Self-Balancing Robot Navigation

Hari Maghfiroh [1*], Henry Probo Santoso [2]
[1, 2, 3] Department of Electrical Engineering, Universitas Sebelas Maret, Surakarta, Indonesia
Email: [1] hari.maghfiroh@staff.uns.ac.id, [2] henryprobo140198@student.uns.ac.id
*Corresponding Author

***Abstract*— Human activity has been increasing, to support the activity, people in the modern era create robots to replace some human activities. The interest in two-wheeled balance robots has continued to increase, this is because it is highly maneuverable, making it efficient for use in various areas. In this study, the online navigation of a two-wheeled self-balancing robot is done. The connection between the robot and online navigation is using a Wi-Fi connection. The world model base on the real room is created by Gazebo and then visualized in RVIZ. The map creation and navigation process are handled by the package provided by ROS. The results of the simulation and real tracking show that the robot can move from the starting point to the destination point in either a straight or a curved path. The difference of the final position of the robot between simulation and real tracking is only (15.4 cm, 4 cm) and (9.6 cm, 43 cm) for the straight and curved path. This result proved that online navigation can be used to navigate an autonomous robot without real navigation sensors.**

Keywords—robot, balancing, control, ROS, navigation

NOMENCLATURE

AMCL	Adaptive Monte Carlo Localization
DWA	Dynamic Window Approach
ROS	Robot Operating system
RVIZ	ROS Visualization
URDF	Unified Robotic Description Format
XML	Extensible Markup Language

I. INTRODUCTION

Human activity has been increasing, to support the activity, people in the modern era create robots to replace some human activities [1]. There are many kinds of robots created in the last decade one of them is a mobile robot. The mobile robot can be used in a variety of applications such as exploration, search and rescue, material handling, and entertainment [2]. It also used in industries as wheeled, crawler, and propulsion which are classified according to their movement modes [3].

In recent decades, two-wheeled balance robots are still an interesting area of research [4][5][6][7]. The interest in two-wheeled balance robots has continued to increase, this is because two-wheeled balance robots are able to become an ideal platform for researching all kinds of filter algorithms and control strategies [3], [8], [9]. This robot is a nonlinear, strongly coupled, multivariable control, and naturally unstable system [3][10].

As a two-wheeled balance robot, the robot has the ability to balance itself on two wheels and rotate on the spot, making it highly maneuverable, making it efficient for use in various areas [4], [11][12]. Therefore, [13], states that a two-wheeled balance robot is a good approach to reduce traffic congestion. On top of that, the balancing robot has a wide range of applications. Segway is one of the robot applications, which is well-known, successful, and has been implemented in various urban areas as a means of transportation where one can ride it instead of conventional polluting vehicles [3], [8], [11], [14], [16], [17]. Besides that, there are also Pegasus and iBot [18]. With these various capabilities and advantages, the two-wheeled balance robot can be arranged into a navigation robot capable of working in various terrains with sharp turns and tight spaces [15], [18], [19].

Previous research on two-wheeled balance robot navigation has been carried out. The author of [20] made a home-made robot called Bimbo, which has a navigation function that is controlled by changing system variables via Bluetooth communication. In [21], shows a vision-based balance robot navigation to detect the trajectory. While [22] presented the design and development of a remote-navigated autonomous two-wheeled balance robot. The author of [23] presents an autonomous two-wheeled balance robot using multilevel PID. Mithil *et al.* [13], made an autonomous navigation system by combining an ultrasonic sensor, camera, and lidar which are processed using OpenCV and processing. Li and Zhou [24] mounted an RGB-D camera on a commercial Segway, which was used to navigate autonomously. The authors in [25] and [26] propose the use of the ultrasonic sensor in two-wheel balancing robot which then the robot avoid obstacle base on sensor information. Whereas [27] proposed a line following system to navigate the two-wheel balancing robot. They conclude that by using PID control the robot can balance itself well while following the track given.

