SINERGI Vol. 27, No. 1, February 2023: 45-56
http://publikasi.mercubuana.ac.id/index.php/sinergi
http://doi.org/10.22441/sinergi. 2023.1.006

Implementation consensus algorithm and leader-follower of multi-robot system formation

Noval Lilansa*, Muhammad Nursyam Rizal, Pipit Anggraeni, Nur Jamiludin Ramadhan
Department of Manufacturing Automation and Mechatronics Engineering, Politeknik Manufaktur Bandung, Indonesia

Abstract
Robot technology has recently been applied to many applications to help human activities. Mobile Robot is one of the most flexible robot technology. This research uses a mobile robot designed using an omnidirectional wheel for the movement mechanism. Coordination and control of multi-robots can be assigned to perform any task from a different kind of field. Therefore, this paper aims to develop a multi-robot system to form a formation to do the task. The multi-robot system consists of three units Mobile Robot. The formation system will be built based on a coordinate point determined by a consensus point. The leader-follower topology is used to determine the orientation of the robot. ROS (Robot Operating System) is used as middleware to create a multi-robot system. The Open Base package in Gazebo Simulator is also used to simulate the movement of the multi-robot. From three test scenarios, this research results show that all the robots can do and follow the tasks simulated in the Gazebo with an average accuracy of 88.14%. Furthermore, no feedback from the robot to the Gazebo Simulator affects the robot's accuracy average below 90%.

Keywords:
Autonomous mobile robot;
Consensus;
Leader-follower;
ROS;

Article History:
Received: March 27, 2022
Revised: September 9, 2022
Accepted: October 10, 2022
Published: February 2, 2023

Corresponding Author:
Muhammad Nursyam Rizal, Department of Manufacturing Automation and Mechatronics Engineering, Politeknik Manufaktur Bandung, Indonesia
Email: nursyam@ae.polman-bandung.ac.id

INTRODUCTION

In recent years, the development and knowledge of robotics have greatly increased especially for the common people. Therefore, robotics can help humans to do tasks even in our households. Therefore, developing robotics technology is important due to chasing extremely fast robot improvement.

Nowadays, robotics is also used to do special tasks with special intelligence required, such as integration between several robots to do the tasks. Furthermore, several robots' interactions are also developed in many applications due to advances in computing, communication, sensing, and actuation devices. These interconnected robots can be assigned to perform civil or military tasks [1, 2, 3].

In implementing an interconnected robot, one of the most implemented interconnected robots is a cooperative logistic robot in industries. In many industries, logistic robots are now included in a production lines. Therefore, those robots are needed to increase productivity by replacing repetitive horizontal transport tasks.

There are several research areas in interconnected robots, such as swarm robotics and multi-agent system. Tan et al. in [4] distinguish those two systems into several categories. Swarm robotics vary in population size, unknown environment, decentralized and autonomous control system, and homogenous. Multi-robot systems have a small range of population size, known environment, and centralized or hierarchical control systems. Multi-robot systems have homogenous or heterogeneous robots. For instance, robot football is a typical multi-robot system application. Each one of the robots has different tasks to get the same purpose which is to get a score. Implementing logistic robots is somehow needed to form a formation when the goods are two times larger than the robots.

One of the steps to implement the multi-robot system is forming a formation. The robots have to move from various points to the point that formed the formation. Multi-robot formation control system for various implementations, such as cooperative area exploration, autonomous vehicles, and security patrols, is a problems that researchers often face [5]. In conventional systems, before producing a formation, each robot has an initial position that is not always the same. Therefore, the robots must communicate with each other so that each robot can find the location of its neighboring robots to overcome the problems. Then, the formation will be easily formed without the misplacement of each robot.

Conceptually, a consensus algorithm [6, 7, 8, 9] can solve forming a multi-robot formation. Consensus means the decision of all agents on certain information based on the reciprocity of decisions and agreements between agents [9].

Network communication is required to control the movement of multi robots from a PC. Anggraeni, in her research [11], researched multi-robots with wireless communication using Multi-master ROS (Robot Operating System). The goal is to implement a multi-master system to manage the communication network between robots wirelessly. It supports to control of the movement of multi robots using only one computer.

Maghenem, in his research [10], researched formation problems consisting of several non-holonomic mobile robots. His research proposes consensus algorithm that consider the kinematic and the dynamic model to move the *N* agents to formation goal with a given orientation. The closed-loop system is calculated using strict Lyapunov function. As a result, the formation can be formed at a desired consensus point with predetermined orientation.

Another approach to solving forming formation problems is using the leader-follower approach [12, 13, 14, 15, 16, 17, 18]. Widyotriatmo et al. in [13] used a leader-follower approach to control a formation of Nonholonomic Mobile Robots Team. Widyotriatmo proposed a role assignment algorithm to move the mobile robots following individual virtual trajectories in a leader-follower formation. The pole-placement method is used to design control parameters for mobile robots to track their own virtual trajectories.

This research focuses on the control of forming multi-agent robot's formation considering the problems and research identification above. The implementation of the consensus algorithm will be combined with the leader-follower approach to solving forming a formation problem.

METHOD

Mobile Robots

Mobile robots are one of the fastest-growing fields in scientific research. The capabilities of mobile robots can replace humans, for example, in moving goods, surveillance, industrial automation, industrial construction, entertainment, and others [11, 19, 20, 21, 22]. The problem of driving on a mobile robot can be solved by understanding the mechanism, kinematics, dynamics, and control. Perception of mobile robots involves signal processing, such as computer vision and sensors. Meanwhile, cognition results from sensor data input analysis to generate action so that the mobile robot can achieve a goal (object). Navigation on a mobile robot requires knowledge of algorithms and the required information. There are three types of mobile robots, namely UGV (Unmanned Ground Vehicles), UAV (Unmanned Aerial Vehicles), and AUV (Autonomous Underwater Vehicles) [9].

Polebot

A mobile robot used in this research is Polebot, which has a UGV mobile robot with three omnidirectional wheels, as shown in Figure 1. *Polebot* has three omni wheels construction which is installed with 1200 between each wheel, where each wheel axis intersects at the center of the robot. The omni wheels installed type can move freely in all directions on a flat surface in either translation or rotation.

Kinematics Model of The Robots

The wheel configuration for the movement of the robot is shown in Figure 2.

