Research on Dynamic Path Planning for Robots Based on the Integration of Improved A* Algorithm and DWA(Dynamic Window Approach) Algorithm

Lu Ke[1], Byung-Won Min[2*]
[1]Ph.D. Student, Department of IT Engineering, Mokwon University
[2]Professor, Department of Game Software Engineering, Mokwon University

개선된 A* 알고리즘과 DWA(Dynamic Window Approach) 알고리즘의 통합에 기반한 로봇의 동적 경로 계획에 관한 연구

노크[1], 민병원[2*]
[1]목원대학교 IT공학과 박사과정 [2]목원대학교 게임소프트웨어공학과 교수

Abstract This paper presents a mobile robot path planning algorithm that combines an improved A* algorithm with the DWA algorithm. The proposed method integrates the global optimality of the A* algorithm with the dynamic obstacle avoidance capability of the DWA algorithm. First, the improved A* algorithm uses obstacle density information to adaptively adjust the cost function in segments, which enhances search efficiency. Second, based on the global path, a DWA evaluation function incorporating key point information is constructed. Key points are then used as intermediate targets for the DWA algorithm to plan the local path, achieving both global optimality and dynamic obstacle avoidance. Finally, simulation experiments verify the effectiveness and feasibility of the proposed algorithm.

Key Words : Mobile Robot, Path Planning, Improved A* Algorithm, DWA Algorithm, Algorithm Integration

요 약 본 논문에서는 개선된 A* 알고리즘과 DWA 알고리즘을 융합한 이동 로봇 경로 계획 알고리즘을 제안한다. 제안된 알고리즘은 A* 알고리즘의 전역 경로 최적화 능력과 DWA 알고리즘의 동적 장애물 회피 능력을 결합하였다. 먼저, 개선된 A* 알고리즘에서는 장애물 밀도 정보를 도입하여 구간별로 비용 함수를 자율적으로 조정함으로써 검색 효율을 향상시켰다. 그다음, 전역 경로를 기반으로 주요 지점 정보를 반영한 DWA 알고리즘 평가 함수를 구성하였으며, 주요 지점을 중간 목표점으로 설정하여 DWA 알고리즘을 적용함으로써 지역 경로를 계획하였다. 이를 통해 전역 경로 최적화와 동적 장애물 회피 기능을 동시에 실현하였다. 마지막으로, 시뮬레이션 실험을 통해 제안된 경로 계획 알고리즘의 효과성과 실현 가능성을 검증하였다.

주제어 : 이동 로봇, 경로 계획, 개선된 A* 알고리즘, DWA 알고리즘, 알고리즘 융합

*교신저자 : 민병원(minfam@mokwon.ac.kr)
접수일: 2025년 03월 10일 수정일: 2025년 04월 04일 심사완료일: 2025년 04월 12일

1. Introduction

Path planning is one of the key performance indicators for mobile robots. It refers to the robot's ability to autonomously plan an optimal path from the start to the target while avoiding all obstacles. Path planning can be divided into two main categories based on the robot's task requirements: global path planning and local path planning [1-7].

Global path planning is based on static environmental information and relies on a high-precision map to plan a smooth, collision-free optimal path between the start and target points. Common global path planning algorithms include the Dijkstra algorithm [8], the A^* algorithm [9,10], the RRT* algorithm [11,12], and the ant colony algorithm [13]. However, traditional global planning methods rely on detailed environmental data and cannot handle dynamic obstacle avoidance for unknown obstacles.

Local path planning uses onboard sensors, such as cameras and LiDAR, to perceive the surrounding environment in real time. Based on the global path outline and the detected distribution of both fixed and moving obstacles, it plans a safe local path. Common local path planning algorithms include the artificial potential field method [14] and the DWA algorithm [15-17]. Due to the lack of global information and limited dynamic correction capability, local planning may result in suboptimal paths or even fail to reach the target in complex environments.