Several previous studies regarding navigation using ROS have also been carried out, such as [28], using Mobile Robot Pioneer 3-DX with the ROS system. Okumus and Kocamaz [29] used ROS to navigate multiple robots via a cloud system. While [30] uses the AMCL algorithm to estimate the location and position, A* to determine global path planning, and DWA to determine local path planning. In this paper, mapping and obstacle detection using gmapping, and all these needs are addressed by ROS. In making a two-wheeled balance robot in this study, RVIZ and Gazebo are used, which are tools for visualization and simulation. The Gazebo was used to create a virtual environment and then visualized it in RVIZ [31][32].

In this study, a two-wheeled balance robot was used, with online navigation using Wi-Fi communication between the real robot and the simulation environment. The world model base on the real room is created by Gazebo and then visualized in RVIZ. The map creation and navigation process are handled by the package provided by ROS. The advantage of online navigation is there is no need for real navigation sensors in the robot such as lidar, ultrasonic, and camera. The sensor is used in simulation mode while the real robot follows the command for the simulation. Therefore, the price to make the navigation system can be reduced.

This paper is organized as follows. Section II presents the self-balancing robot and its navigation. In section III, the result and discussion are presented. Finally, the conclusion is in section IV.

II. Self Balancing Robot and Navigation

A. Design of Self Balancing Robot

The making of a two-wheeled balance robot model is made in the Gazebo software. Robot modeling follows the standard format commonly used in ROS, namely the Unified Robotic Description Format (URDF), which is the XML format used in ROS to describe all robot elements.

In general, URDF consists of three xacro files and one gazebo format. The URDF file describes each element of the robot model created. The first file is about the dimensions of the robot, the second file is about the plugins used by the robot, and the third file is about the color of the robot model. Details of the robot model specifications can be seen in Table 1. Fig. 1 shows the robot model in Gazebo.

TABLE I. The Diminetion of Robot Model

Characteristics	Value
Mass	1.11 Kg
Length	13 cm
Height (from ground)	18 cm
Width	6.5 cm
Wheel Diameter	6.8 cm
Wheel Width	2.6 cm
Distance Between the Wheels	14.6 cm

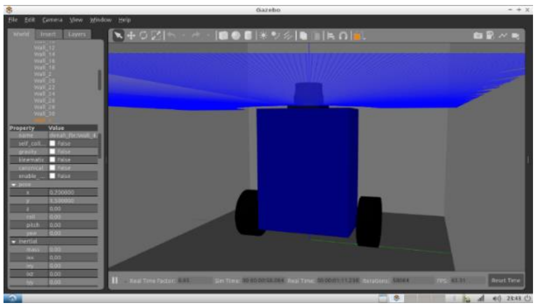


Fig. 1. Model Robot in Gazebo

B. World Model Gazebo

This study uses a world model based on a real room owned by one of the researchers. The real room model is measured and reconstructed in the world model in the Gazebo. Making a world model in the Gazebo uses the building editor feature, by connecting the walls so that they can represent the state of the real room. Fig. 2 shows the building editor features in the Gazebo.

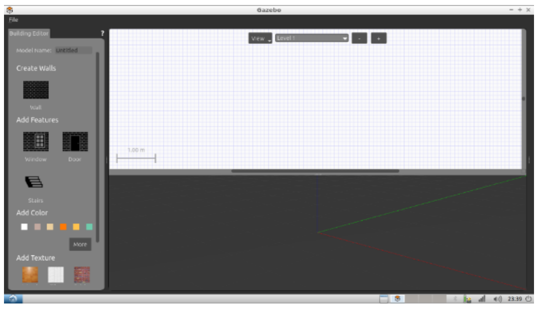


Fig. 2. Building Editor in Gazebo

C. Mapping and Navigation

The map creation and navigation process are handled by the package provided by ROS. To perform the mapping, the gmapping package is used, which by default uses (Simultaneous Localization and Mapping). Fig. 3 shows the flow chart for making a map for navigation. Making maps begins with initiating the Gazebo, initiating RVIZ, launching the gmapping package, and launching the teleoperation package. The teleoperation function makes it possible to perform the movement for the robot by changing the angular or linear velocity of the robot model. By moving the robot model along every corner of the world model, a map of the results of the gmapping function visualized in RVIZ will be formed. When the map is perfectly formed, save the map.

