A UGV Path Planning Algorithm Based on Improved A* with Improved Artificial Potential Field

Xianchen Meng and Xi Fang *

School of Science, Wuhan University of Technology, Wuhan 430070, China; 273197@whut.edu.cn
* Correspondence: fangxi@whut.edu.cn; Tel.: +86-135-5401-6042

Abstract: Aiming at the problem of difficult obstacle avoidance for unmanned ground vehicles (UGVs) in complex dynamic environments, an improved A*-APF algorithm (BA*-MAPF algorithm) is proposed in this paper. Addressing the A* algorithm’s challenges of lengthy paths, excess nodes, and lack of smoothness, the BA*-MAPF algorithm integrates a bidirectional search strategy, applies interpolation to remove redundant nodes, and uses cubic B-spline curves for path smoothing. To rectify the traditional APF algorithm’s issues with local optimization and ineffective dynamic obstacle avoidance, the BA*-MAPF algorithm revises the gravitational field function by incorporating a distance factor, and fine-tunes the repulsive field function to vary with distance. This adjustment ensures a reduction in gravitational force as distance increases and moderates the repulsive force near obstacles, facilitating more effective local path planning and dynamic obstacle navigation. Through our experimental analysis, the BA*-MAPF algorithm has been validated to significantly outperform existing methods in achieving optimal path planning and dynamic obstacle avoidance, thereby markedly boosting path planning efficiency in varied scenarios.

Keywords: Unmanned Ground Vehicle (UGV); A* algorithm; Artificial Potential Field algorithm (APF); BA*-MAPF; path planning; dynamic obstacle avoidance

1. Introduction

In contemporary society, as productivity continues to develop at a rapid pace, people continue to create more and more tools to generate more value for society more conveniently [1]. With the rise in the standard of living, the number of automobiles has also increased, contributing to economic development but also leading to serious safety issues and social challenges. Notably, about one-fifth of the annual death data worldwide is attributed to traffic accidents [2], underscoring the urgency of finding more reliable ways to improve or replace human driving behavior. This quest has made the development of Unmanned Ground Vehicles (UGVs) a research hotspot in the automotive field, especially with the rapid advancement of artificial intelligence technology, crucial for reducing road traffic accidents [3]. UGV technology, evolving within this context, has become a significant focus for both traditional automakers and emerging vehicle builders.

The core functional modules of a UGV include behavioral decision-making, environment sensing, path planning, and tracking control [4]. These modules work in tandem to enable UGVs to provide safe, reliable, and efficient driving behavior in complex environments. The application of UGVs in production and daily life can significantly enhance travel and transportation efficiency, and effectively mitigate the risk of traffic accidents. Path planning technology, relying on the sensing equipment aboard UGVs, plays a critical role in developing a planning scheme from the current location to the target location, based on the collection and analysis of environmental information [5]. With increasing interest in UGVs, the path planning problem has emerged as a challenging task within the field of navigation research [6].

However, heuristic algorithms, as a primary approach to path planning, exhibit significant limitations in addressing the diversity and complexity of practical application.
environments. These traditional heuristic algorithms struggle to meet the requirements of more complex path planning scenarios due to inherent limitations [7–9]. For instance, Shengmin Zhao et al. [7] proposed an innovative full-coverage path planning (CCPP) method, effectively addressing the dynamic tracking problem and solving multi-obstacle challenges in complex environments. This method significantly improves the coverage of the planned path, achieving a high level of 98% through simulation experiments. Similarly, Hub Ali et al. [8] focused on the UAV path planning and obstacle avoidance problems in a 3D terrain, proposing a trajectory planning technique capable of implementing global and local path planning for UAVs under static and dynamic constraints. This method demonstrates superior performance in handling complex terrain situations, significantly outperforming other 3D UAV path planning techniques. Guojun Ma et al. [9] introduced a new path planning algorithm, the Path Planning Probabilistic Smoothing Bidirectional Randomized Tree Algorithm (PSBi-RRT), addressing the issues of the disordered initial solutions and slow convergence found in the Bidirectional Rapid Exploration Randomized Tree (Bi-RRT). The proposed algorithm achieves excellent performance and higher planning efficiency in various environments through several simulation experiments.

