Bulletin of Electrical Engineering and Informatics
Vol. 14, No. 2, April 2025, pp. 931~939
ISSN: 2302-9285, DOI: 10.11591/eei.v14i2.8689

Design and implementation of linear quadratic regulator control for two-wheeled self-balancing robot

Leonardus Gilang Buana Putra, Faisal Wahab, Tua Agustinus Tamba
Department of Electrical Engineering, Parahyangan Catholic University, Bandung, Indonesia

Article Info

Article history:

Received Jul 16, 2024
Revised Nov 15, 2024
Accepted Nov 19, 2024

Keywords:

Inverted pendulum
Linear quadratic regulator
Mathematical model
Self-balancing robot
Simulation

ABSTRACT

This research aimed to develop a control system for a self-balancing robot (SBR) based on the mathematical model of an inverted pendulum on a two-wheeled cart. The linear quadratic regulator (LQR) control was implemented to maintain the SBR's balance under normal conditions. A linearization approach was used to convert the dynamic model into a linear form, enabling the application of LQR. Testing was conducted through simulations and a physical SBR prototype equipped with an MPU6050 sensor and NEMA 17 motor. The test results demonstrated the effectiveness of the LQR control in maintaining the SBR's balance and its responsiveness to disturbances. Although there are differences between the simulations and physical implementation, the system successfully maintained the SBR's balance. In conclusion, the use of the inverted pendulum mathematical model and the implementation of LQR control successfully produced a stable and effective control system for SBR balance. By testing various values of the LQR parameters, optimal robot control parameters can be obtained.

Corresponding Author:

Faisal Wahab
Department of Electrical Engineering, Parahyangan Catholic University
St. Ciumbuleuit no 94, Bandung 40141, Indonesia
Email: faisal.wahab@unpar.ac.id

1. INTRODUCTION

In recent years, developments of robots have grown exponentially, where robots can operate autonomously, semi-autonomously, or full autonomusly. In autonomous systems, many advancements have been made in robots with various functions. Autonomous robot performance relies on sensors for guidance in every movement, allowing the robot to operate without human control. One example of autonomous robots is the self-balancing robot (SBR). The concept of the SBR is similar to inverted pendulum technology, where the wheel's acceleration is based on the angle of pivot point. The SBR uses only two-wheels, and robot must maintain an upright position at 90 degrees. SBR implementations include personal transportation robot [1], humanoid robot posture-balance control [2], [3], and line follower robot [4].

Research on SBR is divided into several areas, namely system modeling, angle control, and the electronic components used. In system modelling, the SBR's system of mathematical equations is designed in the form of a state space [5], [6]. In the field of robotics, electronic components play a crucial role in implementing control systems, as they determine the accuracy and precision of experiments. For the electronics in an SBR, a NEMA 17 stepper motor has been used as the actuator [7], an Arduino microcontroller [8], a myrio device [9], and sensors like gyroscopes and accelerometer [10], [11]. Additionally, the robotic operation system (ROS) has been utilized for simulation purposes [12]. For control methods, several approachers have been implemented in SBRs, including fuzzy logic control (FLC)

[13], [14], proportional integral derivative (PID) control [15], [16], a combination of PID and FLC [17]-[20], pole-placement [21], machine learning [22], and linear quadratic regulator (LQR) [23], [24]. The advantage of LQR control is that it can handle multi input multi output (MIMO) and complex system, on the contrary PID control only single input single output (SISO) for simple system. An SBR can be classified as a MIMO system, where multiple inputs (such as sensor data from accelerometers, gyroscopes, and possibly other sensors) are used to control multiple outputs (such as motor speeds or positions) to maintain balance and stability [25]. Research on the implementation of LQR control in SBRs is typically conducted through simulations, using software such as MATLAB [26] or LABVIEW [27]. However, only a few studies have conducted direct physical implementations.

Compared to other research, the primary objective of this study is to design and implement an effective and stable control system for an SBR, using the mathematical model of an inverted pendulum on a two-wheeled cart and using the LQR method for control. A NEMA 17 stepper motor is used as the actuator, an Arduino Uno as the microcontroller, and an inertial measurement unit (IMU) MPU6050 which includes both a gyroscope and an accelerometer, serves as the sensor. To address the inherent noise in the gyroscope readings, a Kalman filter is implemented to reduce this noise. The Kalman filter is an algorithm that estimates the internal state of a dynamic system based on noisy measurements. However, this paper does not discuss the Kalman filter in detail, focusing instead on the application of the LQR. The advantage to use a stepper motor for the SBR, rather than a DC motor, is based on its superior accuracy. DC motors require an additional encoder to obtain precise velocity information, whereas stepper motors provide greater accuracy due to their precise control of movement through step pulses.

