Development of a ROS 2 Based Path Planning A-star for Opponent Avoidance in Wheeled Soccer Robot

Agus Khumaidi[1], Muhammad Jardin Saputra[2], Dhika Arya Pratama[3], Ryan Yudha Adhitya[4], Sholahuddin Muhammad Irsyad[5]

[1, 2, 3, 4] Automation Engineering Study Program of Politeknik Perkapalan Negeri Surabaya

[5] Marine Electrical Engineering Study Program of Politeknik Perkapalan Negeri Surabaya

E-mail: aguskhumaidi@ppns.ac.id[1], muhammadjardin@student.ppns.ac.id[2], dhikaarya12@student.ppns.ac.id[3], ryanyudhaadhitya@ppns.ac.id[4], muhammad.irsyad@ppns.ac.id[5]

***Abstract* — The Indonesian Wheeled Soccer Robot Contest requires autonomous robots to navigate safely in dynamic environments where opponent robots continuously change their positions. Conventional Single A* path planning generates the navigation path only once before robot movement, making it unable to adapt to obstacle movement during navigation. This study proposes a ROS 2-based Dynamic A-star path planning system for opponent avoidance in wheeled soccer robots. The proposed system performs real-time path replanning whenever the detected obstacle position changes, enabling the robot to generate an alternative collision-free trajectory. The system was evaluated in the Gazebo simulation environment using eight navigation scenarios and compared with the conventional Single A-star approach. The same robot starting positions and goal locations were used in both approaches, while the Dynamic A-star method extended the baseline configuration from one obstacle to two obstacles, including one moving obstacle. Experimental results showed that the proposed Dynamic A-star method achieved a 100% navigation success rate with a 0% collision rate, whereas the conventional Single A-star algorithm achieved a 0% navigation success rate with a 100% collision rate after obstacle movement. The average travel time increased from 19.50 s for Single A-star to 29.95 s for Dynamic A-star, while the average path length increased from 4.08 m to 6.76 m due to additional detours and path replanning. These results demonstrate that Dynamic A-star improves navigation adaptability and collision avoidance in dynamic environments. In addition, the ROS 2 framework provided real-time communication among perception, path planning, and robot control modules, supporting adaptive autonomous navigation for wheeled soccer robots.**

Keywords—A star Algorithm, Dynamic Opponent Avoidance, Path Planning, ROS 2, Wheeled Soccer Robot..

I. INTRODUCTION

The Indonesian Wheeled Soccer Robot Contest (KRSBI-B) requires autonomous robots to navigate efficiently in dynamic environments where opponent robots continuously change their positions. During a match, the robot must reach the target while avoiding collisions with moving opponents. Therefore, an adaptive path planning system is required to generate safe and efficient trajectories under changing environmental conditions [1].

The A-star (A*) algorithm is commonly used for mobile robot navigation because it can determine an optimal path based on travel cost and heuristic estimation [2], [3]. Recent studies have further investigated improved A-star based approaches for dynamic obstacle avoidance and path planning in changing environments [2], [4]. However, previous studies on wheeled soccer robots generally used a single planning approach, where the path is generated only once before the robot starts moving. This limitation makes the robot unable to update its trajectory when an opponent changes position, increasing the risk of collision [5], [6].

To address this problem, this study develops an A-star based path planning system with a dynamic replanning mechanism for opponent avoidance. The proposed system uses an OmniVision 360° camera and You Only Look Once version 5 small (YOLOv5s) object detection to obtain the real-time position of opponent robots. The detected opponent is represented as a dynamic obstacle in the cost map, allowing the A-star algorithm to recalculate the path whenever the opponent position changes [7].

The main contribution of this research is the integration of real-time opponent detection and dynamic A-star path replanning using Robot Operating System 2 (ROS 2) and Gazebo. This approach is expected to improve the navigation performance of wheeled soccer robots by reducing collision risk and enabling more adaptive movement in dynamic match environments [8].

II. METHODOLOGY

This chapter describes the research methodology, including the overall research flow, system architecture, A-star-based path planning algorithm, and experimental procedures used to evaluate the proposed navigation system..

A. Study Literature

This study aims to develop a ROS 2-based A-star path planning system for opponent avoidance in wheeled soccer robots. The research methodology consists of several stages, as illustrated in Fig. 1.

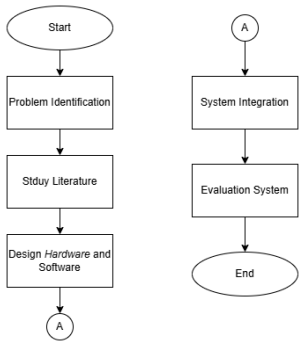


Fig. 1. Study Literature Diagram

@ 2026 Agus Khumaidi, Muhammad Jardin Saputra, Dhika Arya Pratama, Ryan Yudha Adhitya

Based on Figure 1, the research begins with problem identification and literature review to analyze the challenges of robot navigation in dynamic environments and to investigate previous studies related to A-star path planning, ROS 2, and mobile robot navigation [9]. Subsequently, the system block diagram and the system workflow are designed to define the communication between hardware and software modules.

The next stage involves the development of the robot system, including hardware, mechanical, and software design. All components are then integrated through the synchronization process to ensure proper communication between sensors, the embedded controller, and the ROS 2-based navigation system [10]. After the integration process is completed, the proposed system is evaluated through navigation experiments involving dynamic opponent scenarios. If the system does not meet the expected performance, improvements are carried out before repeating the testing process. Finally, the experimental results are analyzed to evaluate the effectiveness of the proposed A-star-based path planning system for opponent avoidance [11].

