IJRCS

International Journal of Robotics and Control Systems
Vol. 5, No. 4, 2025, pp. 2344-2359
ISSN 2775-2658
http://pubs2.ascee.org/index.php/ijrcs

ASCEE

A Delta-Smoothed Path Algorithm: An Improved A* Algorithm for Optimal and Safe Trajectory Planning

Areej Ghazi Abdulshaheed [a,1,*]

[a] Polytechnic College / Al–Qadisiyah, Al-Furat Al- Awsat Technical University, Najaf, Iraq
[1] areej.ghazy.idi25@atu.edu.iq
* Corresponding Author

ARTICLE INFO

Article history
Received September 21, 2025
Revised October 13, 2025
Accepted November 14, 2025

Keywords
Obstacle Avoidance;
Path Planning;
Improved A* Algorithm;
Mobile Robot

ABSTRACT

Path planning plays a crucial role in the real-world applications of mobile robots in cluttered environments. A* algorithm is considered one of the most important algorithms used in path planning. However, it only works with a grid map, and implementing it in real-world applications is challenging. This paper introduces a hybrid algorithm called Delta-Smoothed Path (DSP), a novel algorithm that generates a smooth, safe, and short path. The algorithm technique involves dividing the desired path into straight, long shortcuts in open areas and small, smooth arcs when approaching sharp turns. The suggested method distinguishes itself from previous path smoothing algorithms by employing a critical angle detection mechanism to ensure the avoidance of obstacles and safety zones. Simulations were carried out in three distinct environments—a maze, a complicated environment, and a structured array—to confirm the algorithm's validity. The result shows a significant decrease in path length (up to 12.83%) and a highly reduced number of turning points (by up to 81.81%) compared to the classical A* path. The simulation results demonstrate the effectiveness of the proposed algorithm without any computational complexity, what shows its potential use in real applications.

1. Introduction

Robotic technology is currently advancing at a rapid pace, particularly in the area of mobile robots, which are extensively utilized across various industries [1], including public services, manufacturing [2], the military [3], and healthcare [4]. This highlights the necessity for robots to effectively explore their surroundings and avoid any obstacles [5]. Despite the many challenges it presents, developing mobile robots that are reliable, safe, and efficient is a vital area of robotics research [6]. Common challenges include perception, localization, motion control, and path planning [7], [8]. Among these areas, path planning may be the most critical [9].

Path planning is a technique that determines the optimal route between two places by measuring their separation in space. Multiple input values or factors are typically needed for path planning research, including starting position, goal position, and barriers [10]-[12]. Various methods are used to determine the best course of action [13]-[17]. There are many algorithms that have been applied to find the shortest path. The most important of them are:

Dijkstra algorithm is a technique that calculates the shortest path between any two nodes with values on a weighted network [18]-[22]. One of the many areas that uses algorithms is robot navigation. Although Dijkstra's algorithm is an efficient algorithm for finding the shortest path, it has several limitations. For example, it is not suitable for dynamic environments, as it requires restarting the entire process from scratch whenever the target location or an obstacle changes. Furthermore, its high computational complexity makes it very slow when dealing with large grid map. Additionally, it finds the shortest path, not necessarily the most practical or optimal path.

The A* method is a well-known heuristic approach that works well for determining the shortest pathways. It is most often used for long-term planning [23]. The A* algorithm, on the other hand, has several limitations that restrict its use. The robot finds it challenging to move in its real environment due to the excessive number of twists and inflection points in the system path planning [24], [25]. Additionally, naive smoothing methods run the risk of encroaching on safety zones that are purposefully drawn around barriers to account for uncertainty and dynamic safety margins. On grids, Dijkstra and A* produce sharp, unrealistic turns yet guarantee the shortest pathways [26].

Several path smoothing techniques have been proposed to overcome the problem of A* and Dijkstra's algorithms producing jagged paths. Among the most prominent is the B-Spline method, which generates smoother, more realistic curved paths [27]-[31]. However, it can lead to collisions with obstacles if the turning points are not precisely controlled. On the other hand, the Penalty-Based approach that uses a penalty factor to reduce sharp turns is significantly affected by the selection of the penalty coefficient, making it sensitive and unstable [32]-[35]. On the other hand, the Theta*algorithm attempts to improve the path by reducing turning angles by allowing direct lines of sight between nodes, but it does not always guarantee safe distances from obstacles [36]-[40]. Finally, the recent Hybrid improved A* algorithm combines optimal path finding with geometric smoothing, but these are computationally complex [25], [41]-[44]. Based on the limitations mentioned above, this paper proposes the Delta smoothing path (DSP) algorithm, which balances efficiency, smoothness, safety, and reduces computational complexity. It employs an intelligent mechanism for detecting critical angles and combines the creation of long shortcuts in open areas with the generation of smooth curves when approaching critical angles.

To maintain safety and ensure a collision-free robot path, expanding the distance around obstacles is crucial [45]-[48]. Several methodologies are introduced to declare the effectiveness of the safe zone. One study involves enlarging the configuration of the obstacle to account for its size by generating a buffer area to prevent collisions [49]. For example, the Multi-Robot Path Planning (MRPP) algorithm ensures safe navigation in cluttered terrain by varying the obstacle size to adjust to the robot's size [50]. Another study introduced Artificial Potential Fields (APFs), where the robot pushes away from obstacles due to the repulsive force generated by them. This technique is adopted in many dynamic application workspaces to ensure safe navigation [51], [52]. Additionally, another approach employed a buffer zone to generate space around obstacles and prevent colliding. Normally, this approach is more effective with static settings to create collusion-free routes [53]. Although all these methods differ in their methodologies, they share the same goal of keeping the robot's path free from obstacles.

Geometric smoothing approaches, such as shortcutting or line-of-sight procedures, remove redundant segments by directly linking waypoints that are not next to each other. However, these methods run the danger of cutting too close to obstacles, which could make safety margins less safe [54]. Interpolation methods, including spline fitting using B-splines or Bézier curves, as applied in vision-aided UAV navigation through gradient-based B-spline trajectory optimization [27], [55], [56], offer curvature continuity and aesthetic smoothness but may inadvertently curve into restricted regions if control points are poorly managed. Optimization-based methods define smoothing as mathematical algorithms that minimize curvature or other criteria. These methods are powerful, but they can be too aggressive in their pursuit of smoothness, which could bring the distance to barrier borders closer. A major problem with all of these methods is getting smooth paths without breaking the safety zone [57]. It is still important to find new ways to include obstacle margins in the smoothing process.

Robotic navigation systems require not only short paths but also trajectories that are curvature friendly.

This study addresses these limitations by presenting a novel hybrid DSP algorithm. The proposed algorithm optimizes the raw path generated by the A* planner to reduce the path length. This hybrid method generates smooth, collision-free behavior by utilizing a quadratic Bézier curve and achieving both locally safe and globally efficient trajectories. The main contributions of this study are:

- Develop a new hybrid path planning algorithm (DSP) that combines safe awareness behavior and long path optimization.
- A Critical Corner detection mechanism that controls the shape of the path depending on how close it is to the dangerous zone.
- Evaluate the validity of the algorithm in three different demanding frameworks (a structured array, a complicated obstacle setting, and a maze)

The remainder of this paper is organized as follows. Section 2 details the complete methodology, from environment representation to the implementation of the A* and DSP algorithms. Section 3 presents the simulation results of the three environments and discusses the result. Finally, Section 4 concludes the paper and suggests directions for future work.

2. Method

The proposed path planning framework is a two-stage process. First, an initial, globally optimal path is generated on a discrete grid using the A* search algorithm. Second, this path is refined by the novel DSP (Delta-Smoothed Path) algorithm to produce a shorter, smoother, and kinematically feasible trajectory suitable for a real-world robot.

