IMU Sensor Based Omnidirectional Robot Localization and Rotary Encoder

Setyo Budi Marwanto [1], Riky Dwi Puriyanto [2,*]
[1,2] Department of Electrical Engineering, Universitas Ahmad Dahlan, Yogyakarta, Indonesia
Email: [1] setyo1700022070@webmail.uad.ac.id, rikydp@ee.uad.ac.id
*Corresponding Author

Abstract— **Localization is a technique to determine the position of the robot in an environment. Robot positioning is a basic problem when designing a mobile robot. If the robot does not know its position, then the next robot action will be difficult to determine. To be able to determine the position of the omnidirectional robot requires good speed control on the DC motor. In omnidirectional robots, positioning is through the use of a rotary encoder sensor to count the movement of the omni robot at X and Y coordinates and the IMU sensor to maintain the direction of the robot facing. PID control is also applied to control the rotational speed of each DC motor on the robot wheel. Odometry is the method used in this study. The odometry system aims to estimate the position relative to the initial position of the omni robot to estimate the change in position from time to time. The final result of this research is the application of the odometry method based on a rotary encoder and IMU sensor can produce an effective and stable robot motion and can move in all directions (holonomic) by maintaining the robot's facing direction. The results of the test form simple motions such as forward, backward, right side, and left side motion, as well as forming a box trajectory that has a position error that is not large and quite accurate. The average error value at coordinate X is 1.44 cm and at coordinate Y is 1.67 cm.**

***Keywords*— Localization, Omnidirectional, Odometry**

I. Introduction

Localization is a technique used to determine the position of the robot in the environment [1]-[2]. Omniwheel robot is a type of mobile robot that is capable of moving in any direction depending on the speed of each wheel [3]-[4]. The application of Omnidirectional wheels allows the robot to convert from a non-holonomic robot to a holonomic robot [5]. Movement in indoor or outdoor environments is the most important task that is impossible without proper localization [4].

A mobile robot with holonomic locomotion can also move in any way on the ground plane without changing the orientation of the robot [6]. With rotary speed input from each omnidirectional wheel, the mobile robot can easily move wherever the user wants it to [7].

Robot positioning is a basic problem when designing mobile robots [6], [8]-[9], [1], [10]. If the robot does not know where its position is, then the robot's next action will be difficult to determine [11]-[12]. In omnidirectional robots, common positioning is through the use of rotary encoder sensors in order to obtain the acceleration of rotation of each wheel [13]-[14], as well as a digital compass sensor in the form of a gyroscope in understanding the direction of the robot's face [15]. This direction and speed value will be processed in such a way as to obtain the position of the robot.

Therefore, this research will design the automatic positioning of the robot through the utilization of the odometry system on the robot by using a rotary encoder that functions to detect the number of wheel rotations and the IMU sensor to reach the destination with good positional accuracy.

II. Method

In this research using the odometry method. The odometry system aims to estimate the position relative to the initial position of the omni robot to estimate the change in position over time. The IMU sensor is added to maintain the robot's facing direction [15]. PID control is also applied to control the rotational acceleration of each DC motor on the robot wheel [16]-[17].

A. System Design

The design of the system in research with the title Localization of Omni-Directional Robots Based on IMU Sensors and Rotary Encoder includes the driving part, the physical part or the robot body and the control system and the electronic circuit part. The system block diagram can be seen in Fig. 1.

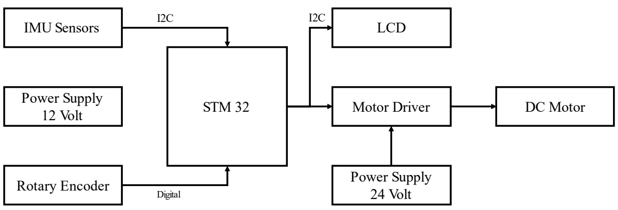

Fig. 1. Block diagram of the system

The STM32 platform works as a system input and output provider that functions to send and receive data from sensors and robot actuators. In addition to providing input and output ports, the STM32 platform also functions to process data from the output of each sensor which becomes a feedback value for the robot's PID control system and robot navigation. The IMU sensor in the block diagram of Fig. 1 is used to obtain the IMU value as a reference point for the direction of the robot and also as a setpoint value for feedback on the PID control of the robot's motion direction which value will be fed to the STM32 platform as the main control of the robot. The movement of the robot can be calculated using a rotary encoder sensor by converting distance units into pulse units so that it can be the setpoint value of PID control and navigation with the odometry method.