This paper is organized as follows: the first section introduces the research aim of the SBR. Section 2 outlines the dynamic model of the inverted pendulum and An explanation of the LQR controller. Section 3 presents the proposed system, including hardware implementation of the SBR. Section 4 discusses the open-loop and closed-loop simulations, as well as the experimental results. Finally, section 5 addresses the advantages and limitations of the proposed controller and presents the conclusions.

2. METHOD

2.1. Inverted pendulum dynamic model

The SBR is an example of an inverted pendulum system. The inverted pendulum model with wheels under the cart, as shown in Figure 1, serves as the basis for deriving the mathematical equations for the SBR. In this study, the inverted pendulum under consideration consists of mechanical components and subsystems to control inputs to the system. This SBR has two degrees of freedom: the horizontal motion of the cart and the rotation of the pendulum rod. In the schematic in Figure 1, the SBR needs to balance itself in an upright position and maintain its position above. Here, m_p represents the mass of the pendulum rod, F_{in} is the external input force, f_x is the friction force, L is half the length of the pendulum rod, θ is the angle between the vertical axis and the pendulum rod, and g is the acceleration due to Earth's gravity.

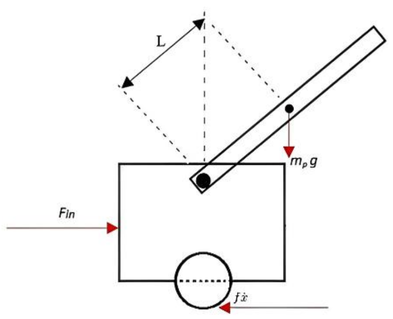


Figure 1. Inverted pendulum system

The equations that can be derived for the cart in Figure 1 along the X-axis using Newton's second law involve both linear and rotational forces.

$$F_{in} = f\dot{x} + N_x + m_c\ddot{x} \tag{1}$$

$$-N_y L \sin\theta - N_x L \cos\theta = I_p\ddot{\theta} \tag{2}$$

F_{in} is the external input force, $f\dot{x}$ represents the frictional force, N_x denotes the normal force at the joint between the cart and the pendulum rod along the x-axis or horizontal direction, m_c is the mass of the cart, x is the linear acceleration, N_y is the force acting in the y-axis or vertical direction, L is the length of the pendulum rod, θ is the angular displacement, and $\dot{\theta}$ represents the angular acceleration.
By combining the two (1) and (2), we derive the mathematical model for $\ddot{x}$ (3) and $\ddot{\theta}$ (4) as:

$$\ddot{x} = \left(\frac{(I_p+m_pL^2)}{(I_p+m_pL^2)(m_c+m_p)-m_p{}^2L^2cos^2\theta}\right)\left(F_{in} - f\dot{x}\{-\frac{m_p{}^2L^2g\sin\theta cos\theta}{I_p+m_pL^2} + m_pL\dot{\theta}^2\sin\theta\right) \tag{3}$$

$$\ddot{\theta} = \left[\frac{1}{(I_p+m_pL^2)(m_c+m_p)-m_p{}^2L^2\cos^2\theta}\right]\left[-m_pL\cos\theta F_{in} + m_pL\cos\theta f\dot{x} - m_p{}^2L^2\sin\theta\cos\theta\dot{\theta}^2 - m_pgL\sin\theta\left(m_c + m_p\right)\right] \tag{4}$$

The following state-space representation (5) is obtained [5]:

$$\begin{bmatrix}\dot{\theta}\\ \ddot{\theta}\\ \dot{x}\\ \ddot{x}\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0\\ 0 & -\frac{(I_p+m_pL^2)f}{(I_p+m_pL^2)(m_c+m_p)-m_p{}^2L^2} & \frac{m_p{}^2gL^2}{(I_p+m_pL^2)(m_c+m_p)-m_p{}^2L^2} & 0\\ 0 & 0 & 0 & 1\\ 0 & \frac{m_pLf}{(I_p+m_pL^2)(m_c+m_p)-m_p{}^2L^2} & -\frac{(m_pgL)(m_c+m_p)}{(I_p+m_pL^2)(m_c+m_p)-m_p{}^2L^2} & 0\end{bmatrix}\begin{bmatrix}\theta\\ \dot{\theta}\\ x\\ \dot{x}\end{bmatrix}$$
$$+\begin{bmatrix}0\\ \frac{I_p+m_pL^2}{(I_p+m_pL^2)(m_c+m_p)-m_p{}^2L^2}\\ 0\\ -\frac{m_pL}{(I_p+m_pL^2)(m_c+m_p)-m_p{}^2L^2}\end{bmatrix}F_{in}(t) \tag{5}$$
$$\begin{bmatrix}\theta\\ x\end{bmatrix} = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}\theta\\ \dot{\theta}\\ x\\ \dot{x}\end{bmatrix} + \begin{bmatrix}0\\ 0\end{bmatrix}u$$

