Model Predictive Control for Rotary Inverted Pendulum: Simulation and Experiment

Phuc-Hoang Huynh [1], Minh-Hanh Nguyen [2], Nguyen-Phat Pham [3], Hoang-Viet-Phuc Duong [4], Huy-Ha Nguyen [5], Duc-Chung Le [6], Minh-Khoa Nguyen [7], Ngoc-Liem Bui [8], Nguyen-Phi-Long Le [9], Van-Dong-Hai Nguyen [10,*]
[1, 2, 3, 4, 5, 6, 7, 8, 10] Faculty of Electrical and Electronics Engineering, Ho Chi Minh City University of Technology and Education (HCMUTE), Vietnam
[9] Faculty of International Education, Ho Chi Minh City University of Technology and Education (HCMUTE), Vietnam
Email: [1] 21151503@student.hcmute.edu.vn, [2] 21142525@student.hcmute.edu.vn, [3] 21161346@student.hcmute.edu.vn, [4] 21161348@student.hcmute.edu.vn, [5] 20151357@student.hcmute.edu.vn, [6] 20151043@student.hcmute.edu.vn, [7] 20142350@student.hcmute.edu.vn, [8] 20161038@student.hcmute.edu.vn, [9] 20151291@student.hcmute.edu.vn, [10] hainvd@hcmute.edu.vn
*Corresponding Author

***Abstract*—Rotary Inverted Pendulum (RIP) is one of the simplest nonlinear systems commonly used for validating control algorithms. In this study, two controllers, Model Predictive Control (MPC) and Linear Quadratic Regulation (LQR), are simulated and experimentally validated. These controllers are executed in real-time on a PC, while the STM32F407 chip handles control and data acquisition from the pendulum using a high-speed USB interface. Due to the custom-built nature of this model, there are inaccuracies in the model and parameter identification. However, results show that the MPC controller is better at trajectory tracking and maintaining balance near the set point compared to the LQR controller. On the other hand, the LQR controller responds more robustly to disturbances and external forces, highlighting distinct differences between MPC's optimization over each prediction horizon and LQR's single-solution approach for the entire prediction horizon.**

***Keywords*—LQR; MPC; Rotary Inverted Pendulum; STM32F4**

I. INTRODUCTION

In nonlinear systems, RIP is considered an easily constructed object with a simple mechanical structure but high nonlinearity [1]. Therefore, this system is commonly used in experiments related to identification and control. Various control algorithms have been applied to the inverted pendulum (IP) model, including methods such as PID control [2], Back-stepping [3], fuzzy control [4], Reinforcement Learning [5], as well as optimal control like LQR [6], yielding significant success. However, MPC control is often used for SISO or SIMO systems. It is not popularly used for SIMO systems, such as RIP. The main difficulty in controlling this model in real models is the challenge of identifying the exact system parameters as the requirement of the MPC method. In [7] Quanser model is used to test MPC control. The experiment is successful due to the standard model of this company. However, this experimental model is expensive and the processor in that research is a professional board that cannot be popularized. Therefore, an MPC control that is successful on a self-made platform that is based on the STM32F4 board can be a solution. In this paper, we propose applying the MPC controller, one of the controllers that are used to manage overall processes in industries such as processing plants, oil refineries, and real-time applications [8][9]. Unlike LQR, MPC is an optimal control technique where control actions are calculated to minimize a cost function for a dynamically constrained system over a finite, receding horizon [10]. To highlight the differences between the two algorithms, we will compare MPC and LQR controllers on RIP to clarify the strengths, weaknesses, and advantages of these two control methods.

II. RIP MODELLING

A. The model's kinetic equations

RIP consists of an arm and a pendulum, with a DC motor mounted at the end of the arm. The pendulum is normally stable in a downward position but unstable in an upright position. Therefore, a controller must be designed to keep the pendulum in an upright position and move it along a predefined trajectory. The specific structure is shown in Fig. 1 [1].

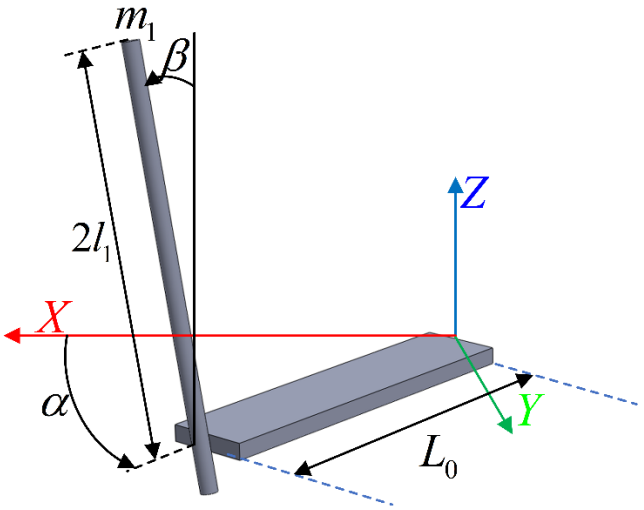

Fig. 1. Mathematical IP model

We use the Parameter Estimator toolbox in MATLAB to estimate system parameters [11]. The parameters of the pendulum are shown in Table 1.