2.1. Environment Representation and Node Generation

The foundation of the planner is the representation of the robot's workspace. The continuous two-dimensional workspace, denoted as $W \subset R^2$, contains a set of obstacles (O) and the robot's start (p_{start}) and goal (p_{goal}) positions. To ensure safe navigation, each obstacle is enclosed by an expanded buffer region, creating a set of safety zones (S). The resulting collision-free configuration space, C_{free}, is the area within the workspace that is outside of both the obstacles and their safety zones.

To prepare for the graph search, this continuous space is discretized into a uniform grid G with a resolution of δ. A validation process then filters this grid, retaining only the nodes that lie within C_{free}. This produces the final set of traversable waypoints, N_{valid}. To guarantee a solution can be found, the start and goal points are explicitly added to this set, creating the final node set N_{final} used for planning. This process is formally defined as follows: Let the workspace be W, the set of obstacles O, and the set of safety zones S. The formula of the free space is:

$$C_{free} = W \setminus (O \cup S) \tag{1}$$

The grid points set G is:

$$G=\{(x,y) \mid x=i\delta,\ y=j\delta;\ i,j\in Z;\ (x,y)\in W\} \tag{2}$$

The formulation of the set of valid nodes:

$$N_{valid}=\{p \mid p \in G \text{ and } p \in C_{free}\} \tag{3}$$

The final set of valid nodes is:

$$N_{valid}=N_{valid} \cup \{p_{start}, p_{goal}\} \tag{4}$$

2.2. A* Path Planner

The A* planner is one of the major classical methods for path planning and graph traversal. This approach was proposed in 1968 to enhance the Dijkstra algorithm. Its evaluation function makes it one of the best first operation searching systems [58], [59]. Using the finalized node set Nfinal , a graph $G=(V,E)$ is constructed where the nodes V are the points in Nfinal. An edge E is created between any two nodes that are within an 8-connected neighborhood (i.e., distance $\leq\sqrt{2}\delta$) and where the straight line connecting them is entirely within Cfree. The A* algorithm is then employed to find the shortest path on this graph. It uses the evaluation function $f(n) = g(n)+h(n)$ where

- $g(n)$ is the known, cumulative cost (sum of Euclidean distances) from the start node p_{start} to node n.
- $h(n)$ is the heuristic estimate of the cost from node n to the goal p_{goal}, calculated as the direct Euclidean distance: $h(n)=\|n-g\|_2$.

This process yields an initial path $P=\{p_1, p_2, \ldots, p_N\}$. Although this path is optimum on the grid map, it is not suitable for the robot's practical operations because it consists of many unnecessary turns and jagged motion. This path P is the input to the optimization stage.

2.3. Path Optimization: The DSP Algorithm

While the A* algorithm provides an optimal path on the discrete grid, its output p is a sequence of waypoints $\{p_1, p_2, \ldots, p_n\}$ that results in a jagged, kinematically inefficient trajectory for a physical robot [60]. The proposed DSP algorithm is a sophisticated post-processing method designed to transform this raw path into a continuous, smooth, and safe trajectory $\tilde{P}(t)$. The algorithm's intelligence lies in its hybrid strategy, which dynamically balances aggressive, long-distance optimization in open spaces with cautious, localized maneuvering in cluttered areas.

This is achieved by iteratively processing the A* path from a current node p_i and deciding between two primary actions: executing a long, straight-line shortcut or inserting a safe, curved arc around a dangerous corner. The DSP mechanism and the flow of the algorithm are shown in Fig. 1 and Fig. 2, respectively.

2.3.1. Critical Corner Detection Mechanism

An aggressive shortcut path can be dangerous, as it may cut the corner of an obstacle, leading to an unwanted collision. To prevent this risk, a crucial safety check is performed first. This step identifies all corners of the safety zones that the A* path passes very close to. These specific danger points are labeled as Critical Corners (*critCorners function*), as defined in Section 3.1.

These points serve as triggers to warn of approaching dangerous areas. When Critical Corner is detected, the smoother algorithm will switch to the safe mode and generate a smooth arc instead of taking a straight, long shortcut.

2.3.2. The Veto and Cautious Arc generation

The smooth trajectories of the DSP algorithm consist of several segments. If the next waypoint in the path $p_{\{i+1\}}$ is indicated as a Critical Node, the algorithm overrides its default "greedy" behavior. It prioritizes safety first and generates smooth, localized curve navigation, see Fig. 1. The three controlled waypoints define the flaunt curve trajectories using quadratic Bézier are:

P_A: refer to the last point confirmed by the optimized path, out(end). This guarantees a fluent, seamless "vortex" between segments.

P_B: It is a Critical point, $p_{\{i+1\}}$. It is the vertex point that controls its shape.

P_C: The next point after the corner, $p_{\{i+2\}}$.

The equation that defines the resulting curve:

$$P(t) = (1\text{-}t)^2 P_A + 2(1\text{-}t)tP_B + t^2 P_C, \quad for \quad t \in [0,t] \tag{5}$$

