Received: 26 May 2024 | Revised: 31 August 2024 | Accepted: 18 September 2024

DOI: 10.1002/msd2.12133

RESEARCH ARTICLE

IJMSD International Journal of Mechanical System Dynamics Open Access WILEY

Analysis of the parametric configuration impact on BallBot control performance

Anh-Duc Pham | Ba Hoa Thai | Phuoc Vinh Dang | Nhu Thanh Vo

Faculty of Mechanical Engineering, The University of Danang – University of Science and Technology, Danang, Vietnam

Correspondence
Dr. Nhu Thanh Vo, Faculty of Mechanical Engineering, The University of Danang – University of Science and Technology, Danang, Vietnam.
Email: vnthanh@dut.udn.vn

Abstract

The BallBot, a versatile robot system, finds applications in various domains of life. It comprises a frame moved by three wheels mounted on a ball. The robot performance is significantly influenced by its parametric configuration, including body mass, chassis size, and ball diameter. This study examines the impact of these configuration parameters on the control of the BallBot. The mathematical model of the BallBot is discussed, considering the assumptions and coordinate systems. To control the robot, a Linear Quadratic Regulator controller is designed. Subsequently, the simulation model is used to assess the effects of changing the initial parametric configuration. It is observed that altering the robot mass has a notable impact on the BallBot response, while changes in the ball diameter have a relatively insignificant effect.

KEYWORDS

BallBot, LQR, BallBot's control performance, structural configuration

1 | INTRODUCTION

Assistant robots are gaining increasing attention from researchers and manufacturers because of rapid advancements in science and technology. These robots are now widely used in various applications, ranging from industrial manipulators to humanoid or bioinspired robots that cater to human needs.[1] Recently, the demand for assistant robots to support daily activities for humans has increased.[2,3] Among such robots, the BallBot stands out as a potential candidate for assisting daily activities, as it can be balanced on a ball and rotate around it.[4]

Despite its advantages over other robots with similar functionality, BallBot possesses highly nonlinear characteristics[5] and presents challenges in control.[4] Unlike conventional robots with two or four wheels that can move in only one direction at a time and require a certain turning radius for navigation,[6,7] the BallBot has all the characteristics of an underactuated system.[8,9] The BallBot can navigate effortlessly in narrow spaces and maintain balance while moving on inclined surfaces owing to its midpoint contact with the floor through the ball that supports it.[5] Three actuators attached between the frame of the robot and the ball can be controlled to move the ball. The fast and sensitive operation of these actuators significantly affects their ability to maintain the balance and stability of the entire BallBot body. However, controlling the BallBot becomes complex since the number of directly controllable inputs is less than the number of degrees of freedom (DOFs) of the robot. Thus, designing and implementing a control strategy for BallBot requires careful consideration.

Research on the BallBot has increased since it was introduced in 2006,[6] and numerous publications have explored its potential. For example, Pham et al.[10] developed a BallBot system capable of carrying people with a total mass of nearly 70 kg. He implemented a combination of Linear Quadratic Regulator (LQR) and Proportional–Integral (PI) controllers to maintain the robot's balance. A feedforward

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.
© 2024 The Author(s). *International Journal of Mechanical System Dynamics* published by John Wiley & Sons Australia, Ltd on behalf of Nanjing University of Science and Technology.

controller was also applied to ensure system stability when the mass changed, or unpredictable disturbances affected the system.[10] In another study, Fankhauser and Gwerder built and controlled a ball-balancing robot with a 20° tilt angle on the vertical axis, achieving a moving speed of up to 2 m/s. The author controlled multiple set points and ensured precise movement along specific trajectories.[11] Lal et al. investigated the structure of the robot base, the part used for motor installation, and the Omni wheel.[12] Using the Omni wheel with two sides increased the likelihood of the robot losing alignment with the ball, leading to system instability compared with the single-sided Omni wheel. Recently, the linearized LQR controllers have been developed for the NXT BallBot model.[13,14] These studies have developed linearized control systems for the NXT BallBot considering the uncertain actuator parameters, which are inertial and friction of the actuator. Essentially, the dynamics model of BallBot has been represented in two-dimensional (2D) systems,[13–15] but this representation ignores the effects of the centrifugal force and applied forces in different planes in the three-dimensional (3D) system. Therefore, a comprehensive 3D dynamics model of the BallBot is necessary to fully understand its performance.[16,17] To control a complex multibody system such as the BallBot,[18] various methods commonly used in advanced robotics including proportional–integral–derivative (PID),[19,20] fuzzy control,[21] neural network models,[22] and sliding control[23] have also been applied for balancing and precision trajectory tracking of the BallBot. Previously, along with passivity-based nonlinear control, the proportional-derivative (PD) feedforward controller was studied to correct the forward motion of the BallBot in the horizontal plane.[20] Moreover, along with the LQR controller, the Kalman filter was utilized due to the high-frequency noise of the inertial measurement unit sensor to estimate the changes in Euler angles and velocities of the BallBot.[24] On the other hand, sliding mode control (SMC), neural network observation, and fuzzy logic control were also used to control the movement of the BallBot precisely. Jang et al.[25] investigated the SMC virtual control input for the BallBot based on neural network observations. Additionally, Nguyen et al.[26] introduced a novel robust controller for the BallBot that used a disturbance observer to reject the effect of an environmental disturbance. The controller was designed using feedback linearization and SMC to ensure precise balancing and trajectory tracking of the BallBot, even in the presence of disturbances. Moreover, hierarchical sliding mode control (HSMC), a special branch of SMC, for a fully nonlinear dynamic ball-balancing robot was also studied by Do et al.[27] Moreover, the adaptive law was employed to compensate for dynamic uncertainties and external disturbances.[27] Another approach using the HSMC method was proposed recently by Vu et al.[28], where a novel HSMC based on an extended state observer was developed to address safety constraints in the body deflection angle of the BallBot. In this research, the path planning algorithm rapidly-exploring random trees (RRT) was adopted to control the system to optimize the trajectory of the BallBot. Furthermore, to increase the robustness of the SMC and improve the transient response, the fuzzy SMC was investigated by Navabi et al.[29] Fuzzy logic control was also adopted for real flatform to investigate the effectiveness of underactuated systems, like, the BallBot.[30,31] Nevertheless, these previous publications lack an evaluation of the impact of changes in robot mass and ball size on BallBot operability.

This study evaluates the influence of parametric configuration, such as the robot mass and ball diameter, on the performance of the BallBot during its control process. First, a detailed mathematical model of the BallBot is presented, based on specific assumptions and selection of coordinate system. The BallBot model is then controlled using an LQR controller. Finally, the impact of changes in the parameter configuration of the BallBot is assessed through numerical simulations. The results show that while changes in ball diameter are insignificant, changes in robot mass significantly influence the BallBot responses. This article makes the following contributions to the field of ball-balancing robot control:

- Mathematical modeling of the 3D dynamics model of BallBot with appropriate parametric configurations using full state variables for LQR control methods.
- Investigation of how parametric configurations affect the control performance of the robot, with notable findings such as a decrease in peak time with an exceedingly high upper body mass of the BallBot.
- Demonstration of the effectiveness of the LQR controller with regard to the system response values and generated moment inputs, demonstrating that the proposed control approach successfully stabilizes and controls the ball-balancing robotic system.
- Validation of the obtained results through simulations of the BallBot confirmed the practical applicability and effectiveness of the LQR control methods despite variations in the parametric configurations.

2 | MATHEMATICAL MODEL

A basic model of the BallBot is shown in Figure 1. The entire body of the robot is placed on a spherical ball, known as the 5-DOFs system. The lower body (main frame) has height l_{B1} and mass m_{B1}, and the upper body (mass) has height l_{B2} and mass m_{B2}. The body of the BallBot with total mass m_B can rotate along three axes (x-, y-, and z-axes) of the 3D coordinate system because the Omni wheel is mounted with three motors at the base. Furthermore, a ball with m_S mass and r_S radius can move in a horizontal plane in the O_x and O_y directions. Thus, to determine all the BallBot configurations, the coordinate vector $\boldsymbol{q} = [x_S\ y_S\ \theta_x\ \theta_y\ \theta_z]$ is used. The movement position of the ball, which is also the position of the BallBot, is determined by two transition variables of the ball in the horizontal plane (x_S, y_S), whereas three variables [θ_x, θ_y, θ_z] are used to define the orientation of the BallBot in the 3D coordinate system.

When the BallBot moves, the change in position and orientation of the BallBot, six coordinate systems (CS_s) are used with the relationships shown in Figure 2. The CS_{or} ($O_{or}x_{or}\ y_{or}z_{or}$),

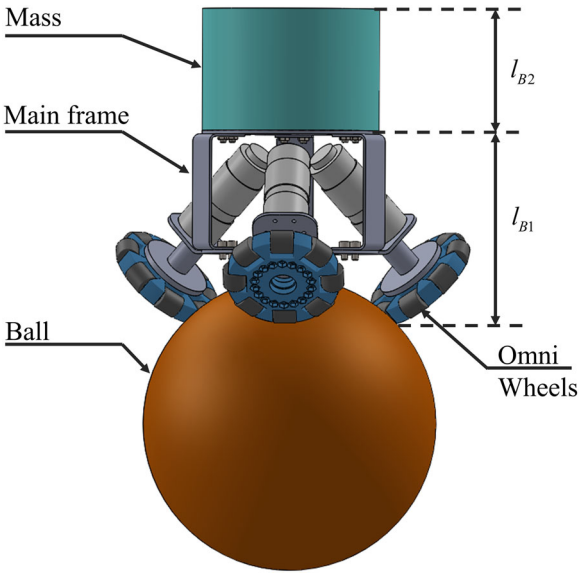

FIGURE 1 Structure of the ball-balancing robot (BallBot).

reference frame is the fixed global frame. The coordinate system CS_1 ($O_1x_1\,y_1z_1$) is determined by a translation of CS_{or} along the x-axis by a distance defined by the x_s-variable. Similarly, by translating the CS_1 coordinate system a distance defined by the y_s-variable along the y-axis of CS_1, the coordinate system CS_2 ($O_2x_2\,y_2z_2$) is determined with an origin placed at the center of the ball. The coordinate systems CS_3 ($O_3x_3\,y_3z_3$), CS_4 ($O_4x_4\,y_4z_4$), and CS_5 ($O_5x_5\,y_5z_5$) are obtained by rotating CS_2 around the z-axis of CS_2 with the θ_z angle, CS_3 around the y-axis of CS_4 with the θ_y angle, and CS_4 around the x-axis of CS_4 with the angle θ_x, respectively. The origin of CS_5 is the mass center of the BallBot body. To determine the relationship between the coordinate systems,[32] transformation matrices are used for the rotation matrix $\boldsymbol{R}_i^j$ (matrix transformation step from CS_j to CS_i, which can be a translation or a rotation), as shown in Table 1. In general, the rotation matrices used to define the ball position $\boldsymbol{R}_{or}^2$ (from CS_2 to CS_{or}) and the configuration of the BallBot $\boldsymbol{R}_{or}^5$ (from CS_5 to CS_{or}) are calculated using Equation (1). The coordinate system CS_2 does not rotate with respect to CS_{or}; thus, $\boldsymbol{R}_{or}^2$ = diag(1, 1, 1), and