2.2. The linear quadratic regulator controller

The LQR method can be used to design optimal controllers for a linear system. The LQR control signal is essentially obtained by minimizing a specific system performance index. In the LQR method, the performance index is characterized by two matrices, Q and R. Matrices Q and R are defined as symmetric positive definite matrices. Matrix Q serves for compensating the state variables of the system, while matrix R serves for compensating the control signals of the system. Review of the general model of a linear system (6) and (7):

$$\dot{x} = \mathrm{A}x + \mathrm{B}u \tag{6}$$

$$y = \mathrm{C}x \tag{7}$$

where A is the system state matrix, B is the system input matrix, y is the system output vector, C is the system output matrix, and u is the control signal vector of the system. In the LQR method, the control signal u is determined as the signal that minimizes the following quadratic performance index (8):

$$J = \int_x^\infty (q^TQq + u^TRu)\,dt \tag{8}$$

To obtain an optimal controller, one must solve the Algebraic Riccati (9):

$$A^TP + PA + PBR^{-1}B^TP + Q = 0 \tag{9}$$

where P is a symmetric positive definite matrix known as the LQR state matrix. Once the matrix P is found, the control signal or LQR controller can be computed as (10):

$$u = -Kx = -(R^{-1}B^TP)x \tag{10}$$

The LQR function in MATLAB can be used to obtain the value of K for the LQR control signal. The values of Q and R can be chosen using the Bryson rule, with Q and R being diagonal matrices with:

$$Q_{(k,k)} = \frac{1}{(q_k^2)_{max}} \quad (11)$$

$$R = \frac{1}{(u_{max}^2)} \quad (12)$$

where $u_{\max}$ is the maximum allowable control signal given to the system, and $(qk)_{\max}$ is the maximum constraint of the system's state variables. Bryson's rule is commonly used for determining the initial values of matrices Q and R in trial-and-error iterations with the aim of obtaining the desired response in a closed-loop system. The Bryson rule helps in setting the initial values for Q and R by considering the desired system behavior, control effort, and constraints. It's a heuristic method used to tune the LQR controller's performance. Adjust those values based on the specific requirements of the system and the desired closed-loop behavior.

3. IMPLEMENTATION

In Figure 2, the design of the SBR is complemented by dimensions presented using an isometric view. The mechanical part of the SBR consists of two levels, The first level accommodates electrical components such as a battery holder and three rechargeable 18650 batteries. The second holds four electronic components, namely Arduino Uno R3 microcontroller, two DRV8825 stepper motor drivers, and an IMU sensor MPU-6050. The designed SBR chassis has dimensions of 180×70×3 mm.

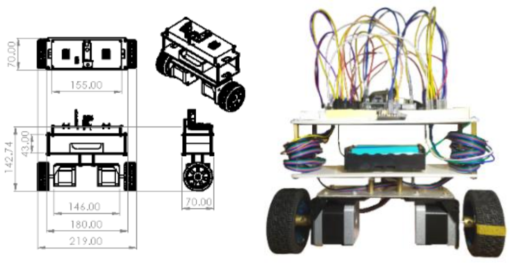


Figure 2. SBR design model diagram

Figure 3 shows the SBR electrical circuit schematic according to the SBR electrical component specifications. This circuit includes the Arduino Uno, which has an input from the IMU sensor to measure the SBR's angular changes and angular acceleration. Two stepper motors can be controlled through a single output. The stepper motors are connected to DRV8825 stepper motor drivers via four pins, including STEP and DIR pins that are connected to the Arduino Uno. The Arduino Uno is powered through the USB port connected to a computer, while the two DRV8825 stepper drivers are powered by three 18650 batteries arranged in series, providing a 12 V voltage. Because the stepper motor input is a pulse signal, then to adjust the speed of the motor is to change the frequency on the PWM output signal on the Arduino.