In summary, to effectively combine global and local path planning enabling the robot to generate a smooth, collision-free global path while dynamically adjusting the path according to the real-time distribution of unknown obstacles this paper proposes a navigation algorithm that fuses an improved A^* algorithm with the DWA algorithm. The main innovations of the proposed method are as follows:

A. Integrate environmental information to optimize the cost function of the traditional A^* algorithm. The efficiency is enhanced by adaptively adjusting the heuristic weight in stages.

B. Optimize global path planning using a key point selection strategy. This approach eliminates redundant collinear nodes and unnecessary turning points, retaining only the essential nodes to improve path effectiveness.

C. Construct an evaluation function based on key point information and apply it to the DWA algorithm. This ensures that the local path planning follows the global path outline, resulting in a smoother path with real-time obstacle avoidance capability.

2. Improved A^* Algorithm

2.1 Principles of the Traditional A^* algorithm

The A^* (A-Star) algorithm is a heuristic search algorithm used to solve the shortest path problem in static environments, with its core based on the cost function:

$$F(n) = G(n) + H(n) \tag{1}$$

Here, $F(n)$ denotes the estimated total cost from the start point to the target through node n. $G(n)$ represents the actual cost from the start point to node n, and $H(n)$ is the heuristic function that estimates the cost from node n to the target.

During the execution of the algorithm, two lists are maintained: an open list and a closed list. The open list stores nodes that are to be processed, while the closed list stores nodes that have already been processed. The algorithm's flow is as follows:

A. Add the starting node to the open list and mark it as visited.

B. Enter the iterative search phase by selecting the node with the smallest $F(n)$ value from the open list for expansion.

C. Calculate $G(n)$ and $F(n)$ for all neighboring nodes of the current node:

a) If the adjacent node has not been visited, add it to the open list.

b) If an adjacent node is already in the open list but the path cost through the current node is lower, update its $G(n)$ and $F(n)$ values.

D. Add the current node to the closed list.

E. Stop the search when the target node is added to the open list, and reconstruct the optimal path from the goal to the start by backtracking through the parent nodes.

The efficiency and search performance of the A* algorithm highly depend on the choice of the heuristic function $H(n)$. When $H(n)$ is underestimated and smaller than the actual cost, the algorithm behaves similarly to breadth-first search, expanding a large number of nodes and reducing search efficiency. When $H(n)$ is overestimated and exceeds the actual cost, the algorithm behaves more like depth-first search, making it prone to getting stuck in local optima and deviating from the globally optimal path. Common heuristic functions include:

A. Manhattan Distance (suitable for four-directional or eight-directional grid maps).

B. Euclidean Distance (suitable for continuous spaces or environments without directional constraints).

Theoretical completeness and optimality make the A* algorithm one of the classic methods for path planning. However, since its performance is still significantly influenced by the heuristic function, $H(n)$ must be carefully designed for specific scenarios to achieve performance optimization.

2.2 Improved A* algorithm

The previous analysis indicates that the heuristic function $H(n)$ plays a crucial role in the A* algorithm, directly affecting search efficiency and flexibility. However, due to the diversity of real-world operating environments, a single $H(n)$ often fails to meet all requirements. Therefore, by adjusting the weight of $H(n)$ at different stages, the A* algorithm can better adapt to various environments, thereby improving overall performance and efficiency.

2.2.1 Heuristic Function Optimization

Considering the impact of $H(n)$ on algorithm performance, its weight should be adjusted based on obstacle density in the map. In high-density obstacle areas, lowering the weight of $H(n)$ helps prevent the algorithm from getting trapped in local optima by placing more emphasis on the actual cost $G(n)$, ensuring global path optimality. Conversely, in low-density obstacle areas, increasing the weight of $H(n)$ reduces redundant searches and accelerates the search process. Therefore, adaptively adjusting the heuristic weight based on obstacle distribution can enhance the robot's path planning capability under different environmental conditions.