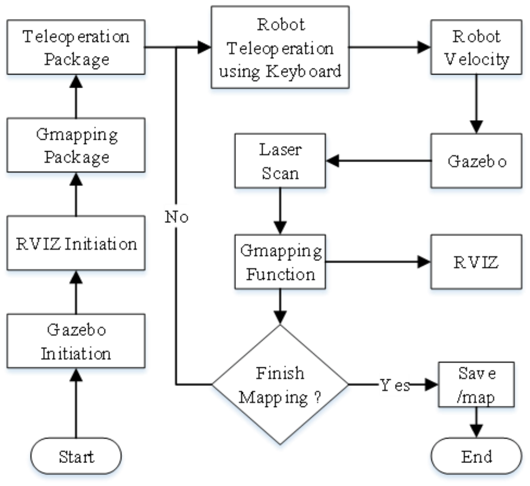


Fig. 3. Mapping Flowchart

The process of making a map and visualizing the map can be seen in Fig. 4. The image on the left shows a mobile robot model with a teleoperation function that is carried out in the Gazebo. The right image shows the map visualized by the Gazebo in RVIZ.

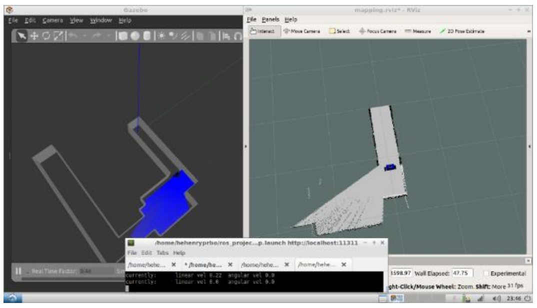


Fig. 4. Mapping Process

ROS also provides navigation packages, one of which is AMCL, which implements the Adaptive Monte Carlo Localization (AMCL) approach, which uses particle filters to track the robot's pose against a known map. As seen in Fig. 5, the robot model in the Gazebo can be navigated by teleoperation using a keyboard or using the 2D Nav Goal function on RVIZ, other methods such as using terminal commands can also be done to navigate the robot.

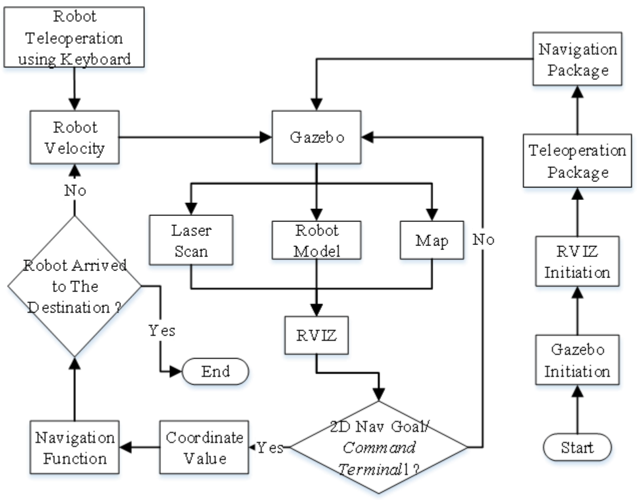


Fig. 5. Navigation Flowchart

To start navigation, it requires Gazebo initiation, RVIZ initiation, launching a teleoperation package, and navigation package. The robot model, data from laser scan, and map data will be visualized in RVIZ. The robot can move to all points on the map manually with the teleoperation function or with the 2D nav goal feature on RVIZ, where we can select points on the map and then the robot will immediately move, or by providing coordinate points via command terminal. The movement of the robot from the starting point to the destination point is influenced by changes in the linear and angular velocity of the robot. This process will continue until the robot reaches its destination point.

D. *Online Navigation*

In the online navigation, the real two-wheel self-balancing robot is connected to the laptop with wireless communication as shown in Fig. 6. The robot is equipped with an IMU sensor, Arduino, and ESP8266 module. The balancing algorithm is processed in the Arduino. Whereas for navigation, the robot is followed online navigation from the Gazebo and RVIZ which run on the laptop. Therefore, if the map of the real room is processed on the laptop, the robot can navigate in the real room correctly.