Table 1. Parameters of RIP model

Sympol	Description	Value	Unit
m_1	Mass of pendulum	0.24297	Kg
l_1	Half-length of pendulum	0.20147	m
L_0	Length of pendulum arm	0.14902	m
J_0	Moment of inertia of arm	0.0045556	$Kg.m^2$
J_1	Inertia moment of pendulum	0.0017725	$Kg.m^2$
C_0	Friction coefficient of arm	0.0063986	$\frac{Kgm^2}{s}$
C_1	Friction coefficient of pendulum	0.0065929	$\frac{Kgm^2}{s}$
g	Gravitational acceleration constant	9.81	$\frac{m}{s^2}$
α	Pendulum arm angle		rad
β	Pendulum angle		rad
V	Armature voltage		V
K_t	Torque constant	0.053344	$\frac{Nm}{A}$
K_b	Back emf constant	0.28834	$\frac{Vs}{rad}$
R_a	Armature resistance	0.72921	Ω
J_m	Moment of inertia of rotor	0.012818	$Kg.m^2$
D_m	Viscous friction constant	0.0033158	$\frac{Nms}{rad}$

According [1], we have mathematical equations describing RIP as follows:

$$\begin{bmatrix} J_0 + m_1L_0^2 + m_1l_1^2\sin^2\beta & -m_1L_0l_1\cos\beta \\ -m_1L_0l_1\cos\beta & J_1 + m_1l_1^2 \end{bmatrix}\begin{bmatrix}\ddot{\alpha}\\ \ddot{\beta}\end{bmatrix} + \begin{bmatrix} C_0 + \frac{1}{2}m_1l_1^2\dot{\beta}\sin 2\beta & m_1L_0l_1\dot{\beta}\sin\beta + \frac{1}{2}m_1l_1^2\dot{\alpha}\sin 2\beta \\ -\frac{1}{2}m_1l_1^2\dot{\alpha}\sin 2\beta & C_1 \end{bmatrix}\begin{bmatrix}\dot{\alpha}\\ \dot{\beta}\end{bmatrix} + \begin{bmatrix}0\\ -m_1gl_1\sin\beta\end{bmatrix} = \begin{bmatrix}\tau\\ 0\end{bmatrix} \quad (1)$$

The control signal here is the torque of the motor (τ). It needs to be converted into voltage to fit the real system. The torque produced by a DC motor is defined as [12].

$$\tau = \frac{K_tV}{R_m} + (\frac{-K_tK_b}{R_m} - D_m)\dot{\alpha} - J_m\ddot{\alpha} \quad (2)$$

Combining equations (1) and (2), we obtain the mathematical equation for RIP in (3)

$$\begin{bmatrix} J_0 + m_1L_0^2 + m_1l_1^2\sin^2\beta + J_m & -m_1L_0l_1\cos\beta \\ -m_1L_0l_1\cos\beta & J_1 + m_1l_1^2 \end{bmatrix}\begin{bmatrix}\ddot{\alpha}\\ \ddot{\beta}\end{bmatrix} + \begin{bmatrix} C_0 + \frac{1}{2}m_1l_1^2\dot{\beta}\sin 2\beta + \frac{K_tK_b}{R_a} + D_m & m_1L_0l_1\dot{\beta}\sin\beta + \frac{1}{2}m_1l_1^2\dot{\alpha}\sin 2\beta \\ -\frac{1}{2}m_1l_1^2\dot{\alpha}\sin 2\beta & C_1 \end{bmatrix}\begin{bmatrix}\dot{\alpha}\\ \dot{\beta}\end{bmatrix} + \begin{bmatrix}0\\ -m_1gl_1\sin\beta\end{bmatrix} = \begin{bmatrix}\frac{K_tV_{in}}{R_a}\\ 0\end{bmatrix} \quad (3)$$

B. *Linearization at working point*

Defining state variables as in (4)

$$x_1(t) = \alpha, x_2(t) = \dot{\alpha}, x_3(t) = \beta, x_4(t) = \dot{\beta} \quad (4)$$

nonlinear state equations of RIP are listed in (5)

$$\begin{cases} \dot{x}_1(t) = \dot{\alpha} = f_1(x_1, x_2, x_3, x_4, V_{in}) \\ \dot{x}_2(t) = \ddot{\alpha} = f_2(x_1, x_2, x_3, x_4, V_{in}) \\ \dot{x}_3(t) = \dot{\beta} = f_3(x_1, x_2, x_3, x_4, V_{in}) \\ \dot{x}_4(t) = \ddot{\beta} = f_4(x_1, x_2, x_3, x_4, V_{in}) \end{cases} \quad (5)$$

The requirement is to control the arm to keep the pendulum balanced in the upright position. The working point is that both the pendulum angle and arm angle are stationary, and no voltage is applied to the motor. It is described in (6) below

$$x = [0 \quad 0 \quad 0 \quad 0]^T, V_{in} = 0 \quad (6)$$

By linearizing RIP around this upright equilibrium point (where deviation angle β is less than 10^0), we obtain linearized state equations for the pendulum system in the following form:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \quad (7)$$

where $x = [\alpha \quad \dot{\alpha} \quad \beta \quad \dot{\beta}]^T = [x_1 \quad x_2 \quad x_3 \quad x_4]^T; u = V_{in};$

$$x = Ax + Bu \quad (1)$$

$$y = [\alpha \quad \beta]^T = [x_1 \quad x_3]^T;$$

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} & \frac{\partial f_1}{\partial x_4} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_2} \\ \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_2} \\ \frac{\partial f_4}{\partial x_2} & \frac{\partial f_4}{\partial x_2} & \frac{\partial f_4}{\partial x_2} & \frac{\partial f_4}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & k_1 & k_2 & k_3 \\ 0 & 0 & 0 & 1 \\ 0 & k_4 & k_5 & k_6 \end{bmatrix};$$