In complex dynamic environments, traditional single-path planning algorithms fail to meet multiple requirements, such as path length, time, and smoothness, rendering them ineffective for real-world applications. This has prompted scholars to explore the combination of two or more algorithms to overcome these challenges [10–12]. Wei Zhang et al. [10] investigated an AUV path planning algorithm based on A* + APF (AplusPF), which not only plans paths safely but also ensures the paths are reliable and trackable, significantly improving the algorithm’s efficiency. Vikas et al. [11] explored a hybrid path planning algorithm for humanoid robots in rugged terrain, combining a memory-based GSA model with a differential perturbation velocity method. This approach yields optimal path planning results, demonstrating high efficiency and feasibility in environments with dynamic obstacles. YuMin Su et al. [12] proposed the Constraint Locked Sweep Method with Velocity Obstacle (CLSM-VO) for unmanned vessel operation in complex dynamic maritime environments, capable of handling complex maritime traffic scenarios and generating smooth and safe trajectory planning.

The motivation for this work is to address the specific limitations and gaps in existing path planning methods. Despite significant advancements made by various researchers [7–12], a comprehensive solution that efficiently and effectively tackles the challenges of dynamic and complex environments remains elusive. This paper introduces the BA*-MAPF algorithm, designed to surmount the limitations of existing path planning algorithms by offering a more adaptable, efficient, and reliable solution for UGVs navigating complex environments.

The main contributions of this paper are as follows:

1. A grid environment model has been developed to tackle the complexities of path planning in environments laden with both dynamic and static obstacles. This model serves as a foundational framework for navigating through intricate spaces, providing a systematic approach to identifying optimal paths amidst varying obstacles;
2. Introducing a bidirectional search strategy into the traditional A* algorithm to overcome the inherent limitations of the A* algorithm in global path planning, such as the tendency to produce excessively long paths, the need for extended search times, and the production of paths with less than optimal smoothness. This approach greatly reduces the search time of the algorithm, improves its efficiency, solves the drawbacks of the traditional algorithm, and optimizes the pathfinding process for better performance and better navigation;
3. We modify the potential field function and introduce a distance factor to address the shortcomings of the APF algorithm in terms of target unreachability and local minima. The simulation results show that the improved algorithm overcomes the problems of local minima and target unreachability, enabling the UGV to safely reach the end point during the obstacle avoidance process of path planning;
4. Utilizing the advantages of the A* algorithm in global path planning, as well as its faster search path and simplicity, and APF algorithm for local path planning with high real-time performance and good local obstacle avoidance effect is combined with the improved A* algorithm (BA* algorithm), and the BA*-MAPF algorithm is proposed. The simulation verification shows that the BA*-MAPF algorithm has good results;

5. The proposed algorithm is compared with the GA algorithm, PSO algorithm, GWO algorithm, A* algorithm and APF algorithm in a predefined grid environment model, and the comprehensive performance of the proposed algorithm is illustrated in terms of evaluation metrics such as the path length, the computation time, the number of corners, the cumulative corners, and the smoothness in terms of the stability and robustness.

Section 2 briefly reviews the existing literature on path planning for intelligent systems. Section 3 describes the UGV path planning model. Section 4 describes the construction details of the BA*-MAPF algorithm. Section 5 presents simulation results and provides a comparative analysis of the discussed algorithms, emphasizing the performance metrics of these algorithms. Finally, Section 6 summarizes the main ideas of the paper and new contributions to the field of path planning.

2. Related Work

Currently, in most cases, various heuristic algorithms are still used to solve the path planning problem. Path planning methods for UGV are usually categorized into two main groups: traditional search algorithms and intelligent bionic algorithms.