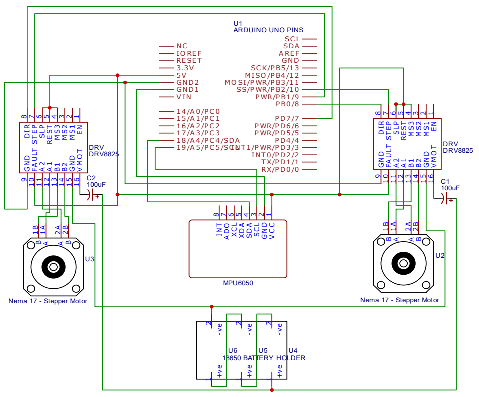


Figure 3. SBR electrical circuit design

The IMU sensor uses the I2C communication protocol connected to the Arduino Uno via pins A4 and A5. The MPU-6050, an inertial sensor, combines an accelerometer and a gyroscope in a single package. Each sensor has its strengths and weaknesses in measuring angles and orientation, but by combining their data, better results can be achieved than by using either sensor alone. The accelerometer and gyroscope data are implemented using a Kalman filter, which leverages the strengths of both sensors while compensating for their individual limitations. The step output results are generated from the control signal derived from (10). This results in a pulse frequency that matches the motor speed measured in steps per second. Based on this data, it can be concluded that the SBR maintains balance by moving forward and backward with minimal pulse frequency adjustments.

4. RESULTS AND DISCUSSION

4.1. Simulation of the open-loop control system

The state-space equations obtained from the mathematical model are simulated in MATLAB using real-world parameters as shown in Table 1. The simulation is carried out by setting the initial conditions to zero: the cart is at position zero, the pendulum angle is zero, and both linear and angular velocities are also set to zero. The simulation models the robot in an upright position on a flat surface. The simulation results are then plotted as shown in Figure 4.

Table 1. SBR parameter table

Parameter	Variable	Value	Parameter
Mass cart	m_c	1.078	Kg
Mass pendulum	m_p	0.169	Kg
Inertia SBR	I	0.0017	Kg/m^2
Half pendulum	L	0.04	m

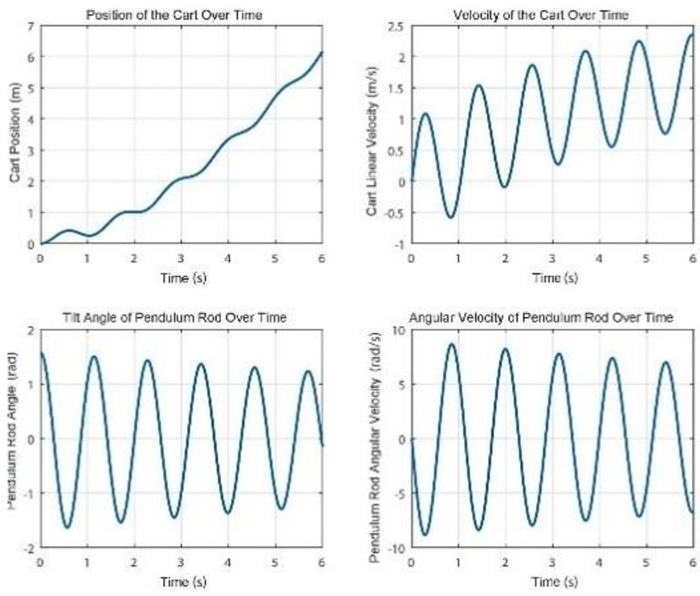


Figure 4. Open-loop system simulation results

In this simulation, an external input force of 1 N is applied to the cart. The position of the cart continues to increase over time, with the positive x-axis indicating the cart moving to the right as depicted in the schematic of Figure 4. The cart's velocity is shown in the graph, initially exhibiting oscillations after the input force is applied. This is due to the influence of the pendulum rod continuously oscillating. Initially, the angle of the pendulum rod is 1.57 radians or 90 degrees. The graph of the cart's velocity displays ongoing oscillations, indicating that the pendulum rod is continuously rotating. These results demonstrate that an open-loop system generates unstable behavior, as the oscillations do not subside within the system.

4.2. Simulation of the closed-loop control system

Before simulating the closed-loop system model, the open-loop system model is first tested to determine whether the system is controllable and observable. This is done by calculating the controllability and observability matrices, with the assistance of MATLAB. The controllability and observability matrices are obtained using:

$$C_k = \begin{bmatrix} 0 & 0.6174 & 0.1906 & -3.9737 \\ 0.6174 & 0.1906 & -3.9737 & 0.0182 \\ 0 & -1.4292 & 0.4412 & 36.6675 \\ -1.4292 & 0.4412 & 36.6675 & -14.1168 \end{bmatrix} \quad (13)$$

$$O_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0.3807 & 2.8215 & 0 \\ 0 & 0.0953 & 0.871 & 2.8215 \end{bmatrix} \quad (14)$$

The controllability and observability matrices have a full ranks, which are four. The calculated determinants using MATLAB are 287.6 for the controllability matrix and 7.96 for the observability matrix. From this, it can be concluded that the system is controllable.

