Developing and Controlling a Raspberry Pi-Powered Two-Wheel Self-Balancing Robot with Python Programming

[1] Dr. Vinod Kumar P, [2]Dr. Kamala N, [3]Dr. Prakash K R

[1] Assistant Professor, Department of EEE, ATMECE
[2]Professor, Department of Electrical Engineering, University of Nevada Los Vegas, USA
[3]Professor, Department of Mechanical Engineering, NIE, Mysuru

Abstract: The paper describes the development and operation of a two-wheeled self-balancing robot with a Raspberry Pi and hub motors. A 3-axis gyroscope and an accelerometer are also included in the system. The paper also presents a complementary filter that can be used to address the issue of gyro drifts. In addition, it explores the design and implementation of LQR and PID control systems on motion linearized by the kinematic and electrical parameters. The results show that the control system can balance properly in proximity to the upright position.

Keywords: Proportional Integral Derivative (PID), Linear Quadratic Regulator(LQR), Two-wheel self-balancing robot (TWSBR).

1. Introduction:

Due to their versatile applications, such as in industrial and commercial settings, two-wheeled and inverted pendulum robots have gained widespread attention. The literature survey seeks to provide a comprehensive overview of recent developments and research findings related to the design, control, and utilization of such devices.

Mobile robots have started to make a transition from military and industrial settings to civilian environments. While some of the most prominent examples of these are vacuum cleaners and robotic dogs, Segway's personal transporter is a fascinating example for people.

This two-wheel, self-balancing robot (TWSBR), despite its inherent mechanical instability, has found applications in law enforcement, tourism, and more. It's fitting to classify the Segway as it relies on sensory capabilities and intellectual controller for maintaining its balance; without these components, the Segway would be unable to remain upright.

Although the Segway has gained recognition as an industrial product, examine on controlling similar mechanical systems has taken various directions.

A TWSBR shares similarities with the inverted pendulum, which has served as a crucial experimental platform for control research and education, as referenced in [1] and [2]. In addition to the Segway's development, there has been a significant body of research on two-wheel self-balancing robots, such as JOE [3] and nBot [4], both of which represent early iterations equipped with inertia sensors, motor encoders, and onboard microcontrollers. Further details can be found on the nBot website [4]. Over time, extensive research has been conducted on control design for these platforms, encompassing classical and linear multivariable control techniques [3], [5], [6], [7], nonlinear backstepping controls [8], [9], fuzzy-neural control [10], and hybrid approaches combining these methods [11]. It is also worth noting related work that addresses the balance of a four-wheeled vehicle on its two side wheels, employing classical control methods [12].

The increasing availability of affordable off-the-shelf sensors and microprocessors has made it easier for academic authors to conduct their studies related to the design and control of mobile robots. Although the former was used for the development of JOE, the latter has started to adopt more modern controllers such as the 68HC11, ARM, and Atmel's ATmega series. For individuals who are interested in learning more about how to develop mobile robots, the open-source software development platform Arduino is a good example. It features a prototyping environment that's based on ATmega processors., making it compatible with a range of off-the-shelf sensors [13]. This versatile platform is increasingly gaining popularity, serving both educational [14] and invention education purposes, with applications spanning from robotics [15], [16] to process control [17], [18], and networked control [19].

In this paper, we present the design, construction, and control of a TWSBR. TWSBR is propelled by two Hub motors as it incorporates a Raspberry Pi, an MPU6050 3-axis gyroscope, and a 3-axis accelerometer sensor for determining its attitude. To address gyro drift issues common in commercial off-the-shelf (COTS) sensors, we employ a filter [20]. Additionally, for the three-axis balancing problem, we consider the significant application of the Kalman filter approach. We adopt two control strategies based on linearized equations of motion for this model: a PID control and LQR design [21][22],[23]. The authors utilized a robust modeling language in their approach to address issues that might arise during the analysis of electrical and kinematic parameters. They were able to achieve a stable upright position by using PID and LQR control. The paper is divided into five sections. The first one covers the introduction and review of the literature. In second section explains about robot's structure and the third section highlights about hardware and software configuration of the robot. The third section gives about the controlling scheme using the Kalman filter, the fourth section talks about balance and control loops and fifth section gives concluding remarks of experiment

2. Two-wheeled Self-balancing Robot Structure

The structure of a self-balancing robot can be classified into three parts: sensors, motor, and motor control, and develop board as shown in figure 1. Section II-A introduces the application and advantage of the sensors on the proposed balancing robot, and how these sensors are employed to obtain measurements of acceleration, distance travelled, and robot tilt angle. Section II-B describes motor selection and control for the balancing robot. Section II-C discusses the reason behind choosing the Arduino develop board, and how it is deployed.