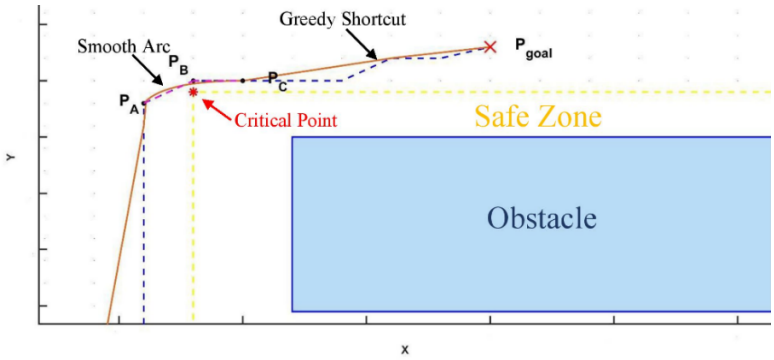

Fig. 1. The veto mechanism of DSP algorithm

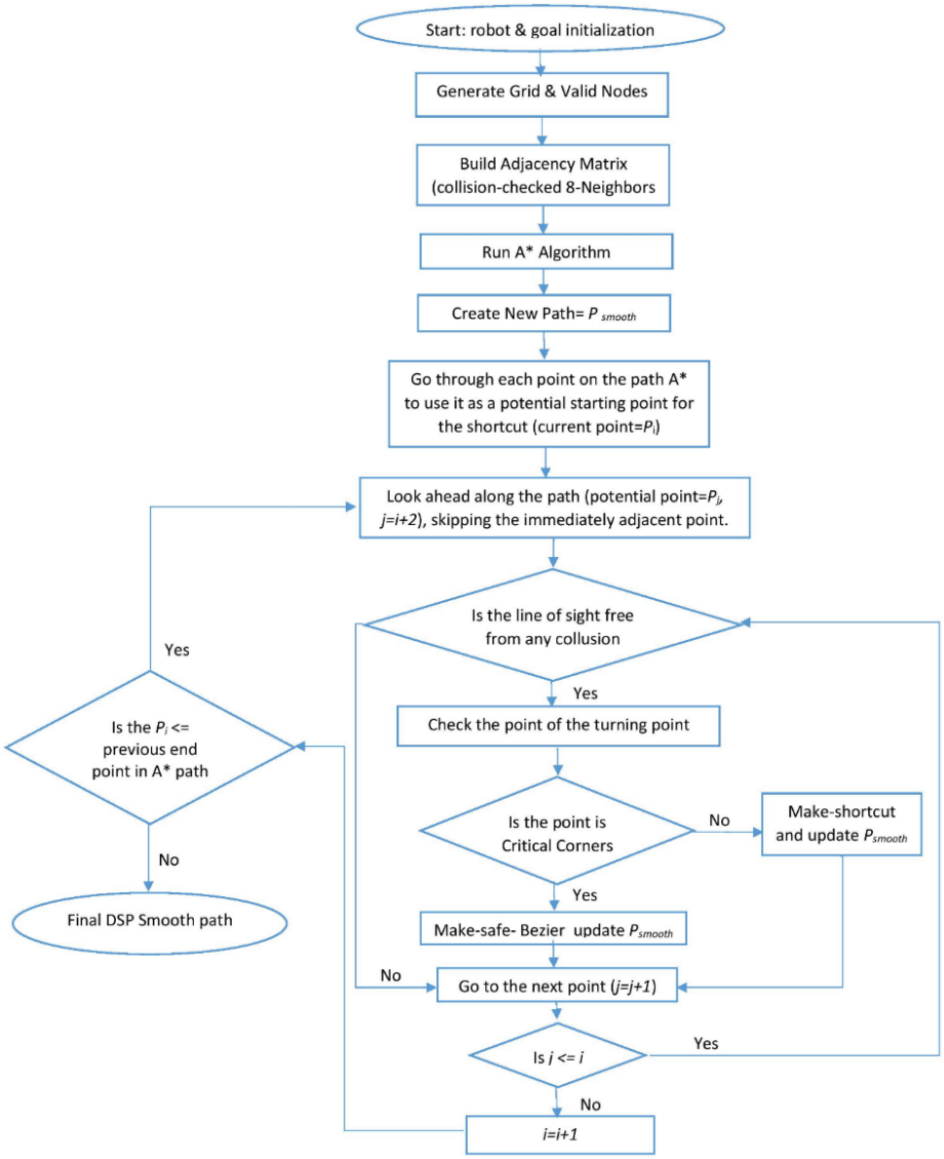

Fig. 2. The flow chart of DSP algorithm

To ensure collusion-free behavior. The initial curve is checked if it goes into a safety zone, where the control point P_B is incrementally pushed outward from the obstacle, along the normal vector with parameter *arcGrowStep*. This process will be stopped when the curve points are verified to be out of the dangerous area. If no safe curve can be found with a maximum offset (*arcMaxOffset parameter*), the default option will be chosen, and the original sharp turn will be executed to ensure safety will not break out.

2.3.3. The Greedy Shortcut path

The DSP algorithm goes into greedy shortcut optimization mode if the upcoming turn is not critical. It executes a "*lookahead*" search. This search starts from the last point of the A* path and goes backward, looking for the farthest possible point p_j that is accessible from the present position {out(end)} by a direct, safe straight line.

The DSP algorithm uses an important strategy, "prudent approach". This strategy depends on preventing the desired path from ending at a sharp corner by connecting the path trajectory to the point, p_{j-1}, which directly precedes the farthest point p_j. This creates more natural, fluid preparation for the next movement. A basic linear interpolation represents the straight-line shortcut:

$$P(t) = (1\text{-}t)\, P_{\{start\}} + tP_{\{end\}}, \qquad for \quad t \in [0,t] \tag{6}$$

Immediately after performing either a shortcut or an arc, the internal pointer of the algorithm i jumps forward to the end of the newly generated segment, and this process continues until the entire path has been optimized. The final hybrid trajectories are composed of long, straight shortcuts in open space and smooth curves in critical, tight areas.

3. Results and Discussion

To rigorously validate the effectiveness and versatility of the proposed DSP algorithm, a series of simulations was conducted. The experiments were run on a Windows 10 computer with an Intel(R) Core (TM) i7-5600U CPU, using MATLAB 2023b as the simulation platform.

3.1. Experimental Setup

To testify the algorithm performance, three different simulation maps are set up under: a structure array, a cultured environment with different shapes, and a maze, see Fig. 3. For each scenario, multiple simulation runs were performed by varying the goal position and fixing the start point at (2,1).

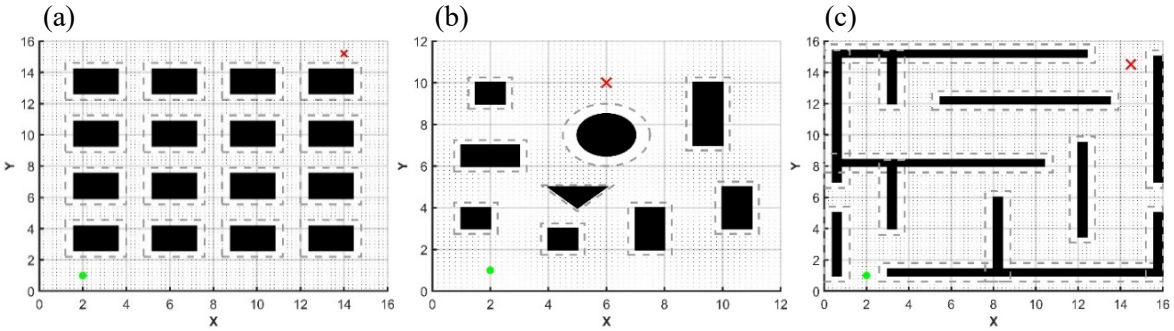

Fig. 3. Three different grid maps: (a) Array map (b) Complex map (c) Maze map

The simulations are applied in a 2D workplace with uniform grid resolution (δ) of 0.2 meters. To ensure collision-free navigation, the obstacles are surrounded by an expanded safety area. The dashed blue line represents the initial path found by the classical A* planner. This path the enhanced by the DSP algorithm to introduce the final optimized solid orange path. The Critical corners is represented by red dots. The critical corners are defined within a sensitivity threshold (*cornerThresh*) of 0.3 of the

A* path. The increment offset (*arcInitOffset*) between the critical corner and the control point P_B is set as 0.12. The maximum arc offset (*arcMaxOffset*) is identified as 0.6.

3.2. Performance Evaluation

Three main indicators are used to evaluate the performance of the A* and DSP algorithms. The first one is the length of the generated path. The shorter path is more efficient. Secondly, the turning points refer to the number of times the robot changes its direction. The lower number of turns is the simpler route for the robot to go through since it's kinematically simpler. Finally, the total turning angle represents the simulation of the angle along the route. A lower value indicates fewer rotations and a smoother trajectory. The performance of the DSP algorithm was tested across three distinct environments, with four goal variations for each, as depicted in Fig. 4, Fig. 5, and Fig. 6.

3.2.1. Case 1: Structured Array Environment

This environment, shown in Fig. 4 structured to examine the ability of the DSP algorithm to improve the raw path generated by A* planner. This map consists of regular passages (4 ɤ 4). The path length is shortened by up to 8.59% as shown in Table 1. The visual result in Fig. 4–(a) to (d), where the blue route of A* planner takes staircase movement compared to the orange path, diagonal straight motion. The remarkable enhancement can be seen in the reduction of the number of turning points, which has been reduced up to 66.66%, as the proposed algorithm combines multiple jagged A* turns into a single, smooth curve. For example, in Fig. 4–(d), the DSP path required only 9 main movement adjustments, while the raw path of A* makes a complicated sequence of 2 turns. The results indicate that the DSP path is much more practical. The quantitative comparison of DSP Algorithm and Algorithm A*'s execution times in Table 1 makes it evident that the DSP Algorithm outperforms the latter by over 93%. Because of the nature of the two algorithms, the suggested method relies on a mechanism to improve the path that already exists, exclude all additional nodes within the safety zone, and limit the search space to finding and identifying critical nodes, whereas Algorithm A* relies on a thorough search throughout the entire grid for valid nodes to reach the goal.