B. Algorithm

The flow of how this system works is that it begins with initializing the system and ensuring that the system is alive and can be used, then the robot will receive input in the form of the position the robot wants to go to (X, Y). Then the input will be processed and processed through inverse kinematic which is a matrix formula to determine linear velocity along with motor speed to get motor speed that already has PID control and responds when there is a disturbance in the motor. After obtaining the data from inverse kinematic processing, if the speed can meet the specified (X, Y) value, the robot will stop, but if the robot has not fulfilled it, the robot will return to check the input processed in inverse kinematic. The flowchart of how this system works can be seen in Fig. 2.

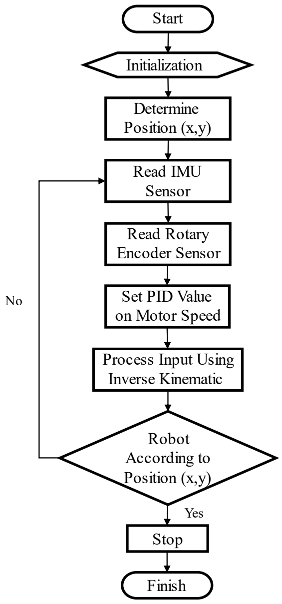

Fig. 2. System flow chart

C. PID Controller

PID (Proportional Integral Derivative) is a controller that is used to select the accuracy of an instrument system and has feedback characteristics in the system. The PID controller component has three types of parameters namely: Proportional, Integrative and Derivative. The three parameters can be used together or individually depending on the response we want to a plant [18]. PID control can be seen in Fig. 3.

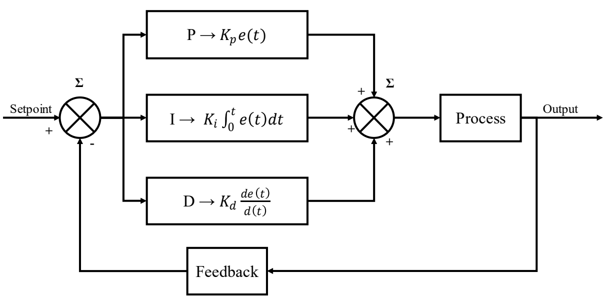

Fig. 3. Block diagram of the PID control

D. Kinematics

In this research, the robot motion system used uses inverse kinematic which functions to get the speed on each wheel with the known position coordinates (X, Y, θ) of the robot. The implementation of inverse kinematic robot movement can be seen in Fig. 4.

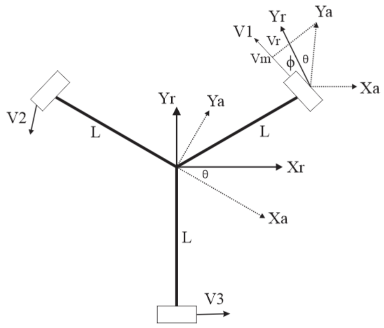

Fig. 4. Kinematics diagram of a three-wheel omnidirectional robot

This omnidirectional robot can move in any direction with kinematical equations. It can be seen that [Xa, Ya] is the world coordinate, then [Xr, Yr] is the representation of the robot frame displacement with respect to the robot center point, then θ is the positive rotation direction angle in the clockwise direction, then L is the distance between the robot center point and the center point in each wheel and [V1, V2, V3] is the translational speed in each wheel. After that, the speed of each wheel can be described in vector-matrix form as follows [11].

$$\begin{bmatrix} V1 \\ V2 \\ V3 \end{bmatrix} = P(\theta).\begin{bmatrix} Xa \\ Ya \\ \theta \end{bmatrix} \tag{1}$$

Where,

$$P(\theta) = \begin{bmatrix} -\sin(\theta + 30^o) & \cos(\theta + 30^o) & L \\ -\sin(30^o - \theta) & -\cos(30^o - \theta) & L \\ \cos\theta & \sin\theta & L \end{bmatrix} \tag{2}$$

In addition, to get the robot speed orientation through other equations as follows.

$$\begin{bmatrix} Xa \\ Ya \\ \theta \end{bmatrix} = P^{-1}(\theta).\begin{bmatrix} V1 \\ V2 \\ V3 \end{bmatrix} \tag{3}$$

Where,

$$P(\theta) = \begin{bmatrix} -\frac{2}{3}\sin(\theta + 30^o) & \frac{2}{3}sin(\theta - 30^o) & \frac{2}{3}\cos\ \theta) \\ \frac{2}{3}\cos(\theta + 30^o) & -\frac{2}{3}\cos(\theta - 30^o) & \frac{2}{3}\sin\ \theta) \\ 1/L & 1/L & 1/L \end{bmatrix} \tag{4}$$

To calculate the rotary pulse into cm, the following equation is needed:

$$Wheel\ circumference\ = \pi 2r \tag{5}$$

$$S\ =\ R \times Wheel\ circumference\ \times ppr \tag{6}$$

III. Result and Discussion

A. BNO055 IMU Sensor Testing

This test is carried out to test the ability of the IMU sensor to read the rotating angle [19]. This test is carried out to determine the direction of the robot's face by rotating the IMU sensor on a certain axis, then the results of the sensor rotation are observed changes in the output value of the IMU sensor after calibration [20]. BNO055 IMU sensor testing can be seen in Fig. 5.

