Journal of Robotics Research (JRR)
Volume 1, Issue 1, Date: 11-7-2024

Evaluating OMNI Robot Navigation with SLAM in CoppeliaSim: Hemangiomas and Nonhomogeneous Paths

Ata Jahangir Moshayedi[1],Yibin Xie[1],Maryam Sharifdoust[3],Amir Sohail Khan[1]
[1]School of Information Engineering Jiangxi University of Science and Technology, Ganzhou, Jiangxi, China.
[2] Department of Mathematics and Statistics,McMasterUniversity,Hamilton,1280,Ontario, Canada.
Email: [1] ajm@jxust.edu.cn ,2469184855@qq.com,mrsohail21@gmail.com, [2] sharifdoost@gmail.com
*Corresponding Author: Ata Jahangir Moshayedi: ajm@jxust.edu.cn

***Abstract*— This study aims to enhance the accuracy and efficiency of Omni robots in navigation, reducing errors and resource use to improve safety and task completion in various applications. Utilizing CoppeliaSim (Vrep) for simulation, the research focuses on the Omni robot's performance over homogeneous and non-homogeneous paths using SLAM. Key parameters such as running time, average velocity, body change, and error are analyzed with statistical methods to ensure robust findings. Results show a direct relationship between speed and tracking error, highlighting the need for optimized speed management. The study provides insights into the robot's performance under different conditions, offering valuable data for further optimization. This research contributes to the development of safer, more reliable robotic systems with applications in industrial automation, healthcare, and service robotics. Virtual prototyping reduces development costs and risks, promoting the adoption of advanced robotic technologies and enhancing productivity and safety in various fields.**

Keywords*— *OMNI robot platform, CoppeliaSim, Vrep, Simultaneous Localization and Mapping, SLAM, Levene's Test ,Independent samples t-test

I. INTRODUCTION

The term "omni" means "all" or "every," which is why OMNI robots are named to emphasize their ability to move in any direction—forward, backward, sideways, and rotationally without changing orientation[1]. This unique capability is typically achieved through specialized wheels, such as Mecanum or omni wheels, allowing the robot to navigate complex environments with high maneuverability and flexibility[2]. This design enhances precision and efficiency in tasks, improves navigation in intricate settings, and increases versatility and adaptability for various applications[3]. Path tracking is a vital task in robotic science to optimizes efficiency by following the most optimal routes, reducing travel time and energy consumption[4][5].Ensuring predictable movements enhances human-robot interaction and minimizes errors[6]. Consistent path tracking supports reliable performance and repeatability, essential for tasks like industrial automation and logistics. The omni platform's special design makes path tracking essential to ensure precision and accuracy in executing tasks, navigating obstacles, and maintaining safety. As the review paper shows, there are eight methods for the OMNI robot platform that can be selected based on the application's requirements for precision, robustness, computational complexity, and environmental conditions[7]. These methods can be described as follows: **Odometry:** Uses data from wheel encoders to estimate the robot's position and orientation by integrating the motion over time. This method is simple and requires minimal computational resources. However, it is prone to errors accumulating over time due to wheel slippage and uneven surfaces[8][9]. **Inertial Navigation System (INS):** Utilizes accelerometers and gyroscopes to track the robot's motion. The data is integrated to estimate the position and orientation. This method can provide high-frequency updates and is immune to wheel slippage, but sensor drift can cause errors to accumulate over time[10]. **Global Navigation Satellite System (GNSS):** Uses GPS or other satellite-based systems to determine the robot's position. It provides absolute positioning with good accuracy in open environments but is limited by signal availability and accuracy in indoor or obstructed environments[11]. **Visual Odometry:** Uses cameras to track features in the environment and estimate the robot's motion. This method can provide accurate position estimates and works well in environments with distinct visual features. However, it is computationally intensive and can be affected by changes in lighting or featureless environments[12]. **Simultaneous Localization and Mapping (SLAM):** Builds a map of the environment while simultaneously keeping track of the robot's position within the map. This method can provide accurate localization in unknown environments and handle dynamic changes but requires significant computational resources and can be complex to implement[13][14]. **LIDAR-Based Tracking:** Uses a LIDAR sensor to scan the environment and track the robot's position relative to detected features. It offers high accuracy and robustness to changes in lighting but can be expensive and data processing can be computationally intensive[15]. **Infrared (IR) Sensors:** Uses IR sensors to detect the position relative to known beacons or markers. This method is simple and cost-effective for specific applications but has limited range and can be affected by environmental conditions[16]. **Path Planning Algorithms:** Uses algorithms like A*, Dijkstra, or Rapidly-exploring Random Tree (RRT)

Received: 25-6-2024
Revised: 9-7-2024
Published: 11-7-2024

to plan the path based on a known map and update the robot's position using various sensors. This method can find optimal paths and handle complex environments but requires prior knowledge of the environment and can be computationally intensive[17]. The review of previous studies shows that among these methods, SLAM is the most popular due to its ability to handle dynamic and unknown environments effectively[18]. Therefore, SLAM is selected as the main focus of this paper for tracking the path. Besides, in this research CoppeliaSim (formerly named Vrep) as the highly ranked simulator among robotics simulators due to its flexibility, extensive features, and real-time capabilities selected to implement the robot platform. CoppeliaSim stands alongside Gazebo and Webots, offering robust support for complex robotic simulations. Its high ranking is attributed to its comprehensive tools and versatility[19].