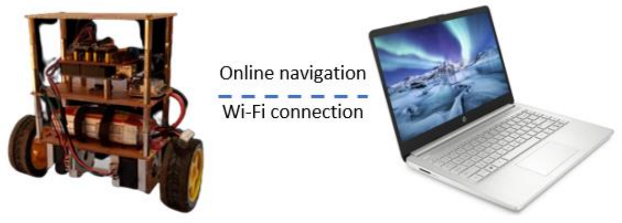


Fig. 6. Online navigation scheme

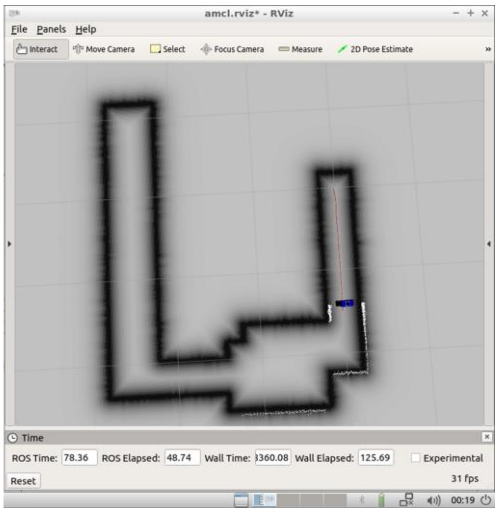
III. Result and Discussion

A. *Tracking simulation*

The tracking simulation test is performed by giving navigation commands via the command terminal to two different destination points (1.5, 0.0) and (2.2, -1), with the same starting point (0, 0). Each destination point is repeated 5 times. The results of the tracking simulation at the first destination point can be seen in Table 2. The robot makes a straight move along one meter from the starting point to the destination point. Robot movement simulation can be seen in Fig. 7.

TABLE II. Tracking Simulation Data on First Destination

No	Destination Coordinate	Final Coordinate	Error X(m)	Error Y(m)
1	(1.5, 0.0)	(1.41, 0.0)	-0.09	0
2	(1.5, 0.0)	(1.40, 0.0)	-0.10	0
3	(1.5, 0.0)	(1.41, 0.0)	-0.09	0
4	(1.5, 0.0)	(1.40, 0.0)	-0.10	0
5	(1.5, 0.0)	(1.41, 0.0)	-0.09	0
	Average Error		-0.094	0.00



ROS Time: 78.36 ROS Elapsed: 48.74 Wall Time: 1360.08 Wall Elapsed: 125.69 Experimental

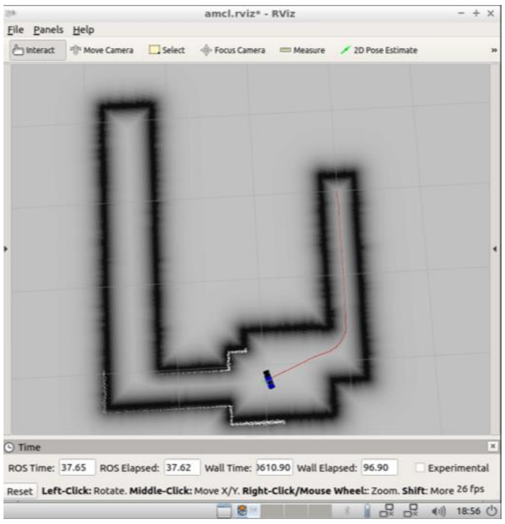
31 fps

Fig. 7. The Displacement to The First Destination Point

The results of the tracking simulation at the second destination point can be seen in Table 3. The robot performs a curved movement from the starting point to the destination point. Fig. 8 shows a simulation illustration of a robot moving curved.

TABLE III. TRACKING SIMULATION DATA ON SECOND DESTINATION

No	Destination Coordinate	Final Coordinate	Error	
			X(m)	Y(m)
1	(2.2, -1)	(2.15, -0.95)	-0.05	-0.05
2	(2.2, -1)	(2.15, -0.95)	-0.05	-0.05
3	(2.2, -1)	(2.14, -0.95)	-0.06	-0.05
4	(2.2, -1)	(2.15, -0.95)	-0.05	-0.05
5	(2.2, -1)	(2.15, -0.95)	-0.05	-0.05
Average Error			-0.052	-0.05



ROS Time: 37.65 ROS Elapsed: 37.62 Wall Time:)610.90 Wall Elapsed: 96.90 Experimental