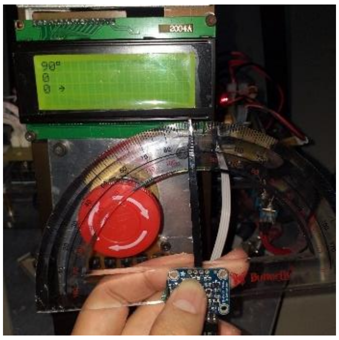

Fig. 5. BNO055 IMU Sensor Testing

Fig. 5 shows the IMU sensor readings displayed in the 20x4 LCD. Table 1 is the test result of the output value of the IMU sensor reading against the actual angle.

Table 1. BNO055 IMU sensor test results

No	Voltage		Angle on arc (˚)	Measured Value on IMU Sensor	Error
	Input (Volt)	Output (Volt)			
1	12.5	5.15	0	0.2	0.2
2	12.5	5.15	10	10.3	0.3
3	12.4	5.16	20	20.1	0.1
4	12.4	5.17	30	30.5	0.5
5	12.3	5.16	40	40.3	0.3
6	12.3	5.16	50	50.3	0.3
7	12.2	5.16	60	60	0
8	12.2	5.16	70	70.3	0.3
9	12.1	5.16	80	80.2	0.2
10	11.9	5.16	90	90	0
		Average error			**0.22**

In Table 1, it can be seen that the input voltage on the IMU sensor of 12 volts can produce an average output voltage of 5.15 volts and has an error value of 0.22, which indicates that the IMU sensor can function properly to be used as a reference for the robot's facing direction.

B. Testing PID Control System Using Matlab

In the PID testing process using matlab software, one formula will be used as a model of the transfer function. The following is the formula for the DC motor PID transfer function model used.

$$\frac{2.181}{s^2 + 99.64s + 40.06} \tag{7}$$

Equation 7 is used as a transfer function model to find the values of KP, KI, and KD. From the results that appear will produce a graph. The resulting graph will be analyzed to see the value of rise time, settling time, over shoot, and also steady state error. Then several PID value tests will be carried out, from testing different KP, KI, and KD values. The following are the results of testing the PID value in matlab shown in Fig. 6, Fig. 7, Fig. 8, Fig. 9.

- RiseTime : 6.8401
- SettlingTime : 18.6553
- Overshoot : 4.2861
- Steady state error : 6.5932e-04

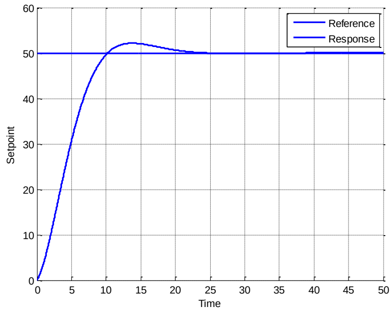

Fig. 6. Value of KP = 2.2; KI = 4.5; KD = 1

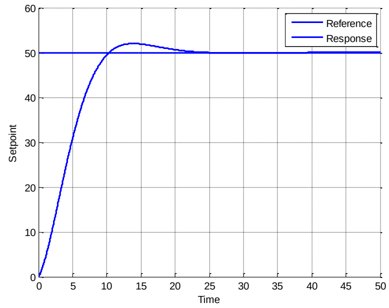

Fig. 7. Value of KP = 2.5; KI = 4.5; KD = 1.2

- Rise Time : 6.9181
- Settling Time : 18.5396
- Overshoot : 3.9692
- Steady state error : 6.8646e-04

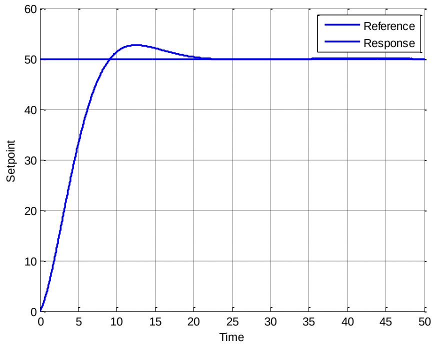

Fig. 8. Value of KP = 2.5; KI = 5; KD = 1

- RiseTime : 6.2285
- SettlingTime : 17.7988
- Overshoot : 5.3905
- Steady state error : 3.1923e-04