Zhang et al. [13] introduced a pioneering real-time navigation strategy for unmanned surface vehicles (USVs) to navigate complex environments with multiple obstacles. Leveraging Artificial Potential Field (APF) technology and Differential Vector Field (DVF) guidance principles, their approach focuses on safe, efficient path planning and multi-obstacle avoidance. This innovative combination enhances USV’s ability to track targets effectively while navigating through unpredictably dynamic spaces, representing a significant advancement in autonomous maritime navigation. Josu Agirrebeitia et al. [14] crafted a novel UAV path planning algorithm utilizing the Artificial Potential Field (APF) method, tailored for obstacle-laden environments. This algorithm stands out for its adept obstacle avoidance in intricate settings, showcasing remarkable efficiency in both two-dimensional and three-dimensional spaces. Their contribution signifies a significant leap in UAV navigation technology, offering enhanced maneuverability across diverse spatial challenges. Wolfgang Fink et al. [15] developed a multi-objective extended Dijkstra algorithm based on terrain data to cope with the problem of rover path planning that requires extensive time-consuming manual surface terrain analysis. The algorithm was optimized for multiple constraint conditions, creating favorable conditions for scientific research and investigation. Yong Ma et al. [16] proposed an IBA* algorithm for the coverage path planning of unmanned ground surveying vehicles, which overcomes the problem of insufficient continuity in the BA* algorithm through task decomposition and dynamic mapping updates. Experiments have shown that this algorithm can effectively ensure the safety of UGV during map mapping. Shanen Yu et al. [17] introduced the SOF-RRT* algorithm, designed to boost UAV navigation efficiency in complex areas. Key benefits include shorter, smoother paths, fewer random tree nodes, and fewer iterations needed for planning. This algorithm significantly cuts down iteration numbers, reduces path lengths, and quickens convergence, marking a notable improvement over traditional path planning methods. SOF-RRT* stands out by offering enhanced navigation efficiency and performance enhancements in UAV path planning. Qingyang Gao et al. [18] developed the BP-RRT algorithm, blending back propagation neural networks with an optimized rapidly expanding random tree (RRT) algorithm for robotic path planning. This approach addresses the inefficiency and high computational demands of navigating a six-degree-of-freedom robotic arm through complex 3D environments filled with obstacles. The BP-RRT algorithm strategically partitions the sampling space and adjusts sampling probabilities for more effective local sampling,
particularly in dynamic or obstacle-rich areas. Experimental simulations have shown its superior performance and adaptability across various conditions, outpacing traditional path planning methods and demonstrating marked improvements in speed against comparable RRT* algorithms. Jiaming Fan et al. [19] innovated a UAV trajectory planning method using a dual-track APF-RRT (Artificial Potential Field-Rapid Exploration Randomized Tree) algorithm to address the limitations of traditional RRT algorithms, such as redundancy, multiple iterations, and excessively long paths. Their approach significantly improves the tree-growth search performance and computational efficiency in path planning, while substantially improving the quality of the planned paths. They used an improved bi-directional APF-RRT strategy to increase efficiency and path planning quality, thus optimizing UAV navigation strategies in complex environments. Yanming Liang et al. [20] introduced the CCPF-RRT* algorithm, an innovative extension of the Rapid Exploration Random Tree Star (RRT*) focused on heuristic optimal path planning. This method is designed to boost robotic navigation efficiency in dense and complex settings by employing a potential function-based strategy for path optimization. Key advancements include the algorithm’s ability to guide robots through crowded spaces with optimal paths, enhancing overall efficiency and minimizing the cost of navigation. The CCPF-RRT* has shown remarkable capabilities in swiftly generating initial solutions, achieving rapid convergence, and crafting low-cost motion paths in intricate environments.