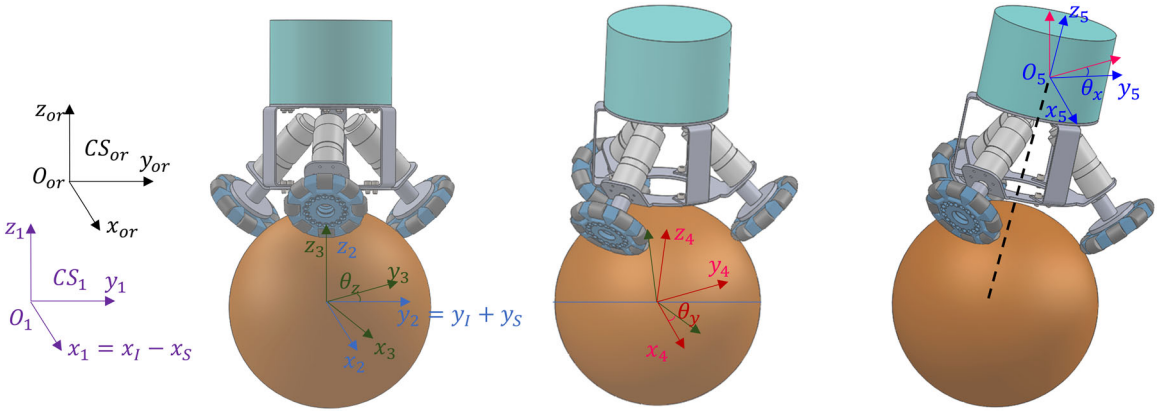

FIGURE 2 Coordinate system of the BallBot.

TABLE 1 Rotation matrices for the coordinate system.

	Equation		
Type	***x*-Axis**	***y*-Axis**	***z*-Axis**
Transition	From CS_1 to CS_{or}: $\boldsymbol{R}_{or}^1$ (transition along $O_{or}x_{or}$ with x_s) $\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}$	From CS_2 to CS_1: $\boldsymbol{R}_1^2$ (transition along $O_1\,y_1$ with y_s) $\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}$	
Rotation	From CS_5 to CS_4: $\boldsymbol{R}_4^5$ (counterclockwise around O_4x_4 amount θ_z) $\begin{bmatrix}\cos(\theta_z)&-\sin(\theta_z)&0\\\sin(\theta_z)&\cos(\theta_z)&0\\0&0&1\end{bmatrix}$	From CS_4 to CS_3: $\boldsymbol{R}_3^4$ (counterclockwise around O_3x_3 amount θ_y) $\begin{bmatrix}\cos(\theta_y)&0&\sin(\theta_y)\\0&1&0\\-\sin(\theta_y)&0&\cos(\theta_y)\end{bmatrix}$	From CS_3 to CS_2: $\boldsymbol{R}_2^3$ (counterclockwise around O_2x_2 amount θ_x) $\begin{bmatrix}1&0&0\\0&\cos(\theta_x)&\sin(\theta_x)\\0&-\sin(\theta_x)&\cos(\theta_x)\end{bmatrix}$

TABLE 2 Position of the origin of CS_j within CS_i.

Position	Value
From CS_1 to CS_{or}	$[x_S \quad 0 \quad 0]^T$
From CS_2 to CS_{or}	$[0 \quad -y_S \quad 0]^T$
From CS_3 to CS_2	$[0 \quad 0 \quad 0]^T$
From CS_4 to CS_3	$[0 \quad 0 \quad 0]^T$
From CS_5 to CS_4	$[0 \quad 0 \quad -l_B]^T$ with the position of the mass center of BallBot: $l_B = \frac{m_{B1}(r_S + l_{B1} + l_{B2}/2) + m_{B2}(r_S + l_{B2}/2)}{m_{B1} + m_{B2}}$

the spatial relationship of the origin of the coordinate system CS_j in the coordinate system CS_i is defined by $\boldsymbol{O}_i^j = \left(Ox_i^j, Oy_i^j, Oz_i^j\right)$, with the specific values of the relative translations between different coordinate systems shown in Table 2. To facilitate writing the transformation matrices between different coordinate systems, the skew-symmetric matrix $\widetilde{\boldsymbol{O}}_i^j$ associated with the vector $\boldsymbol{O}_i^j$ is used, as shown in Equation (2). For example, the origin of the coordinate system CS_5 defined in the coordinate system CS_{or} ($\boldsymbol{O}_{or}^5$) is calculated using Equation (3):

$$\boldsymbol{R}_{or}^2 = \boldsymbol{R}_{or}^1\boldsymbol{R}_1^2 = 1, \quad \boldsymbol{R}_{or}^5 = \boldsymbol{R}_{or}^1\boldsymbol{R}_1^2\boldsymbol{R}_2^3\boldsymbol{R}_3^4\boldsymbol{R}_4^5. \tag{1}$$

where,

$$\boldsymbol{R}_{or}^5 = \begin{bmatrix} \cos(\theta_y)\cos(\theta_z) & \cos(\theta_y)\sin(\theta_z) & -\sin(\theta_y) \\ \begin{pmatrix}\cos(\theta_z)\sin(\theta_x)\sin(\theta_y)\\ -\cos(\theta_x)\sin(\theta_z)\end{pmatrix} & \begin{pmatrix}\cos(\theta_x)\cos(\theta_z)\\ +\sin(\theta_x)\sin(\theta_y)\sin(\theta_z)\end{pmatrix} & \cos(\theta_y)\sin(\theta_x) \\ \begin{pmatrix}\sin(\theta_x)\sin(\theta_z)\\ +\cos(\theta_x)\cos(\theta_z)\sin(\theta_y)\end{pmatrix} & \begin{pmatrix}\cos(\theta_x)\sin(\theta_y)\sin(\theta_z)\\ -\cos(\theta_z)\sin(\theta_x)\end{pmatrix} & \cos(\theta_x)\cos(\theta_y) \end{bmatrix},$$

$$\widetilde{\boldsymbol{O}}_i^j = \begin{bmatrix} 0 & -Oz_i^j & Oy_i^j \\ Oz_i^j & 0 & -Ox_i^j \\ -Oy_i^j & Ox_i^j & 0 \end{bmatrix}, \tag{2}$$

$$\boldsymbol{O}_{or}^5 = \begin{bmatrix} O_{or}^5(x) \\ O_{or}^5(y) \\ O_{or}^5(z) \end{bmatrix} = \begin{bmatrix} x_S + l\sin(\theta_x)\sin(\theta_z) + l\cos(\theta_x)\cos(\theta_z)\sin(\theta_y) \\ y_S - l\cos(\theta_z)\sin(\theta_x) + l\cos(\theta_x)\sin(\theta_y)\sin(\theta_z) \\ l\cos(\theta_x)\cos(\theta_y) \end{bmatrix}. \tag{3}$$

Several assumptions are made for the simulation of the BallBot model.[4] First, the friction in the system can be negligible. Second, the BallBot only moves in the horizontal plane, which means that the potential energy of the ball is zero. Moreover, the effects of the elasticity and damping of the ball are ignored. The ball is assumed not deform when the robot is mounted on it. Additionally, the ball moves without slipping relative to the floor defined by the horizontal plane $O_{or}x_{or}y_{or}$, leading to pure rolling ($P_S = 0$). On the basis of the Euler–Lagrange method, a mathematical model for the energy of the BallBot can be defined by Equation (4). The kinetic energy of the system consists of the kinetic energies of the ball in the O_2x_2 and O_2y_2 directions and the kinetic energies of the robot body and wheels. Moreover, the potential energy of the BallBot is totally based on that of the body.

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{\boldsymbol{q}}}\right) - \frac{\partial L}{\partial \boldsymbol{q}} = \boldsymbol{\tau}_{ext} \tag{4}$$

where $\boldsymbol{q} = [x_s \quad y_s \quad \theta_x \quad \theta_y \quad \theta_z]^T$ is the vector of system coordinates, $\dot{\boldsymbol{q}} = [\dot{x}_s \quad \dot{y}_s \quad \dot{\theta}_x \quad \dot{\theta}_y \quad \dot{\theta}_z]^T$ is the vector of the derivative of system coordinates (or speed of system variables), $\boldsymbol{\tau}_{ext}$ is the system external forces (related to the input moments), $L = K_S + K_B + K_{W_1} + K_{W_2} + K_{W_3} - P_B$ is the total energy of the system, P_B is the potential energy of the body, and K_S, K_B, K_{W_1}, K_{W_2}, and K_{W_3} are the kinetic energies of the ball, body, wheel 1, wheel 2, and wheel 3, respectively. They are calculated in detail as follows:

First, the ball only moves in the horizontal plane (on the floor), so it only possesses kinetic energy, and its potential energy is zero ($V_S = 0$). The kinetic energy of the ball is

$$K_S = \frac{1}{2}m_S\boldsymbol{v}_S^{2,or^T}\boldsymbol{v}_S^{2,or} + \frac{1}{2}\boldsymbol{\omega}_S^{2,or^T}\boldsymbol{I}_S^2\boldsymbol{\omega}_S^{2,or} \tag{5}$$

where $\boldsymbol{v}_S^{2,or} = [\dot{x}_S \quad \dot{y}_S \quad 0]$ is the linear velocity of the ball with respect to CS_{or}, $\boldsymbol{\omega}_S^{2,or} = \left[-\frac{\dot{y}_S}{r_S} \quad \frac{\dot{x}_S}{r_S} \quad 0\right]^T$ is the angular velocity of the ball with respect to CS_{or}, and $\boldsymbol{I}_S = \mathrm{diag}\left[\frac{2m_Sr_S^2}{3} \quad \frac{2m_Sr_S^2}{3} \quad \frac{2m_Sr_S^2}{3}\right]$ is the inertial moment of the spherical ball.