$$f = (J_0J_1 + J_1J_m + J_1L_0{}^2m_1 + J_0l_1{}^2m_1 + J_ml_1{}^2m_1);$$

$$k_1 = \frac{-(m_1l_1{}^2 + J_1)(C_0R_a + D_mR_a + K_bK_t)}{R_af};$$

$$k_2 = \frac{L_0gl_1^2m_1^2}{f}; k_3 = -\frac{C_1L_0l_1m_1}{f};$$

$$k_4 = -\frac{L_0l_1m_1(C_0R_a + D_mR_a + K_bK_t)}{R_af};$$

$$k_5 = \frac{gl_1m_1(m_1L_0{}^2 + J_0 + J_m)}{f};$$

$$k_6 = -\frac{C_1(m_1L_0{}^2 + J_0 + J_m)}{f};$$

$$B = \begin{bmatrix} \frac{\partial f_1}{\partial u} \\ \frac{\partial f_2}{\partial u} \\ \frac{\partial f_3}{\partial u} \\ \frac{\partial f_4}{\partial u} \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{K_t\left(m_1 {l_1}^2 + J_1\right)}{R_a f} \\ 0 \\ \frac{K_t L_0 l_1 m_1}{R_a f} \end{bmatrix}; C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Substituting the parameters from **Error! Reference source not found.** into equations in (7) and we obtain matrix (8) below.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -1.6931 & 16.547 & -0.2272 \\ 0 & 0 & 0 & 1 \\ 0 & -1.0616 & 51.649 & -0.7091 \end{bmatrix}}_{A} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 4.0204 \\ 0 \\ 2.521 \end{bmatrix}}_{B} u \quad (8)$$

III. Design Controller

In this section, we configure parameters for two controllers: LQR and MPC. Results are simulated in MATLAB Simulink and experimentally tested on a real model.

A. LQR controller

According to the LQR control method, to control the pendulum system, we need to design a state feedback controller [1]:

$$u(t) = -Kx(t) \quad (9)$$

where: K is the control matrix, and x(t) is the state variable matrix.

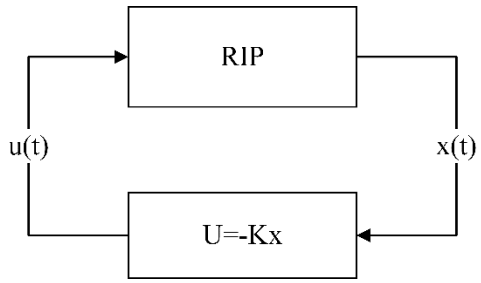

Fig. 2. Structure of the LQR controller

The value of matrix K needs to be optimized, meaning that we must find the value of K minimizing performance index J.

Performance index is chosen to be quadratic, with the final time being: $t_f = \infty$

$$J = \int (x^T Q x + u^T R u) dt \quad (10)$$

Optimal control theory demonstrates that vector K that minimizes quality index (10) is determined by expression:

$$K = R^{-1} B^T P \quad (11)$$

where P is the solution to the Riccati algebraic equation and is computed by solving the Riccati equation:

$$A^T P + PA - PBR^{-1}B^T P = -Q \quad (12)$$

where $Q = \begin{bmatrix} Q_1 & 0 & 0 & 0 \\ 0 & Q_2 & 0 & 0 \\ 0 & 0 & Q_3 & 0 \\ 0 & 0 & 0 & Q_4 \end{bmatrix}, R = a$ are positive definite square matrices used to tune the LQR controller. Here, Q_1, Q_2, Q_3, Q_4 are optimal weights corresponding to state variables $\alpha, \dot{\alpha}, \beta, \dot{\beta}$, respectively.

B. Designed MPC controller

➢ **Predicting future trajectories**

MPC is an optimal control algorithm that considers system constraints, such as its physical limits. As shown in Fig. 3, MPC uses a discrete-time linear model to predict future outputs of the system [13][14]:

$$\begin{cases} x(k+1) = Ax(k) + B_u u(k) + B_v v(k) + B_d n_d(k) \\ y(k) = Cx(k) + D_v v(k) + D_d n_d(k) \end{cases} \quad (13)$$

Where:

$x(k)$ is the state vector, $y(k) = [\alpha(k) \quad \beta(k)]^T$ is vector observed at the system's RIP, $u(k) = V_{in}$, $v(k)$, and $n_d(k)$ are dimensionless manipulated variables, measured disturbances, and unmeasured input disturbances, respectively. Discrete state-space matrices A, B, C, B_u, B_v, B_d, D_v, D_d are computed from the continuous linear model using a discrete sampling time Ts.

Consider the problem of predicting future trajectories of the model performed at time k=0. Set $n_d(i)$=0 for all prediction instants i, and obtain [15] as

$$y(i|0) = C\left[\begin{array}{l} A^i x(0) + \\ + \sum_{h=0}^{i-1} A^{i-1}\left(B_u\left(u(-1) + \sum_{j=0}^{h} \Delta u(j)\right) + \right. \\ \left. + B_v v(h)\right) \\ + D_v v(i) \end{array}\right] \quad (14)$$

The solution to the equation (14) is:

$$\begin{bmatrix} y(1) \\ \dots \\ y(p) \end{bmatrix} = S_x x(0) + S_{u1} u(-1) + S_u \begin{bmatrix} \Delta u(0) \\ \dots \\ \Delta u(p-1) \end{bmatrix} + H_v \begin{bmatrix} v(0) \\ \dots \\ v(p) \end{bmatrix} \quad (15)$$