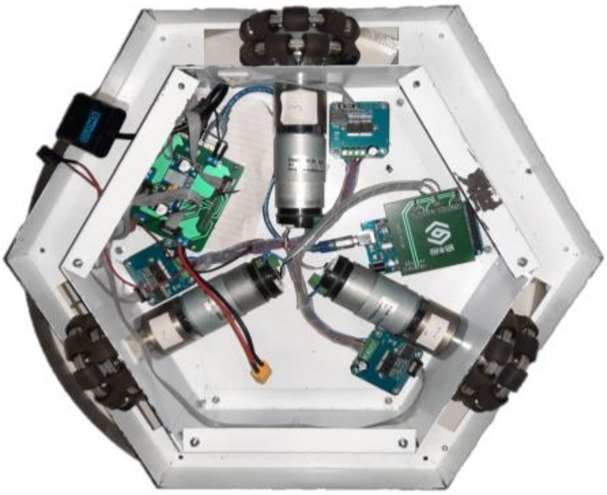

Figure 1. Three Omni Wheel Construction on *Polebot*

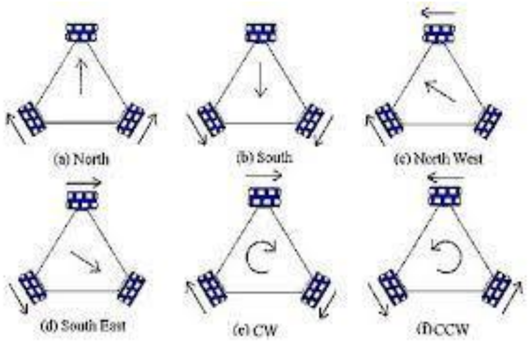

Figure 2. Omni Wheel Robot Movement [23]

As shown in Figure 3, the robot's kinematic model is illustrated. Each motor wheel's speed (rpm) is needed to move this robot in the exact direction. The inverse kinematic of the robot is needed to obtain the rpm. We have already studied this robot and obtained the forward kinematic equations relative to its frame as follows.

$$V_x^m = \frac{2V_2 - V_1 - V_3}{3} \tag{1}$$

$$V_y^m = \frac{\sqrt{3}V_3 - \sqrt{3}V_1}{3} \tag{2}$$

$$\omega_p = \frac{V_1 + V_2 + V_3}{3\mathrm{L}} \tag{3}$$

Then, here are the inverse kinematic equations for this robot:

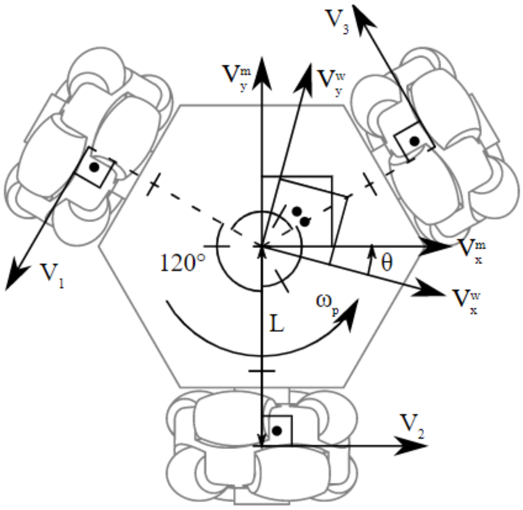

Figure 3. Illustration Geometric Constraints of *Polebot*

$$V_1 = -\frac{V_x^m}{2} - \frac{\sqrt{3}V_y^m}{2} + L\omega_p \tag{4}$$

$$V_2 = V_x^m + L\omega_p \tag{5}$$

$$V_3 = -\frac{V_x^m}{2} + \frac{\sqrt{3}V_y^m}{2} + L\omega_p \tag{6}$$

whereas, V1 is left-side wheel, V2 is back-side wheel, and V3 is right-side wheel.

Odometry System

Odometry uses data from actuator movements to estimate changes in position coordinates over time [24, 25, 26, 27]. Odometry is used to estimate position coordinates relative to the initial position. In the wheeled robot odometry system, the sensor used is a rotary encoder to detect the number of wheel rotations. There are three main parameters in calculating the coordinates of the robot's position, namely the diameter of the wheel ($dW(i)$), the number of encoder resolutions (res_{enc}), and the number of generated rotary encoder pulses per millimeter ($ppm_{(\mathrm{i})}$). To calculate the circumference of the freewheel ($\mathrm{C}(i)$) and ($ppm_{(\mathrm{i})}$), can be used (7) and (8) [23][24].

$$CW_{(\mathrm{i})} = \pi\, dW_{(i)} \tag{7}$$

$$ppm_{(\mathrm{i})} = \frac{res_{enc}}{dW_{(i)}} \tag{8}$$

On omnidirectional wheel robot, (1) to (3) can be used to calculate the X and Y coordinates by replacing the velocity of each wheel ($V_{(\mathrm{i})}$) with a travelled distance of each wheel ($S_{(\mathrm{i})}$) so that equations are obtained as follows [24][25].:

$$S_x^m = \frac{2S_2 - S_1 - S_3}{3} \tag{9}$$

$$S_y^m = \frac{\sqrt{3}S_3 - \sqrt{3}S_1}{3} \tag{10}$$

$$\omega_p = \frac{S_1 + S_2 + S_3}{3\mathrm{L}} \tag{11}$$

(9) to (11) be used to obtain X and Y-traveled coordinates. Those equations also can be used to obtain the position of each robot.

Method

The wheel speed data which have been calculated are sent to the controller. Thus, the controller will generate the Pulse Width Modulation (PWM) to the motor based on the speed of each motor wheel.

In general, the description of the system that will be made is shown in Figure 4. In the designed system, *Polebot* will follow the robot's movement in the *Gazebo Simulation*. The figure describes the relationship between several components that exist in the system. The computer connected to the *Gazebo Simulation* will be connected to the three *Polebots* via a wireless network (*WiFi*) with the help of the multimaster ROS on the laptop/computer and each *Polebot*. This system uses ROS multi-master to create a communication system between the PC and three robots. The ROS multi-master system consists of more than one ROS network with its roscore node each. A package called *multimaster_fkie* is needed to implement this purpose [11]. This package has a set of nodes to build and manage a multi-master network. No or minimal configuration is needed. All of the changes in the system are automatically detected and synchronized. This package also allows the system to send multicast messages periodically to the ROS networks and select which hosts, topics and services should be synchronized or ignored between different roscore. Those two merits are running simultaneously by the multi-master system.

