A Comparison Between PID and LQR Controllers for Stabilization of a Ball Balancing Robot

Shashi Bhushan Sankhyan[1]* and Gunchita Kaur Wadhwa[2]

[1]Department of Mechanical Engineering, Pillai HOC College of Engineering and Technology, University of Mumbai, Rasayani, 410207, Maharashtra, India

[2]Department of Mechanical Engineering, ASET, Amity University Mumbai, Panvel, 410206, Maharashtra, India

*Corresponding author: shashi.sankhyan@gmail.com

Received: 15th March 2023
Revised: 1st June 2023
Accepted: 5th June 2023

OPEN ACCESS

Abstract: Ball balancing robot (BBL) forms a dynamically stable system mounted on a ball, which is in point contact with the ground surface. An omni-directional system for the BBL with maneuvering ability in the horizontal plane is attained as compared to two-wheeled robots, which can only move forward or backward. The stability of the BBL is defined by its capability to retain the upright position under all circumstances. Available literature includes the use of several single controllers to stabilize the BBL. This study performs a comparison of two popular controllers for stability analysis of the BBL, which included two model-based controllers, i.e., Proportional Integral Derivative (PID) and Linear Quadratic Regulator (LQR). A 2D planar model is considered for mathematical modeling at the two vertical planes as well as the horizontal plane. Furthermore, the steady state equations are derived using the Euler-Lagrangian method. PID and LQR controllers are used to provide stability to the BBL using a mathematical toolkit in MATLAB. The results from MATLAB are used to study the differences between PID and LQR for stability of the BBL based on time needed to balance the robot. The settling time for the PID and LQR controllers was 0.79 seconds and 2.25 seconds, respectively. The results illustrate that the PID controller stabilized the BBL in upright position efficiently and more swiftly as compared to the LQR controller.

Keywords: Ballbots; LQR; MATLAB; Mathematical Modeling; PID.

Introduction

Self-balancing robots are well acknowledged for their ability to stabilize themselves using one or two wheels or a ball [1]. The concept of inverted pendulum is applied for either type, i.e., the center of mass of the robot lies above its point of contact with the ground. The pioneering concept based on this method is the two-wheeled robot, which balances itself with a point contact on the ground. This allows it to move freely in forward and backward directions, as used in the popular Segway RMP wheelchair, known as the IBOT [2, 3]. Subsequently, further developments in the same field emerged with tele-presence [4] and UBOT [5, 6, and 7]. A major limitation is that the wheels of the robot were uni-directional due to which the bot is falling in the vertical plane and not permitting sideways movement. This led to the development of the single wheeled robots, which overcame this constraint and could perform several tasks in desired directions with ease.

Ballbots were developed, as the shortcomings of the two-wheeled robots were apprehended [8, 9, 10, and 11]. Unlike two wheeled robots, BBL can balance itself on a ball and moreover it can maneuver in any direction at any instant (omni-directional). The principle on which the BBL works is to keep the center of mass of the robot in line with the point of contact between the ground surface and the ball. Therefore, in order to keep the robot in upright position, the movement of the ball is controlled counter to movement of the body. This is achieved by using omni-directional wheels, which rotate the main ball in contact with ground in the counter direction further maintaining upright position of the robot.

A BBL mainly consists of three mechanical parts: body, undercarriage and the ball. The undercarriage consists of three omni-directional wheels attached to DC motors, which are placed rotation-symmetrically at 120° on the ball. The omni-directional wheels are mounted on a ball that rolls on the ground allowing the robot to move in any direction. The body of the robot is mounted on the undercarriage. The dynamics of the BBL is complex given the ability to move along all directions in the horizontal plane. The first prototype of a BBL was developed by Tom Lauwers and Ralph Hollis. They incorporated rollers instead of omni-directional wheels and a belt drive mechanism, which allowed the ballbot to move forward [12, 13]. Consequently, there were many upcoming researchers, which began to research on this topic [14, 15, 16, and 17].

Laszlo Havasi (2005) autonomously developed a ballbot named ERROSphere, which used optimal control theory using a linear quadratic regulator (LQR) model based on a linear approximation of the system equations. Furthermore, Kumagai (2008) named his work as BallIP [10] at the Tohoku Gakuin University. The model in this case could balance not only the robot itself but also an additional weight of 3 kg, which demonstrated another application of the ballbot in the field of transportation. In 2010, students of mechanical engineering department of a University in Zurich developed a ball-balancing robot named Rezero. The main characteristics of Rezero were that it could maneuver like a human being.