B. Proposed Robosoccer System Architecture

The robosoccer system was designed by integrating computer vision, embedded control, and robot actuators to support autonomous navigation during a soccer match. The system consists of an OmniVision 360° camera, STM32F407VGT6 microcontroller, Arduino Mega 2560, rotary encoders, proximity sensors, HWT101CT inertial measurement unit (IMU), motor drivers, omnidirectional wheel motors, dribbler motors, solenoid, and LCD display. The overall hardware and software integration of the proposed system is shown in Fig. 2.

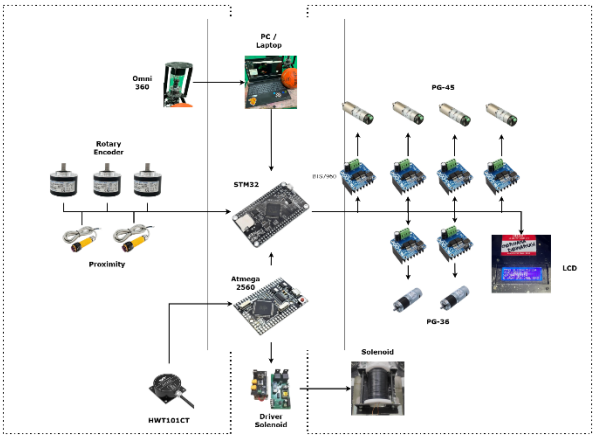


Figure 2. System Design

Based on Fig.2, the OmniVision 360° camera captures the surrounding environment and transmits image data to a laptop running ROS 2. The laptop performs object detection using the YOLOv5s model and generates navigation commands through the A-star-based path planning algorithm. These commands are sent to the STM32F407VGT6 microcontroller via serial communication to control the omnidirectional drive system.

The STM32 receives feedback from rotary encoders and proximity sensors for motion control, while the HWT101CT IMU communicates with the Arduino Mega 2560 to provide robot heading information. Finally, the processed control signals are delivered to the motor drivers, wheel motors, dribbler motors, and solenoid, enabling the robot to execute autonomous navigation and ball-handling tasks during the match.

C. Algorithm Path Planning A-star

The proposed navigation system employs the A-star algorithm to determine a collision-free path from the robot's initial position to the target while avoiding obstacles. The soccer field is represented as a two-dimensional occupancy grid map, where each grid cell corresponds to either free space or an occupied area [12]. The occupancy grid uses a spatial resolution of 250 mm × 250 mm per cell, allowing the environment and detected obstacles to be represented in discrete grid coordinates. The robot's initial position is defined as the start node, while the desired destination is defined as the goal node. Opponent robots detected by the vision system are represented as dynamic obstacles that cannot be traversed during path generation [13].

The A-star algorithm evaluates each candidate node using the following evaluation function:

$$f(n) = g(n) + h(n) \quad (1)$$

where :

- f(n) : total evaluation cost
- g(n) : accumulated travel cost from the start node to the current node
- h(n) : heuristic estimation of the remaining cost from the current node to the goal.

In this study, the A-star planner uses the Euclidean distance as the heuristic function to estimate the cost from the current node to the goal, expressed as:

$$h(n) = \sqrt{(x_g - x_n)^2 + (y_g - y_n)^2} \quad (2)$$

where :

- (x_n, y_n) : current node position
- (x_g, y_g) : goal position

Fig. 3 illustrates an example of path generation using the A-star algorithm. In a static environment (left), the algorithm generates a path from the start position to the goal. When an obstacle blocks the planned route (right), the path is recalculated to produce an alternative trajectory that avoids the obstacle while still reaching the target [14]. This mechanism enables the robot to navigate efficiently in environments containing both static and dynamic obstacles.

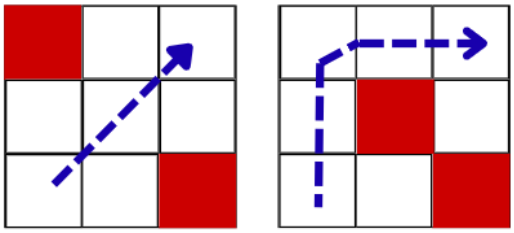


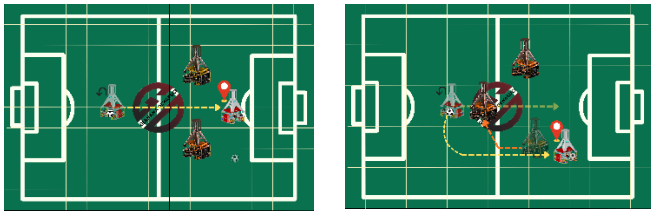
Figure 3. Illustration of A-star Path Planning

D. Opponent Avoidance

The proposed navigation system employs a dynamic opponent avoidance strategy by integrating real-time object detection with the A-star-based path planning algorithm. Opponent robots detected by the OmniVision 360° camera are continuously updated in the occupancy grid map, allowing the navigation system to recognize changes in the surrounding environment [15]. The updated obstacle information is processed by the ROS 2 framework to generate a new navigation path whenever the current trajectory is blocked by a moving opponent [1].

Unlike the conventional A-star algorithm, which performs path planning only once before navigation, the proposed method continuously updates the robot trajectory according to the latest opponent positions [2], [4]. Whenever a dynamic obstacle intersects the planned path, the occupancy grid is reconstructed and the A-star algorithm recalculates an alternative route from the robot's current position to the target [16]. This replanning process enables the robot to avoid collisions while maintaining navigation toward the destination.