The main contributions of the paper are listed as follows: Design and simulation of a real sized along with all features for OMNI robot platform in CoppeliaSim.Utilization of ROS and SLAM for path tracking missions. Comparison of the robot's performance on two types of paths, homogeneous and non-homogeneous, including obstacle navigation, to evaluate and compare the robot's performance with the help of inferential statistics (Levene's Test for Equality of Variances and t-test).The authors believe that the performance of the OMNI platform for navigating paths, hemangiomas, and nonhomogeneous studies in CoppeliaSim demonstrates advanced capabilities and improving safety and task completion in various applications. By leveraging CoppeliaSim (Vrep) as a robust simulation environment, the paper facilitates rapid prototyping and validation of robotic algorithms, significantly reducing development costs and risks before real-world deployment. This provides researchers with valuable insights to enhance the study of the OMNI platform in SLAM applications which based on researcher available resources reported rarely for this platform in the CoppeliaSim. The paper arrangement is as follows: Section 2, Omni Platform Design and Implementation, contains all steps from the implementation of real platform to the control and navigation of the robot, including the environment configuration in CoppeliaSim. Section 3, Experimental Results, presents the results of experiments conducted on the designed hemangiomas and nonhomogeneous paths. Section 4, Conclusion, provides a summary of the key findings and implications of the study.

II. OMNI PLATFORM DESIGN AND IMPLEMENTATION

2.1 Kinematics analysis of OMNI platform

An OMNI structure, in the context of robotics, typically refers to an omnidirectional robotic platform. This platform is characterized by its ability to move in any direction without needing to change its orientation. It features a chassis, individual motors for each wheel, various sensors (Lidar sensor in this research), and a control system for navigation and SLAM. This structure is highly maneuverable and suitable for applications needing precise navigation and control as it is shown in shown in Figure 1(A and B). An omnidirectional robot with four wheels can be mathematically modeled to understand its movement and control dynamics. The robot's coordinate system is centered at the chassis, with four omnidirectional wheels labeled ω_1,ω_2,ω_3 and ω_4, positioned counterclockwise from the upper left.

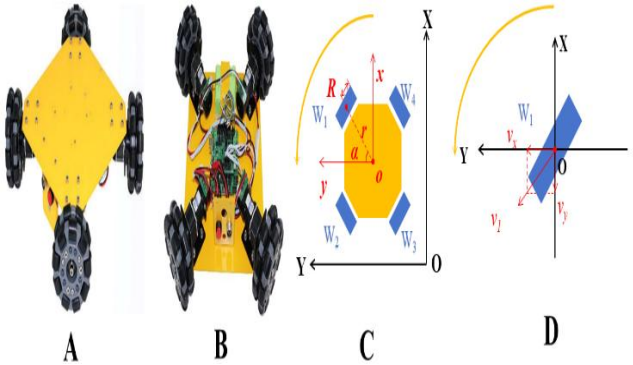

Figure 1.OMNI wheel and motion structure. (a) Real OMNI wheel. (b) Motion structure of the wheel. Figure 4 Kinematics analysis. (a) Chassis analysis. (b) Wheel speed analysis.

The key parameters include the v_w represent wheel linear velocities(Equation.(1)), ω_w indicated wheel angular speeds(Equation.(2)) and v_c shows chassis linear velocity(Equation.(3)).

$$v_w=[v_1,v_2,v_3,v_4]^T \quad (1)$$

$$\omega_w=[\omega_1,\omega_2,\omega_3,\omega_4]^T \quad (2)$$

$$v_c=[v_x,v_y]^T \quad (3)$$

Where in (Equation.(1)), v_1,v_2,v_3, and v_4 represent the linear velocity of one of the four wheels and in Wheel Angular Speeds (ω_w), Each element ω_1,ω_2,ω_3, and ω_4 represents the angular velocity of one of the four wheels. Chassis Linear Velocity (v_c), The elements v_x and v_y represent the velocity components of the chassis along the x and y axes, respectively. This vector describes the overall translational motion of the robot. chassis angular velocity ω_c , wheel radius *R*, and the distance from the chassis center to the wheels *r* as it is shown in Figure 1(C and D). The kinematic equations that relate the chassis velocity to the individual wheel velocities for a 45-degree wheel configuration are given by the transformation matrix (Equation.(4)).

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} = \frac{1}{R} \cdot \begin{bmatrix} 1 & -1 & -r \\ 1 & 1 & r \\ 1 & -1 & r \\ 1 & 1 & -r \end{bmatrix} \cdot \begin{bmatrix} v_x \\ v_y \\ \omega_c \end{bmatrix} \quad (4)$$

To find the chassis velocities from the wheel velocities, the inverse transformation matrix (Equation.(5)).

$$\begin{bmatrix} v_x \\ v_y \\ \omega_c \end{bmatrix} = \frac{R}{4} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ -\frac{1}{r} & \frac{1}{r} & -\frac{1}{r} & \frac{1}{r} \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} \quad (5)$$

These equations allow for the control and analysis of the robot's movements. The chassis linear velocities v_x and v_y represent the robot's movement along the x and y axes, while the chassis angular velocity ω_c represents the rotational speed of the chassis. The wheel velocities v_1,v_2,v_3,v_4 are derived from the combined chassis movements and rotations, influenced by the wheel radius *R* and the distance *r*[20]. This model is essential for precise navigation and control in applications such as SLAM.

2.2 Vrep simulation and implementation

As shown in Figure 2, simulating the OMNI robot platform involves considering mechanical, electrical, and control aspects. The design of an OMNI robot follows these steps: mechanical and electrical design, sensor performance simulation, integration, ROS initialization with CoppeliaSim (Vrep), mapping the robot using classical SLAM with GMapping and RViz, setting Move Base parameters, control system design, and environment configuration. Each of these steps is described in detail below:

1) Step 1: Mechanical and electrical design: The mechanical and electrical design process begins with the construction of an OMNI wheel model. To simplify and efficiently reproduce the omnidirectional wheel characteristics, a set of double-jointed ball-wheel structures is designed, considering its complex structure and dimensions (The radius of 5cm and Thickness of 6cm). Then a square pillar with dimensions of 25 cm*25 cm serves as the chassis. After that four omnidirectional wheels are then connected to the chassis at specific angles, as illustrated in Figure 2.