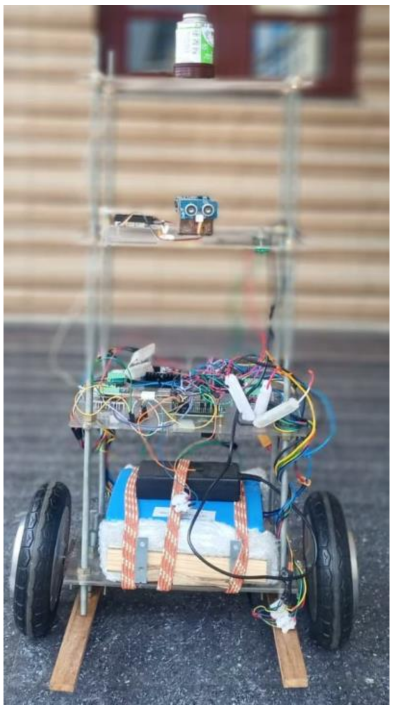


Fig. 1: Balancing Bot

A. Selection and Application of Sensors

To balance the robot, data on the robot’s tilt angle and angular velocity are needed. MPU 6050 has an accelerometer that measures the robot's linear acceleration in three dimensions. This device can be used to manage the robot drive and hold its stability. A gyroscope in the 6050 can measure the movement and rotation of a robot in three dimensions and provide information about its axis and rotation. This data can be utilized to improve the responsiveness of the motors and enable more stable and autonomous operation. The MPU 6050 sensor communicates with the control board of the robot using an I2C interface, allowing it to be easily integrated with other components in the system.

B. Motor and Motor Control Board

Hub motors offer high torque and precise control, making them an ideal choice for self-balancing robots. The advantages of using hub motors are that they can simplify the design of a self-balance robot as it eliminates admits of a separate motor for controlling operation.

Rio Rand Motor Drive provides a range of inputs and outputs that can be used to control and monitor a BLDC hub motor in a self-balancing robot application. These pins provide a high degree of control and flexibility, allowing for precise control of the motor's speed, direction, and braking. Figure 2 shows Rio Rand motor drive connection to the Hub motor.

The author opted for DC Brushless motors equipped with internal sensors, primarily aiming for compatibility with the Raspberry Pi 4. Beyond cost considerations, the research team also factored in power and compatibility aspects. Consequently, they selected two hub motors to meet the demands of 36V and 10A current requirements. Utilizing a speed controller further enhanced the researchers' convenience in managing their robot, with the Rio Rand 350W being an excellent choice for this purpose.