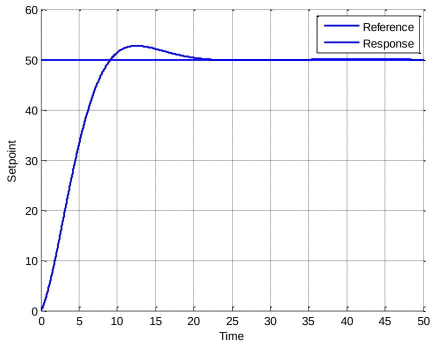

Fig. 9. Value of KP = 2.5; KI = 5; KD = 1.2

- RiseTime : 6.2293

- SettlingTime : 17.8421
- Overshoot : 5.4538
- Steady state error : 3.2940e-04

C. Testing PID Control System on Robot

Motor PID control testing aims to obtain the constant values of Kp, Ki, and Kd which will be used for PID control. The effect of PID control on motor rotation speed is also observed, with good PID control it is expected that the motor can maintain the motor rotation speed without being affected by the load and battery conditions so that it is more adaptive to all conditions. In this test, each motor is given PID control in order to maintain the motor rotational speed within a predetermined rpm. The 50-rpm set test with some PID tuning has previously been tested with matlab software with a comparison of different Kp, Ki, and Kd constant values without load. The following are the results of testing the PID control system on the robot.

Fig. 10 is a test with a set-rpm value setting of 50, a proportional constant value set at 2.2, an integral constant with a value of 4.5 and a derivative constant with a value of 1. The vertical line is the speed while the horizontal line is the time. After the analysis can be seen the average error value on each motor, the average error on motor 1 is 1.86, the average error on motor 2 is 1.94 and the average error on motor 3 is 1.98.

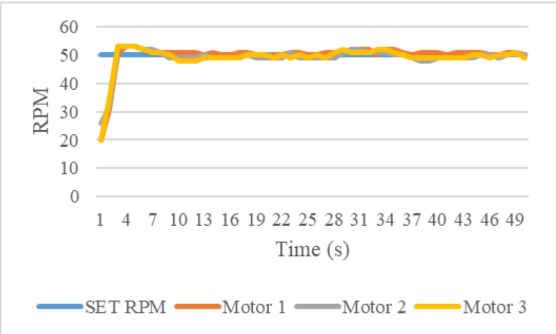

Fig. 10. Test graph with Set-RPM = 50, KP = 2.2; KI = 4.5; KD = 1

Fig. 11 is a test with a set-rpm value of 50, a proportional constant value of 2.5, an integral constant value of 4.5 and a derivative constant value of 1.2. The vertical line is speed while the horizontal line is time. After the analysis can be seen the average error value on each motor, the average error on motor 1 is 2.04, the average error on motor 2 is 1.84 and the average error on motor 3 is 2.02.

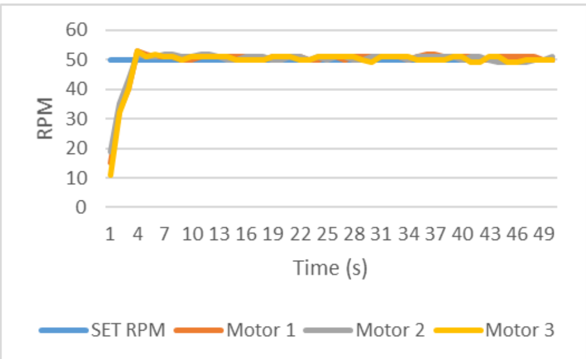

Fig. 11. Test graph with Set-RPM = 50, KP = 2.5; KI = 4.5; KD = 1.2

Fig. 12 is a test with a set-rpm value setting of 50, a proportional constant value set at 2.5, an integral constant with a value of 5 and a derivative constant with a value of 1. The vertical line is the speed while the horizontal line is the time. After the analysis can be seen the average error value on each motor, the average error on motor 1 is 1.82, the average error on motor 2 is 1.82 and the average error on motor 3 is 1.84.

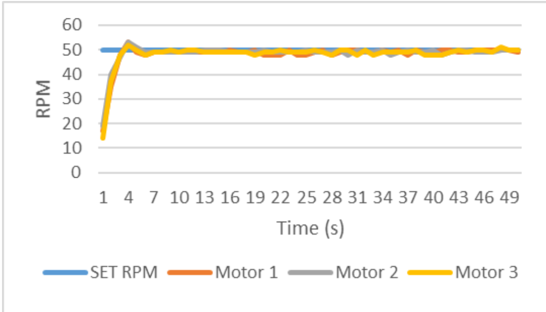

Fig. 12. Test graph with Set-RPM = 50, KP = 2.5; KI = 5; KD = 1

Fig. 13 is a test with a set-rpm value of 50, a proportional constant value of 2.5, an integral constant value of 5 and a derivative constant value of 1.2. The vertical line is speed while the horizontal line is time. After the analysis can be seen the average error value on each motor, the average error on motor 1 is 2.66, the average error on motor 2 is 2.58 and the average error on motor 3 is 2.18.