To achieve this, this paper introduces a criterion for determining the obstacle density of the map. Suppose the total number of grid cells in the rectangular area formed by the start and target points is M, and the number of obstacle grid cells within this area is N, then the obstacle density is expressed as:

$$P = N/M \tag{2}$$

Similarly, let S be the total number of grid cells in the map and T be the total number of obstacle grid cells. Then, the global obstacle density is expressed as:

$$Q = T/S \tag{3}$$

By comparing the environmental obstacle density P and the global obstacle density Q, the area's density can be assessed, and this result can be incorporated into the cost function $F(n)$. Under different density conditions, the weight of the heuristic function ω is adjusted to optimize the cost function. The optimized cost function can be expressed as:

$$F(n) = G(n) + \omega \times H(n) \tag{4}$$

The dynamic adjustment rule for the weight ω is as follows:

$$\omega = \begin{cases} \alpha + \beta \times (Q/P), & P \geq 1.5Q(Dense areas, \omega < 1) \\ \gamma + \delta \times (P/Q), & P < 0.5Q(Sparse areas, \omega > 1) \\ 1, & Other \end{cases} \tag{5}$$

Here, α, β, γ and δ are constants used to dynamically adjust the value of ω to an appropriate range.

In this study, a dynamic weight adjustment strategy based on local obstacle density P and global obstacle density Q is proposed. To distinguish the obstacle density in different environments, two empirical thresholds $1.5Q$ and $0.5Q$ are introduced. Specifically, when the local obstacle density P exceeds $1.5Q$, it indicates a dense obstacle region. To ensure that the actual cost is adequately considered during the search and maintain global path optimality, the heuristic weight for this region should be reduced, resulting in $\omega < 1$. Conversely, when P is lower than $0.5Q$, the local environment is relatively sparse, and increasing the heuristic weight to make $\omega > 1$ helps speed up the search and reduce unnecessary node expansions. In other cases, the heuristic weight is kept at the default value, $\omega = 1$, to balance global optimality and search efficiency.

This threshold setting is based on theoretical analysis and preliminary experimental verification of search behavior in different density environments. While the specific values may vary depending on the application scenario and map resolution, this approach provides a reasonable framework to adaptively adjust the heuristic weight, enabling the algorithm to achieve a good balance between path planning efficiency and optimality across different environments.

2.2.2 Key Point Selection Strategy

Traditional A* algorithms typically generate path nodes at the center of each grid cell in a grid map and then sequentially connect these nodes to form a path. However, this method can lead to paths containing redundant collinear nodes and unnecessary turning points. If the path is used directly for navigation, the robot will need to make heading adjustments at each turning point. These extra maneuvers can negatively impact the robot's operational efficiency and stability. In practical navigation, smoother paths help reduce navigation errors, thereby assisting the robot in completing tasks more precisely.

To optimize the path generated by the traditional A* algorithm, this study proposes a key point selection strategy. Through this strategy, the global path planning is optimized by retaining only the essential path nodes, thus improving the efficiency of the path. The specific steps of the key point selection strategy are as follows:

A. Turning Point Extraction: The area formed by three consecutive points (k-1, k, k+1) in the path is calculated. If the area is non-zero, then k is considered a turning point; otherwise, k is not a turning point. The extraction of turning points starts from the path node closest to the starting point and continues until k is the path node closest to the target point, at which point the extraction process ends. Both the start and target points are also considered turning points.

B. Key Point Selection: Between adjacent turning points (k-1, k, k+1), a line is drawn connecting the non-adjacent points (k-1, k+1). If the connecting line passes through an obstacle area, k is considered a key point; otherwise, k is a redundant point. The key point selection process starts from the turning point closest to the starting point and continues until the turning point closest to the target point is reached, at which point the selection process ends.

Finally, through the key point selection strategy, redundant points are eliminated, and only the essential path nodes are retained. This results in an optimized global path that includes only the start point, key points, and the target point.