Second, the spherical bodies of the BallBot with potential (P_B) and kinetic (K_B) energies are calculated as follows:

$$P_B = m_B[0 \quad 0 \quad g]\boldsymbol{R}_{or}^5[0 \quad 0 \quad l]^T \tag{6}$$

$$K_B = \frac{1}{2}\boldsymbol{\omega}_B^{5,or^T}\boldsymbol{I}_B\boldsymbol{\omega}_B^{5,or} + \frac{1}{2}m_B\boldsymbol{v}_B^{5,or^T}\boldsymbol{v}_B^{5,or}, \tag{7}$$

where $g = 9.81\ \mathrm{m/s_2}$ is the gravity constant, $\boldsymbol{I}_B = \mathrm{diag}[I_{Bx} \quad I_{By} \quad I_{Bz}]$ is the inertial moment of the body of the BallBot, and I_{Bi} is the inertial moment of the body associated with the i-axis ($i = x_5, y_5, z_5$). Depending on the shape of the body (the solid-cuboid, hollow-cuboid, or cylinder), the inertial moment of the body of the BallBot is calculated based on Alrasheed.[33] $\boldsymbol{\omega}_B^{5,or}$ and $\boldsymbol{v}_B^{5,or}$ are, respectively, the angular and linear velocities of the BallBot with respect to the CS_{or} expressed in CS_5; $\boldsymbol{v}_B^{5,or} = \left[\dot{x}_B^5 \quad \dot{y}_B^5\right]^T$ and $\boldsymbol{\omega}_B^{5,or} = \left[\dot{\theta}_x^5 \quad \dot{\theta}_y^5 \quad \dot{\theta}_z^5\right]^T$. $\boldsymbol{\omega}_B^{5,or}$ and $\boldsymbol{v}_B^{5,or}$ can be found using Equation (8) by transforming the twist motion of BallBot from CS_{or} to CS_5.

$$\left[\boldsymbol{\omega}_B^{5,or}, \boldsymbol{v}_B^{5,or}\right]^T = \boldsymbol{T}_B^{5,or} = \begin{bmatrix} \boldsymbol{R}_{or}^5 & 0_{3\times3} \\ \widetilde{\boldsymbol{O}}_{or}^5\boldsymbol{R}_{or}^5 & \boldsymbol{R}_{or}^5 \end{bmatrix}^T \boldsymbol{T}_B^{or,or}, \tag{8}$$

where $T_B^{5,or}$ and $T_B^{or,or}$ are the twist motions of the BallBot with respect to the CS_{or}, expressed in CS_5 and $CS_{or,}$ respectively. In contrast, $T_B^{or,or}$ is calculated by multiplying a Jacobian matrix and the derivative of the system variables ($\dot{q} = [\dot{x}_s \quad \dot{y}_s \quad \dot{\theta}_x \quad \dot{\theta}_y \quad \dot{\theta}_z]^T$), as shown in Equation (9).

$$T_B^{or,or} = J_1 \dot{q} \tag{9}$$

where

$$J_1 = \begin{bmatrix} 0 & 0 & 0 & -\sin(\theta_z) & \cos(\theta_y)\cos(\theta_z) \\ 0 & 0 & 0 & \cos(\theta_z) & \cos(\theta_y)\sin(\theta_z) \\ 0 & 0 & 1 & 0 & -\sin(\theta_y) \\ 1 & 0 & \begin{pmatrix} l(\cos(\theta_z)\sin(\theta_x) \\ -\cos(\theta_x)\sin(\theta_y)\sin(\theta_z)) \end{pmatrix} & l\cos(\theta_x)\cos(\theta_y)\cos(\theta_z) & \begin{pmatrix} l(\cos(\theta_x)\sin(\theta_z) \\ -\cos(\theta_z)\sin(\theta_x)\sin(\theta_y) \end{pmatrix} \\ 0 & 1 & \begin{pmatrix} l(\sin(\theta_x)\sin(\theta_z) \\ +\cos(\theta_x)\cos(\theta_z)\sin(\theta_y)) \end{pmatrix} & l\cos(\theta_x)\cos(\theta_y)\sin(\theta_z) & \begin{pmatrix} -l(\cos(\theta_x)\cos(\theta_z) \\ +\sin(\theta_x)\sin(\theta_y)\sin(\theta_z)) \end{pmatrix} \\ 0 & 0 & 0 & -l\cos(\theta_x)\sin(\theta_y) & -l\cos(\theta_y)\sin(\theta_x) \end{bmatrix}.$$

Third, since the Omni wheels are attached to the robot body, the energy of the wheels only includes rotational kinetic energy (the potential energy of the wheels is zero). The kinetic energy of the ith Omni wheel ($i = 1 \div 3$) is calculated as:

$$K_{W_i} = \frac{1}{2}(I_{OW} + k\, I_M)\dot{\varphi}_i^2, \tag{10}$$

where I_M and I_{OW} are the inertial moments of the motor and the Omni wheel, respectively, k is the reduction ratio of the motor, I_M and k are provided by the motor manufacturer, and I_{OW} is determined as the inertial moment of the circular solid shape.[32]

The velocity of the Omni wheel (also of motor) is calculated using Equation (11) by the radius of the wheel r_{OW}, where the angular velocity $\omega_S^{5,B}$ of the ball with respect to the body (at CS_5) and expressed in CS_5 depends on the direction of the tangential velocity of the ith wheel at the contact point.

$$\dot{\varphi}_i = \frac{1}{r_{OW}}\left(\omega_S^{5,B} \times r_{W_i}\right)^T u_{W_i} \tag{11}$$

where

$$\omega_S^{5,B} = \frac{1}{r_S}\begin{bmatrix} -(\dot{\theta}_x r_S - \dot{\theta}_z r_S \sin(\theta_y) + \dot{y}_S \cos(\theta_y)\cos(\theta_z) - \dot{x}_S \cos(\theta_y)\sin(\theta_z)) \\ -\begin{pmatrix} \dot{\theta}_y r_S \cos(\theta_x) - \dot{x}_S \cos(\theta_x)\cos(\theta_z) - \dot{y}_S \cos(\theta_x)\sin(\theta_z) \\ +\dot{y}_S \cos(\theta_z)\sin(\theta_x)\sin(\theta_y) - \dot{x}_S \sin(\theta_x)\sin(\theta_y)\sin(\theta_z) + \dot{\theta}_z r_S \cos(\theta_y)\sin(\theta_x) \end{pmatrix} \\ -\begin{pmatrix} \dot{x}_S \cos(\theta_z)\sin(\theta_x) - \dot{\theta}_y r_S \sin(\theta_x) + \dot{y}_S \sin(\theta_x)\sin(\theta_z) \\ +\dot{y}_S \cos(\theta_x)\cos(\theta_z)\sin(\theta_y) - \dot{x}_S \cos(\theta_x)\sin(\theta_y)\sin(\theta_z) + \dot{\theta}_z r_S \cos(\theta_x)\cos(\theta_y) \end{pmatrix} \end{bmatrix}.$$

where r_{W_i} is the vector directed from the center of the ball to the contact point between the ball and the ith Omni wheel, as shown in Figure 3A. u_{W_i} is the unit vector, defining the direction of the tangential velocity of the ith Omni wheel ($i = 1 \div 3$) at the contact point, as shown in Figure 3B. These vectors can be defined using the equations shown below:

$$r_{W_1} = r_S[\sin(\alpha) \quad 0 \quad \cos(\alpha)]^T,$$
$$r_{W_2} = r_S\left[-\frac{1}{2}\sin(\alpha) \quad \frac{1}{2}\sqrt{3}\sin(\alpha) \quad \cos(\alpha)\right]^T,$$
$$r_{W_3} = r_S\left[-\frac{1}{2}\sin(\alpha) \quad -\frac{1}{2}\sqrt{3}\sin(\alpha) \quad \cos(\alpha)\right]^T,$$

$$u_{W_1} = [0 \quad 1 \quad 0]^T,$$
$$u_{W_2} = \left[-\frac{1}{2}\sqrt{3} \quad -\frac{1}{2} \quad 0\right]^T,$$
$$u_{W_3} = \left[\frac{1}{2}\sqrt{3} \quad -\frac{1}{2} \quad 0\right]^T.$$

Finally, the external force τ_{ext} is the external force transferred from the motors to the Omni wheels. This force can be calculated by multiplying the matrix of motor forces τ_i ($i = 1 \div 3$) with a Jacobian matrix J_2 created by the derivative of motor speed ($\dot{\varphi}_i$, $i = 1 \div 3$) in terms of the time derivatives of the system coordinates $\dot{q}$. The external force τ_{ext} is shown in Equation (12):

$$\tau_{ext} = [\tau_1 \quad \tau_2 \quad \tau_3]J_2 \tag{12}$$

where

$$J_2 = \begin{bmatrix} \frac{\partial\dot{\varphi}_1}{\partial\dot{x}_S} & \frac{\partial\dot{\varphi}_1}{\partial\dot{y}_S} & \frac{\partial\dot{\varphi}_1}{\partial\dot{\theta}_x} & \frac{\partial\dot{\varphi}_1}{\partial\dot{\theta}_y} & \frac{\partial\dot{\varphi}_1}{\partial\dot{\theta}_z} \\ \frac{\partial\dot{\varphi}_2}{\partial\dot{x}_S} & \frac{\partial\dot{\varphi}_2}{\partial\dot{y}_S} & \frac{\partial\dot{\varphi}_2}{\partial\dot{\theta}_x} & \frac{\partial\dot{\varphi}_2}{\partial\dot{\theta}_y} & \frac{\partial\dot{\varphi}_2}{\partial\dot{\theta}_z} \\ \frac{\partial\dot{\varphi}_3}{\partial\dot{x}_S} & \frac{\partial\dot{\varphi}_3}{\partial\dot{y}_S} & \frac{\partial\dot{\varphi}_3}{\partial\dot{\theta}_x} & \frac{\partial\dot{\varphi}_3}{\partial\dot{\theta}_y} & \frac{\partial\dot{\varphi}_3}{\partial\dot{\theta}_z} \end{bmatrix}.$$

MATLAB software, along with linearization tools and system parameters, was subsequently employed to compute the characteristic matrices of the system and obtain the simulation results.