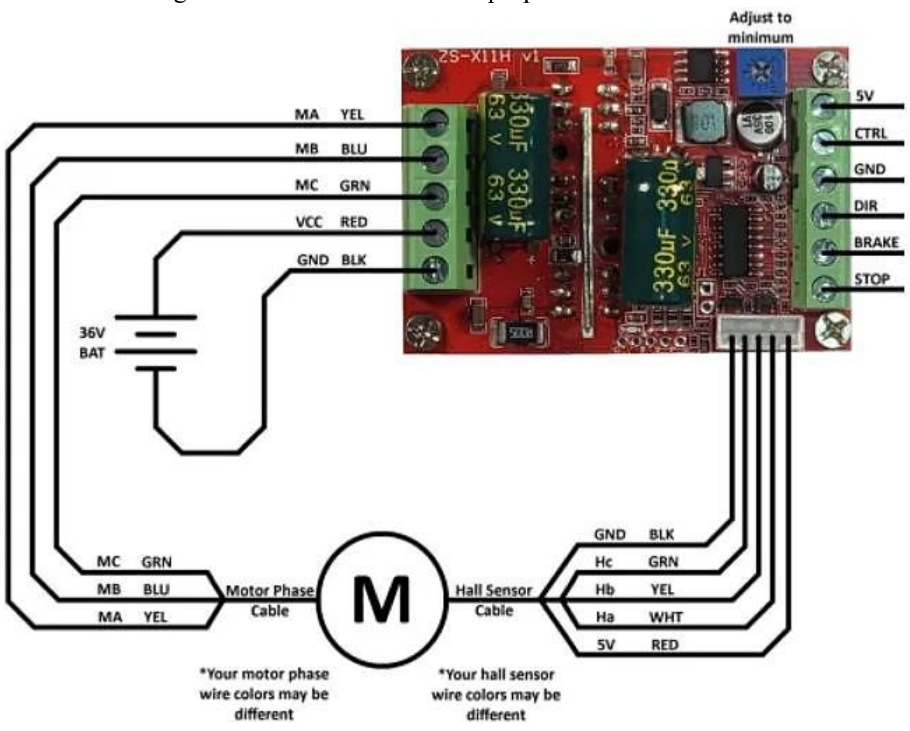


Fig 2: Rio Rand Motor driver connection to Hub motor

C. Raspberry Pi

Constructing a self-balancing robot using a Raspberry Pi involves the connection of diverse components to the computer, encompassing sensors, motor drivers, and various peripherals. The Raspberry Pi can then be programmed to read data from the sensors and use it to control the robot's motors, maintaining its balance and stability. The Pi is ideal for creating a self-balance robot due to its versatile capabilities. It can be programmed with a wide range of programming tools and languages, making it easy for users to modify the device's functions and behaviour. The Pi can also be used to make a variety of sensors and peripherals compatible with its wide range of connectivity. This makes it easy for users to add more features to the device, such as wireless communication and distance sensing.

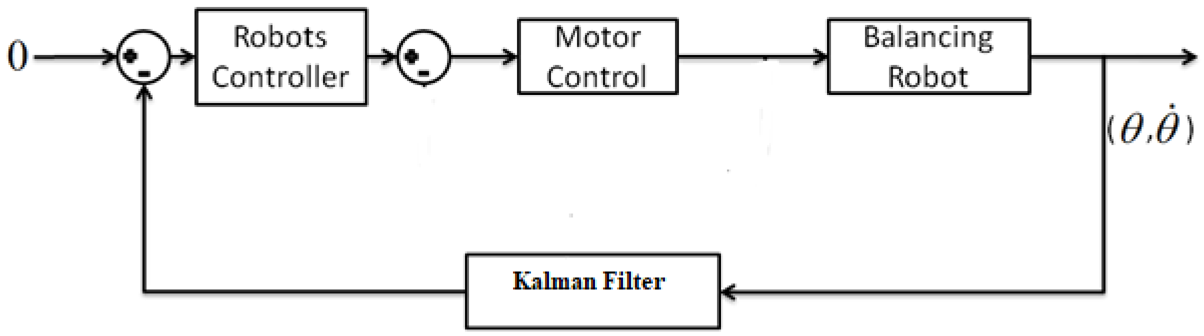


Fig. 3: Architecture of the controlled system

D. Python Programming

The Python code for a self-balancing robot on a Raspberry Pi 4 involves importing necessary libraries, initializing GPIO pins and the MPU6050 sensor, reading sensor data to calculate the robot's angle, and using this information to control the motors. Additionally, ultrasonic sensors can be integrated to detect obstacles, enabling the robot to adapt its behavior accordingly for navigation and collision avoidance.