3. Improved DWA Algorithm

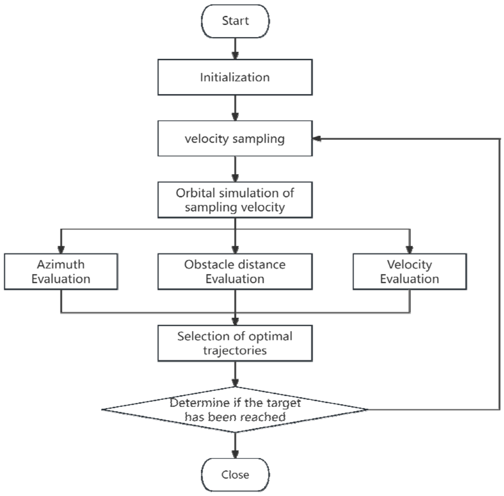
The Dynamic Window Approach (DWA) is an algorithm widely used in local path planning. This algorithm generates local path planning by combining information from global path planning with the positions of unknown and moving obstacles, then outputs velocity commands to guide the robot's motion in a dynamic environment.

The principle of the DWA algorithm is to sample multiple velocity sets in the velocity space (v,w) with a certain resolution and simulate the motion trajectories of these velocities over a specific period. After obtaining the feasible trajectories, they are evaluated using a cost function, and the optimal trajectory is selected. The corresponding (v,w) values are then used to drive the robot's movement. The specific flow of the DWA algorithm is shown in figure 1.

3.1 Velocity Window Calculation

The velocity window (v,w) represents the allowable velocity range for the robot. The robot's velocity is constrained by its dynamics and influenced by obstacles in the environment. Based on the robot's current velocity and the obstacle information detected by sensors, the feasible ranges for linear velocity v and angular velocity w are dynamically calculated.

$$v_{\min} \le v \le v_{\max}, w_{\min} \le w \le w_{\max} \tag{6}$$



[Fig. 1] DWA Algorithm Flowchart

3.2 Trajectory Prediction

Based on position control, for each velocity pair (v,w) within the velocity window, the robot's motion trajectory after a time interval is calculated. The computation follows Equation (7).

$$\begin{cases} x_{t+\Delta t} = x_t + v \times cos(\theta) \times \Delta t \\ y_{t+\Delta t} = y_t + v \times sin(\theta) \times \Delta t \\ \theta_{t+\Delta t} = \theta + \omega \times \Delta t \end{cases} \tag{7}$$

3.3 Evaluation Function of the DWA Algorithm

By evaluating multiple predicted trajectories, the optimal local path is selected. This path should be smooth, safe, and stable. The evaluation function is given as:

$$G(v,w) = \alpha \times Heading(v,w) + \beta \times Dist(v,w) + \gamma \times Vel(v,w) \tag{8}$$

Where $Heading(v,w)$ represents the angular deviation between the predicted trajectory endpoint and the target point, $Dist(v,w)$ denotes the distance between the trajectory and obstacles, and $Vel(v,w)$ is the current velocity. α,β,γ are weight coefficients.

3.4 Improved Dynamic Window Approach

Analysis of the above evaluation function reveals that the DWA algorithm relies solely on a single final target point for guidance. In environments with numerous obstacles, this may make it difficult to find a feasible path to the target. To address this, intermediate target points can be introduced to guide the DWA algorithm, enabling segmented local path planning and preventing navigation failures.

By incorporating the key points extracted from the improved A^* algorithm discussed earlier, these key points serve as intermediate targets to construct an evaluation function that integrates key point information. This ensures that local path planning follows the contour of the global path. The optimized evaluation function is given as:

$$G(v,w) = \alpha \times LHeading(v,w) + \beta \times Dist(v,w) + \gamma \times Vel(v,w) + \delta \times GHeading(v,w) \tag{9}$$