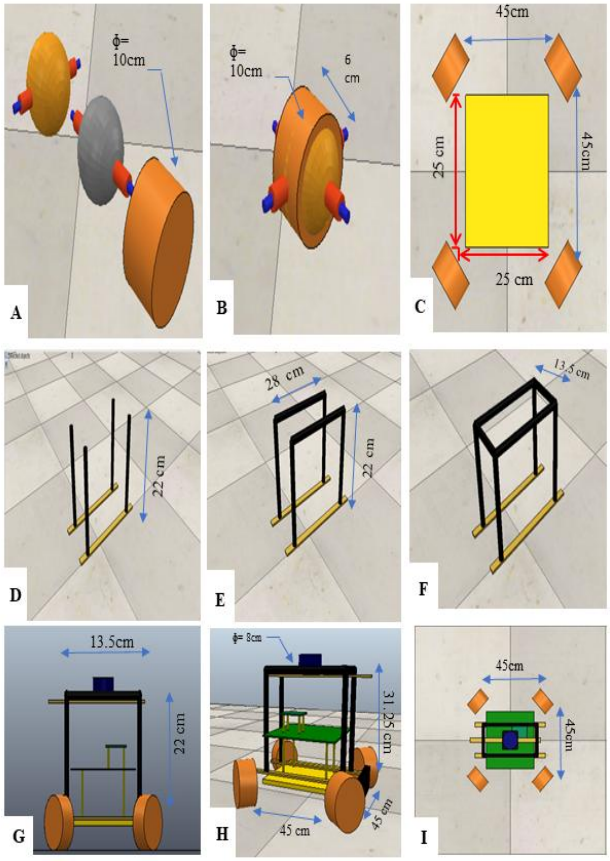

Figure 2. Build an OMNI robot platform. (A&B) wheel disassembled (C) Chaises (D,E,F) Robot Chassis holder.(G) robot Side view. (H) robot Right view (I) robot Top view.

As Figure 2 depicts the angles between neighboring wheels are set at 90 degrees. Then along the wheel and chassis, Completing the robot body structure involves integrating the frame, PCB board, and lidar model. It should be mentioned that all dimensions are based on the actual specifications of the previously implemented OMNI robot platform.

2) Step 2. Sensor Performance Simulation Section: This section focuses on simulating the performance of the Lidar Sensor as the obstacle avoidance and navigator robot integrated into the OMNI robot platform, including their accuracy, response times, and overall effectiveness in real-world scenarios. Then, the radar scripting for the RPLIDAR A1 sensor was finalized by referencing the Hokuyo URG lidar driver script available in the CoppeliaSim model library. Subsequently, the sensor's performance was thoroughly tested, as illustrated in Figure 3.

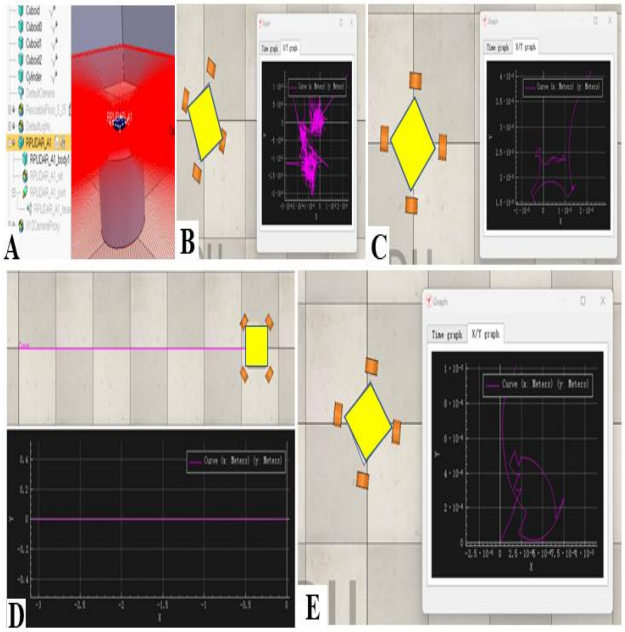

Figure 3.The Lidar sensor RPLIDAR A1 model and test A)The initial model (B,C) the lidar test ,(D,E) The robot movement.

3) Step 3. Integration: The final step involved assembling the platform, ensuring precise alignment and stability of mechanical components according to the design specifications. Following this, all electrical components such as motors, sensors, and controllers were integrated according to the real design platform. The integrated system was thoroughly tested for functionality and adjusted as needed to ensure optimal performance.

4) Step 4: ROS Initialization with CoppeliaSim : In order to navigate the robot Ros [21] used to control the Robot platform and then Installing ROS involves confirming ROS Kinetic installation on Ubuntu 16.04, launching the ROS Master using the command '**roscore**' ,starting CoppeliaSim after **roscore** is running, and verifying successful communication by checking the message "**plugin 'ROSInterface': load succeeded**.

5) Step 5 : Mapping the Robot Using Classical SLAM with GMapping and RViz: The process involves mapping the robot using the classical SLAM algorithm with GMapping and RViz for accurate navigation and positioning in the simulated environment. Generate a map using lidar scan data in RViz and enable navigation and positioning. Transmit real-time

pose data from the robot in the CoppeliaSim simulation to ROS in the odom format and broadcast the coordinate transformation between the robot and the odom. Subscribe to pose data in RViz, including three-dimensional coordinates, directional angle, linear velocity, and rotational angular velocity. Establish initial pose and synchronization using a Dummy object, synchronize with the lidar coordinate system, perform coordinate transformations, and publish results to ROS. Invoke the GMapping function package in ROS for SLAM construction. Then, finally to draw the environment map enable the ROS [21].

Remote keyboard function and subscribe to the **/cmd_vel** topic in the robot script for keyboard control. Manipulate the robot to move in the map and complete the map drawing, showing the movement and drawing process in RViz in real time. Save the map using the command **rosrun map_server map_saver** in the terminal, resulting in the **map.pgm** picture file. After finishing all to draw the map the enable the ROS remote keyboard function to control the robot. In the robot script, subscribe to the **/cmd_vel** topic to receive keyboard control commands.