Next, the design of the LQR controller is carried out for the system in closed-loop mode. The initial step in designing an LQR controller is to determine the matrices Q and R. Matrix Q plays a role in compensating for the system's state variables, while R compensates for the control signal of the system. The selection of Q and R values can follow the Bryson rule, typically using diagonal matrices as the initial values for Q and R. These values are then adjusted through trial and error to achieve the desired response from the closed-loop system. Matrices Q and R reflect the weights of the system's states and inputs.

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, Q_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 50 & 0 \\ 0 & 0 & 0 & 50 \end{bmatrix}, Q_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix}, Q_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 200 & 0 \\ 0 & 0 & 0 & 200 \end{bmatrix}$$

$$R = 1$$

Initially, the matrix Q is taken as a diagonal identity matrix, and the MATLAB command “LQR” is used to obtain the value of K=[3.1623, 2.0655, -10.9006, -2.5584], resulting in the system response as follows in Figure 5. It is observed that the closed-loop system is stable compared to the open-loop system. Even though the closed-loop system response is stable with the obtained gain values of K, it is not guaranteed that these are optimal values. To find the optimal gain values modify the components of matrices Q and R adjust the values of Q_{33} and Q_{44} to 50, 100, and 200, where Q_{33} reflects θ and Q_{44} reflects $\dot{\theta}$.

The MATLAB command “LQR” was used to obtain the values; K1=[22.3607, 26.8683, -95.2838, -15.4266], K2=[31.6228 38.6350 -69.4515 6.6786], and K3=[44.7214, 54.3694, -97.8304, 10.4938]. Then the closed-loop system response is simulated as shown in Figure 6. In Figure 5, different variations of gain K1 result in different responses. It is observed that values K2 and K3 lead to a faster settling time of about 0.8 seconds compared to K, and also result in smaller angular pendulum shifts. This comparison concludes that increasing the values of matrix Q allows the robot to move more stably and remain upright.

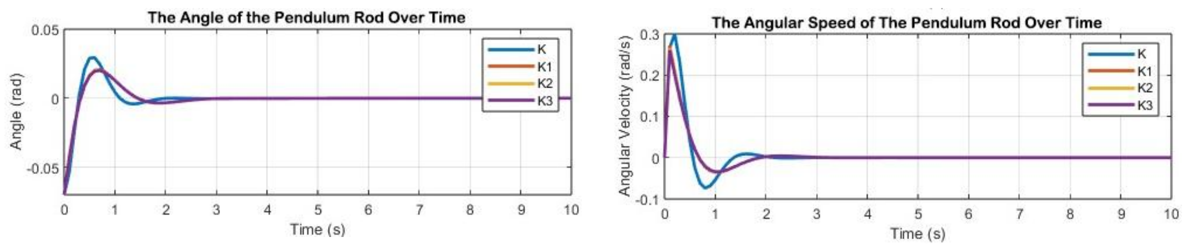


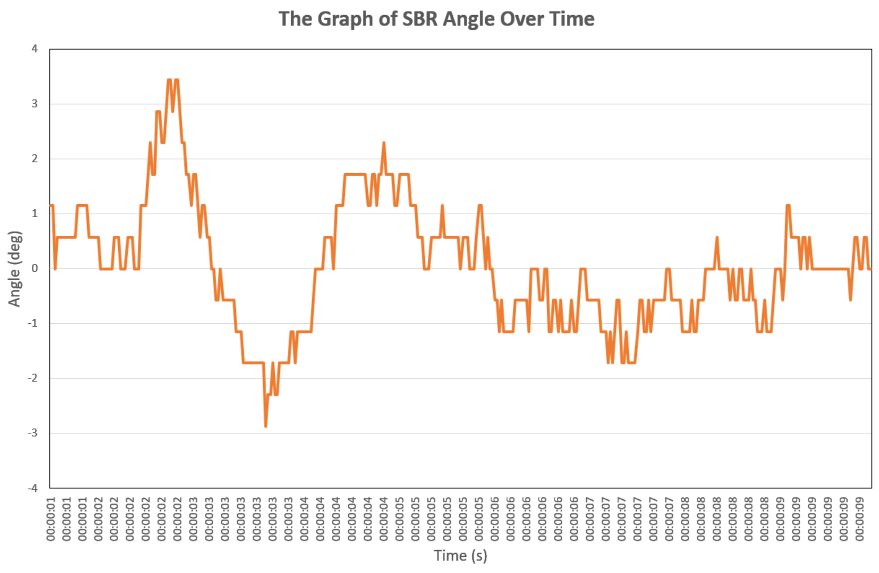
Figure 5. Comparison of closed-loop system simulation results