The overall dynamic opponent avoidance process is illustrated in Fig. 4. The system continuously receives the robot position, goal position, and opponent position from the vision system. The obstacle map is then updated before the A-star algorithm searches for a new collision-free path. The generated waypoints are transmitted to the STM32F407VGT6 controller through ROS 2 communication and executed by the omnidirectional drive system until the robot reaches the target [17].



a) Before Replanning *b) After Replanning*

Figure 4 Dynamic Opponent Avoidance Process

E. System Testing Scenario

The proposed navigation system was evaluated in the Gazebo simulation environment integrated with ROS 2 Humble using a 12 m × 8 m soccer field model. The occupancy grid resolution was set to 250 mm × 250 mm per cell. Eight navigation scenarios were conducted for both Single A-star and Dynamic A-star using the same robot starting positions and goal locations. The Single A-star baseline used one obstacle, while the proposed Dynamic A-star used two obstacles, with one obstacle dynamically moved during robot navigation. The obstacle movement was introduced after the robot began following the initial path, requiring the Dynamic A-star algorithm to update the occupancy grid and generate a new path from the robot's current position to the goal. The system performance was evaluated based on navigation success, collision occurrence, travel time, and path length.

III. Result and Discussion

The proposed ROS 2-based A-star path planning system was evaluated through several navigation scenarios in the Gazebo simulation environment. The experiments were conducted to compare the conventional A-star algorithm and the proposed dynamic replanning method under different opponent configurations. The system performance was evaluated based on navigation success, travel time, path length, and collision occurrence.

A. Single A-star Path Planning

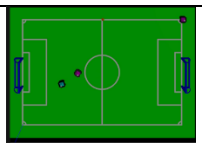
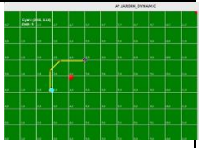
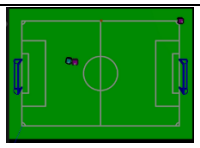
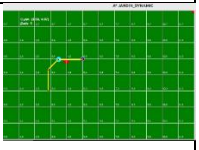
The first experiment was performed using the Single A-star algorithm, where the navigation path was generated only once before the robot started moving. The robot successfully reached the target when no obstacle blocked the planned trajectory.

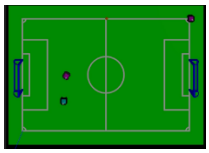
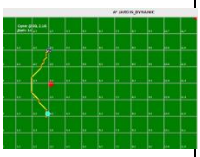
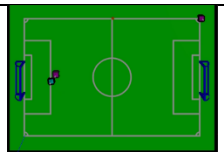
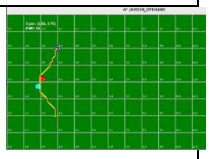
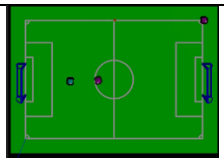
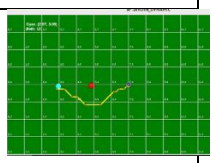
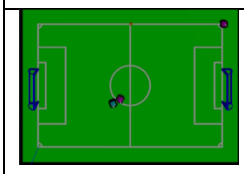
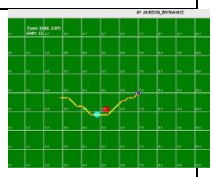
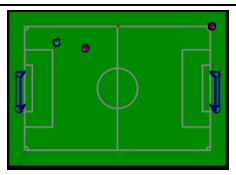
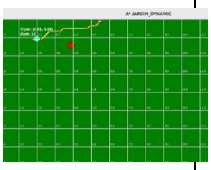
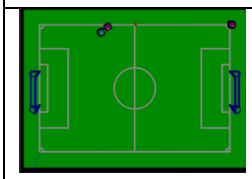
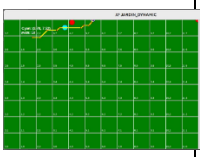
TABLE 1 SINGLE A-STAR PATH PLANNING

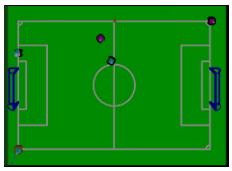
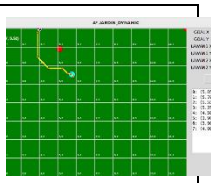
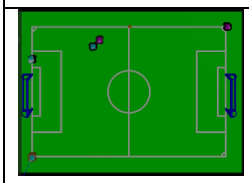
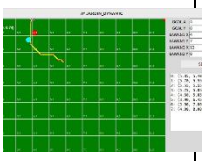
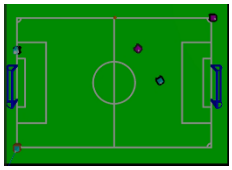
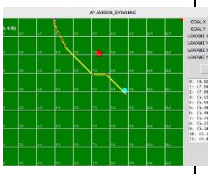
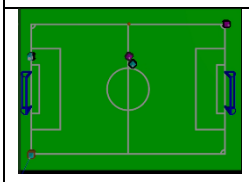
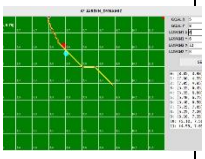
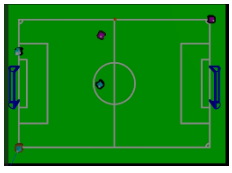
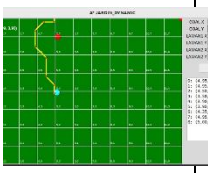
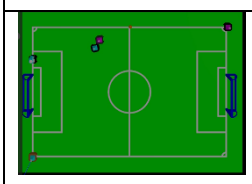
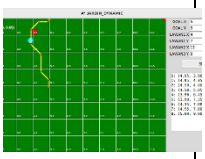
Test	Start (x,y)	Goal (x,y)	Obstacle (x,y)	Obstacle Movement (x,y)	Time (s)	Distance (m)
1	(3, 3)	(5, 5)	(4, 4)	(4, 5)	15.22	2.83
2	(3, 2)	(3, 6)	(3, 4)	(2, 4)	17.81	4.00
3	(3, 4)	(7, 4)	(5, 4)	(5, 3.5)	15.75	4.00
4	(2, 7)	(5.5, 8)	(4, 6.5)	(4, 8)	25.65	3.64
5	(6, 5.5)	(4, 8)	(5, 7)	(4, 7)	21.51	3.54
6	(9, 4)	(5, 8)	(7.5, 6)	(6, 6)	23.32	5.66
7	(5, 4)	(5, 9)	(5, 7)	(4, 7)	19.19	5.00
8	(3, 7)	(7, 7)	(4.4, 6.4)	(4.4, 7.5)	17.51	4.00