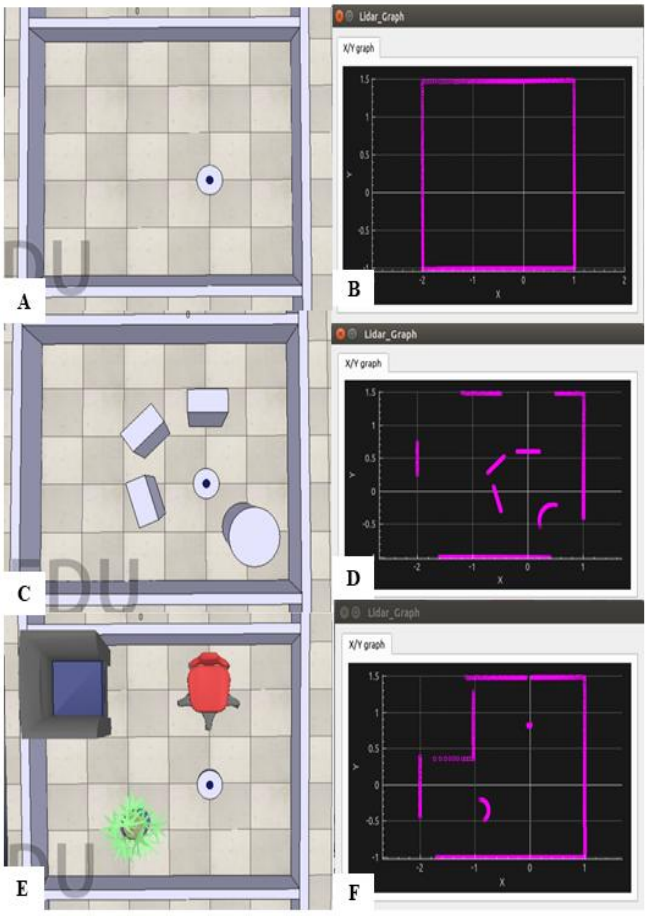

Figure 4.Lidar test A,B)The test in a simple chamber (C,D)with various obstacles. (E,F)in a chamber with various obstacles.

Next, manipulate the robot using the keyboard to move it within the map, which will help in completing the map drawing. The movement and map drawing process will be shown in real time in RViz. Once the map is created, execute the following command in the terminal to save the map: **rosrun map_server map_saver**. This step will generate a map.pgm picture file, as illustrated in Figure 4.

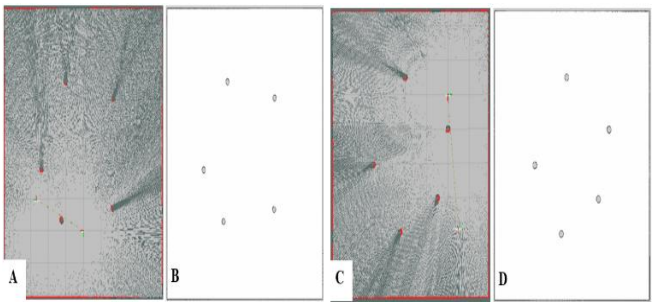

Figure 5. Map construction (CoppeliaSim).Cartography. (a) GMapping running and Map Hemangiomas path in the form of Circle ,(C,D) GMapping running and Map non-Hemangiomas path in the form of Special Path.

Figure 5 illustrates the mapping arena for both Hemangiomas and non-Hemangiomas paths, utilizing GMapping in the CoppeliaSim simulator.

6) Step 6 : Setting Move Base Parameters: The **move_base** package plays a crucial role in ROS navigation, handling global and local path planning by subscribing to Lidar, map data, and Adaptive Monte Carlo Localization (AMCL) positioning data. AMCL is a probabilistic algorithm used for mobile robot localization, which estimates the robot's position and orientation based on sensor data[19]. It translates these paths into speed information, enabling efficient robot navigation. Essential parameter files for **move_baseinclude:base_local_planner_params,yaml** costmap_common_params.yaml,global_costmap_params.yaml,local_costmap_params.yaml,each contributing to specific aspects of navigation. Table 1 summarizes the parameter settings for these files, defining key configurations for move_base functionality. Finally, to test target point navigation using *RViz* and the **move_base** topic control, start by configuring the relevant startup files following the **move_base** topic format.Add the path plug-in to display the pre-planned and action paths in *RViz*, including the pose indicator. Additionally, display the results of the particle filter in the navigation algorithm as arrows. Begin by initiating the CoppeliaSim simulation and launching the appropriate file. Then, use the "**2D Nav Goal**" tool in *RViz* to designate the destination on the map. This action prompts the robot to plan the optimal path, navigate around obstacles, and reach the target point, as depicted in Figure 6.

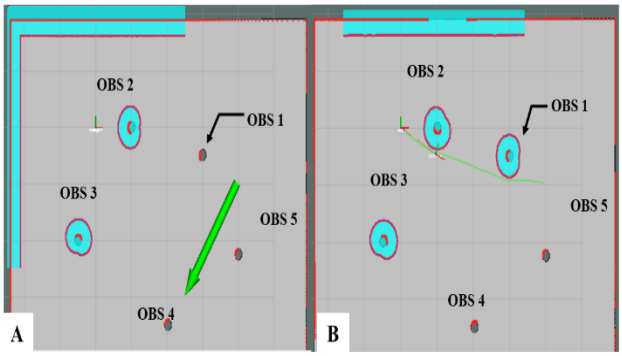

Figure 6. Robot navigation. (a) nav_goal. (b) Navigating.

Alternatively, by using the directly publish the target point coordinates and pose to **move_base** by executing the

command (“**rostopic pub /move_base_simple/goal geometry_msgs/PoseStamped '{header: {frame_id: "map"}, pose: {position: {x: 3.0, y: 0, z: 0}, orientation: {x: 0, y: 0, z: 0, w: 1}}}**”) in the terminal. This command instructs the robot to navigate to the coordinates [x: 3.0, y: 0, z: 0] within the map coordinate system, with the target pose already specified. The robot will move to the specified coordinates, reaching the target pose as instructed.

TABLE 1. MOVE_BASE PARAMETER SETTINGS. BASE_LOCAL_PLANNER_PARAMS(A),COSTMAP_COMMON_ PARAMS(B), GLOBAL_COSTMAP_PARAMS(C), LOCAL_COSTMAP_PARAMS(D).