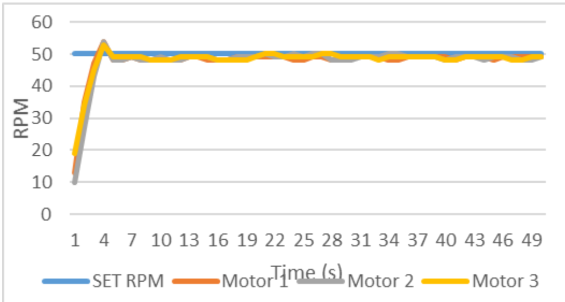

Fig. 13. Test graph with Set-RPM = 50, KP = 2.5; KI = 5; KD = 1.2

Based on the results of testing PID control on the robot, the test results are obtained with a small error value and a large difference between the three motors that will be used in this study, namely with a value of KP = 2.5; KI = 5; KD = 1.

D. Robot Motion Testing

This research is conducted to find out how the localization of Omni-Directional Robots based on IMU and Rotary Encoder Sensors works. So that testing is carried out on the robot as a whole. The test is carried out to find out whether the system is functioning properly. The test carried out is testing the movement of the robot by moving to a predetermined point and then observing the results of the movement whether it has moved well with positional accuracy that has a slight error or a difference in distance from the desired one.

1) Forward Movement

The following are the results of testing the robot moving forward with the initial X coordinate robot position worth 0 cm and the initial Y coordinate worth 0 cm. The resulting coordinates obtained after the robot is run are the final X coordinate worth -1.73 cm and the final Y coordinate worth 100 cm. Then get a large shift or position error that occurs in this forward motion test, namely at the X coordinate of 1.73 cm and at the Y coordinate of 0 cm. Plotting results of the robot moving forward shown in Fig. 14. Forward motion test results shown in Table 2.

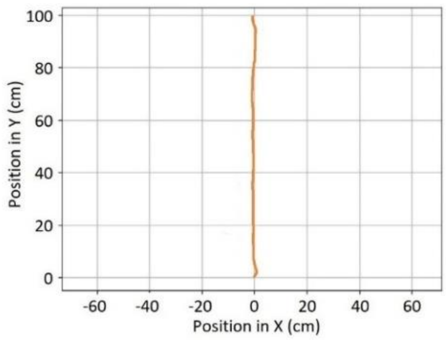

Fig. 14. Plotting results of the robot moving forward

Table 2. Forward motion test results

Starting Coordinates (Cm)		Final Coordinates (Cm)		Error (Cm)	
X	Y	X	Y	X	Y
0	0	-1.73	100	1.73	0

$$Xerror = Xstart - Xfinal$$
$$Xerror = 0 - (-1.73)$$
$$Xerror = 1.73$$

$$Yerror = Distance\ Target - (Yfinal - Ystart)$$
$$Yerror = 100 - (100 - 0)$$
$$Yerror = 0$$

2) Backward Motion

The following are the results of testing the robot moving backwards with the initial X coordinate robot position worth -0.90 cm and the initial Y coordinate worth 0 cm. The resulting coordinates obtained after the robot is run are the final coordinate X worth 0 cm and the final coordinate Y worth -99.86 cm. Then get a large shift or position error that occurs in this backward motion test, namely at the X coordinate of 0.9 cm and at the Y coordinate of 0.14 cm. Plotting results of the robot moving backward shown in Fig. 15. Progression test results shown in Table 3.

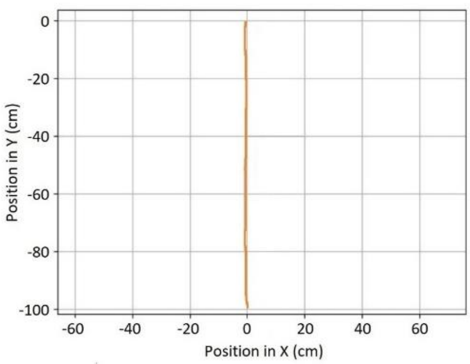

Fig. 15. Plotting results of the robot moving backward

Table 3. Progression test results

Starting Coordinates (Cm)		Final Coordinates (Cm)		Error (Cm)	
X	Y	X	Y	X	Y
-0.90	0	0	-99.86	0.9	0.14

$$Xerror = Xstart - Xfinal$$
$$Xerror = (-0.90) - 0$$
$$Xerror = -0.9$$

$$Yerror = Distance\ Target - (Yfinal - Ystart)$$
$$Yerror = 100 - (0 - (-99.86)$$
$$Yerror = 100 - 99.86$$
$$Yerror = 0.14$$

3) Right Side Movement

The following are the results of testing the robot moving right side with the initial X coordinate robot position worth 0 cm and the initial Y coordinate worth 0 cm. The resulting coordinates obtained after the robot is run are the final X coordinate of 89.85 cm and the final Y coordinate of 0 cm. Then get a large shift or position error that occurs in this right side motion test, namely at the X coordinate of 0.15 cm and at the Y coordinate of 0 cm. Plotting results of the robot in moving right side shown in Fig. 16. Right sideways motion test results shown in Table 4.