Chengzhi Qu et al. [21] developed a novel hybrid algorithm, HSGWO-MSOS, which integrates elements of the gray wolf optimization (GWO) and the symbiotic organisms search (SOS) algorithms. This innovative approach simplifies GWO to maintain its exploratory strength while significantly boosting convergence speed beyond the traditional GWO benchmarks. Additionally, the SOS algorithm’s symbiosis phase has been tailored for enhanced efficiency. The performance of the HSGWO-MSOS algorithm is meticulously evaluated using the linear difference equation method, revealing its exceptional ability to identify optimal paths efficiently. This hybrid algorithm stands out for its rapid execution and superior performance, making it a groundbreaking solution for UAV path planning in challenging environments. Imanshu Gupta et al. [22] proposed a novel hybrid optimization algorithm, known as HCPSOA, where they combined Particle Swarm Optimization (PSO) and Gray Wolf Optimization (COA). The goal of the algorithm is to overcome the difficulty of estimating the flight trajectory of an unmanned aerial vehicle (UAV) under various constraints with acceptable accuracy and time span in a complex 3D environment. It improved convergence speed, reduced the possibility of local minimum value reductions, and overcame the limitations of COA and PSO. The results of the simulation experiments show that HCPSOA estimates better and safer flight paths with fewer iterations, and the ability to search for flight paths has been greatly improved. Yan Ma et al. [23] crafted an advanced hybrid algorithm inspired by the flame ant colony concept, designed specifically for unmanned underwater vehicle (UUV) path planning. This innovative approach aims to address the difficulties UUVs encounter when moving through environments affected by ocean currents and physical obstacles. Through simulation experiments, the algorithm has proven its ability to efficiently determine optimal paths within complex marine settings, significantly improving UUV navigation under challenging underwater conditions. Zheng Wang et al. [24] have developed a novel hybrid algorithm that synergizes global and local path planning techniques, tailored for complex marine navigation. This approach leverages an Enhanced Particle Swarm Optimization (PSO) algorithm to carve out an optimized global route, complemented by an advanced Artificial Potential Field (APF) algorithm for dynamic obstacle avoidance, fine-tuning the path based on global insights. This dual-strategy algorithm adeptly meets the dual challenges of optimizing paths and navigating around dynamic obstacles, significantly improving the operational efficiency of unmanned boats in intricate offshore environments. By integrating these improved algorithms, Zheng Wang et al. offer a sophisticated solution that boosts the navigational capabilities of unmanned marine vessels, facilitating their successful mission execution in demanding marine settings. Zhiheng Yu et al. [25] have developed a unique al-
algorithm that combines the water flow potential field method with the beetle antenna search strategy, aiming to overcome the limitations inherent in the standalone beetle antenna search method, such as local optimization pitfalls, general inefficiency, and inadequate obstacle navigation during path planning. Through rigorous simulation experiments, this integrated approach has proven to significantly enhance path planning capabilities, yielding benefits like shortened planning durations, superior path optimization, and effective circumvention of local optima. This novel algorithm marks a notable advancement in the domain of mobile robot path planning, successfully addressing prevalent challenges by harnessing the synergistic potential of merging two distinct navigational strategies to derive markedly improved outcomes. Lianghan Zeng et al. [26] crafted a path planning methodology designed to anticipate ground threats, employing the Artificial Bee Colony (ABC) algorithm enhanced by a collaborative thinking strategy. This approach focuses on mitigating the adverse effects of incomplete ground threat data on the path planning of Unmanned Aerial Vehicles (UAVs). Through simulation experiments, their method has been validated to secure safe flight paths for UAVs, proving its efficacy in environments where ground threat information is not only incomplete but also subject to dynamic changes. Tiancheng Wang et al. [27] critiqued the conventional Ant Colony Optimization (ACO) algorithms for correlating pheromone concentration with path length, leading to redundancies, and innovatively introduced a Monte Carlo-based Improved Ant Colony Optimization Algorithm (MC-IACO) tailored for welding robot path planning. Through simulation experiments, MC-IACO was shown to significantly outperform both traditional and other enhanced algorithms in optimizing path planning for welding robots, showcasing superior efficiency and precision. This advancement addresses common pitfalls in existing ship path planning algorithms, such as slow convergence, unnecessary detours, and rough path formation, which compromise navigational effectiveness and safety. To address these issues, Qiyong Gu et al. [28] developed the PI-DP-RRT method, a novel approach aimed at enhancing ship path planning. This method promises significant improvements by streamlining the path planning process, notably increasing both efficiency and accuracy, reducing the frequency of turns, and thereby elevating the overall safety and efficiency of ship navigation. Through simulation experiments, the PI-DP-RRT method has demonstrated its ability to effectively resolve the shortcomings of existing algorithms. Xiaohong Li et al. [29] crafted an advanced compression factor particle swarm optimization algorithm tailored for Autonomous Underwater Vehicles (AUVs), designed to navigate through difficult currents and obstacles, producing safe and smooth three-dimensional paths. Their experimental findings underscore substantial enhancements in both the efficiency and quality of path planning, marking a pivotal advancement in AUV navigation by significantly optimizing for safety, smoothness, and expedited planning capabilities. Chao Liu et al. [30] observed that the traditional ACO algorithm suffers from problems such as inefficient search and easy stagnation. Therefore, they proposed a new variant of ACO called Improved Heuristic Mechanism ACO (IHMACO). Simulation experiments show that IHMACO is highly practical and efficient in practical applications, and can obtain the optimal solution of the corresponding path planning problem. This method provides an effective improvement program for path planning.

Researchers have made notable advancements in traditional search and intelligent bionic algorithms for UGV path planning, significantly boosting their efficiency across diverse, complex scenarios. However, despite these improvements, certain algorithms still exhibit drawbacks, such as failing to secure optimal or near-optimal paths in dynamic environments, generating paths that lack smoothness, and featuring excessive rotation angles. These shortcomings underscore the challenges in adapting these algorithms for practical, real-world applications where the demand for efficiency and navigational precision is paramount.
3. UGV Path Planning Modeling

Before a UGV performs path planning, the initial task is processing the surrounding environmental information. Indeed, there is a significant correlation between the accuracy of environmental information and the speed and efficiency of UGV path planning. Therefore, meticulously processing this environmental information is crucial.

The grid method stands out as one of the most prevalent map modeling techniques in static networks. This approach subdivides the 2D map environment into independent cells using polygons of specific shapes, ensuring each cell connects to others without overlap. These cells are commonly referred to as grids. Within the same map environment, utilizing shorter edges for grid segmentation enhances the accuracy of path planning. However, it concurrently extends the computation time required for path search. For instance, as depicted in Figure 1, the white grid marks the feasible region, whereas the black grid signifies the occupied grid, representing obstacles. A blue six-pointed star marks the start point, and a red five-pointed star indicates the end point, with the UGV free to navigate within the feasible area.