	Parameter	Value
A	controller_frequency	3
	max_vel_x , max_vel_y	0.2,-0.6
	min_vel_x, min_vel_y	0.1, -0.2
	max_vel_theta,max_in_place_vel_theta	2
	min_vel_theta, min_in_place_vel_theta	-2
	escape_vel, acc_lim_x, acc_lim_y	-0.1, 2.5
	acc_lim_theta ,holonomic_robot	2, true
	yaw_goal_tolerance	0.17
	xy_goal_tolerance	0.1
	latch_xy_goal_tolerance	true
B	obstacle_range, raytrace_range	2.5
	footprint	[[0, 0.25], [0.2, 0.2], [0.25, 0], [0.2, -0.2],[0, -0.25], [-0.2, -0.2], [-0.25, 0], [-0.2, 0.2]]
	inflation_radius	0.2
	resolution	0.01
	observation_sources	laser_scan_sensor
	laser_scan_sensor	{sensor_frame: laser_link, data_type: LaserScan, topic: /scan, marking: true, clearing: true}
C	global_frame , robot_base_frame	Map, base_link
	update_frequency publish_frequency	2.0, 1.0
	static_map	true
	rolling_window	false
D	global_frame, robot_base_frame	Map, base_link
	update_frequency publish_frequency	5.0, 2.0
	static_map rolling_window	False, true
	Width, height	5

To complete this step, several tests were conducted to control the robot platform based on the parameters listed in Table 1. The robot was operated using a computer keyboard to control and verify its movement (Figure 7).

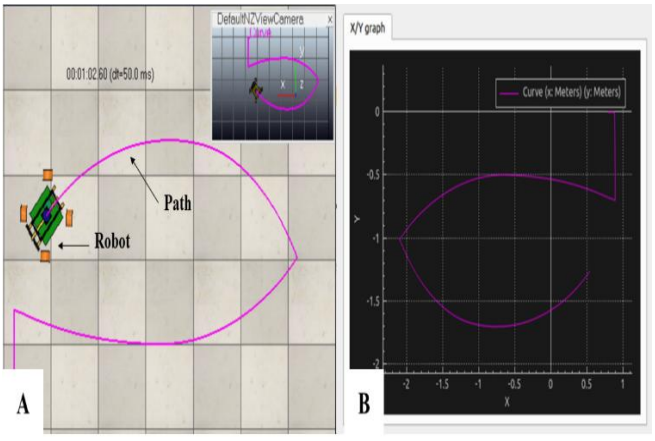

Figure 7. The navigation test over the keyboard and testing the move base parameter.(A) the robot movement (B) the robot movement plot

7) Step 7. Control System Design :

The control loop for the Robot platform shown in Figure 8.

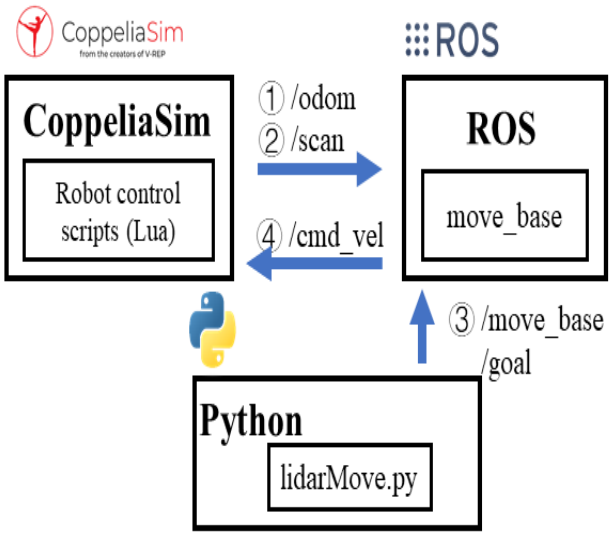

Figure 8.Omni robot Navigation system Block diagram.

As the Figure 8 shown three parts of CoppeliaSim, Ros and Python used to control the robot, The Lua in CoppeliaSim used to Lua scripting in CoppeliaSim empowers users to extend the platform's capabilities beyond basic simulation, facilitating advanced control, automation, and integration tasks crucial for robotics research, development, and education. ROS enhances simulation integration and control, while Python extends Coppeliasim's capabilities through flexible scripting for automation and customization, making both essential tools in robotics research and development within CoppeliaSim environments[19].The robot's Lua control script initially publishes two topics, /scan and /odom, conveying distance information from LIDAR detection in CoppeliaSim and the robot's current attitude information to the ROS move_base function package. Simultaneously, a Python script publishes the target point's coordinates and the robot's pose to move_base via the /move_base/goal topic. Within move_base, this data is analyzed to plan a motion path, and the resulting real-time corrected velocity information is published to the /cmd_vel ROS topic. The robot's control script then subscribes to this topic to obtain velocity information, thus configuring the motors and enabling robot navigation movement.

8) Step 8. Environment Configuration:

As mentioned before , the designed robot is planned to be tested on two distinct paths: the Hemangiomas Path, represented by the Circle Path with the (length 1573 cm ,radius of 2.477m), and the Non-Hemangiomas Path, represented by the Special Path (length 1390 cm) as it shown in Figure 9 and Table 2.