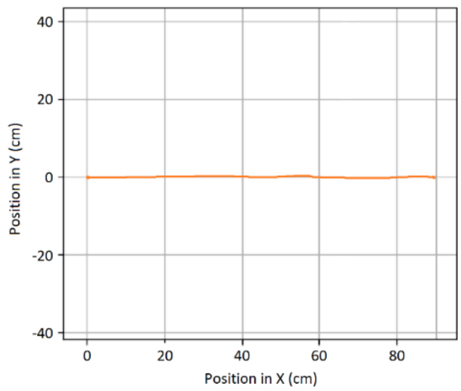

Fig. 16. Plotting results of the robot in moving right side

Table 4. Right sideways motion test results

Starting Coordinates (Cm)		Final Coordinates (Cm)		Error (Cm)	
X	Y	X	Y	X	Y
0	0	89.85	0	0.15	0

$$Xerror = Distance\ Target - (Xfinal - Xstart)$$
$$Xerror = 90 - (89.85 - 0)$$
$$Xerror = 90 - 89.85$$
$$Xerror = 0.15$$

$$Yerror = Ystart - Yfinal$$
$$Yerror = 0 - 0$$
$$Yerror = 0$$

4) Left Side Movement

The following are the results of testing the robot moving left side with the initial X coordinate robot position worth 0 cm and the initial Y coordinate worth 0 cm. The resulting coordinates obtained after the robot is run are the final X coordinate worth -89.90 cm and the final Y coordinate worth -0.13 cm. Then get a large shift or position error that occurs in this left side motion test, namely at the X coordinate of 0.1 cm and at the Y coordinate of 0.13 cm. Plotting results of the robot in left side movement shown in Fig. 17. Left side motion test results shown in Table 5.

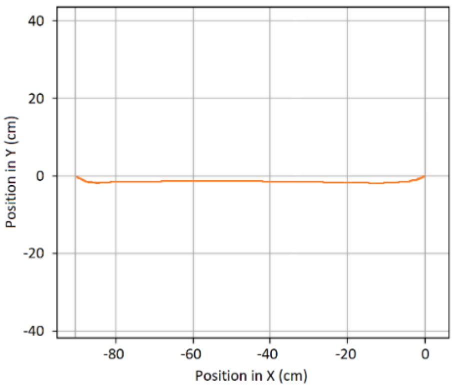

Fig. 17. Plotting results of the robot in left side movement

Table 5. Left side motion test results

Starting Coordinates (Cm)		**Final Coordinates (Cm)**		**Error (Cm)**	
X	Y	X	Y	X	Y
0	0	-89.90	-0.13	0.10	0.13

$$Xerror = Target\ Distance - (Xstart - Xfinal)$$
$$Xerror = 90 - (0 - (-89.80))$$
$$Xerror = 90 - 89.90$$
$$Xerror = 0.10$$

$$Yerror = Ystart - Yfinal$$
$$Yerror = 0 - (-0.13)$$
$$Yerror = 0.13$$

5) Square Motion Movement

By making the trajectory several times, the following are the results of testing the robot's motion to form a box, namely with the position of the X target coordinate robot worth 70 cm; 70 cm; 0 cm; 0 cm and the Y target coordinate worth 0 cm; 70 cm; 70 cm; 0 cm. The resulting actual X and Y coordinates obtained after the robot is run, then obtained the amount of shift or position error that occurs in testing the motion of forming a box with several turns, namely at the X coordinate has an average error value of 1.67 cm and at the Y coordinate has an average error value of 2.21 cm. Plotting results of the robot moving in a square shape Fig. 18. Square motion testing results shown in Table 6.