Where:

$$S_x = \begin{bmatrix} CA \\ CA^2 \\ \dots \\ CA^P \end{bmatrix} \in R^{pn_y \times n_x};$$

$$S_{u1} = \begin{bmatrix} CB_u \\ CB_u + CAB_u \\ \dots \\ \sum_{h=0}^{p-1} CA^h B_u \end{bmatrix} \in R^{pn_y \times n_u};$$

$$S_u = \begin{bmatrix} CB_u & 0 & \dots & 0 \\ CB_u + CAB_u & CB_u & \dots & 0 \\ \dots & \dots & \dots & \dots \\ \sum_{h=0}^{p-1} CA^h B_u & \sum_{h=0}^{p-2} CA^h B_u & \dots & CB_u \end{bmatrix} \in R^{pn_y \times pn_u};$$

$$H_v = \begin{bmatrix} CB_v & D_v & 0 & \dots & 0 \\ CAB_v & CB_v & D_v & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ CA^{p-1}B_v & CA^{p-2}B_v & CA^{p-3}B_v & \dots & D_v \end{bmatrix}; \in R^{pn_y \times (p+1)n_v}$$

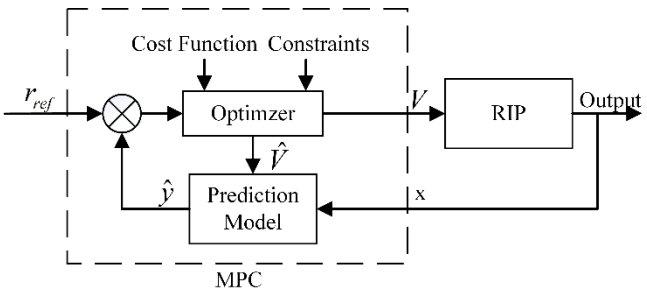

Fig. 3. Structure of MPC controller

➢ **Optimization Variables**

Let *m* be the number of free control moves, and let $z = [z_0; \dots; z_{m-1}]$. Then, it yields

$$\begin{bmatrix} \Delta u(0) \\ \dots \\ \Delta u(p-1) \end{bmatrix} = J_M \begin{bmatrix} z_0 \\ \dots \\ z_{m-1} \end{bmatrix} \quad (16)$$

where J_M depends on the choice of blocking moves

➢ **Standard cost function**

Based on predicted states, MPC calculates optimal control sequences to minimize cost function:

$$J(z, \varepsilon) = \left(\begin{bmatrix} u(0) \\ \dots \\ u(p-1) \end{bmatrix} - \begin{bmatrix} u_{target}(0) \\ \dots \\ u_{target}(p-1) \end{bmatrix} \right)^T \quad (17)$$

$$W_u^2 \left(\begin{bmatrix} u(0) \\ \dots \\ u(p-1) \end{bmatrix} - \begin{bmatrix} u_{target}(0) \\ \dots \\ u_{target}(p-1) \end{bmatrix} \right) + \begin{bmatrix} \Delta u(0) \\ \dots \\ \Delta u(p-1) \end{bmatrix}^T W_{\Delta u}^2 \begin{bmatrix} \Delta u(0) \\ \dots \\ \Delta u(p-1) \end{bmatrix} + \left(\begin{bmatrix} y(1) \\ \dots \\ y(p) \end{bmatrix} - \begin{bmatrix} r(1) \\ \dots \\ r(p) \end{bmatrix} \right)^T W_y^2 \left(\begin{bmatrix} y(1) \\ \dots \\ y(p) \end{bmatrix} - \begin{bmatrix} r(1) \\ \dots \\ r(p) \end{bmatrix} \right) + \rho_\varepsilon \varepsilon^2$$

Where:

$$W_u = L_{S_u}^{-1} diag\left(w_{0,1}^u, w_{0,2}^u, \dots, w_{0,n_u}^u, \dots w_{p-1,1}^u, w_{p-1,2}^u, \dots, w_{p-1,n_u}^u\right)$$

$$W_{\Delta u} = L_{S_u}^{-1} diag\left(w_{0,1}^{\Delta u}, w_{0,2}^{\Delta u}, \dots, w_{0,n_u}^{\Delta u}, \dots w_{p-1,1}^{\Delta u}, w_{p-1,2}^{\Delta u}, \dots, w_{p-1,n_u}^{\Delta u}\right)$$

$$W_y = L_{S_y}^{-1} diag\left(w_{1,1}^y, w_{1,2}^y, \dots, w_{1,n_y}^y, \dots w_{p,1}^y, w_{p,2}^y, \dots, w_{p,n_y}^y\right)$$

L_{Sy} and L_{Su} are diagonal matrices or outputs and MV scaling factors, respectively.

$w_{i,j}^y$: Tuning weight for the *j*th plant output at the *i*th prediction horizon step (dimensionless).

$w_{i,j}^u$: Tuning weight for the *j*th MV at the *i*th prediction horizon step (dimensionless)

$w_{i,j}^{\Delta u}$: Tuning weight for the *j*th MV movement at the *i*th prediction horizon step (dimensionless)

ε_k : Slack variable at the control interval *k* (dimensionless).

ρ_ε : Constraint violation penalty weight (dimensionless).

➢ **Constraints**

When a system contains physical limitations, such as motor voltage V, MPC accounts for these limitations in optimization problems using hard constraints.

$$u(k+i)_{max} \le u \le u(k+i)_{min} \quad (18)$$

➢ **Optimization Variables**

Let *m* be a number of free control moves, and let $z = [z_0; \dots; z_{m-1}]$. Then

$$\begin{bmatrix} \Delta u(0) \\ \dots \\ \Delta u(p-1) \end{bmatrix} = J_M \begin{bmatrix} z_0 \\ \dots \\ z_{m-1} \end{bmatrix} \quad (19)$$

where J_M depends on the choice of blocking moves. Together with the slack variable ε, vectors $z_0, \dots, z_{m-1}$ constitute free optimization variables of the optimization problems.