4.3. Experimental results

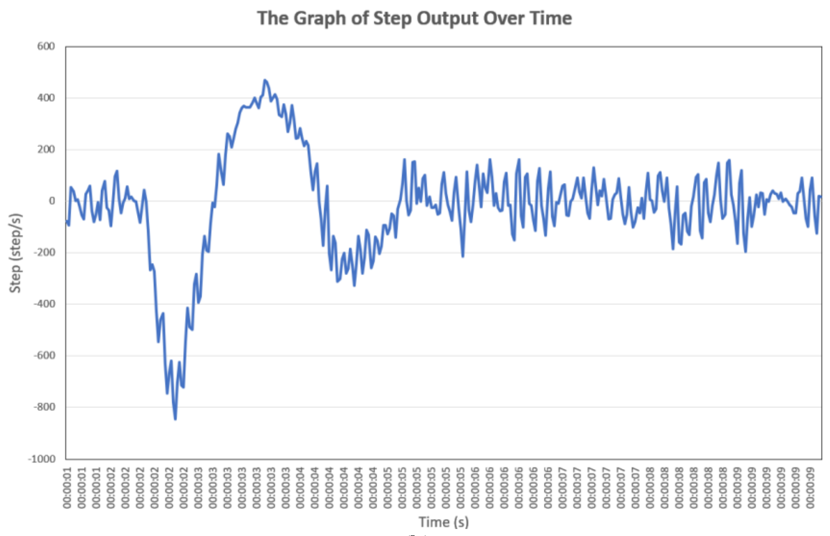
In experimental, the designed robot has been successfully upright and stable condition. To prove that the control that has been designed can work well, the robot is given external interference. The graph in Figure 6(a) displays the SBR's tilt angle before and during an external disturbance.

Initially, the SBR is in a stable balanced position with a low or nearly zero tilt angle. However, when an external disturbance is applied, there is a sharp increase in the SBR's tilt angle. This indicates that the SBR responds to the disturbance by increasing its tilt angle as an effort to regain balance.

The graph in Figure 6(b) shows the number of steps to control the stepper motor when an external disturbance is applied to the SBR. This response includes motor movement controlled by the control system to bring the SBR back to a balanced position. It can be observed that after detecting the external disturbance, the SBR provides a quick and effective response by moving the motor adaptively. The comparison between actual data obtained from the self-balancing SBR and simulation data in Figures 5 and 6 reveals the effectiveness and accuracy of the simulation model. Despite the differences in angle values between the two datasets, the SBR still maintains its self-balance in both conditions, as evident from the data in Figure 7.



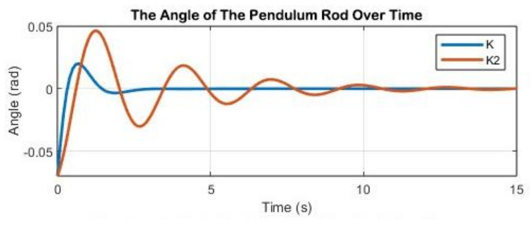
(a)



(b)

Figure 6. The graph of; (a) SBR angle and (b) step output with external disturbance

However, there is a significant difference between the optimized gain value K obtained in the MATLAB simulation and the K1 gain value required in the physical implementation using Arduino. This difference may stem from mechanical, electrical factors, environmental disturbances, and hardware limitations. In this case, a more in-depth analysis of the mechanical and electrical model in the physical implementation could help identify the factors influencing the difference in K gain values and optimize them to achieve the desired balance in the self-balancing SBR.



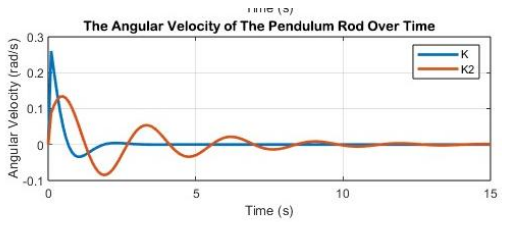


Figure 7. Graph of SBR angle and angular velocity in simulation

5. CONCLUSION

An LQR-based control system has been designed and implemented to optimally maintain the SBR's balance formulated in mathematical model of the inverted pendulum system on a two-wheeled cart derived through linearization and converted into a state-space representation. The use of LQR resulted in stable and optimal responses under normal operating conditions. By fine-tuning the gain values (K) in the LQR control, the robot effectively maintained its balance and responded accurately to setpoints. The LQR control system demonstrated impressive performance in both simulation and physical prototype testing. Simulations showed the robot's ability to maintain balance under various conditions, while physical testing confirmed the robot's capability to balance itself and respond accurately to external disturbances. In the future, SBR development efforts can focus on determining LQR parameters, comparing with other controlling performance, explore their abilities in various environments and situations, in addition, using electronic components that have good accuracy and specifications for the best performance.