Polebot will receive data in the form of wheel speed set points. The wheel speed data will be processed on *Arduino* using PID so that the wheel can rotate at the desired speed. After the data is processed, the data will be republished by *Arduino* to *LattePanda* to drive the motor.

The system created aims to control three *Polebot* robots to move to a destination coordinate point by forming a triangular formation from each different initial position. The coordinates of the triangular consensus will be determined based on the average of the three robots' initial positions. To obtain a consensus, the system requires communication that occurs between agents. Communication is carried out as shown in Figure 5.

In Figure 5, the communication is carried out using a leader-follower topology. *Polebot1* acts as a leader, and *Polebot2* and *Polebot3* act as a follower. *Polebot2* and *Polebot3* will subscribe to the position of *Polebot1* to determine the orientation of the isosceles triangle. The initial position of Polebot1 determines the orientation of the triangle concerning the consensus coordinate point on the Y axis. If the position of *Polebot1* is more positive or equal to the consensus point, the consensus point on the base of the isosceles triangle will be on the lower side. Therefore, *Polebot1* will be in the top corner, *Polebot2* will be in the left corner, and *Polebot3* will be in the right corner. Conversely, if *Polebot1*'s position is more negative, the consensus point at the base of the isosceles triangle will be on the upside. Therefore, *Polebot1* will be in the bottom corner, *Polebot2* will be in the right corner, and *Polebot3* will be in the left corner.

The environment used is 3 m x 3 m, where the length and width of the environment are divided into three squares on the X-axis and three squares on the Y-axis, each measuring 1 m. A square in the Gazebo Simulation represents one meter in the environment. The system description will be explained in Figure 6.

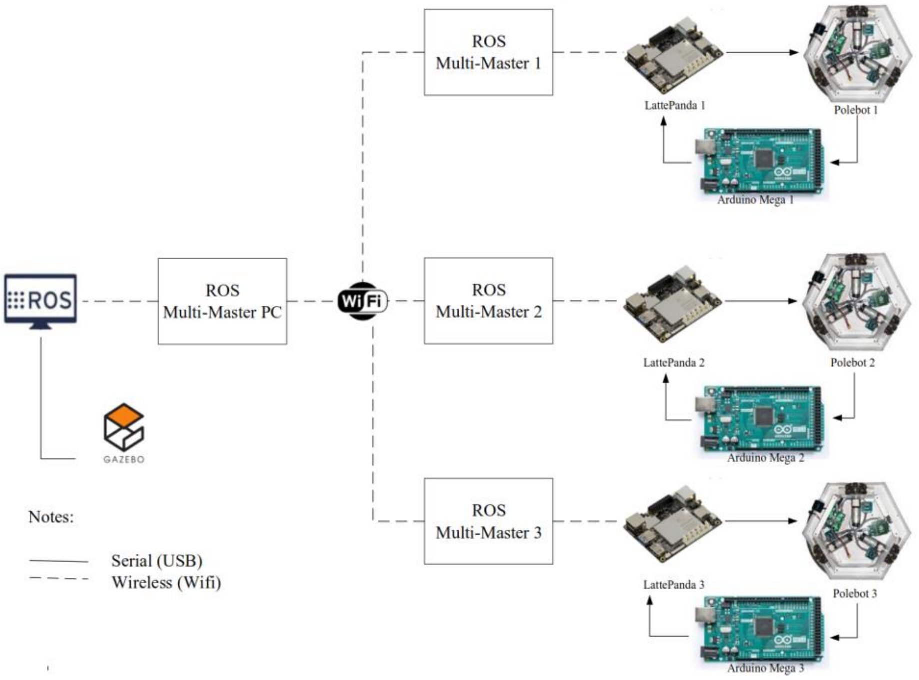

Figure 4. Architecture Diagram of *Polebot* System

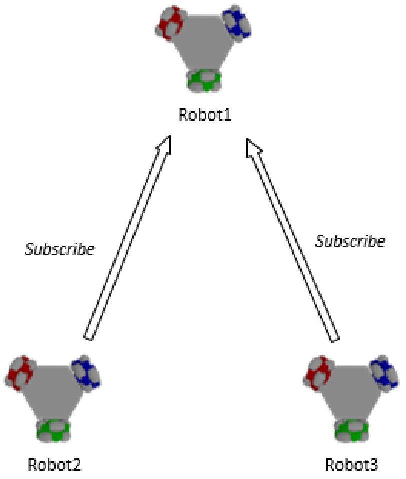

Figure 5. Communication of the Robots

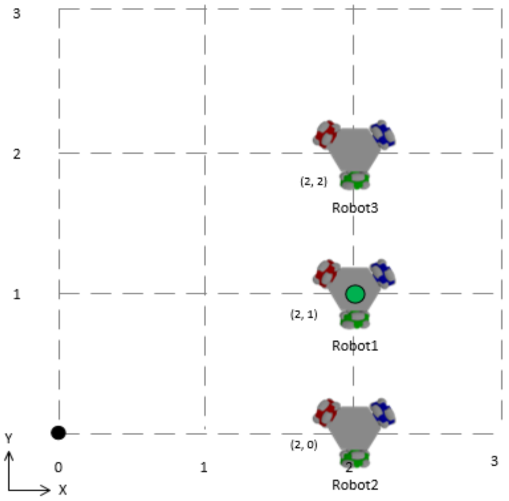

Figure 6. Uses Environment Layout

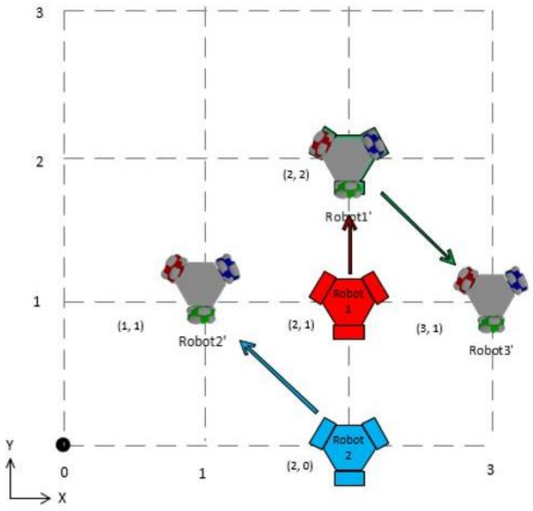

Figure 7. The Expected Formation Result

Figure 6 illustrates the system description where *Polebot1*, *Polebot2*, and *Polebot3* represent each robot. The point (0, 0) on the lower left side of the work area, in Figure 6 indicated by a black dot. In this case, the initial position of *Polebot1* is at coordinates (2, 1), *Polebot2* is at coordinates (2, 0), and *Polebot3* is at coordinates (2, 3). A green dot indicates the coordinates of the goal for the formation of the triangle formation with coordinates (2, 1). This green dot point results from a consensus between the three robots. This point, the average of the X-axis and Y-axis coordinates of the three robots, is calculated by (7) and (8).