As shown in Table 1, eight navigation scenarios with different start positions, goal locations, and obstacle movements were conducted. The average travel time and path length obtained from these experiments were used as quantitative indicators to evaluate the navigation performance of the Single A-star algorithm in dynamic environments.

TABLE 2 RESULT SINGLE A-STAR PATH PLANNING

Test	Image Before	Path Before Movement	Path Before
1		(2.85,3.00) (2.85,4.35) (3.45,4.95) (4.95,4.95) (5.00,5.00)	
Result	**Success**		
	Image After	**Path after Movement**	**Path After**
1		(2.85,3.00) (2.85,4.35) (3.45,4.95)	
Result	**Failed**		
Test	**Image Before**	**Path Before Movement**	**Path Before**

2		(2.85,1.95) (2.85,2.40) (2.40,2.85) (2.40,3.80) (1.95,3.45) (1.95,4.35) (2.55,4.95) (2.55,5.10)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
2		(2.85,1.95) (2.85,2.40) (2.40,2.85)	
Result	**Failed**		
Test	**Image Before**	**Path Before Movement**	**Path Before**
3		(2.85,3.75) (3.30,3.75) (3.75,3.30) (4.85,3.30) (6.45,3.75) (7.00,4.00)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
3		(2.85,3.75) (3.30,3.75) (3.75,3.30)	
Result	**Failed**		
Test	**Image Before**	**Path Before Movement**	**Path Before**
4		(1.80,6.90) (2.25,6.90) (2.70,7.35) (3.30,7.35) (5.25,7.65) (5.40,7.80) (5.50,8.00)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
4		(1.80,6.90) (2.25,6.90) (2.70,7.35) (3.30,7.35)	
Result	**Failed**		
Test	**Image Before**	**Path Before Movement**	**Path Before**

5		(5.85,5.40) (5.75,5.55) (5.55,5.55) (5.25,5.85) (4.50,5.85) (3.90,6.45) (3.90,7.80) (4.80,8.80)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
5		(5.85,5.40) (5.75,5.55) (5.55,5.55) (5.25,5.85) (4.50,5.85)	
Result	**Failed**		
Test	**Image Before**	**Path Before Movement**	**Path Before**
6		(8.85,3.90) (7.80,4.95) (7.05,4.95) (5.55,6.45) (5.55,6.68) (5.46,6.75) (5.48,6.98) (5.15,7.50) (4.95,7.65)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
6		(8.85,3.90) (7.80,4.95) (7.05,4.95) (5.55,6.45)	
Result	**Failed**		
Test	**Image Before**	**Path Before Movement**	**Path Before**
7		(4.95,3.90) (4.95,4.35) (4.50,4.80) (4.50,5.85) (3.90,6.45) (3.90,7.35) (4.35,7.80) (4.95,7.80) (5.00,9.00)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
7		(4.95,3.90) (4.95,4.35) (4.50,4.80) (4.50,5.85) (3.90,6.45)	
Result	**Failed**		
Test	**Image Before**	**Path Before Movement**	**Path Before**

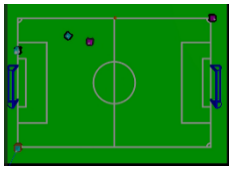
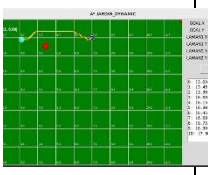
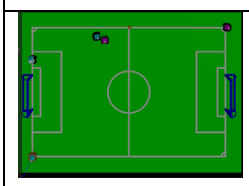
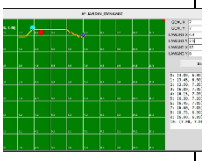
8		(3.00,6.90) (3.45,6.90) (3.90,7.35) (6.00,7.35) (6.15,7.20) (6.30,7.20) (6.45,7.05) (6.60,7.05) (6.75,6.90) (6.90,6.90) (7.00,7.00)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
8		(3.00,6.90) (3.45,6.90) (3.90,7.35) (6.00,7.35)	
Result	**Failed**		

Table 2 presents the navigation results of the Single A-star algorithm before and after obstacle movement. Before the obstacle moved, the robot successfully reached the target by following the initially generated path in all test scenarios. However, after the obstacle changed its position, the algorithm continued following the original trajectory because it did not perform path replanning. As a result, the robot failed to avoid the moved obstacle and collisions occurred in all four experimental scenarios. These results demonstrate that the conventional Single A-star algorithm is unsuitable for navigation in dynamic environments where obstacle positions change.