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More 26 fps

Fig. 8. The Displacement to The Second Destination Point

The results of the tracking simulation show that the robot can move from the starting point to the destination point in either a straight or a curved path. At destination one, there was an error between -0.09 to -0.10 on the X-axis and an error of 0 on the Y-axis. At destination two, there was an error between -0.05 to -0.06 on the X-axis and an error of -0.05 on the Y-axis. The results show that tracking simulations can also be used to track real robots. This is possible by sending the speed data of the simulated robot to the real robot. ROS has a package that supports wireless communication, so what is needed is a real robot capable of receiving data wirelessly. However, it should be noted that the speed data is in the form of linear and angular velocity, so it is necessary to convert the speed data into velocity for each wheel.

B. Hardware Implementation

Online navigation is done after the tracking simulation is successful. Table 4 and Table 5 are shown the result of the online navigation. The coordinate used is the same as the simulation. Table 4 shows that the average error of the X-axis and Y-axis is 0.06 and 0.04, respectively. It means that the delta error of the simulation and real tracking is 0.154 and 0.04 for X-axis and Y-axis, respectively. Since the coordinate unit is in meter; therefore, the difference of final position of the robot between simulation and real tracking is only 15.4 cm and 4 cm for X-axis and Y-axis, respectively.

Table 5 resumes the second coordinate. It informs that, from five experiments, the average error of the X-axis and Y-axis are 0.044 and 0.38, respectively. Where the delta error between simulation and real tracking is 0.096 and 0.43, respectively. This means that the different position between simulation and real is 9.6 cm and 43 cm. The difference occurs since in real tracking there are many factors that affect the robot's movement, for example, the friction between the wheel and the ground.

The small error between simulation and real tracking means that online navigation can be used to navigate the autonomous robot. This method offers a lower price since the real robot does not need any sensor for navigation. However, it only can be done where there is a wireless connection between the robot and the central operator, in this case, a laptop.

TABLE IV. REAL TRACKING DATA ON FIRST DESTINATION

No	Destination Coordinate	Final Coordinate	Error	
			X(m)	Y(m)
1	(1.5, 0.0)	(1.58, 0.05)	0.08	0.05
2	(1.5, 0.0)	(1.55, 0.02)	0.05	0.02
3	(1.5, 0.0)	(1.51, 0.09)	0.01	0.09
4	(1.5, 0.0)	(1.57, 0.03)	0.07	0.03
5	(1.5, 0.0)	(1.59, 0.03)	0.09	0.03
Average Error			0.06	0.04

TABLE V. REAL TRACKING DATA ON SECOND DESTINATION

No	Destination Coordinate	Final Coordinate	Error	
			X(m)	Y(m)
1	(2.2, -1)	(2.21, -1.40)	0.01	0.40
2	(2.2, -1)	(2.30, -1.38)	0.10	0.38
3	(2.2, -1)	(2.25, -1.35)	0.05	0.35
4	(2.2, -1)	(2.22, -1.38)	0.02	0.38
5	(2.2, -1)	(2.24, -1.37)	0.04	0.37
Average Error			0.044	0.38

IV. CONCLUSIONS

The online navigation of a two-wheel self-balancing robot is successfully implemented. The world model base on the real room is created by Gazebo and then visualized in RVIZ. The map creation and navigation process are handled by the package provided by ROS. Online navigation is done using Wi-Fi communication between the robot and simulation in a laptop. The results of the simulation and real tracking show that the robot can move from the starting point to the destination point in either a straight or a curved path. On a straight path, the average error is (-0.094 m, 0 m). Whereas in the curve path the average error is (-0.052 m, -0.05 m). In real tracking, the error for the first path is (0.154 m, 0.04 m). While in the second path is (0.044 m, 0.38 m). The difference of the final position of the robot between simulation and real tracking is only (15.4 cm, 4 cm) and (9.6 cm, 43 cm) for the straight and curved path.

REFERENCES

[1] A. Latif, K. Shankar, P. T. Nguyen, U. Islam, and S. Agung, "Legged Fire Fighter Robot Movement Using PID 1," *J. Robot. Control*, vol. 1, no. 1, pp. 15–18, 2020.