A second prototype of Ballbot was made by Tohoku Gakuin University (TGU) utilizing stepper motors placed at the corners forming a shape of a triangle, which took into account the yaw mechanism [12, 18]. University Of Adelaide demonstrated their ballbot and tried to balance it on using balls of an assortment of sizes [12, 19]. Till date the studies on BBL were carried out using single controller to stabilize the BBL. This paper presents the comparison of two model-based controllers, i.e., PID and LQR.

The current study focuses on the comparison of two controllers, LQR and PID with an objective to develop an optimal system controller. For previously developed balancing robots, researchers have used only a single controller for the system. The authors could not find a comparison study for the two most popular controllers. The objective of the current study is to develop a prototype of BBL, study the two controllers on the system, and suggest the best out of two for better stability of the BBL.

Mathematical Modeling

A 2D model for the BBL is considered for this study. Mathematical model for two planes i.e., vertical plane (YZ/XZ) and horizontal plane (XY) are used for generating the equations of motion.

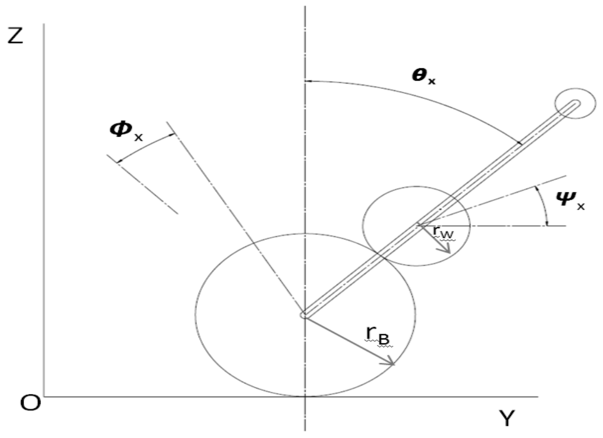


Figure 1. 2D model of the BBL.

Where

r_B	Radius of the ball
r_W	Radius of Omni wheel
I_B	Moment of inertia of the ball
I_W	Moment of inertia of the Omni wheel in the YZ-/XZ-plane
I_{Wxy}	Moment of inertia of the Omni wheel in the XY-plane
I_A	Moment of inertia of the body of the robot in the YZ-/XZ-plane
I_{Axy}	Moment of inertia of the body of the robot in the XY-plane
l	Distance between COM of the ball and COM of the body of the robot

And,
φ_x and φ_y specify the orientation of the ball,
θ_x,θ_y and θ_z specify the orientation of the body and
ψ_x,ψ_y and ψ_z specify the orientation of the virtual actuating wheels.

Mathematical model

Energy in YZ/XZ Plane

Derivation of the kinetic energy and the potential energy of the different parts of the BBL including the equations for the ball, the frame, and the omni-directional wheels were obtained.

The kinetic energy (T) of the ball is given as the summation of translational and rotational energy:

$$T_{B,yz} = \frac{1}{2}.m_B.(r_B^2.\dot{\varphi}_x^2) + \frac{1}{2}.I_B.\dot{\varphi}_x^2 \tag{1}$$

The potential energy (V) of the ball is given by

$$V_B = o \tag{2}$$

The potential energy for the ball is zero as the ball is moving on horizontal surface and therefore has no potential energy.

Similarly, the kinetic and potential energy of the body are given underneath

$$T_{A,yz} = \frac{1}{2}.m_A.(r_B^2.\dot{\varphi}_x^2 + 2.r_B.l.\dot{\varphi}_x.\dot{\theta}_x.cos\theta_x + l^2.\dot{\theta}_x^2 + \frac{1}{2}.I_A.\dot{\theta}_x^2 \tag{3}$$

Whereas the potential energy is

$$V_{A,yz} = m_A.g.l.cos\theta_x \tag{4}$$

Energies of the omnidirectional wheel are denoted as

$$T_{W,yz} = \frac{1}{2}.m_W.[(r_B^2.\dot{\varphi}_x^2 + 2.r_B.(r_B + r_W).\dot{\varphi}_x.\dot{\theta}_x.cos\theta_x + (r_B + r_W)^2.\dot{\theta}_x^2] + \frac{1}{2}.I_W.(\frac{\mathrm{r_B}}{\mathrm{r_W}}.(\dot{\varphi}_x - \dot{\theta}_x))^2 \tag{5}$$

$$V_{W,yz} = m_W.g.(r_B + r_W).cos\theta_x \tag{6}$$

Lagrangian equation for the YZ/XZ Plane is given by summation of kinetic energies for ball, body and omnidirectional wheels and subtracting the summation of potential energies.