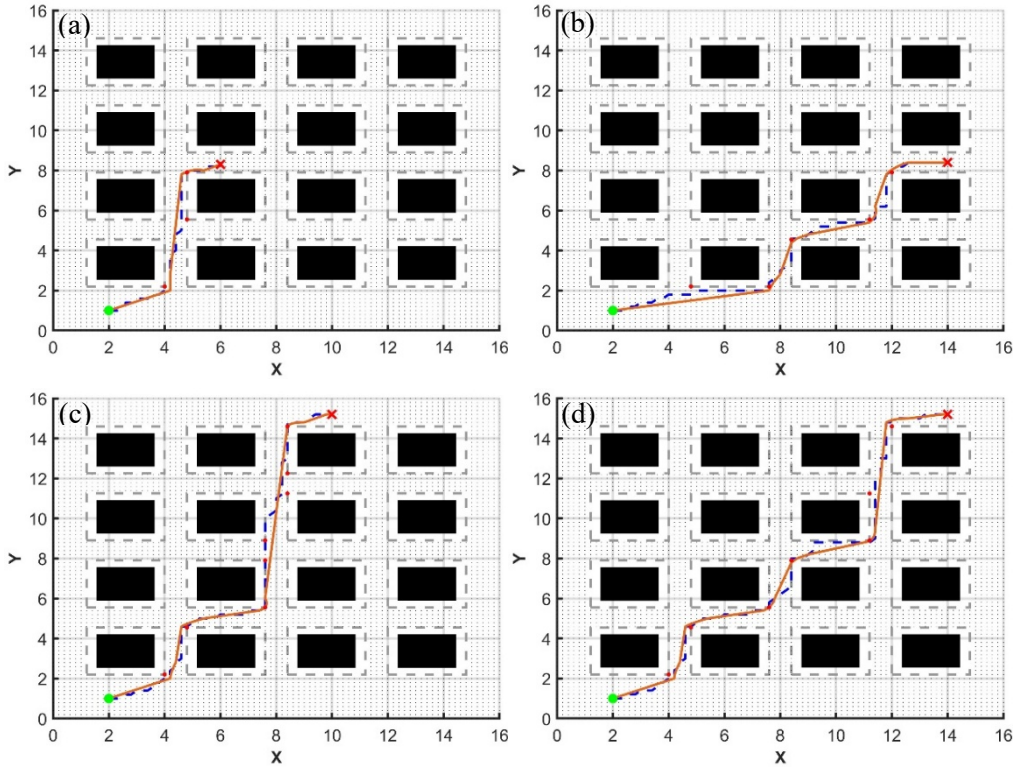

Fig. 4. Simulation results of the DSP algorithm in a structured array environment with uniform corridors for four different goal positions (a-d)

Table 1. Comparison of the structured array environment simulation result

Figure No.	Goal position (X_R, Y_R)	Path parameter	A* path	DSP path	Reduction ratio
2-(a)	6, 8.3	Path length	10.2863	9.6276	6.4%
		Turning point	10	4	60%
		Total Turning Angle	1327.68	589.60	55.59%
		Running time	0.0405 s	0.0026 s	93.58%
2-(b)	14, 8.4	Path length	17.4083	16.1712	7.1%
		Turning point	18	8	55.55%
		Total Turning Angle	2299.55	1180.74	48.65%
		Running time	0.1472 s	0.0170	88.45%
2-(c)	10, 15.2	Path length	20.3255	19.1045	6%
		Turning point	18	6	66.66%
		Total Turning Angle	2320.52	815.01	64.87%
		Running time	0.1750 s	0.0180 s	89.71%
2-(d)	14, 15.2	Path length	23.9740	21.9126	8.59%
		Turning point	27	9	66.66%
		Total Turning Angle	3304.16	1241.05	62.43%
		Running time	0.2591 s	0.0193 s	64.1%

3.2.2. Case 2: Complex Environment with Varied Obstacles

To verify the safety and validity of the proposed algorithm, it was tested in an environment with different shapes (an ellipse, a rectangle, and a triangle), see Fig. 5. The data in Table 2 showed a reduction in the DSP path length up to 10.12% and a remarkable reduction in the number of turning points, reaching 78.94%. The effectiveness concept of the Critical points is demonstrated clearly through these figures. Fig. 5–(a) and Fig. 5–(b) show that the raw A* path crosses the risky zone close to the inverted triangle's sharp corner. The DSP algorithm was able successfully detected this as a dangerous area (identified by a red Critical point) and created a caution, controlled curve for safer navigation. This adaptable, smart attitude is constantly used across all scenarios presented. Table 2 demonstrates that the DSP algorithm's execution time does not exhibit a consistent pattern in every case. In contrast to scenarios (c) and (d), which have execution times that are shorter than those of the A* method, Table 2 shows that in scenarios (a) and (b), the execution time is longer than in the A* path for the same scenario. This is because complex surroundings with a variety of geometric shapes require more computing time. As can be seen in Table 2 scenario (c), the execution time for the DSP path increases by 57.82 % in comparison to A*, making this increase noticeable as the path length grows.

3.2.3. Case 3: Maze Environment

The maze (Fig. 6) is considered one of the most challenging fields for mobile navigation because it combines long passages with narrow, aggressive corners. Table 3 shows that the path length is reduced by up to 12.83% and the number of turning points by up to 81.81%. The orange DSP path in Fig. 6–(b) and Fig. 6–(c) shows successful robot navigation along the long corridors of the maze. The robot in Fig. 6–(a) executes a straight shortcut until it reaches a critical corner, where it switches its strategy. It is forced to make safe Bezier forcing by the Critical points, which act as triggers. However, at some tight corners, the robot is forced to turn aggressively due to the safety check. The DSP algorithm checks the three points that form the curve around the critical point. If one of them intersects with the safety zone surrounding the obstacle, it will abandon the arc shaping. Meanwhile, in other critical points, the function succeeds in forming safe arcs, which are less steep and have a broad space for maneuvering. Across all four maze scenarios, the DSP algorithm proved to be a smart and robust mechanism for cluttered terrain, due to its high capability to balance between high awareness navigation and aggressive optimization. Table 3 demonstrates that the DSP algorithm outperforms the A* method in terms of time by about 88%, suggesting that the system can re-plan rapidly. This indicates a qualitative change in performance efficiency in addition to a straightforward numerical decrease in execution time. As a result, real-time applications can use the suggested algorithm.

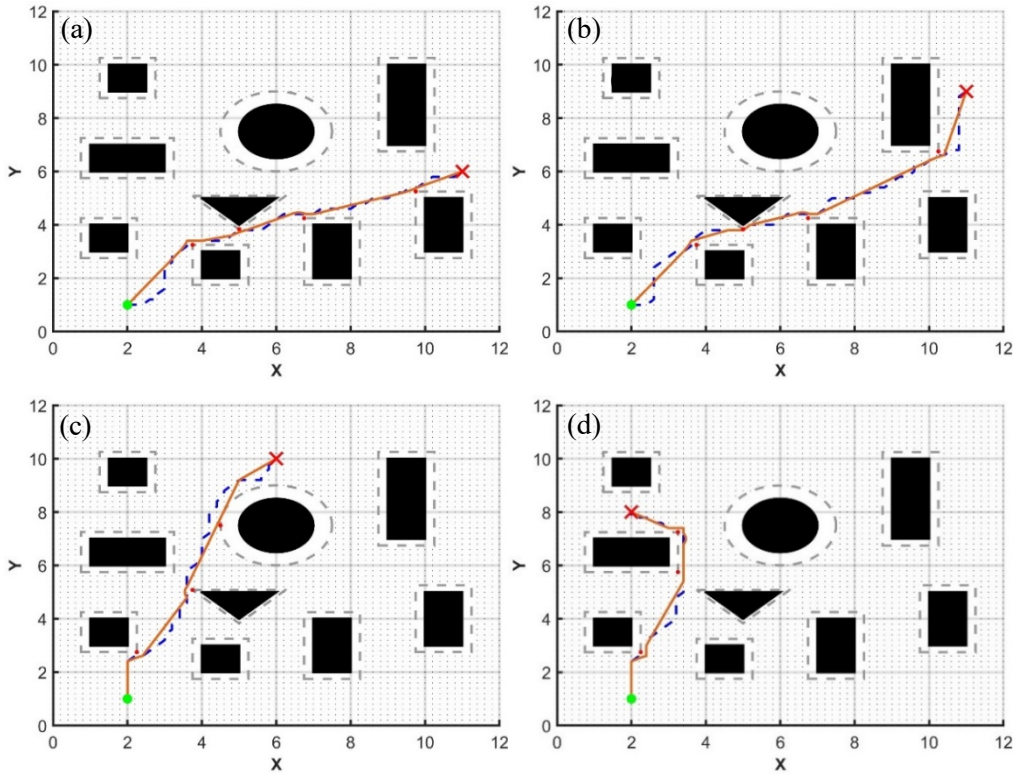

Fig. 5. Simulation results of the DSP algorithm in the complex environment for four different goal positions (a-d)