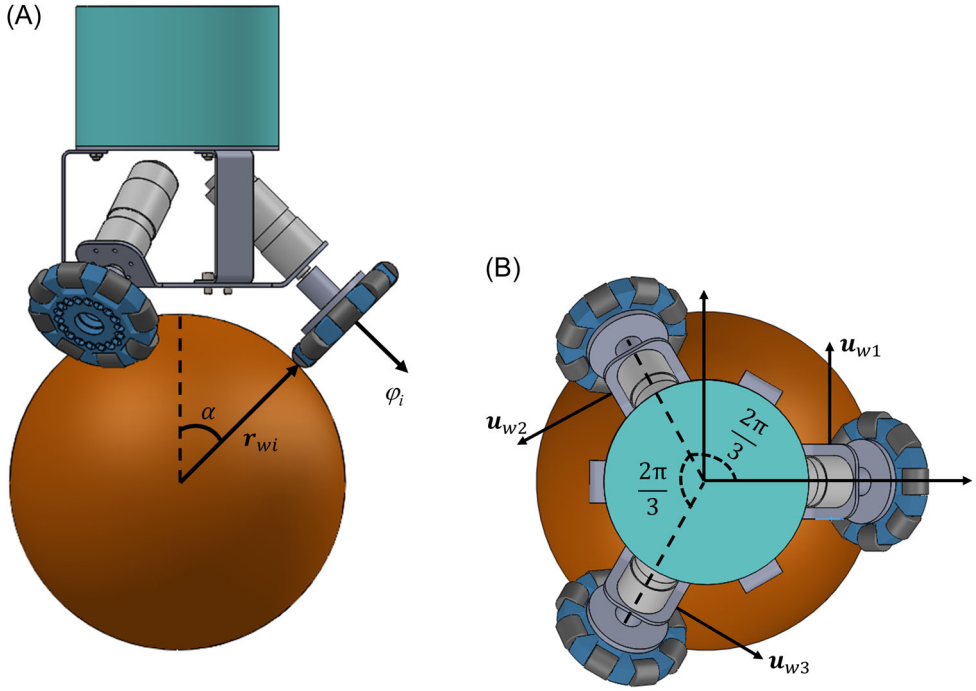

FIGURE 3 Contact points between wheels and ball. (A) Front view, (B) Top view.

3 | CONTROLLER DESIGN

The BallBot is an underactuated multiple-input and multiple-output (MIMO) model with fewer controllable DOFs than the total DOFs of the system. This characteristic, along with its high nonlinearity, makes the balancing control process more challenging than conventional models.[6] To address this issue, the LQR controller was chosen as the most suitable option for the BallBot. First, we linearize the system around its equilibrium position by setting five variables $\boldsymbol{q} = [x_S \quad y_S \quad \theta_x \quad \theta_y \quad \theta_z]$ and their derivatives to zero. Using the mathematical analysis from Section 2 and the system parameters in Table 3, we find the state space model of the BallBot system as shown in Equation (13), where $\boldsymbol{X} = [\boldsymbol{q}, \dot{\boldsymbol{q}}]$, $\boldsymbol{Y}$, $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ represent the state vector, the output vector, the state matrix, the input matrix, and the output matrix, respectively. Initially, we set the values of $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ based on $m_{B2} = 5$ kg to establish the simulation model. Subsequently, m_{B2} is varied to evaluate the effects of changing the robot mass.

$$\begin{cases} \dot{\boldsymbol{X}} = \boldsymbol{AX} + \boldsymbol{BU}, \\ \boldsymbol{Y} = \boldsymbol{CX} + \boldsymbol{DU} \end{cases} \tag{13}$$

where

$$\boldsymbol{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 \\ 0 & 0 & 0 & -14.9026 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 14.9026 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 33.6281 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 33.6281 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\boldsymbol{B} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -6.06 & 6.06 \\ 7.0 & -3.5 & -3.5 \\ 12.52 & -6.26 & -6.26 \\ 0 & 10.84 & -10.84 \\ -26.83 & -26.83 & -26.83 \end{bmatrix}.$$

$\boldsymbol{C} = [1]_{10\times10}$ is the identity matrix; and $\boldsymbol{D} = 0$.

A diagram of system control with LQR for the BallBot is shown in Figure 4. In general, the state vector $\boldsymbol{X}$ of the system compensates for the desired input $\boldsymbol{X}_d$ in the LQR controller to make a control signal $\boldsymbol{U}$.

TABLE 3 Structural parameters of the BallBot.

Parameter	Symbol	Value (unit)
Mass of upper body	m_{B2}	5 ÷ 90 (kg)
Mass of main frame	m_{B1}	3 (kg)
Total of mass	$m_B = m_{B1} + m_{B2}$	Varies with m_{B2}
Position of mass center of BallBot	l_B	Varies with m_{B2}
Height of upper body	l_{B2}	0.15 (m)
Height of main frame	l_{B1}	0.25 (m)
Mass of ball	m_S	0.62 (kg)
Radius of ball	r_S	0.125 (m)
Inertial moment of the ball	I_S	6.5×10^{-3} (kg m^2)
Mass of Omni wheel	m_{OW}	0.12 (m)
Radius of Omni wheel	r_{OW}	0.05 (m)
Inertial moment of Omni wheel	I_{OW}	0.15×10^{-3}
Inertial moment of the motor	I_M	2.42×10^{-6}
Reducing ratio of the motor	k	30
Zenith angle	α	45°

The relationship for these values is shown in Equations (14) and (15), in which the feedback gain matrix of the LQR ($\boldsymbol{K}_{LQR}$) is calculated to minimize the quadratic cost function $\boldsymbol{J}$ of the system. Moreover, the moment values $(\tau_1, \tau_2, \tau_3) = \boldsymbol{\tau}_{ext}$ of the three Omni wheels maintain non-zero values because the control signal $\boldsymbol{U}$ in the LQR controller supports the movement of the BallBot and keeps the system balanced around an angle of ±5°.

$$\boldsymbol{J} = \int_0^{\infty} (\boldsymbol{X}^T \boldsymbol{Q} \boldsymbol{X} + \boldsymbol{U}^T \boldsymbol{R} \boldsymbol{U})\, dt, \tag{14}$$

$$\boldsymbol{U} = \boldsymbol{K}_{LQR}(\boldsymbol{X}_d - \boldsymbol{X}) \tag{15}$$

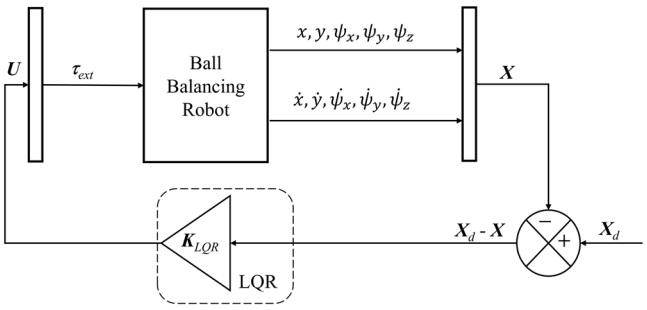

FIGURE 4 Diagram of the BallBot controller. LQR, Linear Quadratic Regulator.

where the state cost matrix $\boldsymbol{Q}$ is defined as:

$$\boldsymbol{Q} = \begin{bmatrix} 4.0000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4.0000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 32.8281 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 32.8281 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1013 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.4444 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.4444 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3.6476 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3.6476 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.0113 \end{bmatrix}.$$

$$\boldsymbol{R} = \begin{bmatrix} 0.0711 & 0 & 0 \\ 0 & 0.0711 & 0 \\ 0 & 0 & 0.0711 \end{bmatrix} \text{ is the input cost matrix.}$$

The stability of the closed-loop system is demonstrated using the Lyapunov criterion. The control input $\boldsymbol{U}$ for the closed-loop system is derived as follows:

$$\boldsymbol{U} = -\boldsymbol{K}_{LQR}\, \boldsymbol{X}_e, \tag{16}$$

where $\boldsymbol{X}_e$ is the output error, $\boldsymbol{X}_e = \boldsymbol{X} - \boldsymbol{X}_d$; and $\boldsymbol{K}_{LQR} = \boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P}$, where $\boldsymbol{P}$ is a positive-definite matrix which is the solution to the continuous time Algebraic Riccati equation.

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{P} + \boldsymbol{Q} = 0.$$

The closed-loop system can be rewritten as $\dot{\boldsymbol{X}}_{\mathrm{e}} = \boldsymbol{A}_{\mathrm{cl}}\boldsymbol{X}_{\mathrm{e}}$, where the closed-loop dynamics matrix $\boldsymbol{A}_{\mathrm{cl}} = \boldsymbol{A} - \boldsymbol{B}\boldsymbol{K}_{\mathrm{LQR}}$.

The Lyapunov candidate $\boldsymbol{V}$ for the closed-loop system is as follows:

$$\boldsymbol{V} = \dot{\boldsymbol{X}}_{\mathrm{e}}^{\mathrm{T}}\boldsymbol{P}\boldsymbol{X}_{\mathrm{e}}. \tag{17}$$

Taking the derivative of $\boldsymbol{V}$ with respect to time leads to

$$\dot{\boldsymbol{V}} = \dot{\boldsymbol{X}}_{\mathrm{e}}^{\mathrm{T}}\boldsymbol{P}\boldsymbol{X}_{\mathrm{e}} + \boldsymbol{X}_{\mathrm{e}}^{\mathrm{T}}\boldsymbol{P}\dot{\boldsymbol{X}}_{\mathrm{e}} = (\boldsymbol{A}_{\mathrm{cl}}\boldsymbol{X}_{\mathrm{e}})^{\mathrm{T}}\boldsymbol{P}\boldsymbol{X}_{\mathrm{e}} + \boldsymbol{X}_{\mathrm{e}}^{\mathrm{T}}\boldsymbol{P}(\boldsymbol{A}_{\mathrm{cl}}\boldsymbol{X}_{\mathrm{e}}) = \boldsymbol{X}_{\mathrm{e}}^{\mathrm{T}}\boldsymbol{A}_{\mathrm{cl}}^{\mathrm{T}}\boldsymbol{P}\boldsymbol{X}_{\mathrm{e}} + \boldsymbol{X}_{\mathrm{e}}^{\mathrm{T}}\boldsymbol{P}\boldsymbol{A}_{\mathrm{cl}}\boldsymbol{X}_{\mathrm{e}}. \tag{18}$$

Because $\boldsymbol{A}_{\mathrm{cl}} = \boldsymbol{A} - \boldsymbol{B}\boldsymbol{K}_{\mathrm{LQR}}$ and $\boldsymbol{K}_{\mathrm{LQR}} = \boldsymbol{R}^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{P}$,

$$\dot{\boldsymbol{V}}(\boldsymbol{X}_{\mathrm{e}}) = \boldsymbol{X}_{\mathrm{e}}^{\mathrm{T}}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{P} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{P})\boldsymbol{X}_{\mathrm{e}}. \tag{19}$$

Using $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{P} + \boldsymbol{Q} = 0$, one obtains

$$\dot{\boldsymbol{V}}(\boldsymbol{X}_{\mathrm{e}}) = \boldsymbol{X}_{\mathrm{e}}^{\mathrm{T}}(-(\boldsymbol{Q} + \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{P}))\boldsymbol{X}_{\mathrm{e}}. \tag{20}$$

Because $\boldsymbol{Q}$ and $\boldsymbol{R}$ are the positive definite matrices that are obtained appropriately and $\boldsymbol{P}$ is the positive solution, $\boldsymbol{V}(\boldsymbol{X}_{\mathrm{e}}) < 0$ is the decreased function. Therefore, the system is asymptotically stable in terms of the Lyapunov criterion.