$$x = \frac{xi_1 + xi_2 + xi_3}{3} \tag{7}$$

$$y = \frac{yi_1 + yi_2 + yi_3}{3} \tag{8}$$

Figure 7 shows the triangle formation result formed from the robot's initial position (see Figure 6). The movement and initial position of *Polebot1* are marked in red. The movement and initial position of *Polebot2* are marked in blue. Finally, the movement and initial position of *Polebot3* are marked in green. The movement of the robot formed is holonomic. Because *Polebot1*'s initial position is more positive or equal to the consensus point on the Y axis, *Polebot1* will be at the same point as the coordinates of the consensus point on the X-axis, and one meter more positive than the consensus point on the Y axis. *Polebot2* will be one meter more negative (on the left) of the consensus point on the X-axis, and will be at the same point as the coordinates of the consensus point on the Y-axis. *Polebot3* will be 1m more positive (on the right) than the consensus point on the X-axis, and will be at the same point as the coordinates of the consensus point on the Y-axis.

Figure 8 is a flow chart for the formation program. The formation program begins with knowing the initial position of each robot in the *Gazebo Simulation*. Next, each *Polebot* will subscribe to the robot's position in the simulation. Finally, each *Polebot* will subscribe to the position data in the simulation to determine the position of the *Polebot* in the actual situation. The consensus point can be determined after the robot's initial position in the simulation is known. The consensus point is obtained from the average starting position of each robot member on the x-axis and y-axis. After the consensus coordinates are obtained, each robot will move to its respective position to form a triangle.

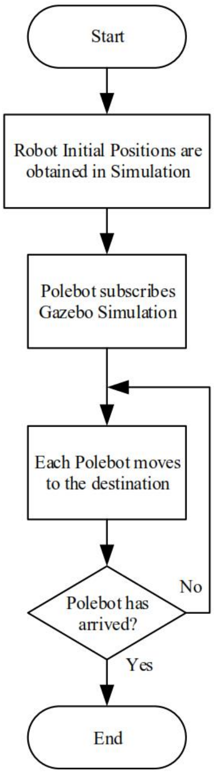
Start
Robot Initial Positions are obtained in Simulation
Polebot subscribes Gazebo Simulation
Each Polebot moves to the destination
Polebot has arrived?
No
Yes
End

Figure 8. Formation Program Flowchart

In general, the system flow diagram as shown in Figure 9, can be explained that the system starts by running a *Gazebo Simulation* on a laptop/computer. Then, run ROS Multimaster on a laptop/computer and each *LattePanda* on a *Polebot*. If Multimaster is connected, then every *LattePanda* on each *Polebot* is connected to *Arduino* using *Rosserial*. Then the program for the formation is run on the laptop/computer. The system is considered complete if all *Polebot*s have stopped at the destination point according to the predetermined formation. The flow chart for the formation program in the above sub-process will be explained in Figure 9.

RESULTS AND DISCUSSION

This test is conducted to prove that *Polebot1*, *Polebot2*, and *Polebot3* can form a triangle formation from different starting positions. The data to be taken is the x-axis and y-axis coordinate data in the *Gazebo Simulation*. Figures 10, Figure 11, and Figure 12 are the graphs of the coordinates generated by the movement of each robot in the *Gazebo Simulation*. A lined circle indicates the starting position of each robot at one end of the line on the graph. A colored circle indicates the robot's destination position at the other end of the line. A red line indicates a blue line indicates the movement of Polebot1, the movement of Polebot2, and the movement of *Polebot3* is indicated by a green line. Finally, the consensus points formed are marked with a yellow circle (see Figure 10).

Formation Test 1

In test 1, the three robots are on the same y-axis coordinates. Table 1 describes the position data taken from the *Gazebo Simulation* in test 1. Figure 10 illustrates how robots can form a triangle from the initial position of each robot being on the same y-axis. Based on the initial position of each robot, the consensus points formed are (2, 1). *Polebot1* will move and stop at coordinates (2, 2), 1 m away from the meeting point on the y-axis. *Polebot2* will stop at coordinates (1, 1), located on the left as far as 1 m from the meeting point on the x-axis. *Polebot3* will remain in its position and will not move at the coordinates (3, 1), being on the right as far as 1 m from the meeting point on the x-axis.

Table 2 explains the measured result coordinates of the *Polebot*'s destination point in the environment. The test was carried out five times and was carried out on a ceramic floor using a measuring instrument in the form of a measuring tape with an accuracy of 0.1 cm.

The resulting *Polebot* movement accuracy is still considered good when the *Polebot* body is still in the destination point area in the actual environment.

Table 1. Coordinate Data in Formation Test 1

Robot	**Initial Position (x,y)**		**Consensus Point (x,y)**		**Destination Position (x,y)**	
	x	**y**	**x**	**y**	**x**	**y**
Polebot1	2.5	1	2	1	2	2
Polebot2	0.5	1			1	1
Polebot3	3	1			3	1

Table 2. Average Results of Formation Test 1

Robot	**Axis**	**Averages**	
		Error (%)	**Accuracy (%)**
Polebot1	X	8.88	91.12
	Y	6.68	93.32
Polebot2	X	11.2	88.8
	Y	7.56	92.44
Polebot3	X	0	100
	Y	0	100