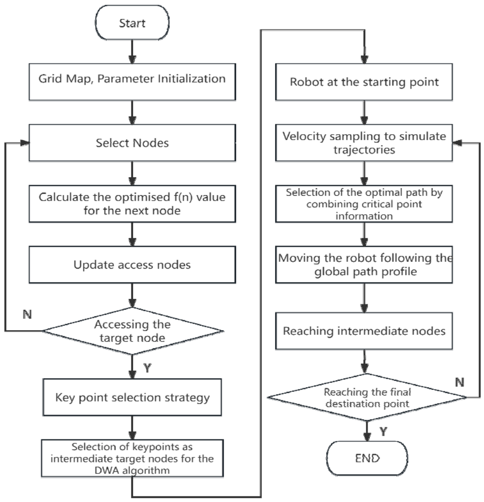
Where $LHeading(v,w)$ represents the angular deviation between the endpoint of the current trajectory and the current intermediate target point, $GHeading(v,w)$ denotes the angular deviation between the endpoint of the current trajectory and the global target point, and δ is a weight coefficient used to control the influence of the global target on local planning.

In equation (9), the setting of the weight coefficients directly affects the balance between path smoothness, obstacle avoidance safety, and global target orientation. In this paper, by fixing the other parameters and adjusting each weight one by one (with a step size of 0.1), the path length, obstacle avoidance success rate, number of turning points, and number of danger points are recorded. Based on Pareto frontier analysis, the parameter combination with the optimal overall performance is selected. This combination achieves an obstacle avoidance success rate of 95.6% in the test scenario.

4. Integrated Path Planning Algorithm

The improved A^* algorithm generates a planned path that consists only of the start point, key points, and the target point. However, it cannot handle unknown obstacles that may appear in the environment. On the other hand, the DWA algorithm has excellent local obstacle avoidance capabilities but relies solely on a single final target point for guidance, making it prone to getting stuck in local optima.

To address these limitations, this study integrates both algorithms by using the key points extracted from the improved A^* algorithm as intermediate targets for the DWA algorithm. The optimized evaluation function ensures that local path planning follows the contour of the globally planned path. By integrating these path planning methods, the proposed approach enables optimal global path planning while incorporating real-time obstacle avoidance for mobile robot navigation. The detailed algorithm flow is illustrated in Figure 2.



[Fig. 2] Flowchart of the Integrated Path Planning Algorithm

A. Global Path Planning and Key Point Extraction: The global path is planned using the improved A* algorithm. The optimized cost function and key point selection strategy are employed to simplify the global path. The improved path retains only the necessary key nodes, the start point, and the target point.

B. Key Point Guidance for Local Path Planning: The extracted global path key points are used as intermediate target points for the Dynamic Window Approach (DWA) algorithm, guiding the local path planning. This allows the robot to approach the target points sequentially in complex environments.

C. Velocity Sampling and Trajectory Simulation: The DWA algorithm samples the robot's linear velocity and angular velocity in the velocity space. Using the robot's kinematic model, the trajectories corresponding to each velocity pair are simulated. These trajectories are then used for subsequent path scoring and optimization.

D. Real-Time Collection and Processing of Unknown Obstacles:

a) Static Obstacle Information Collection: Use sensors such as LiDAR or cameras to collect real-time information about the positions of unknown static obstacles. This data is then transmitted to the DWA algorithm to update the local environment.

b) Dynamic Obstacle Information Prediction: Collect information about the current positions and velocities of unknown dynamic obstacles, and predict their motion trajectories. At each time step, the obstacle position is updated based on the predicted trajectory and time step, and this information is passed to the DWA algorithm to allow it to plan avoidance measures in advance.

E. Trajectory Evaluation and Optimal Trajectory Selection: Combine global path key point information with real-time positions of unknown obstacles to construct a trajectory evaluation function. All sampled trajectories are scored, and the optimal trajectory is selected. The corresponding velocity commands are then generated to control the robot's movement toward the target point.