B. Dynamic Opponent Avoidance Using A-star

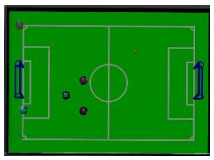
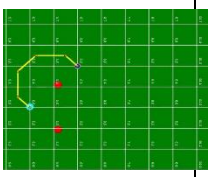
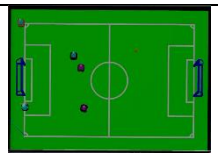
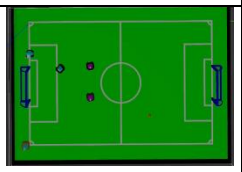
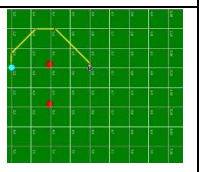
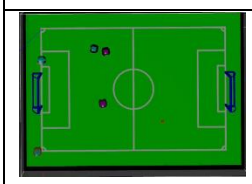
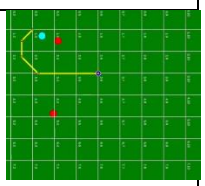
The proposed Dynamic A-star path planning method was developed to overcome the limitations of the Single A-star path planning approach in dynamic environments. In the previous Single A-star implementation, one obstacle was used as the baseline configuration. This study extends the previous approach by introducing two obstacles, one of which moves during robot navigation, to evaluate the effectiveness of the proposed Dynamic A-star method in a more challenging dynamic environment. One of the obstacles was moved during robot navigation to evaluate the ability of the proposed method to update the navigation path according to changes in obstacle positions detected by the OmniVision 360° camera. The opponent coordinates are processed by the ROS 2 framework and converted into an occupancy grid map for path replanning.

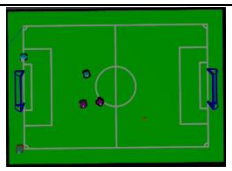
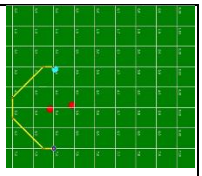
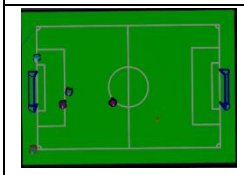
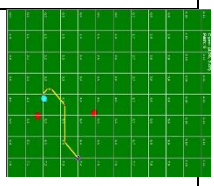
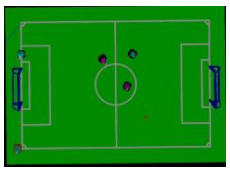
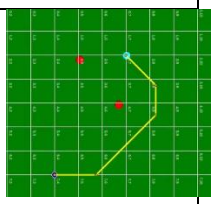
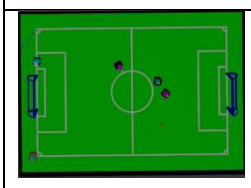
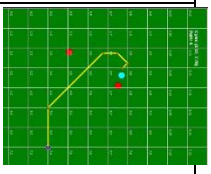
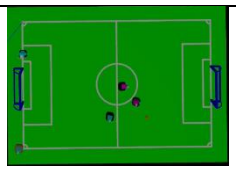
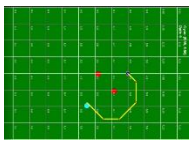
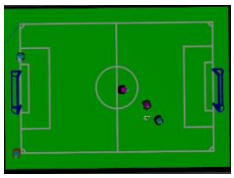
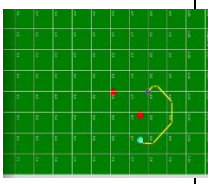
The use of two obstacles in the Dynamic A-star experiments represents a deliberate extension of the previous Single A-star configuration. The Single A-star method was initially implemented using one obstacle as a baseline for static path planning. In this study, the configuration was extended to two obstacles, with one obstacle dynamically moved during navigation, to evaluate whether the proposed Dynamic A-star method can maintain collision-free navigation under a more challenging and changing environment. The same robot starting positions and goal locations were maintained across the Single A-star and Dynamic A-star experiments to provide a consistent basis for evaluating the effect of dynamic obstacle handling.

TABLE 3 DYNAMIC A-STAR PATH PLANNING

Test	Start (x,y)	Goal (x,y)	Obst.1	Obst 2	Obst. 1 Movement	Time (s)	Dist. (m)
1	(3, 3)	(5, 5)	(4, 4)	(2,4)	(5,4)	25,54	5.00
2	(3, 2)	(3, 6)	(3, 4)	(5,4)	(1,4)	25,31	7.00
3	(3, 4)	(7, 4)	(5, 4)	(5,5)	(5,1)	25,36	6.01
4	(2, 7)	(5.5, 8)	(4, 6.5)	(2,5)	(5,8)	38,51	6.32
5	(6, 5.5)	(4, 8)	(5, 7)	(4,6)	(5,8)	27,78	5.44
6	(9, 4)	(5, 8)	(7.5, 6)	(6,5)	(4,4)	39,85	9.51
7	(5, 4)	(5, 9)	(5, 7)	(5,5)	(3,7)	31,60	8.75
8	(3, 7)	(7, 7)	(4.4, 6.4)	(5,7)	(5,8)	25,65	6.08

TABLE 4 DYNAMIC A-STAR PATH PLANNING

Test	Image Before	Path Before Movement	Path Before
1		(3.00,3.00) (3.50,2.50) (4.75,2.50) (5.50,3.25) (5.50,4.50) (5.00,5.00)	
Result	**Success**		
	Image After	**Path after Movement**	**Path After**
1		(5.50,3.25) (5.75,3.00) (6.25,3.50) (6.25,4.25) (5.50,5.00) (5.00,5.00)	
Result	**Success**		
Test	**Image Before**	**Path Before Movement**	**Path Before**
2		(2.75,2.00) (2.25,2.00) (1.00,3.25) (1.00,4.25) (2.75,6.00) (3.00,6.00)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
2		(1.00,3.00) (1.50,2.50) (2.25,2.50) (3.00,3.25) (3.00,6.00)	
Result	**Success**		
Test	**Image Before**	**Path Before Movement**	**Path Before**