E. Power

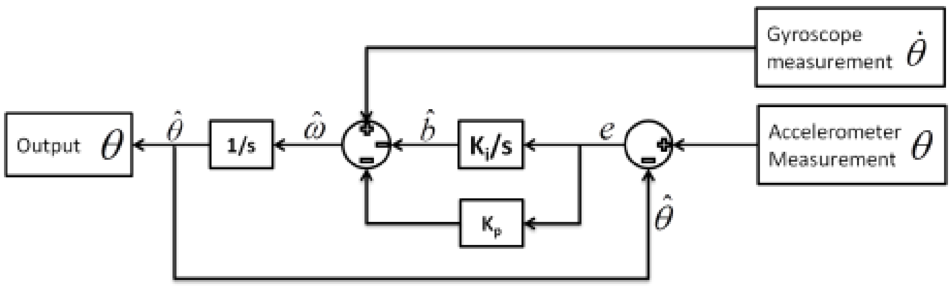
To meet the power requirements of the system, a 36V battery is implemented. A 36V rechargeable lithium battery with a 20AH capacity is selected to power the motors, while voltage regulation is utilized to supply power to other components. The battery's dimensions are 210 x 200 x 70 mm, and it's designed to provide a minimum of 1 hour of operation with a total current requirement of approximately 18.5 amperes, making it a suitable choice for the system.

3. Balancing Control Scheme

The self-balancing robot's system architecture, illustrated in Fig. 3, consists of a forward loop that includes the robot itself and a balancing controller responsible for motor control signals. Feedback is incorporated through a Kalman filter, which serves the purpose of estimating the robot's tilt angle using data from gyroscopes and accelerometers. In the subsequent sections, we will begin by explaining the complementary filter, then move on to the wheel synchronization controller, and conclude with the description of the balancing controller.

In Section II, we utilize a gyroscope to measure the robot's angular velocity, while the Y- and Z-components of gravitational acceleration are captured by an accelerometer. Ensuring precise measurement is essential for accurately correcting the tilt angle θ during the balancing control process. The most straightforward approach involves using the gyroscope to estimate the angle by integrating its angular velocity, but this method is vulnerable to noise, potentially leading to deviations from the correct angle. An alternative method is to directly estimate θ by applying the direction cosine technique to the accelerometer's measurements of gravity components. However, this approach is also susceptible to noise and the robot's acceleration, presenting its own set of challenges.

The author utilized the Kalman filter algorithm to estimate the system's condition. It considers the various uncertainties in the readings and presents a more accurate assessment of the system's overall state. The Kalman filter employs two main steps, namely the prediction and update measures, as shown in Figure 4. The latter updates the system's current state based on the dynamics and the data collected from the sensors. On the other hand, the former updates its status using the error measurements. The Kalman filter can be utilized in various systems, such as robotics and control systems. It can estimate a system's condition by considering the noise from the sensors and the data they collect. For example, it can provide an estimate of the movement velocity and position of the robot based on the data it collects from GPS, IMUs, and other sensors. In a self-balancing robot, the algorithm can also estimate the angle and velocity of the object based on the data collected by the IMU sensor.



θ = measured angle of accelerometer
$\dot{\theta}$ = measured angular velocity of gyroscope
$\hat{\theta}$ = estimated angle of balancing robot
e = estimation error between $\hat{\theta}$ and θ measurement
$\hat{b}$ = estimated gyro bias
$\hat{\omega}$ = estimated angular velocity
k_p = proportional gain
k_i = integral gain