Table 2. Comparison of the complex environment simulation result

Figure No.	Goal position (X_R, Y_R)	Path parameter	A* path	DSP path	Reduction ratio
3-(a)	11, 6	Path length	11.8912	10.8420	8.82%
		Turning point	15	5	66.67%
		Total Turning Angle	2090.60	762.81	63.51%
		Running time	0.0767 s	0.1764 s	-56.51%
3-(b)	11, 9	Path length	14.5397	13.0680	10.12%
		Turning point	19	4	78.94%
		Total Turning Angle	2542.38	576.98	77.30%
		Running time	0.0954 s	0.2262 s	-57.82%
3-(c)	6, 10	Path length	11.1255	10.2972	7.35%
		Turning point	11	5	54.54%
		Total Turning Angle	1556.57	707.39	54.55%
		Running time	0.0493 s	0.0345 s	30%
3-(d)	2, 8	Path length	8.8627	8.2404	7.02%
		Turning point	10	4	60%
		Total Turning Angle	1245.75	553.22	55.59%
		Running time	0.0479 s	0.0371 s	22.54%

The simulation results in the three different types of environments showed a clear superiority of the DSP algorithm over the traditional A* algorithm. According to the quantitative results obtained from the simulation, the DSP algorithm was able to achieve a smooth shortest path (up to 12.83%), using the least number of turns (with a reduction up to 80.65 %). The visual results show the transformation from grid grid-pounded path to a fluid trajectory. The success of the DSP algorithm in implementing the Critical point technique makes it a promising, robust, and more secure method for navigation in a cluttered environment.

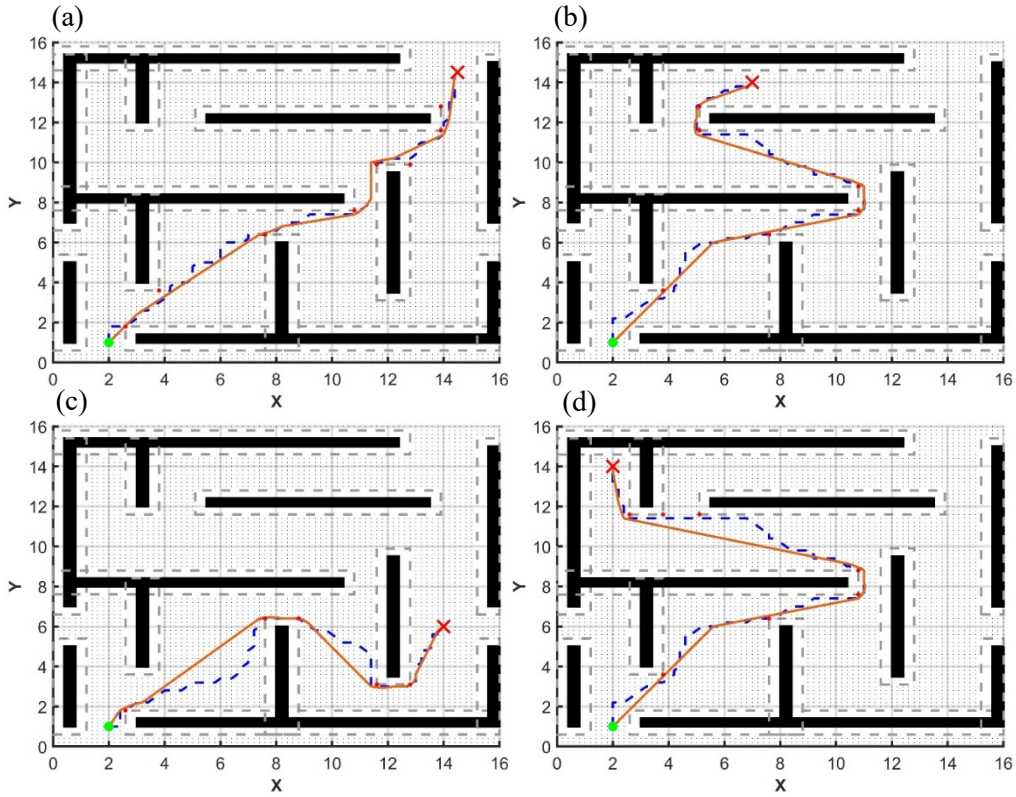

Fig. 6. Simulation results of the DSP algorithm in the highly constrained maze environment for four different goal positions (a-d)

Table 3. Comparison of the MAZE simulation result

Figure No.	Goal position (X_R, Y_R)	Path parameter	A* path	DSP path	Reduction ratio
4-(a)	14.5, 14.5	Path length	22.7782	20.2123	11.26%
		Turning point	29	9	68.96%
		Total Turning Angle	3711.64	1358.76	63.39%
		Running time	0.1385 s	0.0193 s	86%
4-(b)	7, 14	Path length	26.2510	23.2800	11.31%
		Turning point	38	7	81.57%
		Total Turning Angle	4781.57	952.63	80%
		Running time	0.2445 s	0.0610 s	75%
4-(c)	14, 6	Path length	20.7539	18.0898	12.83%
		Turning point	29	10	65.51%
		Total Turning Angle	3722.02	1494.80	59.83%
		Running time	0.1946 s	0.0750 s	61.45%
4-(d)	2, 14	Path length	27.4853	24.6513	10.31%
		Turning point	33	6	81.81%
		Total Turning Angle	4272.75	826.18	80.65%
		Running time	0.2301 s	0.0276 s	88%

Fig. 7 shows a comparison between three optimization methods for Algorithm A*(DSP, Theta*, and Penalty-based approach). From the figure, it can be seen that the length of the DSP path is slightly longer than the path of Theta* approach and shorter than the Penalty-Based path. This is due to the combination of the two mechanisms, namely long shortcuts and smooth Bézier arcs. This combination is useful for shortening the path in wide areas without taking risks on tight curves, while Theta* approach reduces the length significantly, but it passes close to the edges and makes sharp turns, as shown in Fig. 7–(b). The reason why the number of nodes and turning angles in DSP algorithm is greater than Theta is due to the presence of increasing of the number of nodes that shape the smooth arcs in the DSP path, as shown in Table 4. Fig. 7–(c) shows the optimized path using a penalty factor

of 0.4. This path shows an oscillation in the behavior between areas that exhibit sharp angles and a smooth motion in other locations, which is greatly affected by the penalty factor. The increasing of the penalty factor leads to the production of longer paths with lower turning angles. Unlike the execution time, which shows a noticeable superiority of the DSP algorithm over the rest of the smoothing methods, this is due to the balance that the DSP algorithm achieves between maintaining the shortness of the path on the one hand and smoothness and reducing the turning angles on the other hand, without compromising safety requirements or increasing the computational time, which makes it an ideal choice for application in complex environments with a high density of obstacles.