[2] R. Ping, M. Chan, K. A. Stol, and C. R. Halkyard, "Annual Reviews in Control Review of modelling and control of two-wheeled robots," *Annu. Rev. Control J.*, vol. 37, pp. 89–103, 2013.

[3] G. Song, L. Sun, and X. Yang, "Design and Implementation of Self-balancing and Navigation Robot Based on ROS System," *Proc. 31st Chinese Control Decis. Conf. CCDC 2019*, pp. 5597–5602, 2019.

[4] H. Bin, L. W. Zhen, and L. H. Feng, "The Kinematics Model of a Two-wheeled Self-balancing Autonomous Mobile Robot and Its Simulation," in *Second International Conference on Computer Engineering and Applications*, 2010, pp. 64–68.

[5] C. Hsu, C. Su, W. Kao, and B. Lee, "Vision-Based Line-Following Control of a Two-Wheel Self-Balancing Robot," in *2018 International Conference on Machine Learning and Cybernetics (ICMLC)*, 2018, pp. 319–324.

[6] H. Ferdinando, H. Khoswanto, and S. Tjokro, "Design and Evaluation of Two-wheeled Balancing Robot Chassis," in *International Conference on Communications, Computing and Control Applications (CCCA)*, 2011.

[7] Nawawi;, Ahmad;, and Osman;, "Real-Time Control System for a Two-Wheeled Inverted Pendulum Mobile Robot," *Adv. Knowl. Appl. Pract.*, 2010.

[8] S. Xin, M. Gong, Y. Sun, and Z. Zhang, "Control system design for two-wheel self-balanced robot based on Fuzzy-PD control," *5th Int. Conf. Intell. Control Inf. Process. ICICIP 2014 - Proc.*, pp. 169–174, 2015.

[9] G. R. Yu, Y. K. Leu, and H. T. Huang, "PSO-based fuzzy control of a self-balancing two-wheeled robot," *IFSA-SCIS 2017 - Jt. 17th World Congr. Int. Fuzzy Syst. Assoc. 9th Int. Conf. Soft Comput. Intell. Syst.*, pp. 1–5, 2017.

[10] R. Zhang, G. Xiong, C. Cheng, X. Shang, Y. Ma, and Z. Lu, "Control System Design for Two-Wheel Self-Balanced Robot Based on the Stepper Motor," in *Proceedings of 2013 IEEE International Conference on Service Operations and Logistics, and Informatics*, 2013, pp. 241–244.

[11] T. A. Mai, D. N. Anisimov, T. S. Dang, and E. Fedorova, "Fuzzy-PID Controller for Two Wheels Balancing Robot Based on STM32 Microcontroller," *Proc. - 2019 Int. Conf. Eng. Technol. Comput. Sci. Innov. Appl. EnT 2019*, pp. 20–24, 2019.

[12] R. C. Ooi, "Balancing a Two-Wheeled Autonomous Robot," The University of Westerm Australia, 2003.

[13] K. M. Mithil, G. S. Thejas, S. K. Ramani, and S. S. Iyengar, "A Low Cost Multi Sensorial Data Fusion for High Speed Obstacle Avoidance Using 3-D Point Clouds and Image Processing in Self Balancing Robots," *2017 2nd Int. Conf. Emerg. Comput. Inf. Technol. ICECIT 2017*, pp. 1–9, 2018.

[14] J. Wu and W. Zhang, "Design of fuzzy logic controller for two-wheeled self-balancing robot," *Proc. 6th Int. Forum Strateg. Technol. IFOST 2011*, vol. 2, pp. 1266–1270, 2011.

[15] J. Wu, Y. Liang, and Z. Wang, "A robust control method of two-wheeled self-balancing robot," *Proc. 6th Int. Forum Strateg. Technol. IFOST 2011*, vol. 2, pp. 1031–1035, 2011.

[16] H. Ferdinando, H. Khoswanto, D. Purwanto, and S. Tjokro, "Design and evaluation of two-wheeled balancing robot chassis: Case study for Lego bricks," *INISTA 2011 - 2011 Int. Symp. Innov. Intell. Syst. Appl.*, pp. 514–518, 2011.