Fig. 4: Block diagram of Kalman Filter

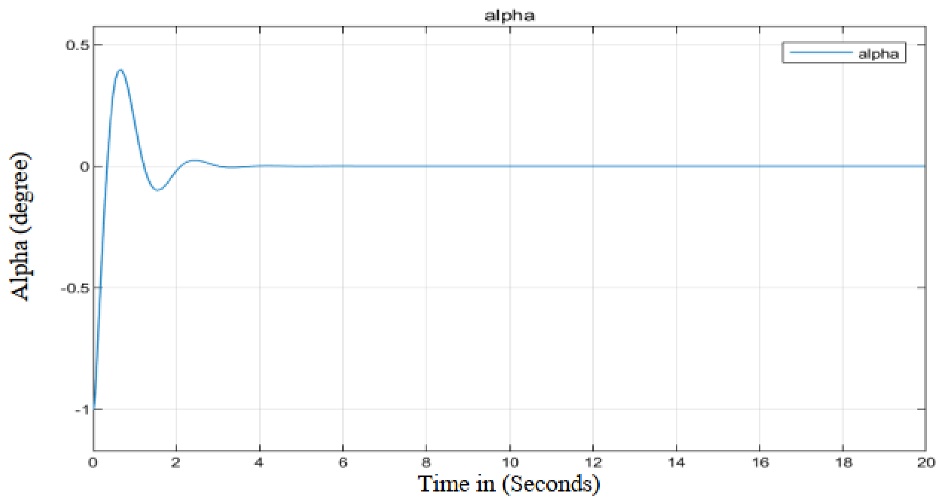


Fig. 5: Robot Pitch Angle of PID controller

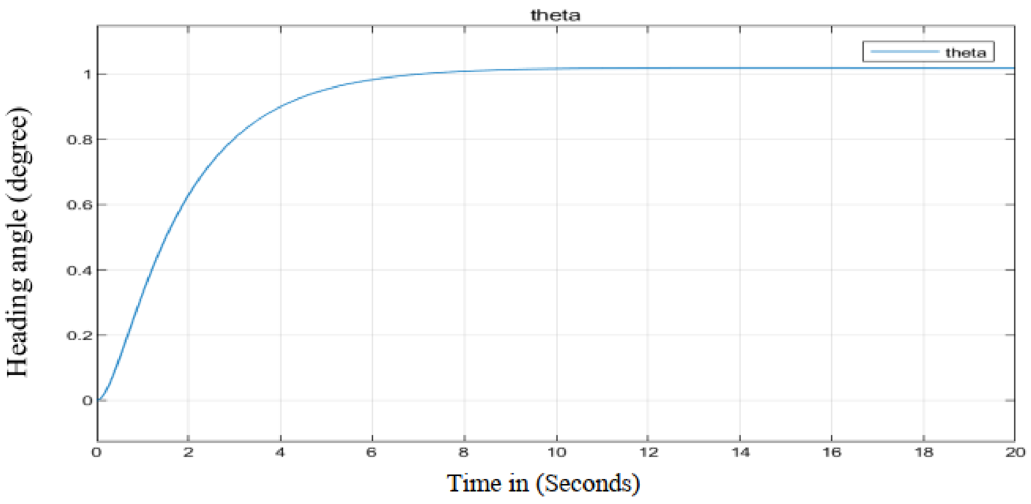


Fig. 6: Robot Heading Angle of PID controller

A Two-Wheeled Self-Balancing Robot employs nuanced motor torque adjustments to uphold stability through pitch angle control and heading angle. Precise PID parameter tuning is vital due to the TWSBR's inherent complexity, ensuring robust responses across pitch, heading, and motion control while effectively countering disturbances for stability in various operational scenarios. Following the adjustment of the filter, its performance in the presence of noise is initially evaluated through simulation, as depicted in Fig. 5. This simulation compares three approaches for estimating θ: direct integration of measured angular velocity leads to divergence, the direction-cosine matrix method yields a stable estimate but is heavily influenced by noise, while the most accurate estimation of the angle θ is achieved through the complementary filter. This consistent outcome is also observed in experimental results, as demonstrated in Fig. 6.