3		(3.00,4.00) (3.00,3.50) (4.50,2.00) (5.50,2.00) (7.00,3.50) (7.00,4.00)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
3		(4.00,2.00) (3.75,2.25) (3.75,2.50) (4.50,3.25) (6.25,3.25) (7.00,4.00)	
Result	**Success**		
Test	**Image Before**	**Path Before Movement**	**Path Before**
4		(2.00,7.00) (3.25,7.25) (4.50,7.25) (5.50,8.00)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
4		(3.00,7.75) (2.75,8.00) (2.25,7.50) (2.25,6.75) (5.50,8.00)	
Result	**Success**		
Test	**Image Before**	**Path Before Movement**	**Path Before**
5		(5.75,5.50) (6.75,6.50) (6.75,7.50) (5.75,8.50) (4.50,8.50) (4.00,8.00)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
5		(6.25,7.50) (6.50,7.75) (6.50,8.25) (5.50,9.25) (4.50,9.25) (3.75,8.50) (3.75,8.25) (4.00,8.00)	
Result	**Success**		
Test	**Image Before**	**Path Before Movement**	**Path Before**

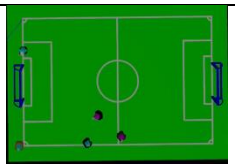
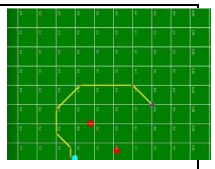
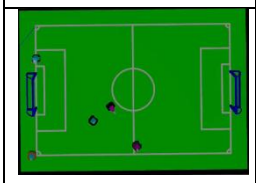
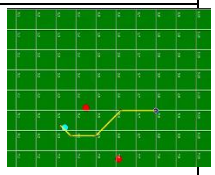
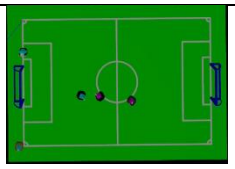
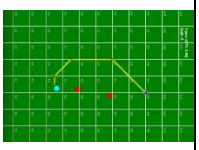
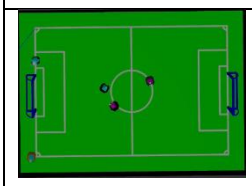
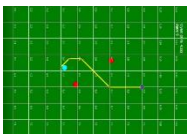
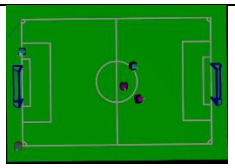
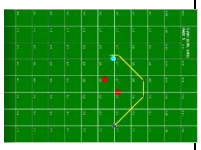
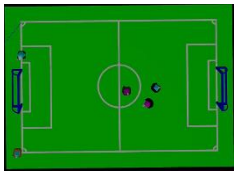
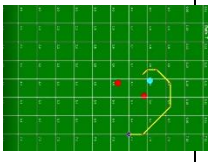
6		(7.75,3.75) (7.25,3.75) (6.50,3.00) (5.25,3.00) (4.00,4.25) (4.00,7.00) (5.00,8.00)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
6		(5.75,3.25) (6.25,3.75) (6.25,5.00) (5.00,6.25) (5.00,8.00)	
Result	**Success**		
Test	**Image Before**	**Path Before Movement**	**Path Before**
7		(4.50,3.50) (4.00,3.50) (3.00,4.50) (3.00,7.00) (5.00,9.00)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
7		(3.75,4.00) (3.25,4.50) (3.25,5.25) (5.00,7.00) (5.00,9.00)	
Result	**Success**		
Test	**Image Before**	**Path Before Movement**	**Path Before**
8		(2.75,6.75) (2.75,7.25) (4.25,8.75) (5.25,8.75) (7.00,7.00)	
Result	**Success**		
	Image After	**Path After Movement**	**Path After**
8		(3.75,7.75) (3.50,8.00) (3.50,8.50) (4.25,9.25) (5.50,9.25) (7.00,7.75) (7.00,7.00)	
Result	**Success**		

Table 3 summarizes the experimental scenarios used to evaluate the proposed Dynamic A-star path planning algorithm. Eight navigation scenarios were conducted using the same robot starting positions and goal locations as the Single A-star experiments, while two obstacles were

introduced in the Dynamic A-star environment. One of the obstacles was moved during navigation to evaluate the ability of the proposed method to adapt to changes in the environment through path replanning. Unlike the conventional Single A-star approach, the proposed method continuously updates the occupancy grid map whenever obstacle positions change, allowing the robot to generate an alternative collision-free trajectory. The travel time and path length were also recorded to quantitatively evaluate the navigation performance.

Table 4 presents the navigation results before and after dynamic path replanning. Before the obstacle movement, the robot followed the initially generated trajectory toward the target. After the obstacle position changed, the proposed Dynamic A-star algorithm successfully recalculated a new collision-free path based on the updated occupancy grid. As shown in all experimental scenarios, the robot adapted its trajectory without colliding with the moving obstacle, demonstrating the effectiveness of the proposed replanning mechanism for navigation in dynamic environments

C. Comparative Analysis

To evaluate the effectiveness of the proposed method, the navigation performance of the Single A-star and Dynamic A-star algorithms was evaluated based on navigation success, travel time, path length, and collision avoidance capability. The Single A-star experiments represent the baseline method from the previous study using one obstacle, whereas the proposed Dynamic A-star method extends this configuration to two obstacles, including one moving obstacle, to evaluate navigation performance under more challenging dynamic conditions. The same robot starting positions and goal locations were maintained for both methods to provide a consistent basis for evaluating the effect of dynamic obstacle handling. Therefore, the comparison is primarily intended to demonstrate the improvement in navigation adaptability and collision avoidance achieved through dynamic replanning. Travel time and path length are presented as supporting quantitative measurements of the navigation behavior rather than as direct performance comparisons under identical obstacle configurations.