[17] P. Govardhan, A. Thakre, N. Shende, N. Phadnis, and S. Muley, "Survey on Self Balancing Two Wheel Electric Prototype," *Int. J. Eng. Res. Gen. Sci.*, vol. 5, no. 5, pp. 32–36, 2017.

[18] H. Ying, G. Yang, H. Lan, and L. Yunbo, "Design and implementation of a multifunctional self-balancing mobile platform," *Proc. 2015 27th Chinese Control Decis. Conf. CCDC 2015*, pp. 5693–5697, 2015.

[19] V. Coelho, S. Liew, K. Stol, and G. Liu, "Development of a Mobile Two-Wheel Balancing Platform for Autonomous Applications," in *15th International Conference on Mechatronics and Machine Vision in Practice*, 2008, pp. 2–4.

[20] R. S. Martins and F. Nunes, "Control System for a Self-Balancing Robot," in *4th Experiment@ International Conference*, 2017, no. Project I, pp. 297–302.

[21] C. F. Hsu, C. T. Su, W. F. Kao, and B. K. Lee, "Vision-Based Line-Following Control of a Two-Wheel Self-Balancing Robot," *Proc. - Int. Conf. Mach. Learn. Cybern.*, vol. 1, pp. 319–324, 2018.

[22] O. Y. Chee and M. S. B. Zainal Abidin, "Design and development of two wheeled autonomous balancing robot," *SCOReD 2006 - Proc. 2006 4th Student Conf. Res. Dev. "Towards Enhancing Res. Excell. Reg.*, no. SCOReD, pp. 169–172, 2006.

[23] E. Philip and S. Golluri, "Implementation of an Autonomous Self-Balancing Robot Using Cascaded PID Strategy," in *6 th International Conference on Control, Automation, and Robotics*, 2020, pp. 74–79.

[24] C. H. G. Li and L. P. Zhou, "Training end-to-end steering of a self-balancing mobile robot based on RGB-D image and deep convNet," *IEEE/ASME Int. Conf. Adv. Intell. Mechatronics, AIM*, vol. 2020-July, pp. 898–903, 2020.

[25] X. Ruan and W. Li, "Ultrasonic Sensor Based Two-wheeled Self-balancing Robot Obstacle Avoidance Control System," in *Proceedings of 2014 IEEE International Conference on Mechatronics and Automation*, 2014, pp. 896–900.

[26] C. Iwendi, M. A. Alqarni, J. H. Anajemba, A. S. Alfakeeh, Z. Zhang, and A. K. Bashir, "Robust Navigational Control of a Two-Wheeled Self- Balancing Robot in a Sensed Environment," *IEEE Access*, vol. 7, 2019.

[27] N. Maniha, A. Ghani, F. Naim, and T. P. Yon, "Two Wheels Balancing Robot with Line Following Capability," *Int. J. Mech. Mechatronics Eng.*, vol. 5, no. 7, 2011.

[28] S. Gatesichapakorn, J. Takamatsu, and M. Ruchanurucks, "ROS based Autonomous Mobile Robot Navigation using 2D LiDAR and RGB-D Camera," *2019 1st Int. Symp. Instrumentation, Control. Artif. Intell. Robot. ICA-SYMP 2019*, pp. 151–154, 2019.

[29] F. Okumus and A. F. Kocamaz, "Cloud based indoor navigation for ros-enabled automated guided vehicles," *2019 Int. Conf. Artif. Intell. Data Process. Symp. IDAP 2019*, pp. 1–4, 2019.

[30] R. Kannan Megalingam, C. Ravi Teja, S. Sreekanth, and A. Raj, "ROS based Autonomous Indoor Navigation Simulation Using SLAM Algorithm," *Int. J. Pure Appl. Math.*, vol. 118, no. 7, pp. 199–205, 2018.

[31] R. K. Megalingam, A. Rajendraprasad, and S. K. Manoharan, "Comparison of Planned Path and Travelled Path Using ROS Navigation Stack," in *International Conference for Emerging Technology (INCET)*, 2020, pp. 1–6.

[32] K. Takaya, T. Asai, V. Kroumov, and F. Smarandache, "Simulation Environment for Mobile Robots Testing Using ROS and Gazebo," in *20th International Conference on System Theory, Control and Computing (ICSTCC)*, 2016, pp. 96–101.