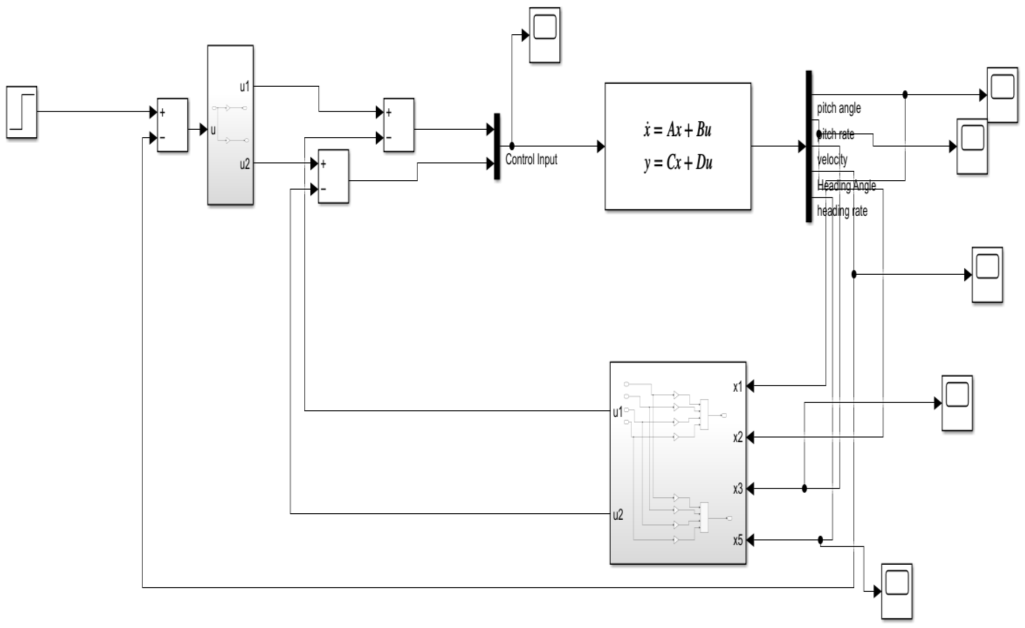


Fig. 7: Control architecture of LQR controller of TWSBR

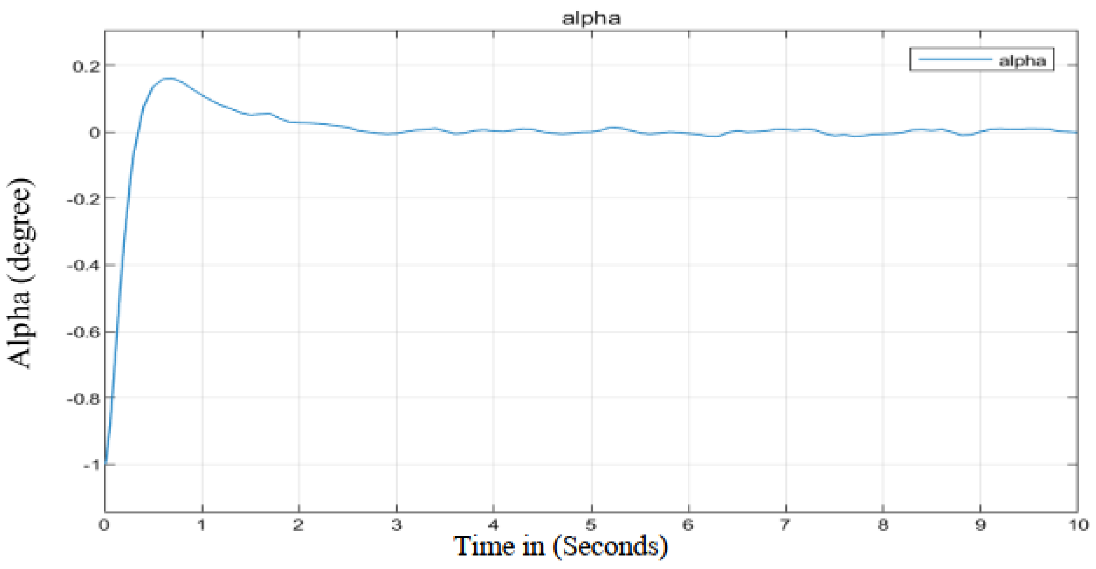


Fig. 8: Robot Pitch angle of LQR controller under disturbance

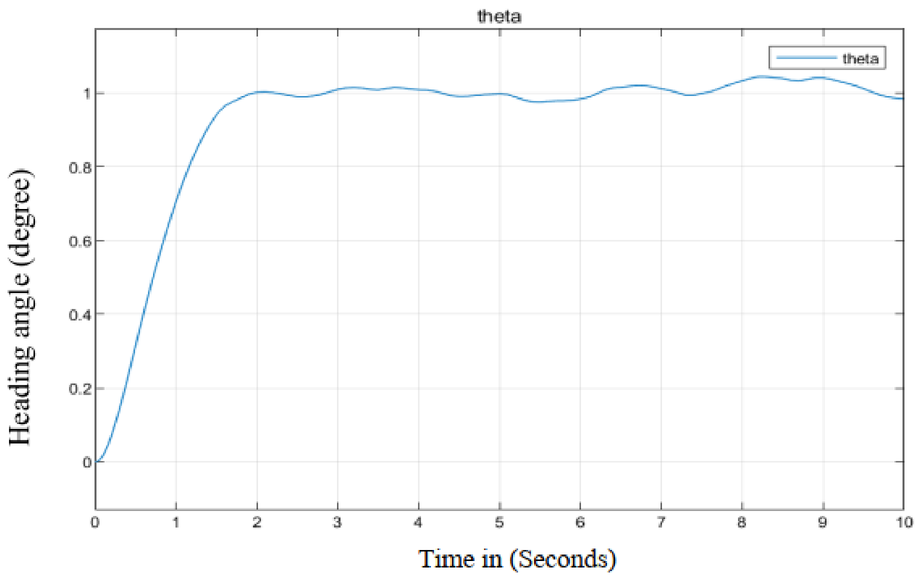


Fig. 9: Robot Heading angle of LQR controller under disturbance

The LQR approach utilized by the proposed controller provided the two output responses for the two-wheel self-balance system. The heading angle and the pitch angle were shown as the system's output values. The value of the pitch angle was programmed to be -1 rad to allow the simulation of the device's performance. It shows that the initial state of a two-wheeled robot is prone to instability. The value of the heading angle's reference signal was set to zero when it was first accessed. The control architecture of the LQR controller is shown below in figure 7.

The computational time of LQR controllers with disturbance input is 2.37 seconds. Incorporating LQR controllers with disturbance handling, the pitch rate and Heading rate control of a robot as depicted in figures 8 and 9. The strategic optimization of motor torques for stable pitch rate responses and precise speed regulation, even in the presence of disturbances. These approaches bolster the robot's stability and controlled motion, enabling it to navigate challenging conditions effectively.

4. Comparison in the experiment

In the experimental comparison between PID and LQR controllers, the results presented indicate that PID control exhibits marginal stability, as angular oscillations surpass the motors' torque limit and become uncontrollable after a few seconds. Conversely, the LQR control demonstrates significantly enhanced stability, mitigating the issue of position drift caused by the center of gravity misalignment and enabling the robot to return to its original position.

5. Concluding Remarks

The concluding section of the study specifies an impression of the key findings and covering up, emphasizing the significance of the investigation and its potential impact. The research was a challenging one, as it required to study various aspects of robotics, including sensing, modeling, and control. The following are some future goals for improvement.

1) **Design improvement:** To enhance the balancing motion, design improvements should focus on optimizing mechanical aspects, which may involve repositioning the center of mass, improving sensor placement, refining the complementary filter to reject disturbances caused by translational motion, and implementing a robust control strategy.
2) **Remote control:** To enhance the robot's capabilities, affordable communication hardware like Xbee can be integrated, enabling remote control of its movements and real-time data retrieval via telemetry. Furthermore, user-initiated actions such as moving forward, backward, turning right, and turning left can also be incorporated.

