

Figure 23: Comparison between the estimated state and the plant

B. To see the tracking performance, the rotation angle of the plant output compared to the tracking objective was illustrated in Fig. 24.

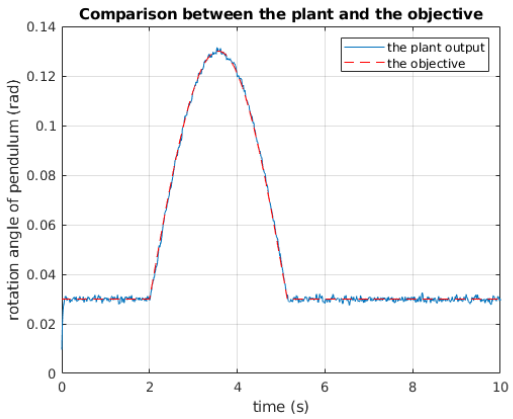

Figure 24: Comparison between the plant state and the objective

6 Conclusions and Future Work

In this study, the MPC controller with state estimator for the mass-spring-damper and the pendulum examples was implemented with cosimulation of Chrono and Matlab/Simulink®. The methods of the model predictive control and the least squares estimation were specified. The plants of these two examples were generated in Chrono and the MPC controller with the estimator was designed in Matlab/Simulink®. In the simulation, the output of the model was compared with that of the plant to make sure the two are approximate. Then, the comparison between the estimated state and the plant output state and the comparison between the plant and the tracking objective was presented under the condition of no disturbance. When disturbances was considered in the control input and the measurement, we compared the results from the estimated state and the plant output state and also compared the results from the plant output and the tracking objective respectively, which illustrated that the displacement of the mass m_1 and the rotation angle of the pendulum tracked the given objective. So far, the model is linear in the controller, the nonlinear model will be considered and the controller design will be extended to more complex situations in the future work.

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] S. Chen and D. Negrut, "The basics of Model Predictive Control in multibody dynamics," Tech. Rep. TR-2019-02: `https://sbel.wisc.edu/wp-content/uploads/sites/569/2019/10/TR-2019-02.pdf`, Simulation-Based Engineering Laboratory, University of Wisconsin-Madison, 2019.

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

[4] 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.

[5] 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.