Before designing the controller, the system controllability and observability must be checked. Using the MATLAB commands "rank (ctrb($\boldsymbol{A}$, $\boldsymbol{B}$))" and "rank(obsv($\boldsymbol{A}$, $\boldsymbol{C}$))" for these characteristics, the results indicate that both ranks equal the number of state variables (10). Therefore, the system is both controllable and observable.

Next, the LQR is an optimal control strategy that aims to minimize a quadratic cost function, which balances state deviations and control effort. By using the $\boldsymbol{K}_{\mathrm{LQR}} = \mathrm{lqr}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{Q}, \boldsymbol{R})$ command in the MATLAB environment, the $\boldsymbol{K}_{\mathrm{LQR}}$ is estimated.

$$\boldsymbol{K}_{\mathrm{LQR}} = \begin{bmatrix} 0.0001 & -6.1237 & 34.1676 & 0.0001 & -0.6892 & 0.0001 & -10.2838 & 11.9778 & 0.0000 & -0.2644 \\ 5.3033 & 3.0619 & -17.0838 & 29.5904 & -0.6891 & 8.9059 & 5.1420 & -5.9889 & 10.3731 & -0.2644 \\ -5.3033 & 3.0619 & -17.0839 & -29.5902 & -0.6892 & -8.9058 & 5.1419 & -5.9889 & -10.3730 & -0.2644 \end{bmatrix}.$$

The stability of the system can be checked by assessing the eigenvalues in Table 4. Owing to all the eigenvalues being negative, the BallBot system easily reaches a steady state.

4 | RESULTS AND DISCUSSION

4.1 | System performance with LQR control

Before investigating the influence of the parametric configuration on the BallBot's control performance, the feasibility of our LQR controller is validated by comparing our simulation results with the experimental results of a similar study by Bhatti et al.[31] under similar input signals. Figure 5 presents our simulation results. Table 5 provides a comparison between simulation and experimental values. The rise time for the roll and pitch angles in their study was 0.5 s, whereas it was 0.12 s and 0.08 s in our simulation. Additionally, Bhatti's system had an overshoot of 12.5% for roll and 6.25% for pitch, whereas our simulation results show 15% and 6.25%, respectively. The differences arise because our simulation model assumes an ideal environment without disturbances or uncertainties, whereas the experiment included external disturbances and approximate system parameter estimation. Despite these differences, our system response closely matches that of previous research when external disturbances and uncertainties are not considered. Thus, our mathematical model and LQR controller implementation were validated as feasible and effective for controlling the BallBot system.

The BallBot can balance itself with the LQR controller when its body is tilted at an angle of approximately 5°, as shown in Figure 6. Assume that the BallBot is initially unbalanced in the linear region (which means that the robot body is tilted only at an angle of approximately 5°). At this time, the system actively controls the robot to move enough for its body to stand upright again in the vertical direction. The robot moves in a direction of 45° from the O_2x_2 direction to reach a new position with coordinates $(x, y) = (0.1, 0.1)$ m. To trace such a path, the ball under the robot's bottom also rolls at an angle of 65°. When the vertical state of the robot is obtained (the tilt angle of the robot body is less than 1°), the robot actively moves back to the position of the original coordinate system. The entire process occurs within 5 s.

Figures 7 and 8 illustrate the simulation of BallBot motion following different trajectories. Each desired trajectory influences the control system through the variable $\boldsymbol{X}_{\mathrm{d}}$. Once the BallBot achieves initial self-balancing, it can accurately follow the desired trajectory. For the square trajectory (2×2 m), the robot encounters challenges at the corners, leading to temporary imbalance. During such instances, the robot actively balances itself, returning to a vertical state before proceeding to the next segment. The rebalancing process typically lasts for approximately 5 s, and its effects can be observed in Figure 7B,C through position and angle overshot. In contrast, when moving along a circular trajectory, the BallBot maintains its equilibrium throughout the motion due to the absence of sudden directional changes observed in the square trajectory. However, when the robot enters the circular trajectory path shown in Figure 8, there is still a slight rebalancing process that takes approximately 2.5 s.

TABLE 4 Eigenvalues used to assess the stability of the BallBot system.

Poles									
−0.89 + 0.73i	−0.89 to 0.73i	−0.89 + 0.73i	−0.89 to 0.73i	−3.01	−3.01	−3.04	−18.2	−112.15	−112.15

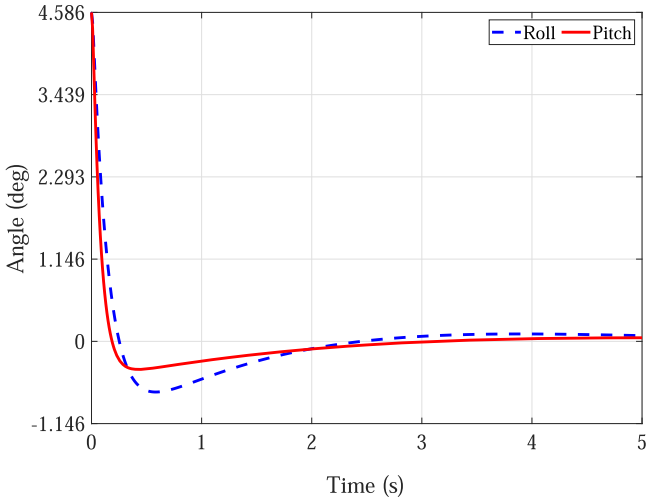

FIGURE 5 Simulation results of the system response with conditions similar to Bhatti et al.[31]

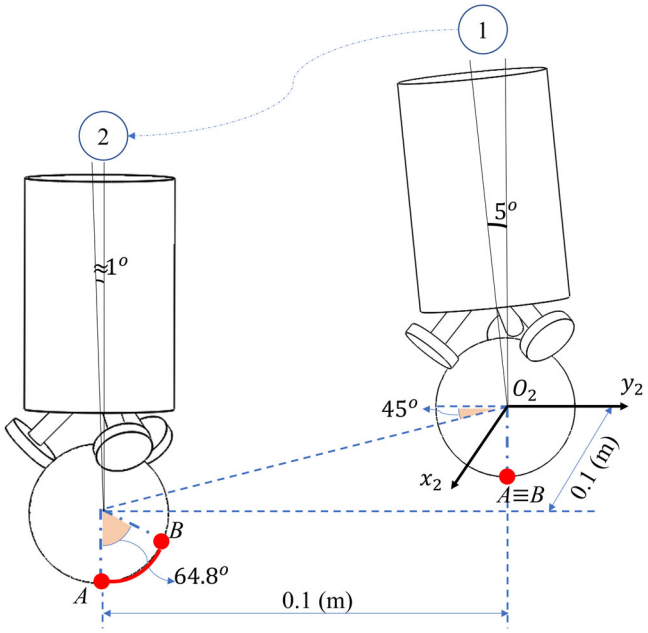

FIGURE 6 Auto-Balance process of the BallBot.

TABLE 5 Comparison of results between Bhatti et al.[31] and the proposed model.

	Roll (reference work)	Pitch (reference work)	Roll	Pitch
Rise time (s)	0.5	0.5	0.12	0.08
Overshoot (%)	12.5	6.25	15	6.25

4.2 | Effect of the BallBot parametric configuration

Figure 9 shows the system performance when the robot mass (the upper part of the robot) changes. In this analysis, the mass of the upper part (m_{B2}) varies from 5 to 90 kg, and the other parameters are unchanged. The resulting system responses in the x_S and y_S directions as well as θ_x and θ_y are similar. Moreover, the system response to the moment value τ_1 represents the response of the input value since the torque from the three motors is similar. On the basis of the results, as the robot mass increases, the settling time (taken within 2% of the maximum value) of x_S, θ_x, and θ_z increases and increases almost linearly. In addition, the overshoot of the angular inclination θ_x decreases linearly as the robot mass increases. However, the response times of the generated torques, which affect the system, are different. When the mass increases above 30 kg, the settling time of τ_1 is almost negligible, whereas the overshoot decreases linearly.

Figures 10–12 show the response values of x_S, θ_x, and τ_1, respectively. These include the rise time (the time the signal takes to increase from 10% to 90% of its maximum value), peak time (the time when the signal reaches its highest value), maximum error (the maximum amplitude of the signal), and overshoot (the maximum percentage of the signal exceeds its steady-state value). In Figure 10, the rise time of x_S is nearly linear as the robot mass increases. However, the peak time and overshoot significantly increase when the mass of m_{B2} changes from 5 to 30 kg. After m_{B2} reaches 30 kg, the peak time and overshoot only slightly increase. The maximum error of x_S decreases from 0.1 to 0.082 m as the robot mass increases, with the rate of decrease diminishing as the mass continues to rise.

The rise times of θ_x and τ_1 increased with increasing upper body mass of the robot, as shown in Figures 11 and 12. However, after the upper body mass reached 50 kg, the rate of increase in the rise time decreased. The peak time for θ_x increased from 0.77 to 0.79 s, and that for τ_1 increased from 0.65 to 0.77 s as the mass changed from 5 to 30 kg. In addition, the peak time decreased linearly from 0.79 to 0.76 s for θ_x and from 0.77 to 0.74 s for τ_1 as the mass increased from 30 to 90 kg.