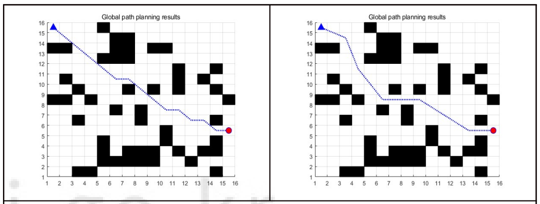
F. Execution of the Integrated Algorithm: Under the integrated navigation framework combining the improved A^* algorithm with the DWA algorithm, the mobile robot follows the global path contours, sequentially approaching or reaching key points. Meanwhile, local path planning via the DWA algorithm enables dynamic obstacle avoidance and path adjustments, ultimately leading the robot to the target point.

5. Simulation Experiments

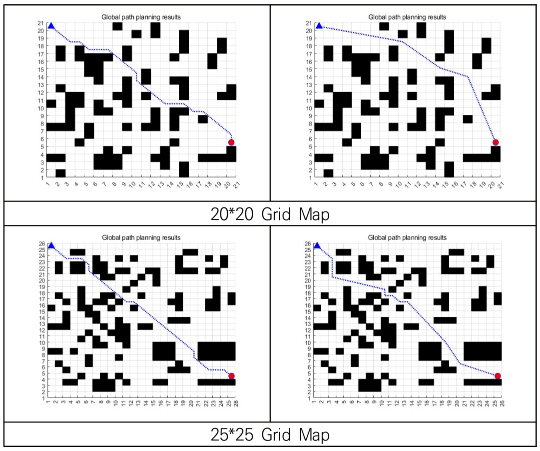
To verify the effectiveness of the proposed algorithm, grid maps were built for simulation experiments. The experimental environment used a 16GB RAM, 64-bit WIN10 operating system, and the platform was MATLAB 2023a. The simulated map designs include three grid map sizes: 15*15, 20*20, and 25*25. In these maps, white cells represent traversable areas, black cells represent obstacle areas, the starting point is located in the upper-left corner, and the goal point is located in the lower-right corner.

5.1 Simulation Comparison Experiment of Improved A^* Algorithm

In the 15*15, 20*20, and 25*25 grid map environments, simulation experiments were conducted using both the traditional A^* algorithm and the improved A^* algorithm in MATLAB. Figure 3 shows a comparison of the paths planned by the traditional A^* algorithm and the improved A^* algorithm. Table 1 presents a comparison of the performance data for the two algorithms.



15*15 Grid Map



[Fig. 3] Comparison of Path Planning Between Traditional A^* Algorithm and Improved A^* Algorithm

<Table 1> Performance Comparison Between Traditional A^* Algorithm and Improved A^* Algorithm

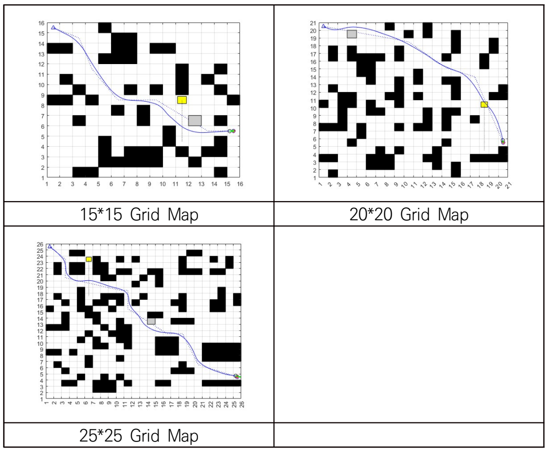
	Path length		breaking point		danger point	
	Improve	Traditional	Improve	Traditional	Improve	Traditional
15*15 MAP	18.8283	18.1421	5	7	0	5
15*15 MAP	26.5173	26.9406	4	11	0	7
25*25 MAP	35.1453	36.7990	8	10	0	7

In the simple terrain simulated with a 15*15 map size, although the traditional A^* algorithm plans a shorter path, it results in unsafe paths that cross obstacle diagonals. The improved A^* algorithm avoids crossing obstacle diagonals, ensuring path safety. At the same time, it significantly outperforms the traditional A^* algorithm in terms of the number of turns and the smoothness of the path. In more complex terrains simulated with 20*20 and 25*25 map sizes, the path planned by the improved A^* algorithm is superior to the traditional A^* algorithm in terms of safety, the number of turns, and path length.