References

[1] R. Fierro, F. Lewis, and A. Lowe, “Hybrid control for a class of underactuated mechanical systems,” IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, vol. 29, no. 6, pp. 649–4, nov 1999.
[2] K. Xu and X.-D. Duan, “Comparative study of control methods of single-rotational inverted pendulum,” in Proceedings of the First International Conference on Machine Learning and Cybernetics, vol. 2, 2002, pp. 776–8.
[3] F. Grasser, A. D’Arrrigo, S. Colombi, and A. C. Rufer, “JOE: A mobile, inverted pendulum,” IEEE Transactions on Industrial Electronics, vol. 49, no. 1, pp. 107–14, 2002.
[4] D. P. Anderson. (2003, Aug.) nBot balancing robot. Online. [Online].
Available: http://www.geology.smu.edu/ dpa-www/robo/nbot
[5] R. C. Ooi, “Balancing a two-wheeled autonomous robot,” Final Year
Thesis, The University of Western Australia, School of Mechanical Engineering, 2003.
[6] N. M. A. Ghani, F. Naim, and P. Y. Tan, “Two wheels balancing robot with line following capability,” World Academy of Science, Engineering and Technology, vol. 55, pp. 634–8, 2011.
[7] X. Ruan, J. Liu, H. Di, and X. Li, “Design and LQ control of a two-wheeled self-balancing robot,” in Control Conference, 2008. CCC 2008. 27th Chinese, july 2008, pp. 275 –279.
[8] T. Nomura, Y. Kitsuka, H. Suemistu, and T. Matsuo, “Adaptive backstepping control for a two-wheeled autonomous robot,” in ICROS-SICE International Joint Conference, Aug. 2009, pp. 4687–92.
[9] G. M. T. Nguyen, H. N. Duong, and H. P. Ngyuen, “A PID backstep-ping controller for two-wheeled self-balancing robot,” in Proceedings fo the 2010 IFOST, 2010.
[10] K.-H. Su and Y.-Y. Chen, “Balance control for two-wheeled robot via neural-fuzzy technique,” in SICE Annual Conference 2010, Proceedings of, aug. 2010, pp. 2838 –2842.
[11] X. Ruan and J. Cai, “Fuzzy backstepping controller for two-wheeled self-balancing robot,” in International Asia Conference on Informatics in Control, Automation and Robotics, 2009, pp. 166–9.
[12] D. Arndt, J. E. Bobrow, S. Peters, K. Iagnemma, and S. Bubowsky, “Two-wheel self-balancing of a four-wheeled vehicle,” IEEE Control Systems Magazine, vol. 31, no. 2, pp. 29–37, April 2011.
[13] Arduino. [Online]. Available: http://arduino.cc/
[14] J. Sarik and I. Kymissis, “Lab kits using the Arduino prototyping
platform,” in IEEE Frontiers in Education Conference, 2010, pp. T3C– 1–5.
[15] G. Guo and W. Yue, “Autonomous platoon control allowing rangelimited sensors,” IEEE Trans. on Vehicular Technology, vol. 61, no. 7, pp. 2901–2912, 2012.
[16] P. A. Vignesh and G. Vignesh, “Relocating vehicles to avoid traffic collision through wireless sensor networks,” in 4th International Conference on Computational Intelligence, Communication Systems and Networks. IEEE, 2012.
[17] M. J. A. Arizaga, J. de la Calleja, R. Hernandez, and A. Benitez, “Automatic control for laboratory sterilization process rsd on Arduino hardware,” in 22nd International Conference on Electrical Communications and Computers, 130-3, Ed., 2012.
[18] S. Krivic, M. Hujdur, A. Mrzic, and S. Konjicija, “Design and implementation of fuzzy controller on embedded computer for water level control,” in MIPRO, Opatija, Croatia, May 21–25 2012, pp. 1747–51.
[19] V. Georgitzikis and I. Akribopoulos, O.and Chatzigiannakis, “Controlling physical objects via the internet using the Arduino platform over 802.15.4 networks,” IEEE Latin America Transactions, vol. 10, no. 3, pp. 1686–9, 2012.
[20] A.-J. Baerveldt and R. Klang, “A low-cost and low-weight attitude estimation system for an autonomous helicopter,” in IEEE International Conference on Intelligent Engineering Systems, sep 1997, pp. 391–5.
[21] Vinod Kumar P, Dr. Kamala N “Design, Dynamic Modelling, and Control of a Two-wheeled Self-Balancing Robot (TWSBR)”. International Journal of Mechanical Engineering (7) Special Issue 5, 650-668, 2022.
[22] Vinod Kumar P, Dr. Kamala N, “Study of Path Finding and Constraints of Two-wheeled Mobile Robots”, International Journal of Mechanical Engineering (7) Special Issue 5, 501-506, 2022
[23] Vinod Kumar P, Dr. Kamala N, “Design of Robust Controller for Two-wheeled Inverted Pendulum System”, Elsevier Journal: Materials Today: Proceeding. (58) Part-1, 191- 198, 2022.