$$L(\varphi_x, \theta_x, \dot{\varphi}_x, \dot{\theta}_x) = T_{B,yz} + T_{A,yz} + T_{W,yz} - V_{B,yz} - V_{A,yz} - V_{W,yz} \tag{7}$$

Further to which as the Lagrangian for the YZ/XZ plane is used to find the equation of motions by using Euler Lagrangian equation, which is given by

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\varphi}_x}\right) - \frac{\partial L}{\partial \varphi_x} = \tau_{\varphi_x} \tag{8}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_x}\right) - \frac{\partial L}{\partial \theta_x} = \tau_{\theta_x} \tag{9}$$

The equations of motion are derived from the above Euler Lagrange further to which result is obtain by combining the equations in the form below

$$M(q_{yz})\ddot{q}_{yz} + C(q_{yz}, \dot{q}_{yz})\dot{q}_{yz} + G(q_{yz}) = \tau_{ext} \tag{10}$$

$$M_{yz} = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \tag{11}$$

$$M_{11} = r_B^2[(m_A + m_B + m_W)] + \frac{r_B^2}{r_W^2}.I_W + I_B \tag{12}$$

$$M_{12} = cos(\theta_x(t)).r_B[r_B.m_W + r_W.m_W + l.m_A\ ] - [\frac{r_B^2}{r_W^2}.I_W] \tag{13}$$

$$M_{21} = cos(\theta_x(t)).r_B[r_B.m_W + r_W.m_W + l.m_A] - [\frac{r_B^2}{r_W^2}.I_W] \tag{14}$$

$$M_{22} = l^2.m_A + m_W(r_B + r_W)^2 + \frac{r_B^2}{r_W^2}.I_W + I_A \tag{15}$$

$$C_{yz} = \begin{bmatrix} 0 & -sin(\theta_x(t)).\dot{\theta}_x.r_B(r_B.m_W + r_W.m_B + l.r_B.m_A) \\ 0 & 0 \end{bmatrix} \tag{16}$$

$$G_{yz} = \begin{bmatrix} 0 \\ -sin(\theta_x(t)).g.((r_B + r_W).m_W + l.m_A) \end{bmatrix} \tag{17}$$

State Space

Applying Lagrange equation to the above model the State Space for YZ/XZ Plane is denoted by

$$M_{11}.\ddot{\varphi}_x + M_{12}.\ddot{\theta}_x + C_{12}.\dot{\theta}_x = \tau_{x1} \tag{18}$$

$$M_{21}.\ddot{\varphi}_x + M_{22}.\ddot{\theta}_x + G_2.\theta_x = \tau_{x2} \tag{19}$$

$$V_1 = \dot{\varphi}_x \tag{20}$$

3D Solid model

The 3D model for the BBL is prepared using commercial software (Solid works 2014 developed by Dassault Systems version 22, release date October 7, 2013). This model represents an overview of the actual model of the Ballbot as shown in Figure 2.

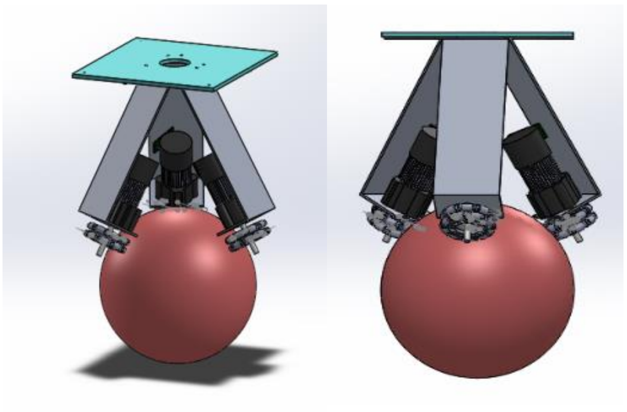


Figure 2. 3D Solid works model showing the BBL in upright position.