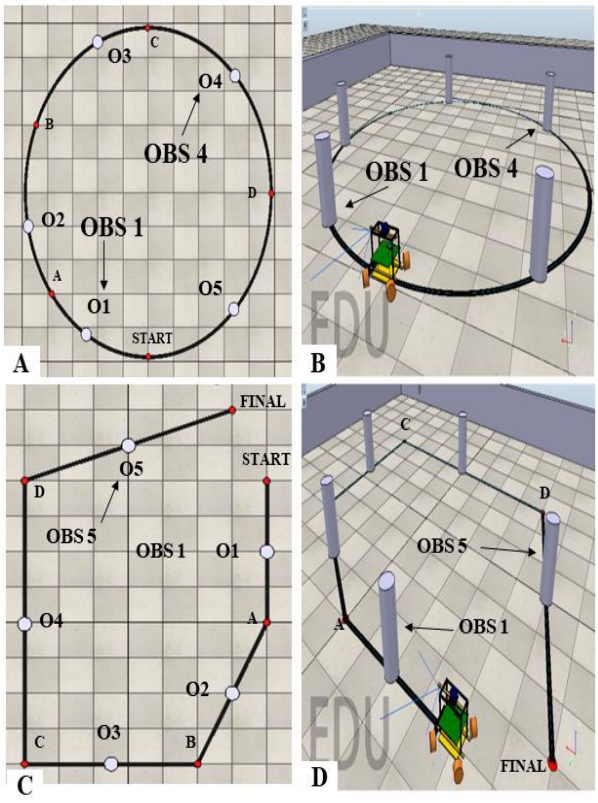

Figure 7.Robomot movement over Hemangiomas (A,B) and non-Hemangiomas(C,D) Path map in CoppeliaSim. Points S, A, B, C, D, and F have specific coordinates, while obstacles O1 to O5 are positioned at different locations.

TABLE 2. HEMANGIOMAS(H) (CIRCLE PATH: 1573 CM), NON_HEMANGIOMAS(NH) (SPECIAL PATH : 1390 CM) POINT NAME , LOCATION AND COORDINATION IN METER , OBSTACLE DIAMETER: 20 CM HEIGHT: 60 CM.

Points		H_Path (x,y) m	NH_Path (x,y) m
Key Point	S	(0.00,0.00)	(0.00,0.00)
	A	(2.00 , 0.00)	(2.00 , 0.00)
	B	(4.00, -1.00)	(4.00, -1.00)
	C	(4.00, -3.50)	(4.00, -3.50)
	D	(0.00, -3.50)	(0.00, -3.50)
	F	(0.00,0.00)	(-1.00, -0.50)
Obstacles	O1	(1.23, 2.10)	(-1.00, 2.00)
	O2	(2.38, 0.50)	(1.00,1.50)
	O3	(1.005, -2.225)	(2.00, -0.25)
	O4	(-1.72, -1.725)	(0.025,-1.50)
	O5	(-1.695, 1.725)	(-2.50, 0.00)

Table 2 shows the coordinates of key points and obstacles in both map environments. Points S, A, B, C, D, and F have specific coordinates, while obstacles O1 to O5 are positioned at different locations.

III. EXPERIMENTS RESULT

Based on the simulated OMNI robot platform in the COPPELIASIM environment, an experiment was conducted to evaluate the robot's performance in the presence of obstacles over two afordement path with the different speed.As the Figure 10 (A and C) shown During the test, the robot's speed varied from 10 cm/s in increments of 5 cm/s. Successful path tracking was reported at each speed. Four key parameters were assessed: Running Time (seconds), Average Velocity (cm/s), Average Body Change (rad), and Average Error (cm). Each path was tested 10 times, with the results summarized in Table 3 and illustrated in Figure 10 (B and D).

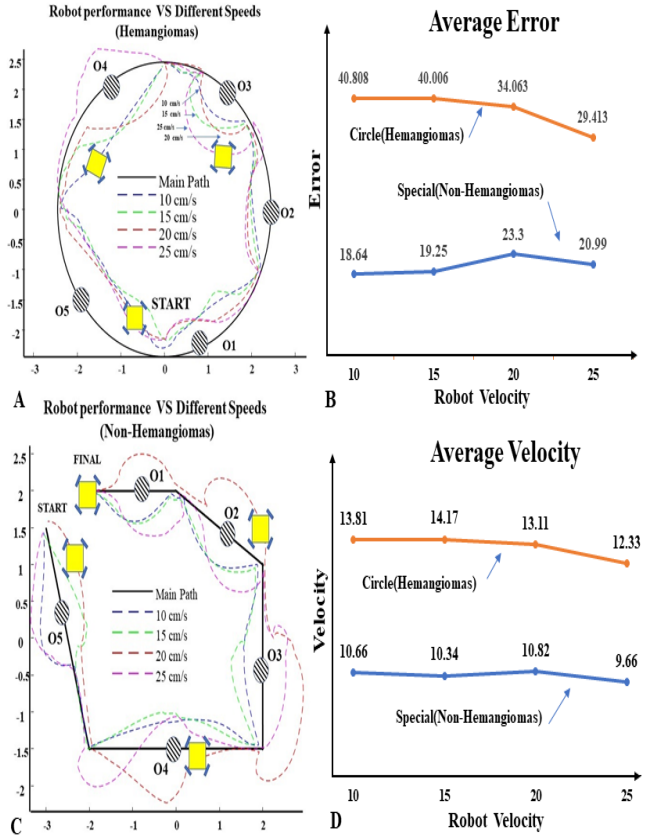

Figure 8. Omnirobot different Path tracking over different speed(A and C). Omni robot performance with respect to Error and Velocity over Hemangiomas and Non_Hemangiomas (B and D).

TABLE 3.HEMANGIOMAS (H) LENGTH 1537 CM AND NON_HEMANGIOMAS (NH) PATH LENGTH 1390 CM WITH FIVE OBSTACLES SUCCESSFUL PATH TRACKING. SECOND(S), AVERAGE_VELOCITY(AV), AVERAGE _ERROR (AE), AVERAGE _BODY CHANGE(ABC).