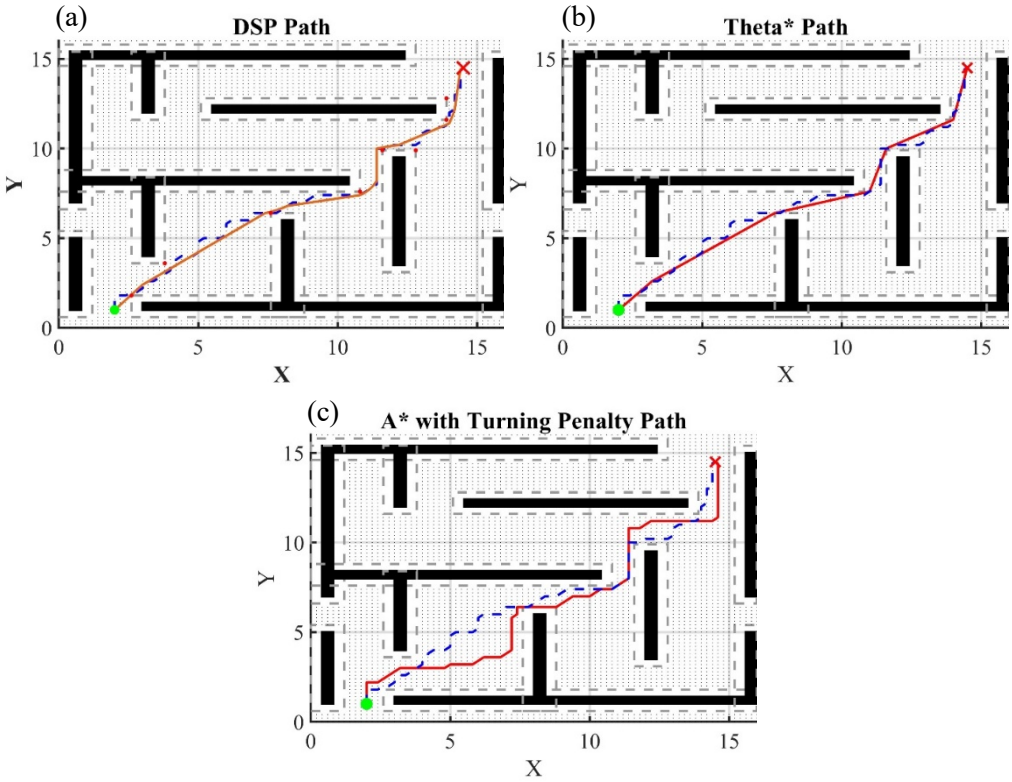

Fig. 7. Comparison between three A* improved path methods in complex environment (a) DSP path (b) Theta* path (c) Penalty based approach path

Table 4. Comparison of the simulation result between different path planning approach

Path parameter	A* path	DSP path	Theta * path	Penalty path
Path length/m	22.7782	20.2123	19.7204	23.6811
Turning point	29	9	5	23
Total Turning Angle	3711.64	1358.76	720.98	2968.15
Running time / s	0.1780	0.0573	1.5941	0.2200

4. Conclusion

This paper presents and evaluates a novel algorithm, the Delta smooth algorithm (DSP), used to optimize the path generated by the A* algorithm. The inherent constraints of the grid-based search algorithm, which generates a jagged, kinematically inefficient trajectory, and is inappropriate for robot practical application, motivated the presentation of this study. These limitations are successfully addressed by the DSP algorithm by smartly optimizing the long, rough distance with safety-aware, cautious navigation.

The validity of the proposed algorithm was demonstrated by conducting a series of simulations in three different challenging environments. The quantitative results showed a consistent and noticeable improvement in the optimized path compared to the raw A* path. The path length and the

number of turns have been reduced significantly using the proposed algorithm. The DSP algorithm's main strength is highlighted by its ability to convert the irregular, jagged path into a smooth, executable path trajectory. Critical Nodes is considered the main innovation of this algorithm, which is used as a trigger to switch from a straight, shortcut path to a controlled and safe Bezier curve. Through the results, the DSP algorithm has been proven to be a strong strategy for maneuvering in a cluttered environment without risking safety.

When comparing the DSP algorithm with the previously presented smoothing methods, we note that the DSP algorithm combines the formation of short shortcut paths, which is characteristic of the Theta* method, with smooth curves, as in the Penalty-Based method. In addition, it outperforms both of them in execution time. Thus, it achieves an ideal balance between path length, execution speed, and path integrity, making it an ideal choice for practical applications.

Despite the high performance of the proposed algorithm, it has some limitations that determine the scope of this study. First, the entire system is tested on a static prior map and is not designed to operate in a dynamic environment. Second, the DSP algorithm cannot find a new route but rather relies on optimizing the initial global path generated by A* planner. Finally, the algorithm may not be suitable for a high-frequency application because it consists of two stages of processing and which may add a computational burden. These limitations can lead to several future work. The framework can be extended to a dynamic platform by integrating the DSP algorithm with a local planner, for example Dynamic Window Approach (DWA). The DWA can handle the obstacle avoidance, while the DSP can provide an overall fluent trajectory.

In conclusion, the DSP algorithm provides a practical, strong, and smart mechanism for optimizing grid-based paths. It successfully could provide a powerful solution for creating smooth, robust trajectories, making a big step forward towards navigation of the robot in a real-world field, by balancing security and efficiency.

Author Contribution: All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding: This research received no external funding

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] M. G. Mohanan and A. Salgaonkar, "Robotic motion planning in dynamic environments and its applications," *International Journal of Robotics and Control Systems*, vol. 2, no. 4, p. 26, 2022, https://doi.org/10.31763/ijrcs.v2i4.816.

[2] J. Jiang and Y. Ma, "Path planning strategies to optimize accuracy, quality, build time and material use in additive manufacturing: A review," *Micromachines*, vol. 11, no. 7, p. 633, 2020, https://doi.org/10.3390/mi11070633.

[3] H. S. Yahia and A. S. Mohammed, "Path planning optimization in unmanned aerial vehicles using meta-heuristic algorithms: A systematic review," *Environmental Monitoring and Assessment*, vol. 195, no. 1, p. 30, 2023, https://doi.org/10.1007/s10661-022-10590-y.

[4] R. Raj and A. Kos, "A comprehensive study of mobile robot: History, developments, applications, and future research perspectives," *Applied Sciences*, vol. 12, no. 14, p. 6951, 2022, https://doi.org/10.3390/app12146951.

[5] M. B. Alatise and G. P. Hancke, "A review on challenges of autonomous mobile robot and sensor fusion methods," *IEEE Access*, vol. 8, pp. 39830–39846, 2020, https://doi.org/10.1109/ACCESS.2020.2975643.

[6] R. Hercik, R. Byrtus, R. Jaros, and J. Koziorek, "Implementation of autonomous mobile robot in smart factory," *Applied Sciences*, vol. 12, no. 17, p. 8912, 2022, https://doi.org/10.3390/app12178912.

[7] K. Karur, N. Sharma, C. Dharmatti, and J. E. Siegel, "A survey of path planning algorithms for mobile robots," *Vehicles*, vol. 3, no. 3, pp. 448–468, 2021, https://doi.org/10.3390/vehicles3030027.

[8] C. S. Tan, R. Mohd-Mokhtar, and M. R. Arshad, "A comprehensive review of coverage path planning in robotics using classical and heuristic algorithms," *IEEE Access*, vol. 9, pp. 119310–119342, 2021, https://doi.org/10.1109/ACCESS.2021.3108177.

[9] H. Wang, X. Qi, S. Lou, J. Jing, H. He, and W. Liu, "An efficient and robust improved A* algorithm for path planning," *Symmetry*, vol. 13, no. 11, p. 2213, 2021, https://doi.org/10.3390/sym13112213.