$$\dot{V}_1 = \ddot{\varphi}_x \tag{21}$$

$$\dot{V}_1 = (\tau_{x1} - M_{12}.\dot{V}_2 - C_{12}.V_2)/M_{11} \tag{22}$$

$$V_2 = \dot{\theta}_x \tag{23}$$

$$\dot{V}_2 = (\tau_{x2} - M_{21}.\dot{V}_1 - G_2.\theta_x)/M_{22} \tag{24}$$

$$\dot{X} = Ax + Bu \tag{25}$$

$$y = Cx + Du \tag{26}$$

Table 1. Parameters derived from 3D model

Parameter	Description	Value
mb	mass of ball.	0.181[kg]
ma	mass of body	1.64 [kg]
mw	mass of omnidirectional wheel	0.00782 [kg]
rb	radius of ball	0.062 [m]
rw	radius of omnidirectional wheel	0.0225 [m]
l	length of end of body to c.o.g.	0.219 [m]
Ib	moment of inertia of ball	4.63 x 10^-4 [kg-m^2]
Iw	moment of inertia of omnidirectional wheel	1.98 x 10^-5 [kg-m^2]
Ia	moment of inertia of body	5.4 x 10^-3 [kg-m^2]
g	gravitational acceleration	9.81 [m/s/s]

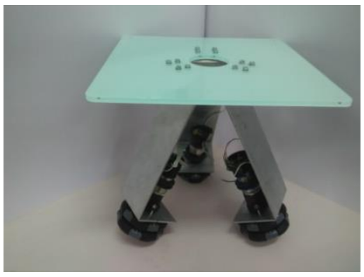
$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -510.6 & 0.9157 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 176.1 & -0.2408 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ -144.7 \\ 0 \\ 49.6 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

A



B

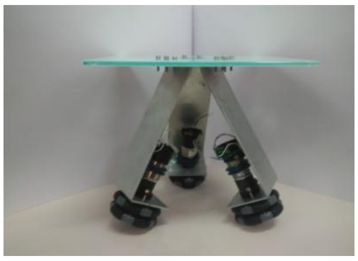


Figure 2. Actual prototype A) Isometric view showing the acrylic plate, B) Front view showing the body and omnidirectional wheel

Controller design

Proportional–Integral–Derivative (PID) controller is a control loop feedback mechanism (controller) commonly used in control systems. A PID controller continuously calculates an error value *e(t)* as the difference between a desired set point and a measured process variable and applies a correction based on proportional (K_p), integral (K_i), and derivative *(Kd)* terms.

The controller tries to minimize the error over time by tuning of a control variable *u(t)*, which is further dependent on coefficients of proportional, integral and derivative terms given by the following formula.

$$u(t) = \overbrace{K_p e(t)}^{\text{Proportional}} + \overbrace{K_i \int_0^t e(\tau)d\tau}^{\text{Integral}} + \overbrace{K_d \frac{d}{dt} e(t)}^{\text{Derivative}}$$

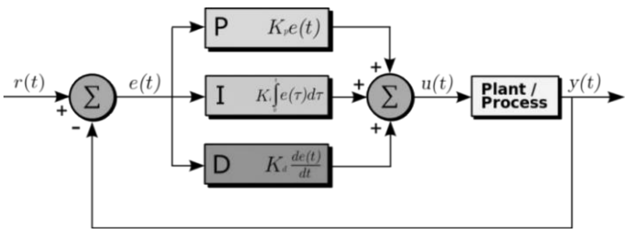


Figure 3. A block diagram of a PID controller in a feedback loop

There are some common methods for obtaining PID control parameters:

Manual Tuning: This method involves manually adjusting the parameters based on the system's response to disturbances. It usually starts with setting the integral and derivative gains to zero (Ki = Kd = 0) and gradually increasing the proportional gain (Kp) until the system starts to oscillate. The integral and derivative gains are then adjusted to reduce oscillations and improve stability. Manual tuning is an iterative process and requires expert knowledge and experience.

Ziegler-Nichols Method: This method provides a systematic approach to tune PID parameters based on the system's response to step inputs. It involves initially setting Ki and Kd to zero and increasing Kp until the system exhibits sustained oscillations. The critical gain value (Kcu) and the corresponding oscillation period (Pu) are noted. Based on these values, the ultimate gain (Ku) and ultimate period (Tu) are calculated. The final PID parameters are obtained by applying specific formulas based on the control type (P, PI, or PID) recommended by Ziegler-Nichols.