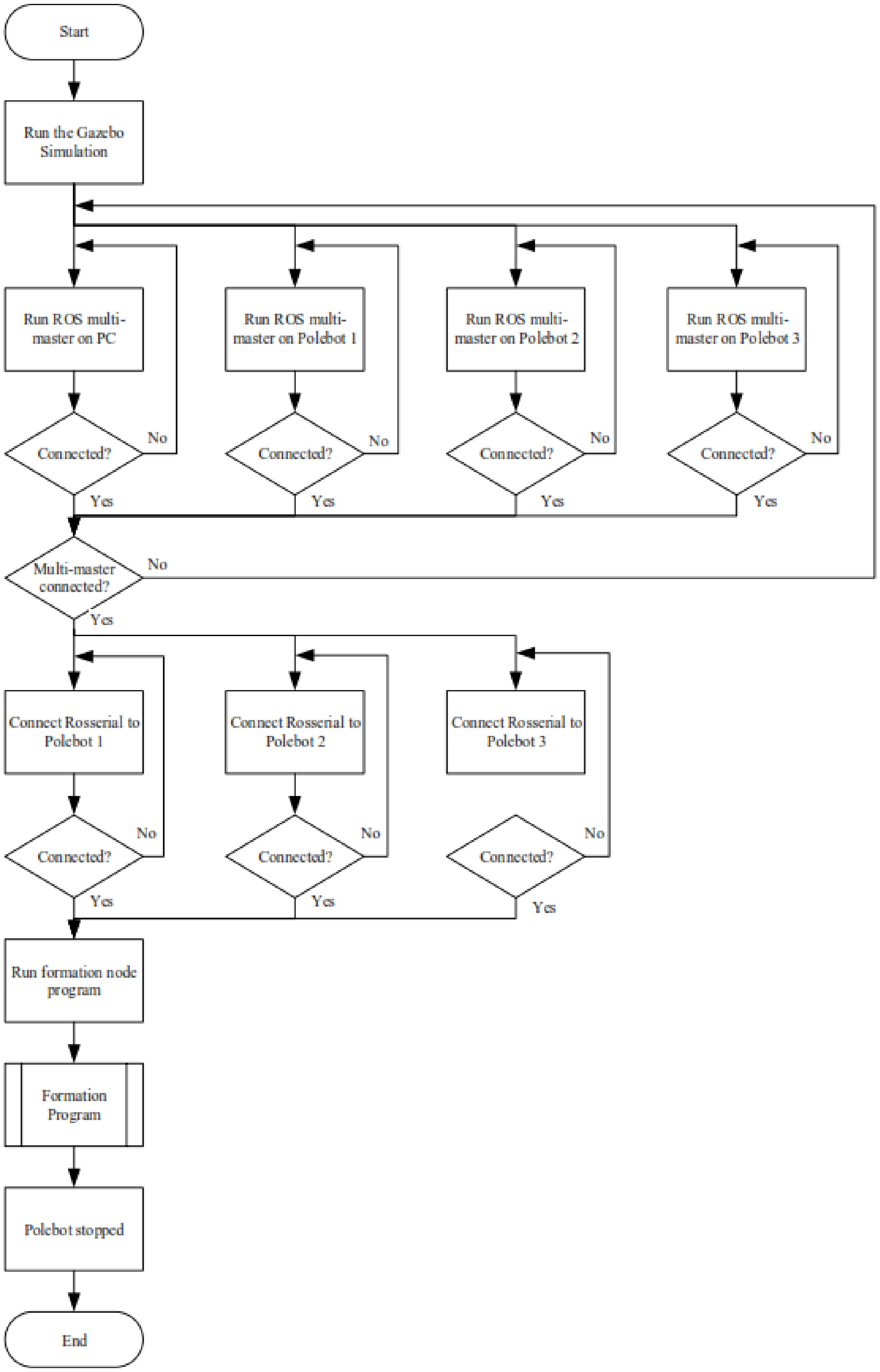
Start
Run the Gazebo Simulation
Run ROS multi-master on PC
Run ROS multi-master on Polebot 1
Run ROS multi-master on Polebot 2
Run ROS multi-master on Polebot 3
Connected?
No
Yes
Multi-master connected?
Connect Rosserial to Polebot 1
Connect Rosserial to Polebot 2
Connect Rosserial to Polebot 3
Run formation node program
Formation Program
Polebot stopped
End

Figure 9. Flowchart of *Polebot* Formation System

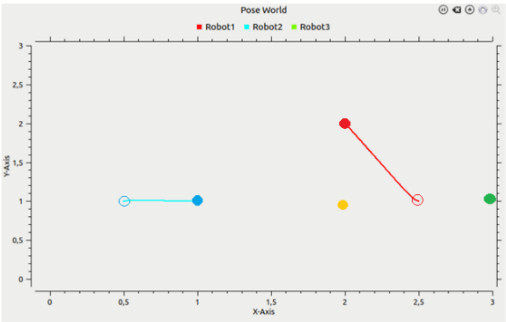

Figure 10. Position Graph Coordinates of Formation Test 1 Results

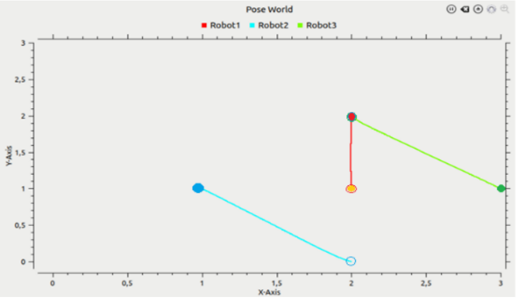

Figure 11. Position Graph Coordinates of Formation Test 2 Results

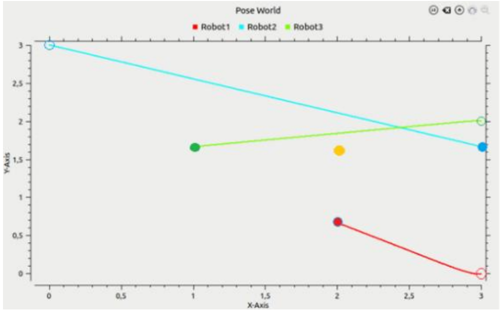

Figure 12. Position Graph Coordinates of Formation Test 3 Results

Table 3 shows the measurement results of the coordinates of the *Polebot* destination point. From the measurement results, it is found that *Polebot*1 has an average accuracy of 91.12% on the x-axis and 93.32% on the y-axis. *Polebot*2 has an average accuracy of 88.8% on the x-axis and 92.44% on the y-axis. Finally, because *Polebot*3 remains in position and does not move, *Polebot*3 has an accuracy of 100% on the x-axis and 100% on the y-axis.