[10] A. G. Abdulshaheed and F. Kamil, "Path planning control in known environments using turning points," *International Journal of Advanced and Applied Sciences*, vol. 11, no. 12, pp. 129–139, 2024, https://doi.org/10.21833/ijaas.2024.12.015.

[11] J. R. Sánchez-Ibáñez, C. J. Pérez-del-Pulgar, and A. García-Cerezo, "Path planning for autonomous mobile robots: A review," *Sensors*, vol. 21, no. 23, p. 7898, 2021, https://doi.org/10.3390/s21237898.

[12] J. Qi, H. Yang, and H. Sun, "MOD-RRT*: A sampling-based algorithm for robot path planning in dynamic environment," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 8, pp. 7244–7251, 2020, https://doi.org/10.1109/TIE.2020.2998740.

[13] J. A. Abdulsaheb and D. J. Kadhim, "Classical and heuristic approaches for mobile robot path planning: A survey," *Robotics*, vol. 12, no. 4, p. 93, 2023, https://doi.org/10.3390/robotics12040093.

[14] R. Singh, J. Ren, and X. Lin, "A review of deep reinforcement learning algorithms for mobile robot path planning," *Vehicles*, vol. 5, no. 4, pp. 1423–1451, 2023, https://doi.org/10.3390/vehicles5040078.

[15] Y. Xu, Q. Li, X. Xu, J. Yang, and Y. Chen, "Research progress of nature-inspired metaheuristic algorithms in mobile robot path planning," *Electronics*, vol. 12, no. 15, p. 3263, 2023, https://doi.org/10.3390/electronics12153263.

[16] H. Liu, Y. Shen, S. Yu, Z. Gao, and T. Wu, "Deep reinforcement learning for mobile robot path planning," *arXiv preprint* arXiv:2404.06974, 2024, https://doi.org/10.48550/arXiv.2404.06974.

[17] M. Jones, S. Djahel, and K. Welsh, "Path-planning for unmanned aerial vehicles with environment complexity considerations: A survey," *ACM Computing Surveys*, vol. 55, no. 11, pp. 1–39, 2023, https://doi.org/10.1145/3570723.

[18] M. Luo, X. Hou, and J. Yang, "Surface optimal path planning using an extended Dijkstra algorithm," *IEEE Access*, vol. 8, pp. 147827–147838, 2020, https://doi.org/10.1109/ACCESS.2020.3015976.

[19] S. Sundarraj, R. V. K. Reddy, M. B. Basam, G. H. Lokesh, F. Flammini, and R. Natarajan, "Route planning for an autonomous robotic vehicle employing a weight-controlled particle swarm-optimized Dijkstra algorithm," *IEEE Access*, vol. 11, pp. 92433–92442, 2023, https://doi.org/10.1109/ACCESS.2023.3302698.

[20] R. S. Wijaya, A. Mahendra, S. Prayoga, and A. Wibisana, "Path planning application using Dijkstra algorithm on service robot," in *Proc. 7th Int. Conf. Appl. Eng. (ICAE),* 2024, pp. 262–271, https://doi.org/10.2991/978-94-6463-620-8_20.

[21] N. Abu, W. Bukhari, M. Adli, and A. Ma'arif, "Optimization of an autonomous mobile robot path planning based on improved genetic algorithms," *Journal of Robotics and Control (JRC)*, vol. 4, no. 4, pp. 557–571, 2023, https://doi.org/10.18196/jrc.v4i4.19306.

[22] A. N. A. Rafai, N. Adzhar, and N. I. Jaini, "A review on path planning and obstacle avoidance algorithms for autonomous mobile robots," *Journal of Robotics*, vol. 2022, no. 1, p. 2538220, 2022, https://doi.org/10.1155/2022/2538220.

[23] D. Mandloi, R. Arya, and A. K. Verma, "Unmanned aerial vehicle path planning based on A* algorithm and its variants in 3D environment," *International Journal of System Assurance Engineering and Management*, vol. 12, no. 5, pp. 990–1000, 2021, https://doi.org/10.1007/s13198-021-01186-9.

[24] H. Qin, S. Shao, T. Wang, X. Yu, Y. Jiang, and Z. Cao, "Review of autonomous path planning algorithms for mobile robots," *Drones*, vol. 7, no. 3, p. 211, 2023, https://doi.org/10.3390/drones7030211.

[25] H. Sang, Y. You, X. Sun, Y. Zhou, and F. Liu, "The hybrid path planning algorithm based on improved A* and artificial potential field for unmanned surface vehicle formations," *Ocean Engineering*, vol. 223, p. 108709, 2021, https://doi.org/10.1016/j.oceaneng.2021.108709.

[26] E. M. Miyombo, Y. Liu, M. M. Chishinga, A. Siamulonga, M. C. Kabanda, P. Shaba, C. Xi, and A. Ayodeji, "Optimal path planning in a real-world radioactive environment: A comparative study of A-star and Dijkstra algorithms," *Nuclear Engineering and Design*, vol. 420, p. 113039, 2024, https://doi.org/10.1016/j.nucengdes.2024.113039.

[27] F. Huo, S. Zhu, H. Dong, and W. Ren, "A new approach to smooth path planning of Ackerman mobile robot based on improved ACO algorithm and B-spline curve," *Robotics and Autonomous Systems*, vol. 175, p. 104655, 2024, https://doi.org/10.1016/j.robot.2024.104655.

[28] M. Liu, H. Zhang, J. Yang, T. Zhang, C. Zhang, and L. Bo, "A path planning algorithm for three-dimensional collision avoidance based on potential field and B-spline boundary curve," *Aerospace Science and Technology*, vol. 144, p. 108763, 2024, https://doi.org/10.1016/j.ast.2023.108763.

[29] S. Eshtehardian and S. Khodaygan, "A continuous RRT*-based path planning method for non-holonomic mobile robots using B-spline curves," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 7, pp. 8693–8702, 2023, https://doi.org/10.1007/s12652-021-03625-8.

[30] P. Wang, J. Yang, Y. Zhang, Q. Wang, B. Sun, and D. Guo, "Obstacle-avoidance path-planning algorithm for autonomous vehicles based on B-spline algorithm," *World Electric Vehicle Journal*, vol. 13, no. 12, p. 233, 2022, https://doi.org/10.3390/wevj13120233.

[31] N. T. Nguyen, P. T. Gangavarapu, N. F. Kompe, G. Schildbach, and F. Ernst, "Navigation with polytopes: A toolbox for optimal path planning with polytope maps and B-spline curves," *Sensors*, vol. 23, no. 7, p. 3532, 2023, https://doi.org/10.3390/s23073532.

[32] K. K. Pandey, C. Kumbhar, D. R. Parhi, S. K. Mathivanan, P. Jayagopal, and A. Haque, "Trajectory planning and collision control of a mobile robot: A penalty-based PSO approach," *Mathematical Problems in Engineering*, vol. 2023, no. 1, p. 1040461, 2023, https://doi.org/10.1155/2023/1040461.

[33] S.-L. Jeng and C. Chiang, "End-to-end autonomous navigation based on deep reinforcement learning with a survival penalty function," *Sensors*, vol. 23, no. 20, p. 8651, 2023, https://doi.org/10.3390/s23208651.

[34] S. G. S. Pula, S. A. Kumar, S. Jha, and A. Ramanathan, "Enhanced penalty-based bidirectional reinforcement learning algorithms," *arXiv preprint* arXiv:2504.03163, 2025, https://doi.org/10.48550/arXiv.2504.03163.

[35] Z. Ding, J. Liu, W. Chi, J. Wang, G. Chen, and L. Sun, "PRTIRL based socially adaptive path planning for mobile robots," *International Journal of Social Robotics*, vol. 15, no. 2, pp. 129–142, 2023, https://doi.org/10.1007/s12369-022-00924-8.