Auto-tuning Algorithms: Several automated tuning algorithms, such as the Ziegler-Nichols ultimate gain method, Cohen-Coon method, and others, have been developed to determine PID parameters automatically. These algorithms often involve applying a series of predefined test signals to the system and analyzing the response data to compute optimal PID parameters. Auto-tuning algorithms can save time and effort compared to manual tuning but may not always provide the best results in complex systems.

Model-based Tuning: This method involves developing a mathematical model of the system and using it to determine PID parameters. The model can be derived from system equations or identified from experimental data. Model-based tuning techniques, such as pole placement or optimization-based methods, use the system model's characteristics to calculate PID parameters that meet specific control objectives.

The selection of the tuning method depends on the system's complexity, available resources, time constraints, and the desired control performance. Additionally, tuning PID parameters may require multiple iterations and fine-tuning to achieve the desired system response.

Linear–quadratic regulator (LQR) is a method to define state-feedback control gain matrix$\mathbb{K}$.
In LQR controller two parameters, R and Q, are considered which balances the control effort (u) and error, respectively. The simplest case is to assume R=1 and Q=C'*C.

The LQR method basically allows for the control of both outputs (the body angle and the ball position).
So as the value of Q is given by C'*C, Q is represented by a 4x4 matrix as

$$Q=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}$$

The element (1, 1) in the above matrix denotes the weight on the ball's position and the element (3, 3) denotes the weight on the body's angle. The input weight value R is considered at 1. Now further the value of K which is given by k=lqr (A,B,Q,R) is plotted in the graph shown in Figure 7.

Results and Discussion

The performance of both, PID and LQR controllers for the BBL are presented in this section by comparing the settling time and peak amplitude.

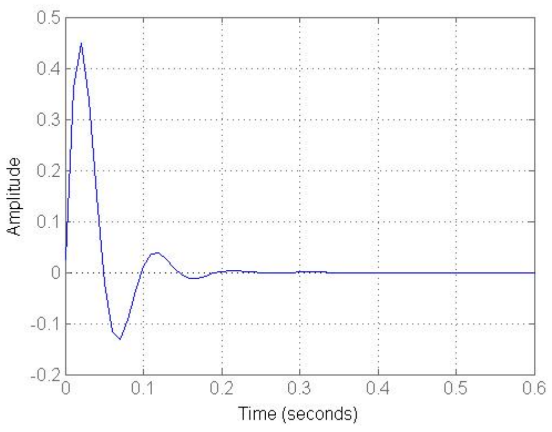


Figure 5. Response of the body position to an impulse disturbance under PID control

The response of the body position to an impulse disturbance under PID control when the values of Kp =100, Ki =1 and Kd =1 is illustrated in Fig. 5.
The amplitude vs. time graph shows that the Settling time for the system is 0.175 seconds and the peak amplitude is 0.449 radians after 0.02 seconds. The settling time of the response is determined to be 0.175 seconds, which is less than 2 seconds and which is well within the accepted limit. As per the literature the limit for settling time for the robot is 2 seconds and tilt angle is 5° [14, 15]. Since the steady-state error approaches to zero in a sufficiently swift manner, no further integral tuning is needed. The peak response, however, is larger than the needed value of 0.08 radians (5ʃ). Therefore, the overshoot can be controlled by increasing and tuning the amount of derivative control. Hence, at Kd=12 proper response is achieved and graph is plotted showing its characteristics as shown in Figure 6.

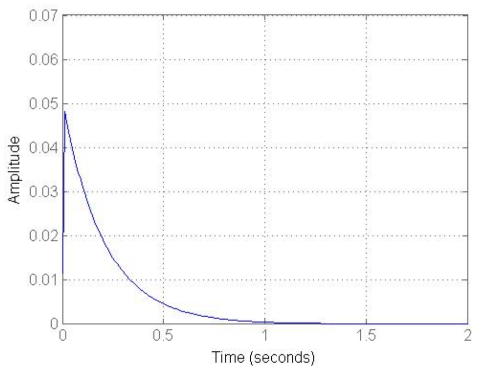


Figure 6. Response of the body position to an impulse disturbance under PID control

Figure 6 depicts the response after changing the derivative control Kd=12, the overshoot has been reduced so that the body does not move more than 5°away from the vertical axis. Additionally, it is observed that the settling time for the system is 0.483 seconds and the overshoot is controlled, as the peak amplitude value is 0.048 radians after 0.01 seconds.