➢ **QP Solvers**

Model predictive controller QP solver converts a linear MPC optimization problem to the general form QP problem [16]:

$$\underset{x}{Min}\left(\frac{1}{2}x^T Hx + f^T x\right) \quad (20)$$

Subject to linear inequality constraints

$$Ax \leq b \quad (21)$$

where

- x is the solution vector.
- H is the Hessian matrix. This matrix is constant when your prediction model and tuning weights do not change at run time.
- A is a matrix of linear constraint coefficients. This matrix is constant when your prediction model does not change at run time.
- b and f are vectors.

IV. CONTROL RESULTS AND DISCUSSION

In this section, we simulate RIP tracking the set-point signal at arm angle. The experiment will use Simulink Real-Time Target with STM32F407VE control chip. The sampling time for the system is Ts=0.01s, with control parameters as follows:

➢ **LQR**

Because we use a microcontroller to control with a sampling time $T_s = 0.01(s)$, the discrete-time matrices A, B and the weighting matrices for the LQR controller have the following values:

$$A_d = \begin{bmatrix} 1 & 0.01 & 0.0008 & 0 \\ 0 & 0.9832 & 0.1636 & -0.0014 \\ 0 & -0.0001 & 1.0026 & 0.01 \\ 0 & -0.0105 & 0.5142 & 0.9955 \end{bmatrix} \quad (22)$$

$$B_d = [0.0002 \quad 0.0398 \quad 0.0001 \quad 0.0249]^T \quad (23)$$

$$Q = \begin{bmatrix} 30 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 50 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (24)$$

$$R = 0.7 \quad (25)$$

From (22) to (25) we obtain the matrix K:

$$K = [-4.998 \quad -3.713 \quad 91.02 \quad 12.872] \quad (26)$$

➢ **MPC**

The configuration of the system's input and output measurements is shown Fig. 4, along with the controller parameters as follows:

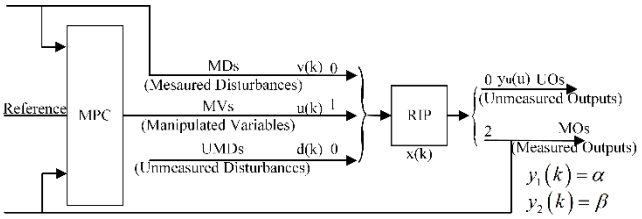

Fig. 4. Configuration of MPC controller for RIP

- Sample time: Ts = 0.01s
- Prediction horizon: P = 50
- Control horizon: m = 3
- $L_{s_y} = \begin{bmatrix} 1 & 0 \\ 0 & 0.02 \end{bmatrix}; L_{s_u} = 1$
- $w_{i,1}^y = 10; w_{i,2}^y = 0.5$
- $w_{i,1}^u = 0; w_{i,1}^{\Delta u} = 0.01$
- $-12 \leq u(k+i) \leq 12$

A. Results in Simulation

We use MATLAB/Simulink to simulate the output responses of two controllers. The simulation diagram is shown in Fig. 5. Simulation results of MPC and LQR controllers with white noise (Noise power = [0.000001]) and arm angle tracking set-point signal f=1 (rad) are shown in Fig. 6.

Simulation results show that the output responses of the arm and pendulum under both LQR and MPC controllers are similar. However, a control signal for the LQR controller has a larger amplitude oscillation, ranging from [−40, 40] (V), whereas the control signal for the MPC controller is [−12, 12] (V).

⇨ The results indicate that the control quality of the MPC controller is better. It also ensures that the operating voltage threshold of the motor is maintained even under significant disturbances affecting the system.

Simulation results of two controllers with white noise (Noise power = [0.0000001]) and arm angle tracking set-point signal f=sin(0.2πt) (rad) are shown in Fig. 7.

Simulation results show that the output response at the pendulum angle is similar for both controllers. However, the response at the arm angle is better with the MPC controller compared to the LQR controller.