[36] W. Zhao, L. Li, Y. Wang, H. Zhan, Y. Fu, and Y. Song, "Research on a global path-planning algorithm for unmanned aerial vehicle swarm in three-dimensional space based on Theta*-artificial potential field method," *Drones*, vol. 8, no. 4, p. 125, 2024, https://doi.org/10.3390/drones8040125.

[37] X. Han and X. Zhang, "Multi-scale Theta* algorithm for the path planning of unmanned surface vehicle," *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, vol. 236, no. 2, pp. 427–435, 2022, https://doi.org/10.1177/14750902211039650.

[38] M.-S. Yuan, T.-L. Zhou, and M. Chen, "Improved lazy Theta* algorithm based on octree map for path planning of UAV," *Defence Technology*, vol. 23, pp. 8–18, 2023, https://doi.org/10.1016/j.dt.2022.01.006.

[39] L. Han, L. He, X. Sun, Z. Li, and Y. Zhang, "An enhanced adaptive 3D path planning algorithm for mobile robots with obstacle buffering and improved Theta* using minimum snap trajectory smoothing," *Journal of King Saud University–Computer and Information Sciences*, vol. 35, no. 10, p. 101844, 2023, https://doi.org/10.1016/j.jksuci.2023.101844.

[40] Y. Zhang, Y. Hu, J. Lu, and Z. Shi, "Research on path planning of mobile robot based on improved Theta* algorithm," *Algorithms*, vol. 15, no. 12, p. 477, 2022, https://doi.org/10.3390/a15120477.

[41] B. Tang, K. Hirota, X. Wu, Y. Dai, and Z. Jia, "Path planning based on improved hybrid A* algorithm," *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 25, no. 1, pp. 64–72, 2021, https://doi.org/10.20965/jaciii.2021.p0064.

[42] X. Liu, Z. Degan, T. Zhang, C. Yuya, L. Chen, and S. Liu, "Novel best path selection approach based on hybrid improved A* algorithm and reinforcement learning," *Applied Intelligence*, vol. 51, no. 12, pp. 9015–9029, 2021, https://doi.org/10.1007/s10489-021-02303-8.

[43] D. Li, W. Yin, W. E. Wong, M. Jian, and M. Chau, "Quality-oriented hybrid path planning based on A* and Q-learning for unmanned aerial vehicle," *IEEE Access*, vol. 10, pp. 7664–7674, 2021, https://doi.org/10.1109/ACCESS.2021.3139534.

[44] X. Tong, S. Yu, G. Liu, X. Niu, C. Xia, J. Chen, Z. Yang, and Y. Sun, "A hybrid formation path planning based on A* and multi-target improved artificial potential field algorithm in the 2D random environments," *Advanced Engineering Informatics*, vol. 54, p. 101755, 2022, https://doi.org/10.1016/j.aei.2022.101755.

[45] A. Vagale, R. T. Bye, R. Oucheikh, O. L. Osen, and T. I. Fossen, "Path planning and collision avoidance for autonomous surface vehicles II: a comparative study of algorithms," *Journal of Marine Science and Technology*, vol. 26, no. 4, pp. 1307–1323, 2021, https://doi.org/10.1007/s00773-020-00790-x.

[46] Y. Ding, W. Zhuang, L. Wang, J. Liu, L. Guvenc, and Z. Li, "Safe and optimal lane-change path planning for automated driving," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 235, no. 4, pp. 1070–1083, 2021, https://doi.org/10.1177/0954407020913735.

[47] T. Zhao, H. Li, and S. Dian, "Multi-robot path planning based on improved artificial potential field and fuzzy inference system," *Journal of Intelligent & Fuzzy Systems*, vol. 39, no. 5, pp. 7621–7637, 2020, https://doi.org/10.3233/JIFS-200869.

[48] C. Kim, Y. Kim, and H. Yi, "Fuzzy analytic hierarchy process-based mobile robot path planning," *Electronics*, vol. 9, no. 2, p. 290, 2020, https://doi.org/10.3390/electronics9020290.

[49] L. Wellhausen, R. Ranftl, and M. Hutter, "Safe robot navigation via multi-modal anomaly detection," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1326–1333, 2020, https://doi.org/10.1109/LRA.2020.2967706.

[50] Y. Gao and S. Li, "Obstacle avoidance path planning for UAV applied to photovoltaic stations based on improved dynamic window method," *Electronics*, vol. 14, no. 10, p. 1963, 2025, https://doi.org/10.3390/electronics14101963.

[51] Q. Xing, S. Xu, H. Wang, J. Wang, W. Zhao, and H. Xu, "Path planning of a mobile robot using an improved mixed-method of potential field and wall following," *International Journal of Advanced Robotic Systems*, vol. 20, no. 3, 2023, https://doi.org/10.1177/17298806231169186.

[52] M. M. J. Samodro, R. D. Puriyanto, and W. Caesarendra, "Artificial potential field path planning algorithm in differential drive mobile robot platform for dynamic environment," *International Journal of Robotics and Control Systems*, vol. 3, no. 2, p. 10, 2023, https://doi.org/10.31763/ijrcs.v3i2.944.

[53] S. T. Shahid, S. M. A. Siddique, and M. M. Alam, "UAV survey coverage path planning of complex regions containing exclusion zones," *arXiv preprint arXiv:2411.07053*, Nov. 2024, https://arxiv.org/abs/2411.07053.

[54] M. K. Ouach, T. Eren, and E. Özcan, "PRM path smoothening by circular arc fillet method for mobile robot navigation," *International Journal of Engineering Research and Development*, vol. 16, no. 1, pp. 1–19, 2024, https://doi.org/10.29137/umagd.1278980.

[55] Z. Xu, Y. Xiu, X. Zhan, B. Chen, and K. Shimada, "Vision-aided UAV navigation and dynamic obstacle avoidance using gradient-based B-spline trajectory optimization," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2023, pp. 1214–1220, https://doi.org/10.1109/ICRA48891.2023.10160638.

[56] Y. Zhang, P. Wang, K. Cui, H. Zhou, J. Yang, and X. Kong, "An obstacle avoidance path planning and evaluation method for intelligent vehicles based on the B-spline algorithm," *Sensors*, vol. 23, no. 19, p. 8151, 2023, https://doi.org/10.3390/s23198151.

[57] R. Sukwadi, G. Airlangga, W. W. Basuki, Y. Kristian, R. Rahmananta, L. F. Sugianto, and O. I. A. Nugroho, "Comparative analysis of path planning algorithms for multi-UAV systems in dynamic and cluttered environments: A focus on efficiency, smoothness, and collision avoidance," *International Journal of Robotics and Control Systems*, vol. 4, no. 4, 2024, https://doi.org/10.31763/ijrcs.v4i4.1555.

[58] D. Xiang, H. Lin, J. Ouyang, and D. Huang, "Combined improved A* and greedy algorithm for path planning of multi-objective mobile robot," *Scientific Reports*, vol. 12, no. 1, p. 13273, 2022, https://doi.org/10.1038/s41598-022-17684-0.

[59] X. Bai, H. Jiang, J. Cui, K. Lu, P. Chen, and M. Zhang, "UAV path planning based on improved A* and DWA algorithms," *International Journal of Aerospace Engineering*, vol. 2021, no. 1, p. 4511252, 2021, https://doi.org/10.1155/2021/4511252.

[60] B. Guo, Z. Kuang, J. Guan, M. Hu, L. Rao, and X. Sun, "An improved A-star algorithm for complete coverage path planning of unmanned ships," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 36, no. 3, p. 2259009, 2022, https://doi.org/10.1142/S0218001422590091.