REFERENCES

[1] J. Zhang, G. Li, F. Liu, and Y. Liu, "Design of a two-wheeled self-balance personal transportation robot," in *2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA)*, 2016, pp. 225–228, doi: 10.1109/ICIEA.2016.7603583.

[2] H.-I. Lin and X.-A. Nguyen, "Humanoid robot posture-balance control," in *2016 55th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, 2016, pp. 160–165, doi: 10.1109/SICE.2016.7749249.

[3] S. L. Podvalny and E. M. Vasiljev, "Modeling of Human-Robot Physical Interaction for Case of Mobile Self-Balanced Robot," in *2019 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*, 2019, pp. 1–5, doi: 10.1109/ICIEAM.2019.8742942.

[4] C. -F. Hsu, C. -T. Su, W. -F. Kao, and B. -K. Lee, "Vision-Based Line-Following Control of a Two-Wheel Self-Balancing Robot," *2018 International Conference on Machine Learning and Cybernetics (ICMLC)*, Chengdu, China, 2018, pp. 319-324, doi: 10.1109/ICMLC.2018.8526952."

[5] O. Jamil, M. Jamil, Y. Ayaz, and K. Ahmad, "Modeling, control of a two-wheeled self-balancing robot," in *2014 International Conference on Robotics and Emerging Allied Technologies in Engineering (iCREATE)*, 2014, pp. 191–199, doi: 10.1109/iCREATE.2014.6828364.

[6] H. Bin, L. W. Zhen, and L. H. Feng, "The kinematics model of a two-wheeled self-balancing autonomous mobile robot and its simulation," in *2010 2nd International Conference on Computer Engineering and Applications, ICCEA 2010*, 2010, pp. 64–68, doi: 10.1109/ICCEA.2010.169.

[7] R. Zhang, G. Xiong, C. Cheng, X. Shang, Y. Ma, and Z. Lu, "Control system design for two-wheel self-balanced robot based on the stepper motor," in *Proceedings of 2013 IEEE International Conference on Service Operations and Logistics, and Informatics*, 2013, pp. 241–244, doi: 10.1109/SOLI.2013.6611417.

[8] A. S. Shekhawat and Y. Rohilla, "Design and Control of Two-wheeled Self-Balancing Robot using Arduino," in *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, 2020, pp. 1025–1030, doi: 10.1109/ICOSEC49089.2020.9215421.

[9] S. Sarathy, M. M. M. Hibah, S. Anusooya, and S. Kalaivani, "Implementation of Efficient Self Balancing Robot," in *2018 International Conference on Recent Trends in Electrical, Control and Communication (RTECC)*, 2018, pp. 65–70, doi: 10.1109/RTECC.2018.8625624.

[10] V. P. Kumar, K. N., and S. R., "Controlling a Two-Wheeled Self-balancing Robot (TWSBR) with Gyroscope" *Grenze International Journal of Engineering and Technology GIJET*, pp. 1-5, 2023.

[11] F. Dai, F. Li, Y. Bai, W. Guo, C. Zong, and X. Gao, "Development of a coaxial self-balancing robot based on sliding mode control," in *2012 IEEE International Conference on Mechatronics and Automation*, 2012, pp. 1241–1246, doi: 10.1109/ICMA.2012.6283528.

[12] G. Song, L. Sun, and X. Yang, "Design and Implementation of Self-balancing and Navigation Robot Based on ROS System," in *2019 Chinese Control And Decision Conference (CCDC)*, 2019, pp. 5597–5602, doi: 10.1109/CCDC.2019.8832808.

[13] M. Rahmawaty, "Modeling, Simulation, and Stabilization of A Two-Wheeled Inverted Pendulum Robot Using Hybrid Fuzzy Control," *Indonesian Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 3, no. 3, pp. 93–98, Jun. 2021, doi: 10.35882/ijeeemi.v3i3.2.

[14] N. Yusuke, "2017 Joint 17th World Congress of International Fuzzy Systems Association and 9th International Conference on Soft Computing and Intelligent Systems (IFSA SCIS)," *Holding Report. Intelligence and information*, vol. 29, no. 5, pp. 173-174, 2017, doi: 10.3156/jsoft.29.5_173.

[15] J. Velagić, I. Kovač, A. Panjević, and A. Osmanović, "Design and Control of Two-Wheeled and Self-Balancing Mobile Robot," in *2021 International Symposium ELMAR*, 2021, pp. 77–82, doi: 10.1109/ELMAR52657.2021.9550938.