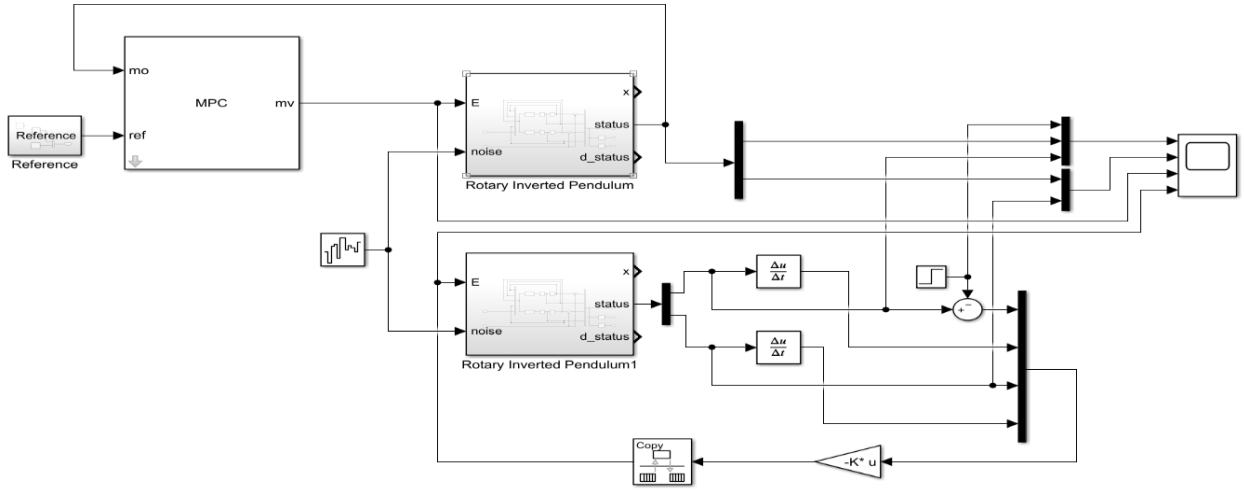

Fig. 5. Simulation diagram of MPC and LQR controllers

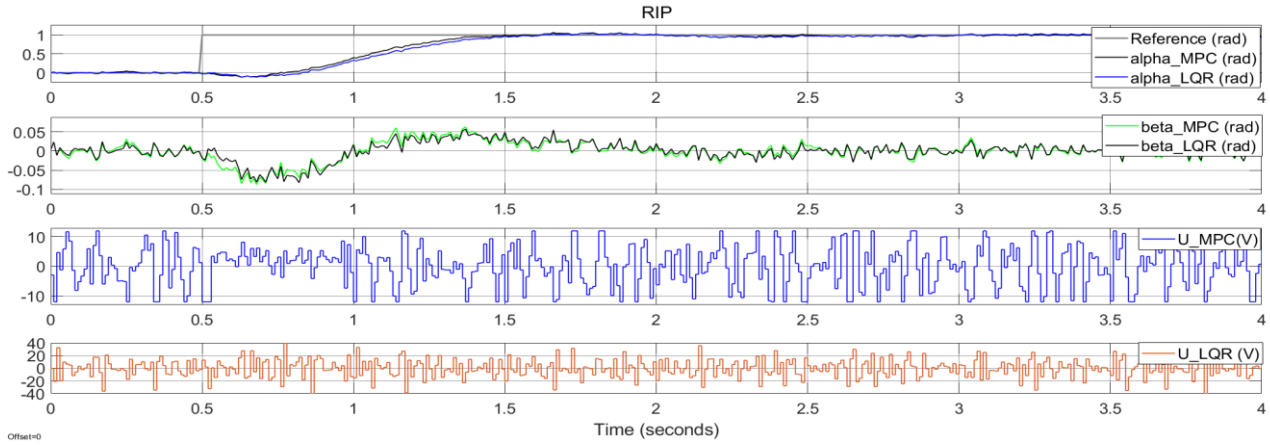

Fig. 6. Output response of RIP tracking a step signal

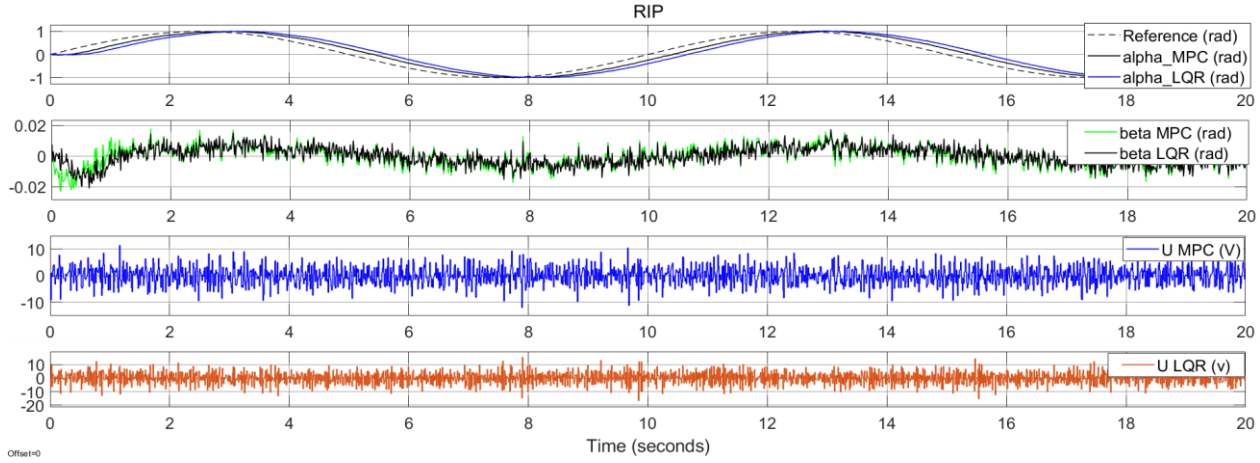

Fig. 7. Output response of RIP tracking a sine wave signal

B. Results in Experiment

The RIP system model is shown in Fig. 8. We set the reference signal for the arm angle of the MPC and LQR controllers from f=0 (rad) to f=1 (rad) at the 20th second. Experimental results for LQR and MPC controller are shown from Fig. 9 to Fig. 11.