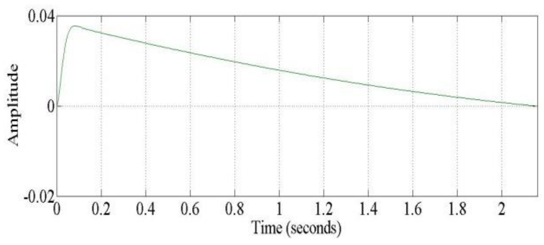


Figure 7. Step response with LQR

Figure 7 illustrates the peak amplitude for LQR controller as 0.025 m which is lower as compared to earlier results of PID controller. However, the time taken to balance the system is 2.25 seconds, which is higher as compared to PID controller.

Table 2. Comparison between PID and LQR controller

Controllers	Rise Time(s)	Settling Time(s)	Peak Amplitude (m)	Peak time(s)
PID	0.004	0.798	0.0481	0.01
LQR	0.004	2.25	0.025	0.1

The comparison in the Table 2 shows that PID controller has more overshoot in the beginning but it controls and balances the system in 0.798 seconds as compared to LQR controller for which overshoot is less but settling time for stabilization is 2.25 seconds. Thus for this system PID controller gives better results for stabilization as compared to LQR controller.

The superiority of the PID controller over the LQR controller in this scenario can be attributed to several factors. Firstly, the PID controller is capable of handling a wide range of system dynamics, making it suitable for diverse applications. In contrast, the LQR controller relies on a linear model that may not accurately capture complex or nonlinear system behaviors.
Secondly, the PID controller offers greater flexibility in parameter tuning. Its proportional, integral, and derivative terms can be adjusted to optimize performance for specific system characteristics. On the other hand, the LQR controller requires careful selection of weighting

matrices, which can be more challenging and may not capture the system dynamics as effectively.

Furthermore, the PID controller demonstrates robustness by effectively handling uncertainties and disturbances through its integral and derivative terms. This adaptability allows the controller to respond to changing conditions in real-time applications. In contrast, the LQR controller may not perform as well in the presence of uncertainties.

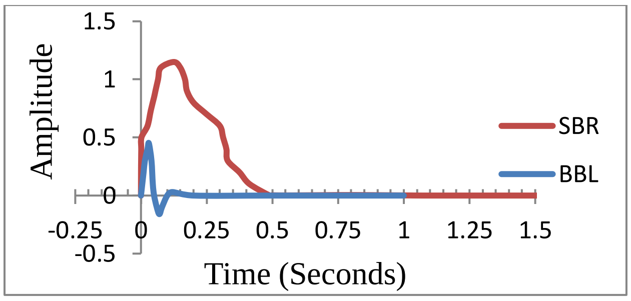


Figure 8. Comparison of tilt angle response using PID controller between SBR (Self-balancing Robot) and BBL

The comparison between BBL and SBR in the aspects of the performance of the control system is shown in Figure 8 [22]. The results show that the response time to get BBL stable is less as compared to SBR. Moreover, PID controller has a higher overshoot in case of SBR. As illustrated by Wei An for a control system on a two wheeled self-balancing robot and using PID controller studied the response performance [22]. Similarly, the current study illustrates that the BBL is more efficient than SBR when compared for stabilization of the robotic system.

Conclusion and Future Work

In this paper, a detailed 2D mathematical model of a ball-balancing robot, named BBL, has been presented. The dynamic model of BBL mobile robot with nonlinear equations has been derived using Lagrange's method. The equations derived are then linearized using the Euler-Lagrangian approach and further analyzed for controller design. The linearized equations have been analyzed to see whether the system is controllable and observable, and can be stabilized. State space model has been derived to get final equations for further assessment.
Comparison between the two model-based controllers, i.e., PID and LQR for balancing the robot has been presented. Experiments have been carried out to test the controllers and the results are presented for stabilization time and swift movement. A comparative study based on the system between PID and LQR illustrates that the PID controller stabilizes the BBL in upright position more efficiently and faster as compared to the LQR controller.
In future work, 3D mathematical modeling will be taken into account and the controllers will be compared in real time.