From these tests, the precision value of *Polebot*1 with movement on the x-axis is 0.038 and the precision value of movement on the y-axis is 0.047. The precision value of *Polebot*2 with movement on the x-axis is 0.080 and the precision value of movement on the y-axis is 0.052. The precision value of *Polebot*3 with movement on the x-axis is 0 and the precision value of movement on the y-axis is 0.

Formation Test 2

In test 2, the three robots are on the same x-axis coordinates. Table 4 explains the position data taken from the *Gazebo Simulation* in test 2.

Figure 11 illustrates the formation of a triangle from the initial position of each robot on the same x-axis. Based on the initial position of each robot, the consensus points formed are (2, 1). *Polebot1* will move and stop at coordinates (2, 2), one meter away from the meeting point on the y-axis. *Polebot2* will stop at coordinates (1, 1), located on the left as far as one meter from the meeting point on the x-axis. *Polebot3* will stop at coordinates (3, 1), located on the right as far as one meter from the meeting point on the x-axis.

Table 3. Measured Position of *Polebot* in Formation Test 1

No	Robot	Destination Point Coordinates				Error (%)		Accuracy (%)	
		Correct Point		Meas. Point					
		X	Y	X	Y	X	Y	X	Y
1	*Polebot1*	2	2	2.185	2.05	9.1	2.5	90.9	97.5
	Polebot2	1	1	0.95	0.851	5	14.9	95	85.1
	Polebot3	3	1	3	1	0	0	100	100
2	*Polebot1*	2	2	2.164	2.167	8.2	8.35	91.8	91.65
	Polebot2	1	1	1.111	0.998	11.1	0.2	88.9	99.8
	Polebot3	3	1	3	1	0	0	100	100
3	*Polebot1*	2	2	2.232	2.154	11.6	7.7	88.4	92.3
	Polebot2	1	1	1.124	0.913	12.4	8.7	87.6	91.3
	Polebot3	3	1	3	1	0	0	100	100
4	*Polebot1*	2	2	2.126	2.156	6.3	7.8	93.7	92.2
	Polebot2	1	1	1.141	0.928	14.1	7.2	85.9	92.8
	Polebot3	3	1	3	1	0	0	100	100
5	*Polebot1*	2	2	2.184	2.141	9.2	7.05	90.8	92.95
	Polebot2	1	1	1.134	0.932	13.4	6.8	86.6	93.2
	Polebot3	3	1	3	1	0	0	100	100

Table 4. Coordinate Data in Formation Test 2

Robot	Initial Position (x,y)		Consensus Point (x,y)		Destination Position (x,y)	
	x	y	x	y	x	y
Polebot1	2	1	2	1	2	2
Polebot2	2	0			1	1
Polebot3	2	2			3	1

Table 5 explains the measured result coordinates of the *Polebot*'s destination point in the environment. The test was carried out five times and was carried out on a ceramic floor using a measuring instrument in the form of a measuring tape with an accuracy of 0.1 cm. The resulting *Polebot* movement accuracy is still considered good when the *Polebot* body is still in the destination point area in the actual environment.

Table 6 shows the measurement results of the *Polebot* coordinates destination point. From the measurement results, it is found that *Polebot*1 has an average accuracy of 96.26% on the x-axis and 98.28% on the y-axis. *Polebot*2 has an average accuracy of 75.56% on the x-axis and 95.16% on the y-axis. Because *Polebot*3 remains in position and does not move, *Polebot*3 has an accuracy of 97.6% on the x-axis and 80.78% on the y-axis. The accuracy of 75.56% in *Polebot*2 is caused if the set point given by the simulation is sometimes too small in the forward diagonal movement. Because of the difference in the number of passive omni-wheels between the simulation robot model and the *Polebot* model, the minimum output set point from the simulation must be limited to 3 rad/s to overcome unnecessary wheel rotation in a forward movement.

From these tests, the precision value of *Polebot*1 with movement on the x-axis is 0.0108 and the precision value of movement on the y-axis is 0.0098. The precision value of *Polebot*2 with movement on the x-axis is 0.009, and the precision value of movement on the y-axis is 0.0038. The precision value of *Polebot*3 with movement on the x-axis is 0.0061 and the precision value of movement on the y-axis is 0.021.

Formation Test 3

In test 3, the three robots are on different x-axis and y-axis coordinates. Table 7 shows the position data taken from the *Gazebo Simulation* in test 3.

Figure 12 describes the formation of a triangle from the initial position of each robot on a different x-axis and y-axis. Based on the initial position of each robot, the consensus point formed is (2, 1.67). *Polebot1* will move and stop at the coordinates (2, 0.67), one meter away from the meeting point on the y-axis. *Polebot2* will stop at the coordinates (3, 1.67), located on the right as far as one meter from the meeting point on the x-axis. *Polebot3* will stop at the coordinates (1, 1.67), located on the left as far as one meter from the meeting point on the x-axis.

Table 8 explains the measured result coordinates of the *Polebot*'s destination point in the environment. The test was carried out five times and was carried out on a ceramic floor using a measuring instrument in the form of a measuring tape with an accuracy of 0.1 cm. The resulting *Polebot* movement accuracy is still considered good when the *Polebot* body is still in the destination point area in the actual environment.

Table 9 shows the Polebot coordinates destination point measurement results in Formation Test 3. From the measurement results, it is found that *Polebot*1 has an average accuracy of 88.6% on the x-axis and 73.134% on the y-axis.