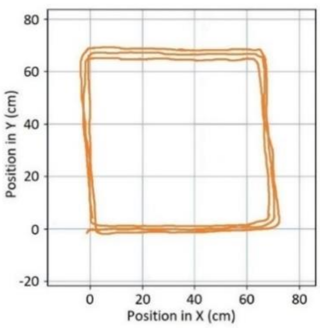

Fig. 18. Plotting results of the robot moving in a square shape

Table 6. Square motion testing results

Round to	Target Coordinate (Cm)		Actual Coordinate (Cm)		Error (Cm)	
	X	Y	X	Y	X	Y
1	70	0	70	2.57	0	2.57
	70	70	67.72	66.50	2.28	3.5
	0	70	-1.86	66.38	1.86	3.62
	0	0	2.50	-2.18	2.5	2.18
2	70	0	72.50	2.50	2.5	2.5
	70	70	68.37	69.14	1.63	0.86
	0	70	0	69.85	0	0.15
	0	0	2.45	0	2.45	0
3	70	0	68.75	4	1.25	4
	70	70	66.39	65.42	3.61	4.58
	0	70	-0.78	67.59	0.78	2.41
	0	0	1.26	4.15	1.26	4.15
Average Error					1.67	2.21

$$Average\ Error = \frac{Amount\ of\ Data}{Number\ of\ Data}$$

$$Average\ X\ Error = \frac{20.12}{12} = 1.67$$

$$Average\ Y\ Error = \frac{26.52}{12} = 2.21$$

6) Triangle Shaping Motion

By making the trajectory several times, the following are the results of testing the robot's motion to form a triangle, namely with the robot position X target coordinate worth 0 cm; 70 cm; 0 cm and Y target coordinate worth 70 cm; 70 cm; 70 cm. The actual X and Y coordinates obtained after the robot is run. Motion to form a triangle has a large enough distance difference because the X coordinate does not move straight ahead, but tilts at an angle of 30 degrees, so the average error value at the X coordinate is 8.07 cm and at the Y coordinate is 2.40 cm. Plotting results of the robot moving in a triangular shape shown in Fig. 19. Triangular motion test results Table 7.

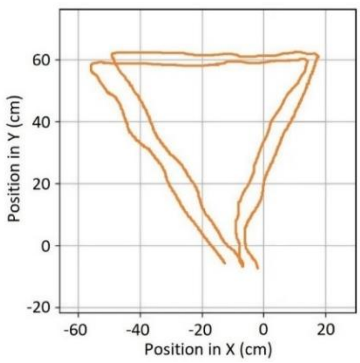

Fig. 19. Plotting results of the robot moving in a triangular shape

Table 7. Triangular motion test results

Round to	Target Coordinate (Cm)		Actual Coordinate (Cm)		Error (Cm)	
	X	Y	X	Y	X	Y
1	0	70	20	68.91	20	1.09
	70	70	67.9	68.2	2.1	2.8
	0	70	-6	69.8	6	0.2
2	0	70	-6	67.54	6	2.46
	70	70	68.15	66.68	1.82	3.32
	0	70	-12.50	65.32	12.5	4.58
Rata-rata Error					8.07	2.40

$$Average\ Error = \frac{Amount\ of\ Data}{Number\ of\ Data}$$

$$Average\ X\ Error = \frac{48.42}{6} = 8.07$$

$$Average\ Y\ Error = \frac{14.45}{6} = 2.40$$

IV. CONCLUSIONS

The IMU sensor can maintain the robot's facing direction according to the program entered, and the Rotary Encoder can work properly according to the desired distance. The test results on simple movements such as forward motion have an error value at the X coordinate of 1.73 cm and at the Y coordinate of 0 cm. The backward motion has an error value at the X coordinate of 0.9 cm and at the Y coordinate of 0.14 cm. The right side motion has an error value at the X coordinate of 0.15 cm and at the Y coordinate of 0 cm. And the left side motion has an error value at the X coordinate of 0.1 cm and at the Y coordinate of 0.13 cm. While in the motion of the robot forming a box has an average error value at the X coordinate of 1.67 cm and at the Y coordinate of 2.21 cm. While the motion of forming a triangle has a large enough distance difference because the X coordinate does not move straight ahead, but tilts at an angle of 30 degrees, the results obtained at the X coordinate have an average error value of 8.07 cm and at the Y coordinate have an average error value of 2.40 cm.

REFERENCES

[1] B. Miodrag, M. Lukić, J. Bajić, B. Dakić and M. Vukadinović, "Hardware realization of autonomous robot localization system," *2012 Proceedings of the 35th International Convention MIPRO*, pp. 146-150, 2012, https://ieeexplore.ieee.org/document/6240630.

[2] Kok Seng Chong and L. Kleeman, "Accurate odometry and error modelling for a mobile robot," *Proceedings of International Conference on Robotics and Automation*, pp. 2783-2788 vol.4, 1997, https://doi.org/10.1109/ROBOT.1997.606708.

[3] R. H. Abiyev, I. S. Günsel, N. Akkaya, E. Aytac, A. Çağman, and S. Abizada, "Fuzzy control of omnidirectional robot," *Procedia Comput. Sci.*, vol. 120, no. 10, pp. 608–616, 2017, https://doi.org/10.1016/j.procs.2017.11.286.