References

[1] Golem Krang, M. Stilman, J. Olson,and W. Gloss, *"Dynamically stable humanoid robot for mobile manipulation"* in Proc. IEEE Int'l Conf. on Robotics and Automation, 2010,pp. 3304–3309. https://doi.org/10.1109/ROBOT.2010.5509593

[2] Suomela, Tomi Ylikorpi and Jussi **"**Ball-shaped Robots". [book auth.] Houxiang Zhang. *Climbing & Walking Robots, Towards New Applications,* Vienna, Austria: Itech Education and Publishing, 2007, pp. 546. https://doi.org/10.5772/5083

[3] Kumar, Vivek "Self Balancing Bots". Kanpur: IIT, 2012.

[4] H. G. Nguyen, J. Morrell, K. Mullens, A. Burmeister,S. Miles, N. Farrington, K. Thomas, and D. Gage "Segway robotic mobility platform" in SPIE Proc.5609: Mobile Robots XVII,2004. https://doi.org/10.1117/12.571750

[5] Anybots. [Online] 2010. [Cited: November 28, 2014.] https://anybots.com

[6] P. Deegan, B. Thibodeau, and R.Grupen "Designing a self-stabilizing robot for dynamic mobile manipulation" Robotics: Science and Systems - Workshop on Manipulation for Human Environments, 2006. https://doi.org/10.21236/ADA459932

[7] P. Deegan, R. Grupen, A. Hanson, E.Horrell, S. Ou,E. Riseman, S. Sen, B. Thibodeau, A. Williams, and D. Xie "Mobile manipulators for assisted living in residential settings" Autonomous Robots, Special Issue on Socially Assistive Robotics,2008, pp. 24(2):179–192. https://doi.org/10.1007/s10514-007-9061-8

[8] T.B.Lauwers, G.A. Kantor,R.L.Hollis "A Dynamically Stable Single -Wheeled Mobile Robot with Inverse Mouse Ball Drive" 2006.

[9] M. Neunert, P. Fankhauser, S. Leutenegger, C. Pradalier, F. Colas,and R. Siegwart "Ballbot Rezero: Mechanical design, system modeling and control",

2013, IEEE Robotics and Automation Magazine.

[10] M.Kumagai, T.Ochiai “Development of a Robot Balancing on a Ball”,2008.

[11] C.-W. Liao, C.-C.Tsai, Y. Y. Li, and C.-K. Chan “Dynamic modeling and sliding-mode control of a ball robot with inverse mouse-ball drive” Aug. SICE Annual Conference, pp. 2951 – 2955, 2008.

[12] Vijay Ramnath, Swarnim Raizada, “Kugel Balance: Ballbots-the future of personal transport?" 2013, p.3.

[13] Tom Lauwers, George Kantor and Ralph Hollis “One is enough”, 12th International Symposium on Robotics Research, 2005.

[14] M. Kumagai and T. Ochiai “Development of a robot balanced on a ball- first report, implementation of the robot and basic control”, Journal of Robotics and Mechatronics, vol. 22, no. 3, pp. 348–355, 2010. https://doi.org/10.20965/jrm.2010.p0348

[15] Havasi., L. “ERROSphere: an equilibrator robot”, Int’l Conf. on Control and Automation, pp. 971-976,2005.

[16] Ochiai., M. Kumaga and T. “Development of a robot balanced on a ball — Application of passive motion to transport” Int. Conf. Robotics and Automation ICRA ’09., pp. 4106-4111, 2009. https://doi.org/10.1109/ROBOT.2009.5152324

[17] Rezero. [Online] 2010. https://www.rezero.ethz.ch

[18] Ochiai, Masaaki Kumagaind Takaya “Development of a Robot Balanced on a ball-First report, Implementation of the robot and basic control”, Journal of robotics and Mechatronics, 2010. https://doi.org/10.20965/jrm.2010.p0348

[19] J.Fong, S.Uppill “Design and Build a Ballbot” Adelaide, Australia : s.n., 2009.

[20] Wei An and Yangmin Li. “Simulation and control of a two wheeled self-balancing robot”, ROBIO, china DEC2013. https://doi.org/10.1109/ROBIO.2013.6739501

[21] R.Hollis. Ballbots. Scientific American,. [Online] October 23, 2006. [Cited: December 23, 2014.]

[22] Xin Yeoh Jia, “Balancing Ballbot”.Universiti Teknologi Malaysia 2012, p. 99.

Publisher: Chinese Institute of Automation Engineers (CIAE)

ISSN: 2223-9766 (Online)