Table 5. Measured Position of *Polebot* in Formation Test 2

No	Robot	Destination Point Coordinates				Error (%)		Accuracy (%)	
		Correct Point		Meas. Point					
		X	Y	X	Y	X	Y	X	Y
1	*Polebot1*	2	2	2.025	2.04	1.25	2	98.75	98
	Polebot2	1	1	1.242	1.043	24.2	4.3	75.8	95.7
	Polebot3	3	1	2.911	1.204	2.97	20.4	97.03	79.6
2	*Polebot1*	2	2	2.028	2.021	1.4	1.05	98.6	98.95
	Polebot2	1	1	1.232	1.048	23.2	4.8	76.8	95.2
	Polebot3	3	1	2.949	1.18	1.7	18	98.3	82
3	*Polebot1*	2	2	2.26	2.029	13	1.45	87	98.55
	Polebot2	1	1	1.257	1.047	25.7	4.7	74.3	95.3
	Polebot3	3	1	3.071	1.16	2.37	16	97.63	84
4	*Polebot1*	2	2	2.05	2.047	2.5	2.35	97.5	97.65
	Polebot2	1	1	1.244	1.051	24.4	5.1	75.6	94.9
	Polebot3	3	1	2.924	1.212	2.533	21.2	97.47	78.8
5	*Polebot1*	2	2	1.989	2.035	0.55	1.75	99.45	98.25
	Polebot2	1	1	1.247	1.053	24.7	5.3	75.3	94.7
	Polebot3	3	1	2.927	1.205	2.433	20.5	97.57	79.5

Table 6. Average Results of Formation Test 2

Robot	Axis	Averages	
		Error (%)	Accuracy (%)
Polebot1	X	3.74	96.26
	Y	1.72	98.28
Polebot2	X	24.44	75.56
	Y	4.84	95.16
Polebot3	X	2.4	97.6
	Y	19.22	80.78

Table 6. Coordinate Data of Formation Test 3

Robot	Initial Position (x,y)		Consensus Point (x,y)		Destination Position (x,y)	
	x	y	x	y	x	y
Polebot1	3	0	2	1.67	2	0.67
Polebot2	0	3			3	1.67
Polebot3	3	2			1	1.67

Table 7. Measured Position of *Polebot* in Formation Test 3

No	Robot	Destination Point Coordinates				Error (%)		Accuracy (%)	
		Correct Point		Meas. Point					
		X	Y	X	Y	X	Y	X	Y
1	*Polebot1*	2	0.67	2.128	0.795	6.4	18.65	93.6	81.34
	Polebot2	3	1.67	3.225	1.298	7.5	22.27	92.5	77.72
	Polebot3	1	1.67	1.192	1.887	19.2	12.99	80.8	87.01
2	*Polebot1*	2	0.67	2.116	0.78	5.8	16.42	94.2	83.58
	Polebot2	3	1.67	3.151	1.301	5.03	22.10	94.97	77.90
	Polebot3	1	1.67	1.128	1.87	12.8	11.98	87.2	88.02
3	*Polebot1*	2	0.67	2.118	0.798	5.9	19.10	94.1	80.90
	Polebot2	3	1.67	3.154	1.289	5.13	22.81	94.87	77.19
	Polebot3	1	1.67	1.187	1.869	18.7	11.92	81.3	88.08
4	*Polebot1*	2	0.67	2.151	0.816	7.55	21.79	92.45	78.21
	Polebot2	3	1.67	3.147	1.295	4.9	22.45	95.1	77.55
	Polebot3	1	1.67	1.174	1.854	17.4	11.02	82.6	88.98
5	*Polebot1*	2	0.67	2.123	0.785	6.15	17.16	93.85	82.84
	Polebot2	3	1.67	3.157	1.297	5.23	22.33	94.77	77.66
	Polebot3	1	1.67	1.186	1.849	18.6	10.72	81.4	89.28

Table 8. Average Results of Formation Test 3

Robot	Axis	Averages	
		Error (%)	Accuracy (%)
Polebot1	X	6.36	93.64
	Y	18.62	81.37
Polebot2	X	5.56	94.44
	Y	22.39	77.60
Polebot3	X	17.34	82.66
	Y	11.72	88.27

*Polebot*2 has an average accuracy of 79.167% on the x-axis and 77.724% on the y-axis. Because *Polebot*3 remains in position and does not move, *Polebot*3 has an accuracy of 70.8% on the x-axis and 95.029% on the y-axis.

From these tests, the precision value of *Polebot*1 with movement on the x-axis is 0.014 and the precision value of movement on the y-axis is 0.0139. The precision value of *Polebot*2 with movement on the x-axis is 0.0327 and the precision value of movement on the y-axis is 0.0044. The precision value of *Polebot*3 with movement on the x-axis is 0.0262 and the precision value of movement on the y-axis is 0.0149.

Those three formations tests prove that the consensus point algorithm can solve the formation problem on *Polebot*s. All of the robots can move to make a triangle formation. The leader-follower approach can also solve the orientation problem so that all the robots can form the formation with the same orientation as the leader.

CONCLUSION

In this research, we can conclude that each *Polebot* can follow the movement according to the simulation on the *Gazebo Simulation*. The coordinates of the formation are obtained from the consensus between each robot. The consensus point is obtained from the average initial position of the three robots. A leader-follower topology is used to determine the orientation of the triangular formation. *Polebot1,* which acts as the leader, will determine the formation's orientation by comparing its initial position with the consensus coordinate point on the Y axis. Then *Polebot2* and *Polebot3,* which act as the follower, will fill the formation according to the predetermined position. The results show that the *Polebot1* has 90.67% accuracy on x-axis and 89.21% accuracy on y-axis. *Polebot2* has 83.32% accuracy on x-axis and 86.17% on y-axis. *Polebot3* has 89.27% accuracy on x-axis and 90.2% accuracy on y-axis.

REFERENCES

[1] A. Patil and G. Shah, “Multi-Robot Trajectory Tracking and Rendezvous Algorithm,” *IETE Journal of Research,* vol. 68, no. 6, pp. 1–7, Aug. 2020, doi: 10.1080/03772063.2020.1800521.

[2] B. A. Issa and A. T. Rashid, “A Survey of Multi-mobile Robot Formation Control,” *International Journal of Computer Applications*, vol. 181, no. 48, pp. 975–8887, Apr. 2019, doi: 10.5120/ijca2019918651.

[3] H. Rezaee and F. Abdollahi, "Consensus Problem Over High-Order Multiagent Systems with Uncertain Nonlinearities Under Deterministic and Stochastic Topologies," in *IEEE Transactions on Cybernetics*, vol. 47, no. 8, pp. 2079-2088, Aug. 2017, doi: 10.1109/TCYB.2016.2628811.

[4] Y. Tan and Z. yang Zheng, “Research Advance in Swarm Robotics,” *Defence Technology*, vol. 9, no. 1, pp. 18–39, Mar. 2013, doi: 10.1016/j.dt.2013.03.001.

[5] Y. Guo, *Distributed cooperative control: emerging applications*. Hoboken, NJ, USA: John Wiley & Sons, Inc, 2017.