Velocity (cm/s)	Running Time (s)		AV (cm/s)	
	H	NH	H	NH
10	111.3	130.39	13.81	10.66
15	108.4	134.46	14.17	10.34
20	117.2	128.46	13.11	10.82
25	124.6	143.91	12.33	9.66
Velocity (cm/s)	**ABC (rad)**		**AE (cm)**	
	H	**NH**	**H**	**NH**
10	-0.069	-0.165	40.66	18.64
15	-0.271	-0.13	40.12	19.25
20	-0.215	0.09	34.86	22.3
25	-0.056	0.23	30.7	20.99

As the Table 3 results shown, the robot exhibited different movement trajectories across various paths. Based on the importance of velocity and path tracking error, an

investigation was initiated using statistical analysis to evaluate these parameters and shown in Table 4 and Figure 10 (B and D).As it depicted in Figure 10, the average error for the Hemangiomas path represented by circle is higher than that for the non-Hemangiomas(special) path at all robot velocities. Moreover, although the average error for the Hemangiomas path decreases as robot velocity increases, the average error for the non-Hemangiomas path shows a fluctuating pattern under the Hemangiomas path. A similar trend is observed for average velocity based on robot velocity and the paths. Therefore, inferential statistics are required to draw conclusions. As it shown in Table 4 and Figure 12, Considering the Velocity parameter: In the Hemangiomas Path, the robot moves faster with a mean velocity of 13.355 units, but with greater variability (standard deviation of 0.81279 units) and less precision (standard error of the mean at 0.40640 units).

TABLE 4 . HEMANGIOMAS(H) AND NON_HEMANGIOMAS(NH) PATH AND OMNI ROBOT PERFORMANCE OVER VELOCITY AND ERROR

Parameter	Path	Mean	Std. Deviation	Std. Error Mean
Velocity	H	13.3550	0.81279	0.40640
	NH	10.3700	0.51368	0.25684
Error	H	36.58500	4.715602	2.357801
	NH	20.29500	1.666743	0.833372

In contrast, in the Non-Hemangiomas Path, the robot moves slower with a mean velocity of 10.370 units, exhibiting less variability (standard deviation of 0.51368 units) and greater precision (standard error of the mean at 0.25684 units).Considering the Error parameter: In the Hemangiomas Path, the robot has higher positional errors with a mean error of 36.58500 units, more variability (standard deviation of 4.715602 units), and less precision (standard error of the mean at 2.357801 units). In the Non-Hemangiomas Path, the robot has lower positional errors with a mean error of 20.29500 units, less variability (standard deviation of 1.666743 units), and greater precision.

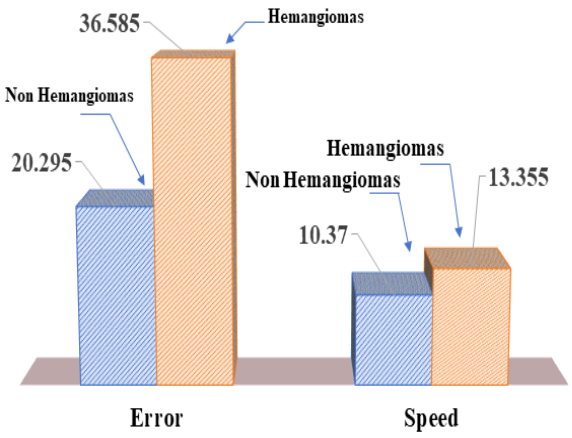

Figure 9.The robot performance analysis over different path VS error and speed in path tracking mission

IV. CONCLUSION

This paper describes the steps involved in simulating and implementing an OMNI robot in CoppeliaSim, focusing on path tracking over Hemangiomas and Non-Hemangiomas paths. The analysis centers on two key parameters: average error and average velocity with the help of Levene's Test for Equality of Variances and t-test statistical analysis .Path tracking for an OMNI robot using SLAM as the main target of this paper involves navigating through a pre-mapped environment while continuously updating the robot's position and orientation. The effectiveness of the OMNI robot's path tracking is assessed by measuring the average positional error and the average velocity, ensuring precise and efficient movement along the designated paths. As the results shown, Over the Non-Hemangiomas Path robot Provides a slower but more consistent and precise navigation experience with significantly lower errors, making it preferable for applications where accuracy is crucial. Overall, the choice of path type should consider the trade-off between speed and accuracy, with the Hemangiomas path favoring speed and the non-Hemangiomas path favoring precision. As expected, the mean velocity and error are higher for the circle path compared to the special path. Based on Table 2, these differences are significant at the 5 percent level (P-value = 0.001). Analysis of Independent Samples T-Test Results concerning the Velocity shows over the Levene's Test for Equality of Variances. The significance value (0.285) is greater than 0.05, indicating that the assumption of equal variances holds true. In the equal variation assumed, the test shows a significant difference in velocity between the two path types, with p-value 0.001, well below the threshold of 0.05. The mean velocity difference of 2.98500 units is statistically significant, with the confidence interval indicating that the true mean difference lies between 1.80864 and 4.16136 units (assuming equal variances. With respect to Error, Levine's Test for Equality of Variances shows that the significance value (0.035) is less than 0.05, indicating that the assumption of equal variances does not hold true. By using the t-test for Equality of Means, in case not assuming equal variances, shows a significant difference in error between the two path types, with p-value 0.004, well below the threshold of 0.05. The mean error difference of 16.290 units is statistically significant, with the confidence interval indicating that the true mean difference lies between 9.15066 and 23.42934 units. These results indicate that while the robot moves faster on the Circle path, it also incurs significantly higher positional errors compared to the Special path. This trade-off between speed and accuracy should be considered when choosing path types for specific applications. As the result shows, according to the average speed of 10.37 for a specific route, with 95% confidence, it can be stated that the speed on the circular route is 20% to 40% higher than on the non-homogeneous path. When comparing the amount of error in path tracking, the error has increased by 50% to 100% on the homogeneous path. This indicates that although the speed is lower on the non-homogeneous path, the accuracy of path tracking is better. Overall, the test results show that while the SLAM method can successfully reach the initial and final points, it has limitations in achieving complete route tracking according to the designed path, especially in the presence of obstacles. It appears that integrated methods might offer a solution, and this approach will be considered for future work in this research.

V. REFERENCE

[1] Ioan Doroftei, V. Grosu, and Veaceslav Spinu, “Design and Control of an Omni-directional Mobile Robot,” Springer eBooks, pp. 105–110, Aug. 2008, doi: https://doi.org/10.1007/978-1-4020-8737-0_19.

[2] A. Eirale, M. Martini, L. Tagliavini, D. Gandini, M. Chiaberge, and G. Quaglia, “Marvin: an Innovative Omni-Directional Robotic Assistant for Domestic Environments,” Sensors, vol. 22, no. 14, p. 5261, Jul. 2022, doi: https://doi.org/10.3390/s22145261.