The results of the simulation experiments show that, under different environmental conditions, the proposed improved A^* algorithm performs better than the traditional A^* algorithm in terms of path length, path safety, the number of turns in the path, and path smoothness. This indicates that the improved A^* algorithm is capable of planning a safer and more optimal path.

5.2 Integrated Algorithm Dynamic Obstacle Avoidance Simulation Results

To verify the dynamic obstacle avoidance capability of the proposed method, an environment with both unknown dynamic and static obstacles is constructed. Unknown static and dynamic obstacles are added to 15*15, 20*20, and 25*25 map environments. The simulation results show the paths planned by the improved A^* algorithm and the actual traveling paths of the fusion algorithm. The simulation results are shown in Figure 4.



[Fig. 4] Dynamic Obstacle Avoidance Fusion Algorithm for Path Planning

In the figure, the solid line represents the actual robot path, the dashed line represents the path planned by the improved A^* algorithm, small squares represent unknown moving obstacles and their movement paths, and large squares represent unknown fixed obstacles. From Figure 4, it can be observed that in all three map sizes, the fusion algorithm can guide the robot from the starting point to the goal point according to the key points generated by the improved A^* algorithm. During the journey, it successfully avoids both unknown dynamic and static obstacles.

The simulation results demonstrate that by constructing a combined key point-based evaluation function, each key point serves as an intermediate target for the DWA algorithm, thereby enabling the planning of local paths based on the global path. In the simulated environment, the fusion path planning algorithm achieves both global path optimization and dynamic obstacle avoidance for mobile robots.

6. Conclusion

This paper proposes a mobile robot path planning algorithm based on the integration of the improved A^* algorithm and the DWA algorithm, to meet the requirements of global path optimization and dynamic obstacle avoidance during the path planning process. The algorithm incorporates environmental information to optimize the cost function of the traditional A^* algorithm, allowing the cost function to adaptively adjust in segments based on the robot's position. A key point selection strategy is also used to remove redundant points and retain necessary path nodes. Simulation results show that the improved A^* algorithm effectively enhances the search efficiency and path effectiveness, resulting in a global path consisting solely of key points. Moreover, an evaluation function combining key point information is constructed, using key points as intermediate goal points for the DWA algorithm. Under the guidance of the global path, local path planning is performed to improve path smoothness while avoiding static and dynamic unknown obstacles in the environment. MATLAB simulation experiments validate that the proposed fusion navigation algorithm achieves global path optimization and dynamic obstacle avoidance for mobile robots, demonstrating practical feasibility.

Since mobile robot navigation must adapt to various scenarios, how to set the thresholds for the improved A^* algorithm and balance the weight coefficients in the DWA algorithm for different scenarios is an issue that requires careful consideration and further in-depth research in the future.

REFERENCES

[1] M. Reda, and A. Onsy, “Path planning algorithms in the autonomous driving system: A comprehensive review”, “Robotics and Autonomous Systems”, Volume 174, 2024.

[2] J.R. Sánchez-Ibáñez, “Path Planning for Autonomous Mobile Robots: A Review”, “Sensors”, 21.7898, 2021, DOI: https://doi.org/10.3390/s21237898

[3] X. J. ZHA, H. YE, and W. JIA. “A review of research on AGV path planning and obstacle avoidance algorithms”. “Department of Small Microcomputer System,2024,45(03):529-541. DOI: 10.20009.

[4] C. NING, J. FAN, and S. SUN, “A review of multi-UAV cooperative planning research”, “Computer Engineering and Applications”, 61(01):42-58, 2025.