[4] N. Ushimi and Y. Aoba, "Development of a two-wheels caster type of odometer with a dual shaft for Omnidirectional Mobile Robots," *Procedia Eng.*, vol. 41, pp. 163–169, 2012, https://doi.org/10.1016/j.proeng.2012.07.157.

[5] W. Li, C. Yang, Y. Jiang, X. Liu, and C. Y. Su, "Motion planning for omnidirectional wheeled mobile robot by potential field method," *J. Adv. Transp.*, vol. 2017, no. 2, pp. 1–11, 2017, https://doi.org/10.1155/2017/4961383.

[6] F. Tajti, G. Szayer, B. Kovács, P. Barna, and P. Korondi, "Optical flow based odometry for mobile robots supported by multiple sensors and sensor fusion," *Automatika*, vol. 57, no. 1, pp. 201–211, 2017, https://doi.org/10.7305/automatika.2016.07.886.

[7] A. A. Santoso Gunawan *et al*., "Line Follower Smart Trolley System V2 using RFID," *2021 1st International Conference on Computer Science and Artificial Intelligence (ICCSAI)*, pp. 17-21, 2021, https://doi.org/10.1109/ICCSAI53272.2021.9609710.

[8] J. A. Batlle and A. Barjau, "Holonomy in mobile robots," *Rob. Auton. Syst.*, vol. 57, no. 4, pp. 433–440, 2009, https://doi.org/10.1016/j.robot.2008.06.001.

[9] A. F. Tehrani, A. M. Doosthosseini, H. R. Moballegh, P. Amini, and M. M. DaneshPanah, "A new odometry system to reduce asymmetric errors for omnidirectional mobile robots," *RoboCup 2003: Robot Soccer World Cup VII*, vol. 3020, pp. 600–610, 2004, https://doi.org/10.1007/978-3-540-25940-4_57.

[10] H. Y. Lin and C. H. He, "Mobile robot self-localization using omnidirectional vision with feature matching from real and virtual spaces," *Appl. Sci.*, vol. 11, no. 8, pp. 1–14, 2021, https://doi.org/10.3390/app11083360.

[11] H. Min, Y. Lin, S. Wang, F. Wu, and X. Shen, "Path planning of mobile robot by mixing experience with modified artificial potential field method," *Adv. Mech. Eng.*, vol. 7, no. 12, pp. 1–17, 2015, https://doi.org/10.1177/1687814015619276.

[12] J. Inthiam and C. Deelertpaiboon, "Self-localization and navigation of holonomic mobile robot using omni-directional wheel odometry," *TENCON 2014 - 2014 IEEE Region 10 Conference*, pp. 1-5, 2014, https://doi.org/10.1109/TENCON.2014.7022281.

[13] M. Taufiqqurohman and N. F. Sari, "Odometry Method and Rotary Encoder for Wheeled Soccer Robot," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 407, no. 1, 2018, https://doi.org/10.1088/1757-899X/407/1/012103.

[14] A. Martinelli, "The accuracy on the parameter estimation of an odometry system of a mobile robot," *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, vol. 2, pp. 1378-1383, 2002, https://doi.org/10.1109/ROBOT.2002.1014735.

[15] D. Tak, A. Jain, P. S. Savnani and D. Akash Mecwan, "Path Tracing in Holonomic Drive System with Reduced Overshoot using Rotary Encoders," *2020 7th International Conference on Signal Processing and Integrated Networks (SPIN)*, pp. 343-348, 2020, https://doi.org/10.1109/SPIN48934.2020.9071196.

[16] J. Moreno et al., "Design, implementation and validation of the three-wheel holonomic motion system of the assistant personal robot (APR)," *Sensors*, vol. 16, no. 10, pp. 1–21, 2017, https://doi.org/10.3390/s16101658.

[17] I. Hassani, I. Maalej, and C. Rekik, "Holomonic Control of a Robot with an Omni-Directional Drive," *Math. Probl. Eng.*, vol. 3, no. 11, pp. 1–7, 2006, https://doi.org/10.1155/2018/2163278.

[18] C. M. S. Francisco G Rossomando, "Identification and control of nonlinear dynamics of a mobile robot in discrete time using an adaptive technique based on neural PID," *Neural Comput. Appl.*, vol. 26, pp. 1179–1191, 2015, https://doi.org/10.1007/s00521-014-1805-8.

[19] T. Gujarathi and K. Bhole, "Gait Analysis Using Imu Sensor," *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1-5, 2019, https://doi.org/10.1109/ICCCNT45670.2019.8944545.

[20] Y. Jiang and S. Yin, "Recent Advances in Key-Performance-Indicator Oriented Prognosis and Diagnosis With a MATLAB Toolbox: DB-KIT," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 5, pp. 2849-2858, 2019, https://doi.org/10.1109/TII.2018.2875067.