[3] C. Prados Sesmero, L. R. Buonocore, and M. Di Castro, “Omnidirectional Robotic Platform for Surveillance of Particle Accelerator Environments with Limited Space Areas,” Applied Sciences, vol. 11, no. 14, p. 6631, Jul. 2021, doi: https://doi.org/10.3390/app11146631.

[4] Ata Jahangir Moshayedi, Atanu Shuvam Roy, L. Liao, Amir Sohail Khan, Amin Kolahdooz, and A. Eftekhari, “Design and Development of Foodiebot Robot: from Simulation to Design,” IEEE access, pp. 1–1, Jan. 2024, doi: https://doi.org/10.1109/access.2024.3355278.

[5] A. Eirale, M. Martini, L. Tagliavini, D. Gandini, M. Chiaberge, and G. Quaglia, “Marvin: an Innovative Omni-Directional Robotic Assistant for Domestic Environments,” Sensors, vol. 22, no. 14, p. 5261, Jul. 2022, doi: https://doi.org/10.3390/s22145261.

[6] Mostafa Mo. Massoud, A. Abdellatif, and Mostafa, “Different Path Planning Techniques for an Indoor Omni-Wheeled Mobile Robot: Experimental Implementation, Comparison and Optimization,” Applied sciences, vol. 12, no. 24, pp. 12951–12951, Dec. 2022, doi: https://doi.org/10.3390/app122412951.

[7] B. Fares, Haïfa Souifi, Mohsen Ghribi, and Yassine Bouslimani, “Omnidirectional Platform for Autonomous Mobile Industrial Robot,” 2021 IEEE 3rd Eurasia Conference on IOT, Communication and Engineering (ECICE), Oct. 2021, doi: https://doi.org/10.1109/ecice52819.2021.9645621.

[8] D. Nister, O. Naroditsky and J. Bergen, "Visual odometry," Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004., Washington, DC, USA, 2004, pp. I-I, doi: 10.1109/CVPR.2004.1315094.

[9] P. Jain, "Odometry and motion planning for omni drive robots," 2014 Innovative Applications of Computational Intelligence on Power, Energy and Controls with their impact on Humanity (CIPECH), Ghaziabad, India, 2014, pp. 164-168, doi: 10.1109/CIPECH.2014.7019080..

[10] P. Li, B. Leng and H. Fu, "Autonomous positioning of omnidirectional mobile robot based on visual inertial navigation," 2020 39th Chinese Control Conference (CCC), Shenyang, China, 2020, pp. 3753-3758, doi: 10.23919/CCC50068.2020.9189018.

[11] Y. Yu, W. Gao, C. Liu, S. Shen and M. Liu, "A GPS-aided Omnidirectional Visual-Inertial State Estimator in Ubiquitous Environments," 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 2019, pp. 7750-7755, doi: 10.1109/IROS40897.2019.8968519.

[12] T. Yokota, K. Watanabe, K. Kobayashi and Y. Kurihara, "Development of visual odometry component by using omni-directional camera," SICE Annual Conference 2011, Tokyo, 2011, pp. 2149-2151.

[13] D. Burschka and G. D. Hager, "V-GPS(SLAM): vision-based inertial system for mobile robots," IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004, New Orleans, LA, USA, 2004, pp. 409-415 Vol.1, doi: 10.1109/ROBOT.2004.1307184.

[14] A. M. Derbas and T. A. Tutunji, "SLAM Algorithm for Omni-Directional Robots based on ANN and EKF," 2023 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), Amman, Jordan, 2023, pp. 80-86, doi: 10.1109/JEEIT58638.2023.10185708.

[15] A. S. Kundu, O. Mazumder, A. Dhar, P. K. Lenka, and S. Bhaumik, “Scanning Camera and Augmented Reality Based Localization of Omnidirectional Robot for Indoor Application,” Procedia Computer Science, vol. 105, pp. 27–33, 2017, doi: https://doi.org/10.1016/j.procs.2017.01.183.

[16] Durst, V., Hagel, D., Vander, J., Blaich, M., Bittel, O. (2011). Designing an Omni-Directional Infrared Sensor and Beacon System for the Eurobot Competition. In: Obdržálek, D., Gottscheber, A. (eds) Research and Education in Robotics - EUROBOT 2011. EUROBOT 2011. Communications in Computer and Information Science, vol 161. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-21975-7_10

[17] Massoud, M.M.; Abdellatif, A.; Atia, M.R.A. Different Path Planning Techniques for an Indoor Omni-Wheeled Mobile Robot: Experimental Implementation, Comparison and Optimization. Appl. Sci. 2022, 12, 12951. https://doi.org/10.3390/app122412951].

[18] Z. Wang and M. Feng, "Research on Omnidirectional SLAM based on Vehicle-mounted Multi-Camera System," 2021 6th International Symposium on Computer and Information Processing Technology (ISCIPT), Changsha, China, 2021, pp. 798-802, doi: 10.1109/ISCIPT53667.2021.00167

[19] .A. J. Moshayedi, S. M. Zanjani, D. Xu, X. Chen, G. Wang and S. Yang, "Fusion BASED AGV Robot Navigation Solution Comparative Analysis and Vrep Simulation," 2022 8th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS), Behshahr, Iran, Islamic Republic of, 2022, pp. 1-11, doi: 10.1109/ICSPIS56952.2022.10044044

[20] Azizi, M.R.; Rastegarpanah, A.; Stolkin, R. Motion Planning and Control of an Omnidirectional Mobile Robot in Dynamic Environments. Robotics 2021, 10, 48. https://doi.org/10.3390/robotics10010048

[21]. A. Jahangir Moshayedi, K. S. Reza, A. Sohail Khan, and A. Nawaz, “ Integrating Virtual Reality and Robotic Operation System (ROS) for AGV Navigation”, EAI Endorsed Trans AI Robotics, vol. 2, Apr. 2023.