[6] P. Anggraeni, M. Defoort, M. Djemai, A. Subiantoro, P. Anggraeni Michael Defoort, and A. Muis, “Consensus of Double-Integrator Multi-Agent Systems Using Decentralized Model Predictive Control,” in *5th International Conference on Control Engineering & Information Technology (CEIT-2017)*, Dec. 2017, pp. 28–32.

[7] T. V. Pham, N. Messai, and N. Manamanni, “Consensus of Multi-Agent Systems in Clustered Networks,” in *2019 18th European Control Conference (ECC)*, Jun. 2019, pp. 1085-1090, doi: 10.23919/ECC.2019. 8795970.

[8] H. Wei, Q. Lv, N. Duo, G. S. Wang, and B. Liang, “Consensus algorithms based multi-robot formation control under noise and time delay conditions,” *Applied Sciences (Switzerland)*, vol. 9, no. 5, 2019, doi: 10.3390/app9051004.

[9] D. Koung, I. Fantoni, O. Kermorgant and L. Belouaer, "Consensus-based formation control and obstacle avoidance for nonholonomic multi-robot system," *2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2020, pp. 92-97, doi: 10.1109/ ICARCV50220.2020.9305426.

[10] M. Maghenem, A. Bautista-Castillo, and E. Nuno, A. Loria, and E. Panteley, “Consensus-based Formation Control of Nonholonomic Robots using a Strict Lyapunov Function,” *IFAC-PapersOnLine*, vol. 9, no. 1, 2017, doi: 10.1016/j.ifacol.2017.08.406

[11] P. Anggraeni, “Decentralized leader-follower consensus for multiple cooperative robots under temporal constraints,” *Doctoral Thesis*, Université de Valenciennes et du Hainaut-Cambresis, Valenciennes, 2019.

[12] W. He, G. Chen, Q. Han, and F. Qian, “Network-based Leader-following Consensus of Nonlinear Multi-Agent System via Distributed Impulsive Control,” *Information Sciences,* vol. 380, 2017, doi: 10.1016/j.ins. 2015.06.005.

[13] A. Widyotriatmo, E. Joelianto, Y. Y. Nazaruddin, A. Prasdianto, and H. Bahtiar, “Implementation of leader-follower formation control of a team of non-holonomic mobile robots,” *International Journal of Computers, Communications and Control*, vol. 12, no. 6, pp. 871–885, 2017, doi: 10.15837/ijccc. 2017.6.2774.

[14] J. Hirata-Acosta, J. Pliego-Jiménez, C. Cruz-Hernádez, and R. Martínez-Clark, “Leader-follower formation control of wheeled mobile robots without attitude measurements,” *Applied Sciences (Switzerland)*, vol. 11, no. 12, Jun. 2021, doi: 10.3390/app11125639.

[15] H. Zhuang, K. Dong, N. Wang, and L. Dong, “Multi-Robot Leader Grouping Consistent Formation Control Method Research with Low Convergence Time Based on Nonholonomic Constraints,” *Applied Sciences (Switzerland)*, vol. 12, no. 5, Mar. 2022, doi: 10.3390/app12052300.

[16] T. Agustinah, “Formation Control of Multi-Robot using Virtual Structures with a Linear Algebra Approach,” *JAREE-Journal on Advance Research in Electrical Engineering*, vol. 4, no. 1, pp. 45–50, Apr. 2020.

[17] K. Cao, Y. Chen, S. Gao, H. Zhang, and H. Dang, “Multi-Robot Formation Control Based on CVT Algorithm and Health Optimization Management,” *Applied Sciences (Switzerland)*, vol. 12, no. 2, Jan. 2022, doi: 10.3390/app12020755.

[18] J. Qin, W. X. Zheng, H. Gao, Q. Ma, and W. Fu, “Containment Control for Second-Order Multiagent Systems Communicating over Heterogeneous Networks,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 9, pp. 2143–2155, Sep. 2017, doi: 10.1109/TNNLS.2016.2574830.

[19] I. Nielsen, Q. V. Dang, G. Bocewicz, and Z. Banaszak, “A methodology for implementation of mobile robot in adaptive manufacturing environments,” *Journal of Intelligent Manufacturing*, vol. 28, no. 5, pp. 1171–1188, Jun. 2017, doi: 10.1007/s10845-015-1072-2.

[20] F. Rubio, F. Valero, and C. Llopis-Albert, “A review of mobile robots: Concepts, methods, theoretical framework, and applications,” *International Journal of Advanced Robotic Systems*, vol. 16, no. 2. Mar. 01, 2019. doi: 10.1177/1729881419839596.

[21] P. Anggraeni, M. Mrabet, M. Defoort and M. Djemai, "Development of a wireless communication platform for multiple-mobile

robots using ROS," in *2018 6th International Conference on Control Engineering & Information Technology (CEIT)*, 2018, pp. 1-6, doi: 10.1109/CEIT.2018.8751845.

[22] J. J. Parmar and C. v Savant, “Selection of Wheels in Robotics,” *International Journal of Scientific & Engineering Research*, vol. 5, no. 10, 2014.

[23] G. Priyandoko, C. K. Wei, and M. S. H. Achmad, “Human following on ROS framework a mobile robot,” *SINERGI*, vol. 22, no. 2, pp. 77-82, 2021, doi: 10.22441/sinergi. 2018.2.002

[24] R. A. Priambudi and M. Mobed Bachtiar, “Penentuan Posisi Menggunakan Odometry Omniwheel,” in *The Indonesian Symposium on Robotic Systems and Control*, 2018, pp. 1–3.

[25] P. Lin, D. Liu, D. Yang, Q. Zou, Y. Du and M. Cong, "Calibration for Odometry of Omnidirectional Mobile Robots Based on Kinematic Correction," *2019 14th International Conference on Computer Science & Education (ICCSE)*, 2019, pp. 139-144, doi: 10.1109/ICCSE.2019.8845402.

[26] A. Jamalipour and C. B. Papadias, “Weighted Online Calibration for Odometry of Mobile Robots,” in *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, May 2017, pp. 1036–1042. doi: 10.1109/ICCW.2017.7962795.

[27] Sheldon Ijau Winston, Annisa Jamali, “Axis manipulation to solve Inverse Kinematics of Hyper-Redundant Robot in 3D Space,” *Journal of Integrated and Advanced Engineering (JIAE)*, vol. 2, no. 2, pp. 113-122, 2022, doi: 10.51662/jiae.v2i2.81