[16] T. Nikita and K. T. Prajwal, “PID Controller Based Two Wheeled Self Balancing Robot,” in *Proceedings of the 5th International Conference on Trends in Electronics and Informatics, ICOEI 2021, Institute of Electrical and Electronics Engineers Inc*., Jun. 2021, pp. 1–4, doi: 10.1109/ICOEI51242.2021.9453091.
[17] R. Sadeghian and M. T. Masoule, “An experimental study on the PID and Fuzzy-PID controllers on a designed two-wheeled self-balancing autonomous robot,” in *2016 4th International Conference on Control, Instrumentation, and Automation (ICCIA)*, 2016, pp. 313–318, doi: 10.1109/ICCIAutom.2016.7483180.
[18] X. Su, C. Wang, W. Su, and Y. Ding, “Control of balancing mobile robot on a ball with fuzzy self-adjusting PID,” in *2016 Chinese Control and Decision Conference (CCDC)*, 2016, pp. 5258–5262, doi: 10.1109/CCDC.2016.7531938.
[19] J. Zhang, T. Zhao, B. Guo, and S. Dian, “Fuzzy fractional-order PID control for two-wheeled self-balancing robots on inclined road surface,” *Systems Science and Control Engineering*, vol. 10, no. 1, pp. 289–299, 2022, doi: 10.1080/21642583.2021.2001768.
[20] Q. Yong, L. Yanlong, Z. Xizhe, and L. Ji, “Balance control of two-wheeled self-balancing mobile robot based on TS fuzzy model,” in *Proceedings of 2011 6th International Forum on Strategic Technology*, 2011, pp. 406–409, doi: 10.1109/IFOST.2011.6021051.
[21] O. Aburas, H. Ahmed, L. El-Khoms, and Libya, “A Guide to Implement a Two Wheeled Robot Using Pole-Placement on Arduino,” *The International Journal of Engineering and Information Technology (IJEIT)*, vol. 6, no. 2, p. 183, 2020.
[22] S. R. Garces, M. Y. Beb, A. Boubakari, H. H. Ammar, and M. A. W. Shalaby, “Hybrid Self-Balancing and object Tracking Robot Using Artificial Intelligence and Machine Vision,” in *2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES)*, 2020, pp. 264–269, doi: 10.1109/NILES50944.2020.9257916.
[23] Jianwei and Xiaogang, “The LQR control and design of dual-wheel upright self-balance Robot,” in *2008 7th World Congress on Intelligent Control and Automation*, 2008, pp. 4864–4869, doi: 10.1109/WCICA.2008.4593712.
[24] I. Mohammed and A. I. Abdullah, “Balancing a Segway robot using LQR controller based on genetic and bacteria foraging optimization algorithms,” *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 18, no. 5, pp. 2642-2653, Jun. 2020, doi: 10.12928/telkomnika.v18i5.14717.
[25] J. Qiu *et al*., “Two-wheeled self-balancing robot modeling and nonlinear control,” in *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, 2017, pp. 734–739, doi: 10.1109/URAI.2017.7992813.
[26] S. Wenxia and C. Wei, “Simulation and debugging of LQR control for two-wheeled self-balanced robot,” in *2017 Chinese Automation Congress (CAC)*, 2017, pp. 2391–2395, doi: 10.1109/CAC.2017.8243176.
[27] C. Xu, M. Li, and F. Pan, “The system design and LQR control of a two-wheels self-balancing mobile robot,” in *2011 International Conference on Electrical and Control Engineering*, 2011, pp. 2786–2789, doi: 10.1109/ICECENG.2011.6057680.

BIOGRAPHIES OF AUTHORS

Leonardus Gilang Buana Putra received the bachelor’s degree in Mechatronics Engineering from Parahyangan Catholic University. He is currently working as a Staff Production Engineer. His research interests include robotics, control systems, and dynamic stabilization. He can be contacted at email: leonardusgilangbp@gmail.com.

Faisal Wahab is an assistant professor in the Department of Electrical Engineering at Parahyangan Catholic University, Indonesia. He received the B.Eng. degree in Electrical Engineering Education from Indonesia University of Education (Indonesia) in 2011 and the master’s degrees in School of Electrical Engineering and Informatics at Bandung Institute of Technology (Indonesia) in 2015. His research interests include robotics, modelling system, electronics, intelligent control system, and mechatronics. He can be contacted at email: faisal.wahab@unpar.ac.id.

Tua Agustinus Tamba is an associate professor in the Department of Electrical Engineering at Parahyangan Catholic University, Indonesia. He received both MSEE and Ph.D. in Electrical Engineering from University of Notre Dame (USA) in 2016, M.Sc. in Mechanical Engineering from Pusan National University (Republic of Korea) in 2009, and BEng in Engineering Physics from Institut Teknologi Bandung (Indonesia) in 2006. His main research interests include dynamical systems, control theory, and optimization with applications in mechatronics, robotics, automation systems, and systems biology. He can be contacted at email: ttamba@unpar.ac.id.