TABLE 5 COMPARISON PATH PLANNING

Parameter	Single A-star	Dynamic A-star
Tested Scenarios	8	8
Successful Navigation	0/8	8/8
Navigation Success Rate	0%	100%
Collision Rate	100%	0%
Average Travel Time	19.50 s	29.95 s
Average Path Length	4.08 m	6.76 m

As shown in Table 5, the proposed Dynamic A-star algorithm successfully completed all eight navigation scenarios, achieving a 100% navigation success rate and a 0% collision rate, whereas the conventional Single A-star algorithm failed to adapt to the changes in obstacle positions. The results indicate that Single A-star, which generates the path without subsequent replanning, is unable to maintain a collision-free trajectory when the obstacle configuration changes. In contrast, Dynamic A-star continuously updates the path based on the changed obstacle positions, allowing the robot to generate an alternative collision-free trajectory.

The Dynamic A-star algorithm resulted in a longer average travel time and path length than Single A-star. The average travel time increased from 19.50 s to 29.95 s, while the average path length increased from 4.08 m to 6.76 m. This increase occurred because the robot had to perform additional detours when the moving obstacle blocked the initially planned trajectory. The obstacle movement caused the previously generated path to become unsuitable, requiring the occupancy grid to be updated and a new collision-free trajectory to be generated from the robot's current position. Consequently, the additional path distance and trajectory changes increased the travel time.

A limitation of this evaluation is that the two methods were intentionally tested with different obstacle complexities. The Single A-star baseline used one obstacle, whereas the proposed Dynamic A-star extended the configuration to two obstacles, including one moving obstacle. Therefore, the travel time and path length should not be interpreted as direct quantitative comparisons under identical obstacle conditions. Instead, these results primarily demonstrate the improved adaptability and collision avoidance capability provided by dynamic replanning. Future experiments should evaluate Single A-star and Dynamic A-star under identical multi-obstacle configurations to provide a more controlled quantitative comparison.

D. ROS 2 Communication Integration

The integration of the proposed navigation system was implemented using the ROS 2 Humble middleware to facilitate communication between perception, path planning, and robot control modules. Figure 5 illustrates the ROS 2 flow graph generated during system execution.

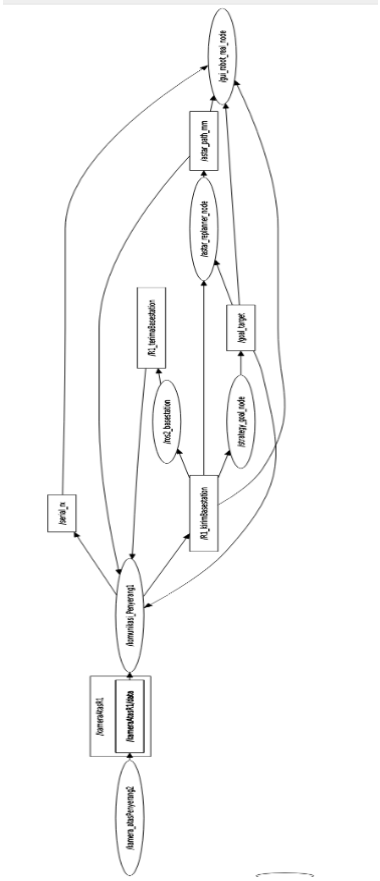


Figure 5. ROS 2 Communication Flow Graph

Based on Fig. 5, the proposed ROS 2 framework successfully established communication among all navigation modules. The camera node publishes image data that are processed by the object detection node to identify the positions of the ball and opponent robots. The detected object coordinates are then transmitted to the goal strategy and A-star-based path planning nodes to generate navigation waypoints.

Furthermore, the generated path is published to the robot control node through ROS 2 topics and transmitted to the STM32 controller via serial communication. The experimental results show that all nodes communicated properly without communication failure during navigation, allowing the robot to update its trajectory continuously according to changes in opponent positions. This confirms that the ROS 2 framework provides reliable real-time communication for implementing dynamic A-star-based path planning in wheeled soccer robots.

IV. Conclusion

This study successfully developed a ROS 2-based Dynamic A-star path planning system for opponent avoidance in wheeled soccer robots. The proposed method enables the robot to dynamically update its navigation path whenever the positions of opponent robots change, allowing collision-free navigation in dynamic environments. Experimental results from eight navigation scenarios demonstrated that the proposed Dynamic A-star algorithm achieved a 100% navigation success rate (8/8 successful scenarios) with a 0% collision rate, whereas the conventional Single A-star algorithm failed to adapt to obstacle movement, resulting in a 0% navigation success rate and a 100% collision rate. Although the proposed method increased the average travel time from 19.50 s to 29.95 s and the average path length from 4.08 m to 6.76 m due to the path replanning process, it successfully generated alternative collision-free trajectories under dynamic conditions. Furthermore, the ROS 2 communication framework successfully provided reliable real-time communication between the navigation modules, enabling continuous path updates during robot operation. Overall, the proposed Dynamic A-star approach improves navigation robustness and adaptability, making it suitable for autonomous wheeled soccer robots operating in dynamic environments.

References

[1] M. A. Y. Yaqin, Agus Khumaidi, Ryan Yudha Adhitya, Noorman Rinanto, Evi Nafiatus Sholikhah, and Riko Satrya Fajar Jaelani Putra, "Implementasi Path Planning A-star Untuk Menghindari Robot lawan Mengunakan Robot Operating System pada KRSBI-Beroda," *J. Elektron. dan Otomasi Ind.*, vol. 12, no. 1, 2025, doi: 10.33795/elkolind.v12i1.7268.

[2] M. Yao, X. Feng, H. Deng, P. Li, H. Liu, and A. Wang, "Results in Engineering A dynamically hybrid path planning and obstacle avoidance algorithm for mobile robots based on improved A-star and dynamic windows approach," *Results Eng.*, vol. 29, no. December 2025, p. 108924, 2026, doi: 10.1016/j.rineng.2025.108924.