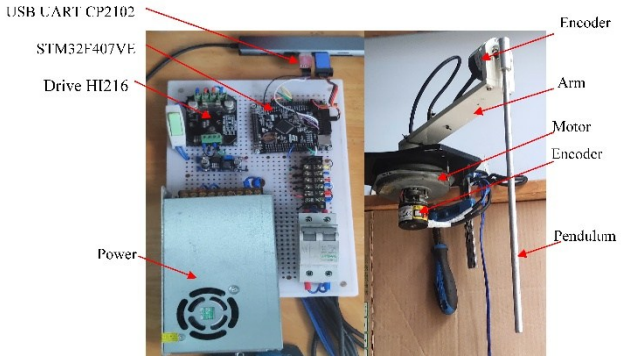

Fig. 8. Hardware platform of RIP

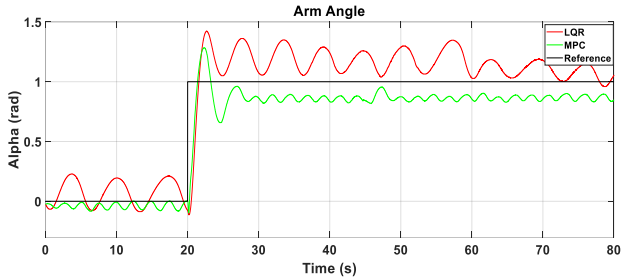

Fig. 9. Arm response under LQR and MPC controllers

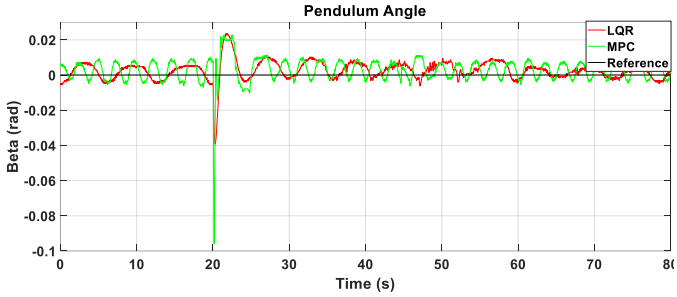

Fig. 10. Pendulum response under LQR and MPC controllers

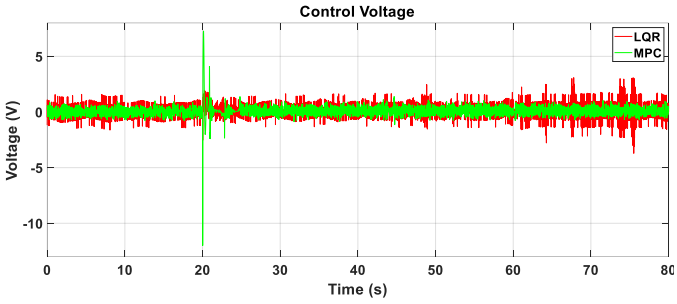

Fig. 11. Control voltage of the LQR and MPC controllers

The performance of two controllers, when the arm angle tracks set-point signal f=1 (rad), is shown in Table 2 and Table 3.

Table 2. Quality of the RIP

	Average pendulum angle (rad)			Average arm angle (rad)	Control voltage (V)	t_{xl} (s) (5%)	σ_{max}(%)
	α_{xl}	e_{xl}	Range	β_{xl}			
LQR	1.15	0.15	[1; 1.2]	0.002	[1.6; -1.5]	38.7	23.5
MPC	0.87	0.13	[0.9; 0.84]	0.002	[0.9; -0.7]	7.5	47.7

Table 3. Quality of RIP according to Root Mean Square Error standard

	Arm angle (rad)	Pendulum angle (rad)	Control voltage (V)
LQR	0.2020	0.8795	0.9950
MPC	0.1560	0.8634	0.9214

⇨ Experimental results show that:

- At steady state, MPC tracks set-point signal at arm angle better than LQR, with an oscillation amplitude that is three times smaller, a settling time that is five times faster, and a control voltage fluctuation that is half as large.
- From RMSE values, we see that the MPC controller performs significantly better in tracking reference signals for arm angle and pendulum angle compared to the LQR controller, although the control voltage signals of the two controllers are nearly identical.
- During a transient period, LQR provides an arm angle response that is twice as small and a control voltage that is four times smaller.

⇨ From experimental results, we observe that the arm and pendulum oscillate along a curve around the reference signal because:

- Dynamic equation described in (3) is for the ideal model. However, experimental model during fabrication inevitably has errors. This leads to a mathematical equation that describes the physical characteristics of the experimental model as being relatively accurate, but not entirely precise.
- Encoder signal wire attached to the pendulum unintentionally creates an additional resistance force, which we can refer to as "Input Disturbance".
- Control system, originally designed as SISO for a SIMO system, makes stabilizing two variables simultaneously challenging.

These three reasons cause the controller to attempt to reduce reference signal error, leading to curve oscillation.

V. CONCLUSIONS

After evaluating the simulation and experimental results of MPC and LQR controllers under different operating conditions on RIP, we find that MPC frequently computes new solutions, whereas LQR uses the same single (optimal) solution for the entire time horizon [17]. For this reason, in terms of control quality, MPC performs well for trajectory tracking and handling system constraints, while LQR provides strong responses to system disturbances, external forces, and unforeseen system changes. Additionally, MPC is characterized by smooth changes in the control signal, whereas LQR produces rapid changes in the control signal, which is a significant drawback due to its substantial impact on actuator wear.

Regarding controller processing, the size of the control matrix for the LQR controller depends only on the number of internal states of the system. In contrast, the MPC controller sets up a control matrix for the entire prediction horizon. Therefore, size of the MPC control matrix not only depends on the number of internal states but also increases proportionally with extension of the prediction horizon and reduction in sampling time [18]. This result in an increasing number of calculations required to generate a control signal,

limiting potential applications of the MPC algorithm and relying on the capabilities of the controller used.

At each step, an MPC controller receives or estimates the current state of the plant. It then calculates a sequence of control actions that minimize cost over the horizon by solving a constrained optimization problem that relies on an internal plant model and depends on the current system state. The controller then applies only the first computed control action to the plant, disregarding subsequent ones. The process repeats in the following time step [13]. Therefore, an accurate mathematical model is necessary, considering the uncertainties [19][20] and disturbance rejection [21].