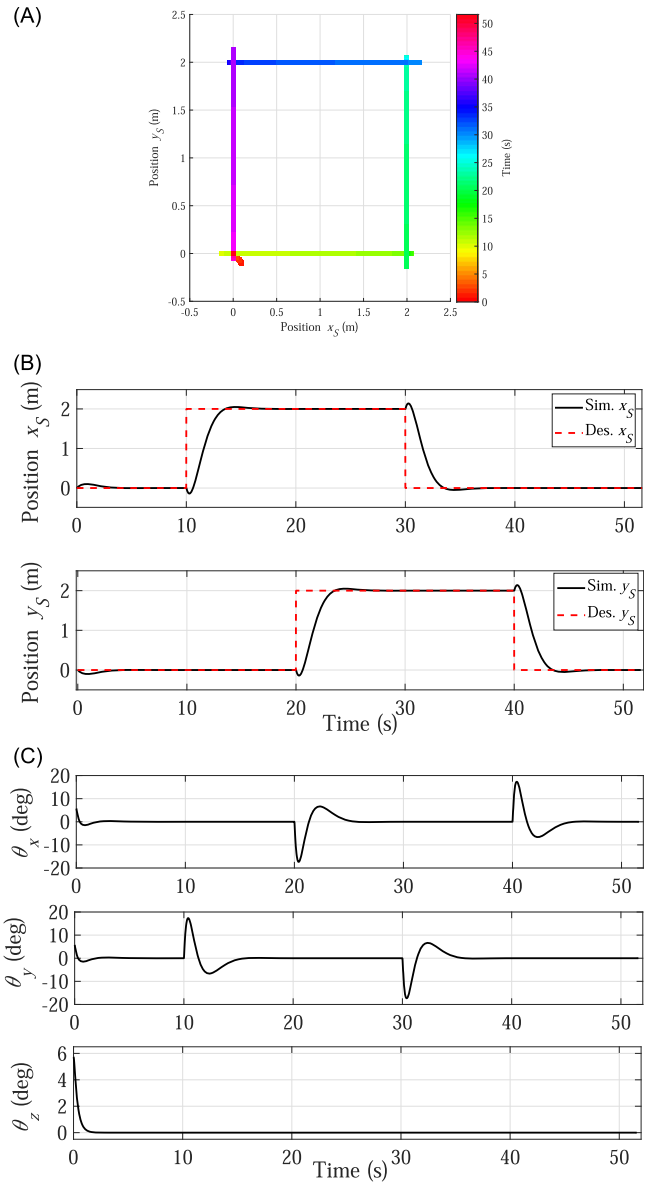

FIGURE 7 Simulation results of the BallBot with a square trajectory: (A) moving trajectory, (B) performance of the BallBot following O_2x_2 and O_2y_2, and (C) tilting angles of the BallBot.

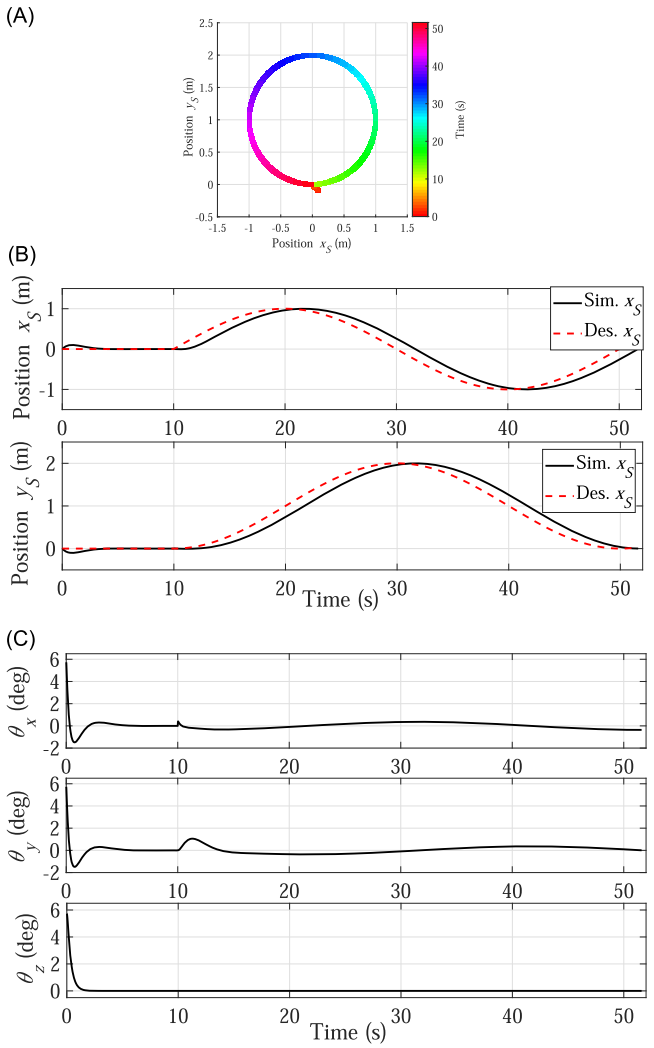

FIGURE 8 Simulation results of the BallBot with a circular trajectory: (A) moving trajectory, (B) performance of the BallBot following O_2x_2 and O_2y_2, and (C) tilting angles of the BallBot.

The maximum error and overshoot of θ_x decreased linearly, whereas for τ_1, they increased linearly with increasing body mass. This is because the state cost matrix $\boldsymbol{Q}$ has a larger value, making the controller more aggressive in minimizing the state errors of θ_x and θ_y. The peak time increases from 5 to 30 kg, followed by a decrease from 30 to 90 kg, indicating that the upper body mass significantly affects the system response time. The slowest response time occurred when the upper body mass was approximately 30 kg, approximately 10 times greater than the robot main frame mass.

Additionally, the peak time variation for θ_x was within a smaller range (0.76–0.79 s) than that for τ_1 (0.65–0.77 s), indicating that even with a maximum controller signal change, the peak time for the controller signal was still greater for smaller upper body masses. The overshoot of θ_x decreased linearly from 20.5% to 16.8% as the mass increased from 5 to 90 kg. The maximum moment signal τ_1 increased from 3.2 to 5.2 N·m, a 62.5% increase. This shows that with an increase in upper body mass, the controller effort must significantly increase to maintain system stability and control variables within the designed range. In all cases, the rise time, which indicates the time at which the system reaches a steady state, always increases linearly as the upper body mass increases.

Furthermore, Figure 13 illustrates the response of the BallBot system as the characteristics of the ball are changed. Specifically, Table 6 presents the radius and mass of the ball used in the BallBot, which are based on commercial products.[34] Similar to the analysis of the effect of the BallBot mass, the system response in the O_2x_2 and O_2y_2 directions,

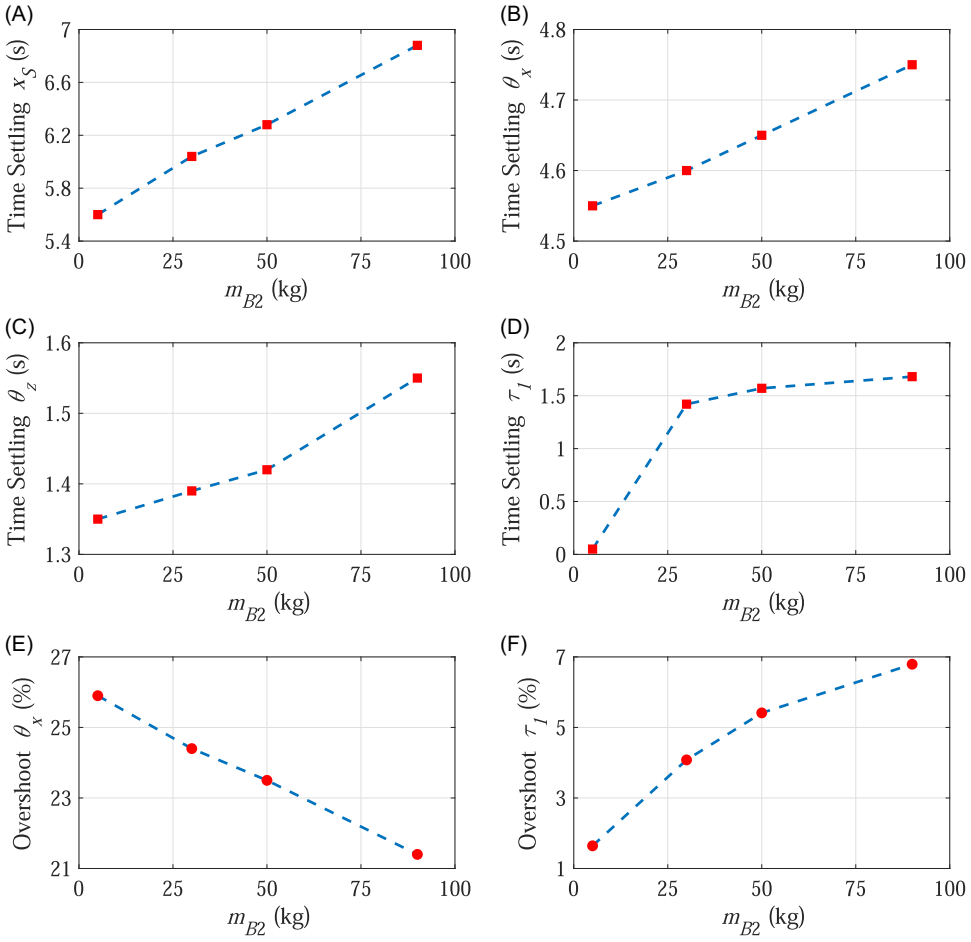

FIGURE 9 Settling time and overshoot with varying upper body masses of the robot. (A-D) Settling time of x_S, θ_x, θ_z, and τ_1 in respectively. (E) and (F) Overshoot of θ_x and τ_1 in respectively.

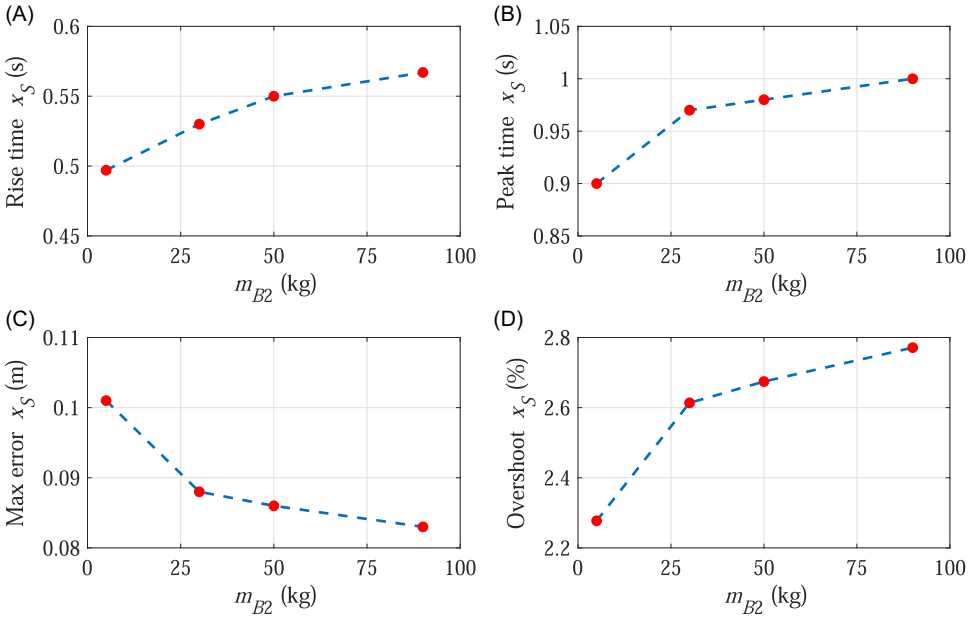

FIGURE 10 Response values of x_s with varying upper body mass of the robot. (A) Rise time, (B) Peak time, (C) Maximum error, and (D) Overshoot.