[3] A. Alexander, K. Venkatesan, J. Mounsef, and K. Ramanujam, "A Comprehensive Survey of Path Planning Algorithms for Autonomous Systems and Mobile Robots : Traditional and Modern Approaches," *IEEE Access*, vol. 13, no. October, pp. 176287–176326, 2025, doi: 10.1109/ACCESS.2025.3618837.

[4] J. Wang, *A* Path Planning Algorithm Optimization via Unscented Kalman Filter in Dynamic Environments*, no. Iceegt 2025. Atlantis Press International BV, 2026. doi: 10.2991/978-94-6463-986-5.

[5] L. I. Safitri, J. Sahertian, and D. W. Widodo, "Implementasi Algoritma Path Planning A-star Pada Base Station Robot Sepak Bola Beroda," *Gener. J.*, vol. 7, no. 3, pp. 56–63, 2023, doi: 10.29407/gj.v7i3.20545.

[6] W. Darmawan, M. Basuki Rahmat, A. Khumaidi, R. Yudha Adhitya, and D. Pristovani Riananda, "Perancangan Strategi Keputusan Robot Sepak Bola Beroda menggunakan Metode Decision Tree," *J. Elektron. dan Otomasi Ind.*, vol. 10, no. 2, pp. 175–182, 2023, doi: 10.33795/elkolind.v10i2.3020.

[7] M. J. Saputra, A. Khumaidi, R. Satrya, and F. Jaelani, "Implementasi Estimasi Jarak Relatif Objek Berbasis ROS2 pada Robot Sepak Bola Menggunakan YOLO dan Kamera," *J. Elektron.*

dan Otomasi Ind., vol. 13, no. 1, pp. 110–121, 2026, doi: 10.33795/elkolind.v13i1.9813.

[8] F. D. Ikram, A. Khumaidi, M. B. Rahmat, J. Endrasmono, M. Syai, and D. Pristovani, "Deteksi Objek di Lapangan pada Robot Sepakbola Beroda Menggunakan Metode YoloV5," *J. Elektron. dan Otomasi Ind.*, vol. 11, no. 2, pp. 604–611, 2024, doi: 10.33795/elkolind.v11i2.5235.

[9] Wahid Hasyim, I. Siradjuddin, and L. Kamajaya, "Robot Beroda Pemindah Barang otomatis Dengan Metode Path Planning," *J. Elektron. dan Otomasi Ind.*, vol. 12, no. 2, pp. 244–253, 2025, doi: 10.33795/elkolind.v12i2.4014.

[10] A. A. Wahyudi, Agus Khumaidi, Mohammad Basuki Rahmat, Dimas Pristovani Riananda, Mat Syai'in, and Joko Endrasmono, "Implementasi Robot Operating System (ROS) Untuk Meningkatkan Akurasi Deteksi Bola Menggunakan YOLO V5 Pada KRSBI-Beroda," *J. Elektron. dan Otomasi Ind.*, vol. 11, no. 2, pp. 648–661, 2024, doi: 10.33795/elkolind.v11i2.5234.

[11] S. S. Hidayatullah and E. Saputra, "Perancangan Omni Vision Camera sebagai Sistem Pendeteksi Objek untuk Robot Sepak Bola," *Jambura J. Electr. Electron. Eng.*, vol. 4, no. 2, pp. 229–235, 2022, doi: 10.37905/jjeee.v4i2.14495.

[12] M. Basavanna., M. Shivakumar., K. R. Prakash., and P. Bhomkar, "ROS Based 3D Mapping of an Indoor Environment Using Fusion of Orbbec Astra Camera and Lidar on Turtlebot Mobile Robot," *2021 5th Int. Conf. Electr. Electron. Commun. Comput. Technol. Optim. Tech. ICEECCOT 2021 - Proc.*, no. October, pp. 323–327, 2021, doi: 10.1109/ICEECCOT52851.2021.9708058.

[13] A. Khumaidi, M. A. Yaqin, R. Y. Adhitya, S. M. Irsyad, and M. J. Saputra, "Penerapan Rotary Encoder sebagai Sistem Odometry untuk Navigasi Robot Sepak Bola Beroda," *JURITEK J. Ilm. Tek. Mesin, Elektro dan Komput.*, vol. 5, no. 3, pp. 291–301, 2025, doi: 10.51903/juritek.v5i3.6043.

[14] A. Jayadi, P. A. Pratomo, and E. R. Dalimunthe, "Implementasi Algoritma A-Star untuk Perencanaan Jalur Robot Menggunakan Deteksi Warna Berbasis Kamera," *Blend Sains J. Tek.*, vol. 4, no. 1, pp. 116–127, 2025, doi: 10.56211/blendsains.v4i1.987.

[15] S. M. Irsyad, A. Khumaidi, R. Y. Aditya, and A. R. Ramadhan, "Predicting Real Distance for Wheeled Soccer Robot using YOLO Architecture," *Int. J. Informatics Comput.*, vol. 7, no. 2, pp. 806–821, 2025, doi: 10.35842/ijicom.v7i2.200.

[16] A. Ghabussi, "Motion Control and Path Planning and Tracking Algorithm of Football Robot," *Int. J. Multimed. Comput.*, vol. 1, no. 1, pp. 15–30, 2020, doi: 10.38007/ijmc.2020.010102.

[17] A. Bonci, F. Gaudeni, M. C. Giannini, and S. Longhi, "Robot Operating System 2 (ROS2)-Based Frameworks for Increasing Robot Autonomy: A Survey," *Appl. Sci.*, vol. 13, no. 23, pp. 1–31, 2023, doi: 10.3390/app132312796.