[5] W. Cui, F. C. Zhu, “A research review on path planning algorithms for robot navigation”, “Computer Engineering and Application”,59(19):10-20, 2023.

[6] H. J. Wang, L. N. Wang, “A review of robot path planning algorithms”, “Journal of Guilin University of Technology”, 43(01):137-147, 2023.

[7] X.X.LI, X.L.MA, X. P. WANG, “A review of path planning algorithms for mobile robots”, “Computerized Measurement and Control”, 30(07):9-19, 2022, DOI: 10.16526/j.cnki.11-4762/tp.2022.07.002.

[8] C. Zeng, Z. H. Cui, “A path optimization model for accessing vehicles in a horizontal circulation class of stereo garage based on improved Dijkstra's algorithm”. “Journal of Chongqing Jiaotong University”, 42(02):93-98, 2023.

[9] X. Xu, J. Z. Zeng, Y. Zhao, and X.S. Lü, “Research on global path planning algorithm for mobile robots based on improved A^*”, “Expert Systems With Applications”, 243 (2024) 122922, DOI: https://doi.org/10.1016/j.eswa.2023.122922

[10] C.W.ZHANG, S.Y. L, P.L. QIN, “Global path planning algorithm for mining vehicles integrating simplified visibility graph and A^* algorithm”, Journal of Mine Automation , 50 (10) : 12-20 , 2024.

[11] G.P. LUO, G.L. ZHANG, M. H. YANG, “RRT global path planning algorithm based on reinforcement learning method”, “Journal of Sichuan University of

Science & Engineering (Natural Science Edition)", 37(2):57-63, 2024

[12] E.R.YANG , Z.Q. WANG , X. SUN, "Global Path Planning Algorithm for Substation Patrol Robots Based on Modified RRT", "Automation Application", Vol.65 No.7 Apr, 2024, DOI : 10.19769/j.zdhy.2024.07.012

[13] Y.L. CHEN, G.Q.BAI, Y. ZHAN, X.HU, and J. LIU, "Path Planning and Obstacle Avoiding of the USV Based on Improved ACO-APF Hybrid Algorithm with Adaptive Early-Warning", "IEEE Access", 9: 40728-40742, 2021.

[14] X. Gong, Y. Ga, F. Wang, D. Zhu, W. Zhao, F. Wang, and Y. Li, "A Local Path Planning Algorithm for Robots Based on Improved DWA", "Electronics, 2024, DOI : https:// doi.org/10.3390/electronics13152965

[15] T.R .Diao, "Research on Path Planning of Logistics Robots in Park Environment" , "Civil Aviation University of China", 2023, DOI:10.27627/d.cnki.gzmhy.2023.000504.

[16] X. B. JIAN, M. W. WANG, C.M. YANG, "Orchard spraying robot path planning based on A* and APF algorithms", "Sensors and Microsystems", 2024,43(12):145-149, DOI:10.13873/J.1000-9787(2024)12-0145-05.

[17] W. M. ZHANG, S. X. Fu, "Mobile robot path planning based on improved RRT* algorithm", "Journal of Huazhong University of Science and Technology (Natural Science Edition)",49(01):31-36, 2021, DOI: 10.13245/j.hust.210101.

노 크(Lu Ke) [정회원]

- June 2000, PLA Information Engineering University, Computer Applications (Bachelor of Engineering)
- May 2008, China University of Mining and Technology (Beijing), computer engineering technology field (Master of Engineering)
- September 2022-Present, has been pursuing Ph.D in Intelligent Fusion in IT at Mokwon University (student)

〈관심분야〉
Computer Networks, Path Planning

민 병 원(Byung-Won Min) [정회원]

- He received M.S. degree in computer software from Chungang University, Seoul, Korea in 2005.
- He received Ph.D. degree in the dept. of Information and Communication Engineering, Mokwon University, Daejeon, Korea, in 2010.
- He is currently a professor of Mokwon University since 2010.

〈관심분야〉
digital communication systems, Big Data