ACKNOWLEDGMENT

This paper belongs to the project for students in HCMUTE for the year 2025. It is funded by HCMUTE. We, the authors, are grateful for this support. The operation of the system is shown in the link: https://www.youtube.com/watch?v=woHq5wdWdEM

REFERENCES

[1] S. N. Vassilyev, A. Y. Kelina, Y. I. Kudinov, and F. F. Pashchenko, “Intelligent control systems,” *Procedia Computer Science*, vol. 103, pp. 623-628, 2017, https://doi.org/10.1016/j.procs.2017.01.088.

[2] N. V. Đ. Hải and N. V. Thuyên, “PID-neuron controller design for rotary inverted pendulum system”, *JTE*, vol. 7, no. 4, pp. 37–43, 2012, https://jte.edu.vn/index.php/jte/article/view/824.

[3] M. T. Vo, “Back-stepping control for rotary inverted pendulum”, *JTE*, vol. 15, no. 4, pp. 93–101, 2020. https://jte.edu.vn/index.php/jte/article/view/110.

[4] Haiyan Wang and Yu Bai, "Application of fuzzy control in the inverted pendulum," *Proceedings of 2013 2nd International Conference on Measurement, Information and Control*, pp. 1354-1357, 2013, https://doi.org/10.1109/MIC.2013.6758210.

[5] J.-B. Kim, H.-K. Lim, C.-M. Kim, M.-S. Kim, Y.-G. Hong, and Y.-H. Han, "Imitation Reinforcement Learning-Based Remote Rotary Inverted Pendulum Control in OpenFlow Network," *IEEE Access,* vol. 7, pp. 36682-36690, 2019, https://doi.org/10.1109/ACCESS.2019.2905621.

[6] Y.-J. K. Minho Park, Ju-Jang Lee, "Swing-up and LQR stabilization of rotary inverted pendulum," *The Sixteenth International Symposium on Artificial Life and Robotics,* vol. 16, pp. 94–97, 2011, https://doi.org/10.1007/s10015-011-0897-9.

[7] K. R. S. K N Deepak, T Ananthan, "Model Predictive Control for rotary inverted pendulum using LabVIEW," *IOP Publishing Ltd,* 2019, https://doi.org/10.1088/1757-899X/577/1/012113.

[8] D. M. M. Ganga G, "MPC controller for trajectory tracking control of quadcopter," *International Conference on Circuit, Power and Computing Technologies (ICCPCT),* pp. 1-6, 2017, https://doi.org/10.1109/ICCPCT.2017.8074380.

[9] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Engineering Practice,* vol. 11, no. 7, pp. 733-764, 2003, https://doi.org/10.1016/s0967-0661(02)00186-7.

[10] J. B. Rawlings, "Tutorial: model predictive control technology," *Proceedings of the 1999 American Control Conference (Cat. No. 99CH36251)*, pp. 662-676 vol.1, 1999, https://doi.org/10.1109/ACC.1999.782911.

[11] G. V. Troshina, A. A. Voevoda and K. M. Bobobekov, "The parameters determination of the inverted pendulum model in the automatic control system," *2016 13th International Scientific-Technical Conference on Actual Problems of Electronics Instrument Engineering (APEIE)*, pp. 180-182, 2016, https://doi.org/10.1109/APEIE.2016.7807049.

[12] N.-C. Tran, "Double Rotary Inverted Pendulum LQR," *Journal of Fuzzy Systems and Control,* vol. 2, pp. 104–10, 2024, https://doi.org/10.59247/jfsc.v2i2.212.

[13] A. Bemporad, "Model Predictive Control Design: New Trends and Tools," *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 6678-6683, 2006, https://doi.org/10.1109/CDC.2006.377490.

[14] T.-D. Chu and C.-K. Chen, "Design and Implementation of Model Predictive Control for a Gyroscopic Inverted Pendulum," *Applied Sciences,* vol. 7, no. 12, 2017, https://doi.org/10.3390/app7121272.

[15] C. Gambella, B. Ghaddar, and J. Naoum-Sawaya, “Optimization problems for machine learning: A survey,” *European Journal of Operational Research*, vol. 290, no. 3, pp. 807-828, 2021, https://doi.org/10.1016/j.ejor.2020.08.045.

[16] C. Schmid and L. T. Biegler, "Quadratic programming methods for reduced hessian SQP," *Computers & chemical engineering,* vol. 18, no. 9, pp. 817-832, 1994, https://doi.org/10.1016/0098-1354(94)E0001-4.

[17] L. Wang. *Model predictive control system design and implementation using MATLAB*, vol. 3. London: springer. 2009. https://link.springer.com/book/10.1007/978-1-84882-331-0.

[18] A. Jezierski, J. Mozaryn, and D. Suski, "A Comparison of LQR and MPC Control Algorithms of an Inverted Pendulum," In *Polish Control Conference*, pp. 65-76, 2017, https://doi.org/10.1007/978-3-319-60699-6_8.

[19] M. M. J. Grimm G, Tuna S E, and Teel A R, "Nominally robust model predictive control with state constraints," *IEEE transactions on Automatic control,* vol. 52, pp. 1856-1870, 2007, https://doi.org/10.1109/TAC.2007.906187.

[20] S. M. M. a. R. S. V. Mayne D Q, "Robust model predictive contol of constrained linear systems with bounded disturbances," *Automatica,* vol. 41, pp. 219-224, 2005, https://doi.org/10.1016/j.automatica.2004.08.019.

[21] Z. F. a. G. Y. P, "A simplified predictive control algorithm for disturbance rejection," *ISA transactions,* vol. 44, pp. 187-198, https://doi.org/10.1016/S0019-0578(07)60177-3.