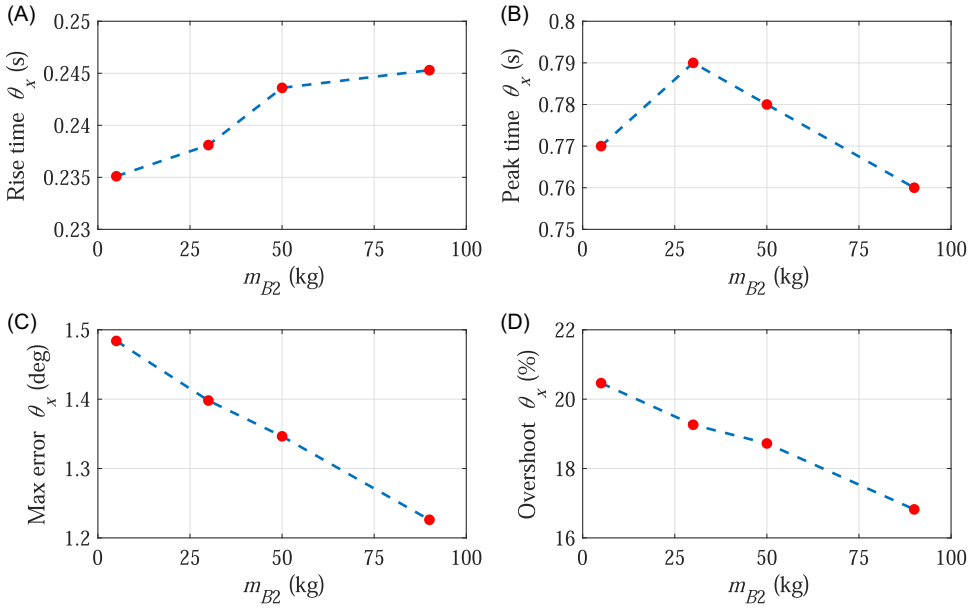

FIGURE 11 Response values of θ_x with varying upper body masses of the robot. (A) Rise time, (B) Peak time, (C) Maximum error, and (D) Overshoot.

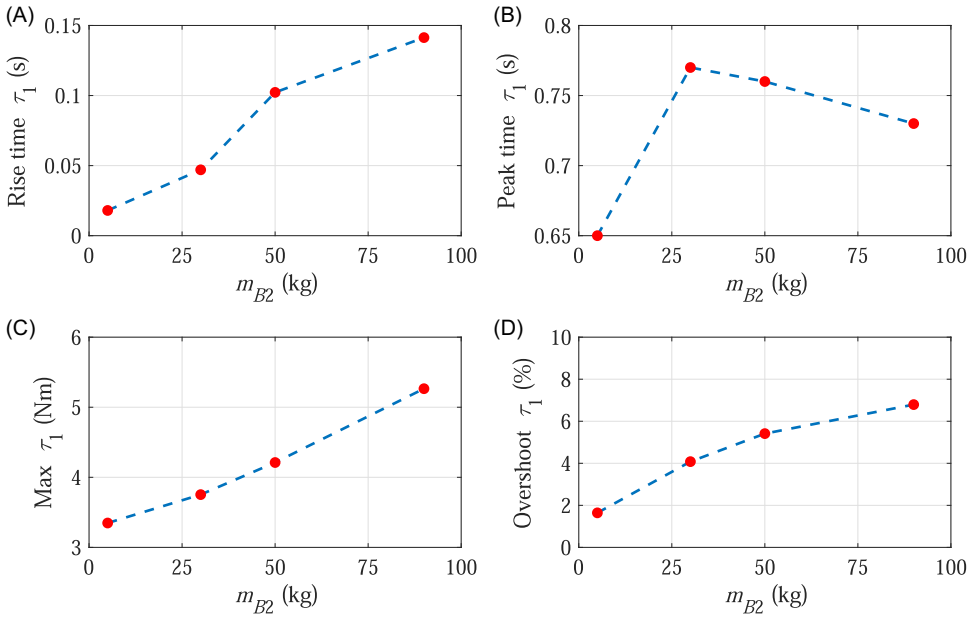

FIGURE 12 Response values of τ_1 with varying upper body masses of the robot. (A) Rise time, (B) Peak time, (C) Maximum error, and (D) Overshoot.

as well as θ_x and θ_y, exhibit similar patterns. Figure 14 shows the trajectory error x_S, angle errors of θ_x, θ_y, and θ_z between ball sizes of 6 and 5 and between ball sizes of 7 and 5. The maximum error of x_S is approximately 2.5×10^{-3} m at a peak time of approximately 0.8 s, whereas the maximum errors of θ_x and θ_y are very similar: approximately 2.5×10^{-3} ° also occur at a peak time of 0.8 s. Moreover, based on these results, it can be concluded that the system has not undergone significant changes as the ball parameters vary.

5 | CONCLUSION

The LQR controller was effectively applied to control the BallBot, a highly underactuated MIMO system with nonlinear characteristics. The control diagram in Figure 4 shows the relationship between the state vector $\boldsymbol{X}$ and the desired input $\boldsymbol{X}_d$, resulting in the control signal $\boldsymbol{U}$. The moments τ_{ext} generated by the LQR controller drive the BallBot's movement and maintain its balance within an angle of ±5°.

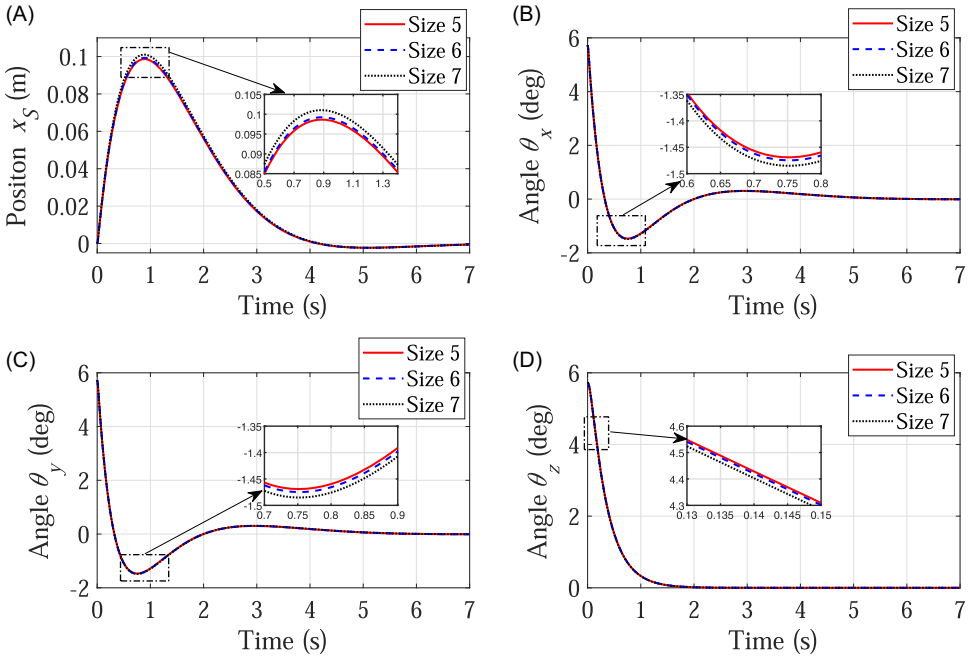

FIGURE 13 System response when the parameters of the ball change. (A) System response of posistion x_S, (B-D) System responses of angles θ_x, θ_y, and θ_z.

TABLE 6 Parameters of ball.[34]

	Radius of ball (m)	Mass of ball (kg)
Size 5	0.222	0.46
Size 6	0.23	0.57
Size 7	0.25	0.62

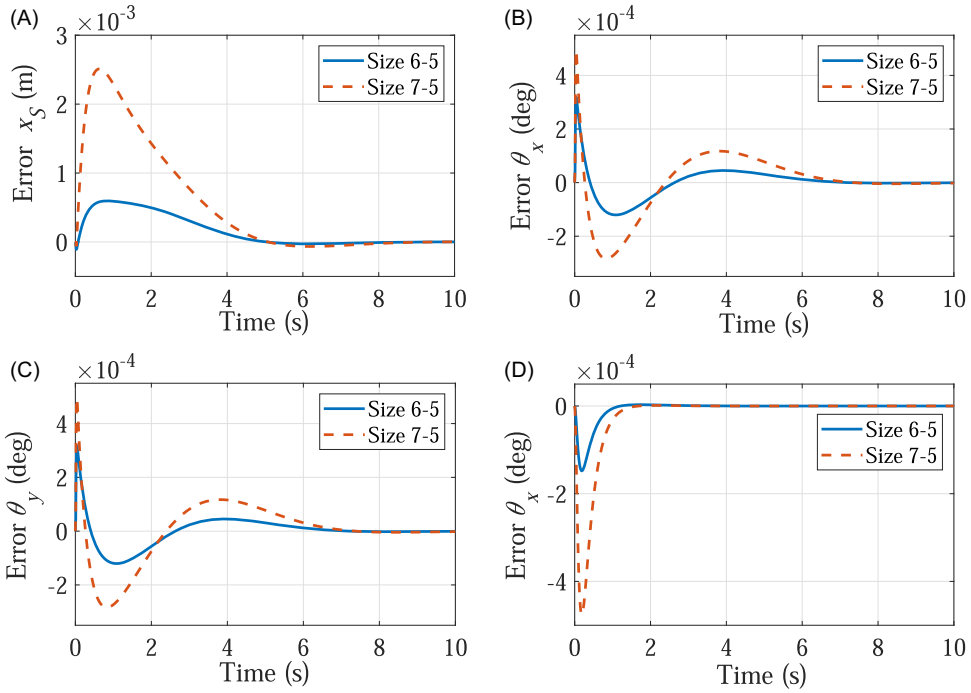

FIGURE 14 Errors when different sizes of balls are compared. (A) Trajectory error of x_S, (B-D) Angle errors of θ_x, θ_y, and θ_z.

The simulations revealed that the BallBot could balance itself with the LQR controller when tilted at an angle of approximately 5°. The robot controlled its movement to return to a vertical state when initially unbalanced, followed a trajectory to reach a new position, and returned to its original coordinate within 5 s. This study also examined the effects of varying the BallBot upper body mass and ball characteristics on control performance. As the upper body mass (m_{B2}) increased from 5 to 90 kg, the settling times of x_S and θ_x increased almost linearly, whereas the overshoot of θ_x decreased linearly. The rise times of θ_x and τ_1 also increased with mass but at a slower rate above 50 kg. The peak times initially increased (from 0.77 to 0.79 s for θ_x and from 0.65 to 0.77 s for τ_1 when m_{B2} changed from 5 to 30 kg) and then decreased linearly with further mass increases. The maximum error of x_S decreased from 0.1 to 0.082 m as the mass increased.

Figures 7 and 8 show that the BallBot could accurately follow the desired paths once it achieved initial self-balancing. For a square trajectory, a temporary imbalance occurred at the corners, prompting the robot to actively rebalance before proceeding. The rebalancing process typically lasted approximately 5 s. In contrast, the BallBot maintained equilibrium throughout a circular trajectory with a slight 2.5 s rebalancing process upon entry.

The analysis indicated that the controller effort had to increase significantly to maintain stability as the upper body mass increased. Additionally, varying the ball radius and mass resulted in consistent system responses for x_S and θ_x, with maximum errors of approximately 2.5×10^{-3} m for x_S and 2.5×10^{-3} ° for θ_x at a peak time of 0.8 s.

Overall, the results demonstrate that upper body mass significantly impacts the system's response time and stability, whereas ball parameters have a less pronounced effect. The implementation of the LQR controller in the BallBot indicated effective control performance, enabling smooth motion and robust balancing even with varying trajectories and mass changes.

AUTHOR CONTRIBUTIONS

Nhu Thanh Vo: Conceptualization; methodology; investigation; data analyzing; writing—original draft; writing—review and editing. **Anh-Duc Pham**: Supervision; writing—review and editing; analysis and investigation. **Ba Hoa Thai**: Data collection; investigation. **Phuoc Vinh Dang**: Investigation.

ACKNOWLEDGMENTS

The authors would like to thank all the members of the Mechatronics Division, University of Science and Technology, The University of Da Nang, for their support and feedback.

CONFLICT OF INTEREST STATEMENT

The author declares no conflict of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Anh-Duc Pham http://orcid.org/0000-0002-0400-9200

Nhu Thanh Vo http://orcid.org/0000-0001-5383-3119

REFERENCES

1. Elliott A. *The Culture of AI: Everyday Life and the Digital Revolution*. Routledge; 2019.
2. Strazdas D, Hintz J, Khalifa A, Abdelrahman AA, Hempel T, Al-Hamadi A. Robot system assistant (RoSA): towards intuitive multi-modal and multi-device human–robot interaction. *Sensors*. 2022;22(3):923.
3. Thanh VN, Vinh DP, Nghi NT, Nam LH, Toan DLH. Restaurant serving robot with double line sensors following approach. Paper presented at: 2019 IEEE International Conference on Mechatronics and Automation; 2019:235-239; Tianjin, China.
4. Pham DB, Kim H, Kim J, Lee SG. Balancing and transferring control of a ball segway using a double-loop approach [applications of control]. *IEEE Control Syst Mag*. 2018;38(2):15-37.
5. Cai C, Lu J, Li Z. Kinematic analysis and control algorithm for the BallBot. *IEEE Access*. 2019;7:38314-38321.
6. Lauwers TB, Kantor GA, Hollis RL. A dynamically stable single-wheeled mobile robot with inverse mouse-ball drive. Paper presented at: 2006 IEEE International Conference on Robotics and Automation; 2006:2884-2889; Orlando, FL.
7. Thanh VN, Vinh DP, Nam LH, Nghi NT, Anh DL. Reinforcement Q-learning PID controller for a restaurant mobile robot with double line-sensors Association for Computing Machinery. Paper presented at: 4th International Conference on Machine Learning and Soft Computing ICMLSC; 2020:164-167; New York, NY.
8. Le HN, Dang PV, Pham AD, Vo NT. System identifications of a 2DOF pendulum controlled by QUBE-servo and its unwanted oscillation factors. *Arch Mech Eng*. 2020;67(4):435-450.
9. Pham AD, Ahn HJ. Evaluation of input shaping methods for the nonlinear vibration system using a furuta pendulum. *J Korean Soc Precis Eng*. 2020;37(11):827-833.
10. Pham DB, Weon IS, Lee SG. Partial feedback linearization double-loop control for a Pseudo-2D ridable BallBot. *Int J Control Autom Syst*. 2020;18:1310-1323.
11. Fankhauser P, Gwerder C. *Modeling and Control of a BallBot*. Dissertation. ETH Zurich; 2010.
12. Lal I, Nicoara M, Codrean A, Busoniu L. Hardware and control design of a ball balancing robot. Paper presented at: IEEE 22nd International Symposium on Design and Diagnostics of Electronic Circuits & Systems; 2019:1-6; Cluj-Napoca, Romania.
13. Garcia-Garcia RA, Arias-Montiel M. Linear controllers for the NXT BallBot with parameter variations using linear matrix inequalities [lecture notes]. *IEEE Control Syst Mag*. 2016;36(3):121-136.
14. Ortiz I, Jurado F, Ollervides-Vazquez E. Discrete-time linear quadratic regulator for a NXT BallBot system. Paper presented at: 2022 International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE); 2022:95-100; Cuernavaca, Mexico.
15. Ba Pham D, Quang Duong X, Sang Nguyen D, et al. Balancing and tracking control of BallBot mobile robots using a novel synchronization controller along with online system identification. *IEEE Trans Ind Electron*. 2023;70(1):657-668.
16. Nguyen DN, Cao TB, Pham TH, Vo NT, Dang PV, Le HN. Design and control of a Ball-balancing robot. Paper presented at: 4th International Conference on Green Technology and Sustainable Development; 2018:317-322; Ho Chi Minh City, Vietnam.

17. Blonk JW. *Modeling and Control of a Ball-balancing robot*. Dissertation. University of Twente; 2014.
18. Wang J, Rui X, Wang X, Zhang J, Zhou Q, Gu J. Eigenvalue analysis of planar linear multibody system under conservative force based on the transfer matrix method. *Int J Mech Syst Dyn*. 2023;3:12-24.
19. Le HN, Vo NT. Modeling and analysis of an RUU delta robot using SolidWorks and SimMechanics. *Int J Dyn Control*. 2024;12(7): 2467-2479.
20. Do VT, Lee SG, Gwak KW. Passivity-based nonlinear control for a BallBot to balance and transfer. *Int J Control Autom Syst*. 2019;17: 2929-2939.
21. Wang P, Rui X, Liu F, et al. Generation mechanism and control of high-frequency vibration for tracked vehicles. *Int J Mech Syst Dyn*. 2023;3:146-161.
22. Huang Y, Ding J. A multiscale differential-algebraic neural network-based method for learning dynamical systems. *Int J Mech Syst Dyn*. 2024;4:77-87.
23. Do VT, Lee SG, Kim JH. Robust integral backstepping hierarchical sliding mode controller for a BallBot system. *Mech Syst Signal Process*. 2020;144:106866.
24. Jespersen TK. *Kugle—Modelling and Control of a Ball-balancing Robot*. Master's thesis. Aalborg University; 2019.
25. Jang H-G, Hyun C-H, Park BS. Virtual angle-based adaptive control for trajectory tracking and balancing of ball-balancing robots without velocity measurements. *Int J Adapt Control Signal Process*. 2023; 37(8):2204-2215.
26. Nguyen XM, Pham DB, Duong XQ, et al. Intelligent robust disturbance rejection control of a BallBot drive micromobility vehicle. *IEEE Trans Intell Veh*. 2024;9(2):3617-3629.
27. Do V-T, Lee S-G, Van M. Adaptive hierarchical sliding mode control for full nonlinear dynamics of uncertain ridable BallBots under input saturation. *Int J Robust Nonlinear Control*. 2021;31: 2882-2904.
28. Vu DC, Pham MD, Nguyen TTH, Nguyen TVA, Nguyen TL. Time-optimal trajectory generation and observer-based hierarchical sliding mode control for BallBots with system constraints. *Int J Robust Nonlinear Control*. 2024;34(11):7580-7610.
29. Navabi H, Sadeghnejad S, Ramezani S, Baltes J. Position control of the single spherical wheel mobile robot by using the fuzzy sliding mode controller. *Adv Fuzzy Syst*. 2017:2651976.
30. Odry Á, Fullér R, Rudas IJ, Odry P. Fuzzy control of self-balancing robots: a control laboratory project. *Comput Appl Eng Educ*. 2020;28: 512-535.
31. Bhatti OS, Tariq OB, Manzar A, Khan OA. Adaptive intelligent cascade control of a ball-riding robot for optimal balancing and station-keeping. *Adv Robot*. 2017;32(2):63-76.
32. Rui X, Zhang J, Wang X, Rong B, He B, Jin Z. Multibody system transfer matrix method: the past, the present, and the future. *Int J Mech Syst Dyn*. 2022;2:3-26.
33. Alrasheed S. Rotation of rigid bodies. In: Alrasheed S, ed. *Principles of Mechanics. Advances in Science, Technology & Innovation*. Springer; 2019:103-122.
34. Jeff H. *Basketball Size Chart—Recommended Sizes for Kids & Adults*. Breakthrough Basketball, LLC. Accessed November 6, 2024. https://www.breakthroughbasketball.com/basketballs/size-chart.html

AUTHOR BIOGRAPHIES

Anh-Duc Pham received a PhD degree from Soongsil University, Korea, in 2018. He was a research associate at IMSLab – Soongsil University and is currently a researcher/lecturer at the Faculty of Mechanical Engineering, University of Science and Technology – The University of Danang. Dr. Pham's research interests are in the areas of mechatronics, control and precision machine design, sustainable design and manufacturing, and applications of nanomaterials.

Ba Hoa Thai received his Bachelor degree in Mechatronics Engineering at University of Science and Technology – The University of Danang. He is currently a Master student at Chungnam National University. His research interests are in the area of mechatronics and control systems.

Phuoc Vinh Dang is an associate professor at the Faculty of Mechanical Engineering, The University of Danang – University of Science and Technology in Danang, Viet Nam. He completed his PhD at Politecnico di Milano, Italy, from 2013 to 2017, specializing in fluid journal bearings, tilting pad journal bearings, dynamics and vibration, robotics, automatic control, and microcontrollers. His research and teaching continue to advance in these fields, contributing to mechanical engineering education and innovation.

Nhu Thanh Vo is the head of the Mechatronics Division of the Faculty of Mechanical Engineering, The University of Danang – University of Science and Technology in Danang, Viet Nam. He received his PhD in Intelligent Mechanical System Engineering at Kagawa University, Japan, in 2017. His research interests include robotics, control systems, machine learning, sound and image processing, and automation.

How to cite this article: Pham A-D, Thai BH, Dang PV, Vo NT. Analysis of the parametric configuration impact on BallBot control performance. *Int J Mech Syst Dyn*. 2024;4:446-460. doi:10.1002/msd2.12133