![Figure 1. Raster map.](image1)

In the 3D space, the whole plane is divided by a square grid, the height value of the grid is reset, and the numbers of columns, rows and layers of the grid are determined by combining with the actual terrain data. Figure 2 shows the valleys and peaks in a matrix, and the different values on the matrix indicate the elevation at the coordinates, after which the terrain is smoothed using the interpolation method.

In grid maps, we generally categorize grids into two types: white grids (free passage is allowed) and black grids (free passage is prohibited). There are two general search methods for UGV in grid maps, namely, four-neighbor search and eight-neighbor search, where the black dots indicate the optional nodes of the UGV and the arrows indicate the movable directions of the UGV, as shown in Figure 3. As shown in Figure 3, from the perspective of path search time, four-neighbor search entails half the computational cost of eight-neighbor search, and therefore takes less time. From the perspective of the feasibility and length of the path, the four-neighbor search path has more turning curves than the eight-neighbor search path, and each turning is at a 90 degree right angle, which is not friendly to the actual motion of the UGV. The eight-neighbor search path can be planned through the
diagonal lines of the grid, and the path length is better than that of the four-neighbor search path. Therefore, in this study, the grid map adopts an eight-neighbor search method.

<table>
<thead>
<tr>
<th>X(m)</th>
<th>0</th>
<th>0</th>
<th>28</th>
<th>41</th>
<th>60</th>
<th>92</th>
<th>110</th>
<th>115</th>
<th>116</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y(m)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>30</td>
<td>50</td>
<td>73</td>
<td>43</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>48</td>
<td>36</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>18</td>
<td>62</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td></td>
<td>53</td>
<td>142</td>
<td>109</td>
<td>61</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>88</td>
<td></td>
</tr>
<tr>
<td></td>
<td>84</td>
<td>136</td>
<td>13</td>
<td>75</td>
<td>0</td>
<td>32</td>
<td>91</td>
<td>31</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>83</td>
<td>134</td>
<td>41</td>
<td>0</td>
<td>39</td>
<td>139</td>
<td>128</td>
<td>73</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>94</td>
<td>149</td>
<td>90</td>
<td>0</td>
<td>0</td>
<td>73</td>
<td>154</td>
<td>73</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>124</td>
<td>170</td>
<td>85</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>50</td>
<td>121</td>
<td>93</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>151</td>
<td>175</td>
<td>86</td>
<td>13</td>
<td>0</td>
<td>19</td>
<td>70</td>
<td>124</td>
<td>160</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>199</td>
<td>154</td>
<td>68</td>
<td>0</td>
<td>0</td>
<td>167</td>
<td>148</td>
<td>183</td>
<td>158</td>
<td>0</td>
</tr>
</tbody>
</table>

**Figure 2.** 3D digital terrain coding matrix.

![Diagram](image-url)

**Figure 3.** A* algorithm for searching neighborhoods. (a) Four neighbors per search. (b) Eight neighbors per search.

The operational dynamics of UGV hold significant importance in the development of an effective path for obstacle avoidance during the path planning process. To facilitate the simulation of the UGV motion, a coordinate system designated as OXY is established, with its operational mechanics illustrated in Figure 4. Within this framework, UGVs are conceptualized as particles executing motion in a stepwise manner, their movement confined to adjacent units within the environmental layout showcased in Figures 1 and 2. The strategy for navigating through this environment incorporates the eight-neighborhood search method, granting the UGV a spectrum of eight possible directions for movement, namely, up, down, left, right, and the four diagonals (up-left, down-left, up-right, down-right), as meticulously outlined in Figure 3. This methodological approach to UGV path planning...
underscores the pivotal role of the vehicle’s motion state in circumventing obstacles, employing a systematic progression within a meticulously defined coordinate system, and capitalizing on the flexibility afforded by an eight-directional movement capacity.

Figure 4. Schematic diagram of UGV motion.

4. Design of the BA*-MAPF Algorithm


4.1.1. Traditional A* Algorithm

The A* algorithm represents a sophisticated evolution of Dijkstra’s algorithm and the BFS (breadth-first search) algorithm, ingeniously amalgamating their best attributes to efficiently ascertain the shortest path, although it is traditionally not as efficient as its individual predecessors. Unlike the BFS algorithm, which adopts a greedy search strategy that favors nodes nearer to the goal purely based on distance—thereby hastening the search but at the risk of settling for less-than-ideal paths—the A* algorithm introduces a nuanced cost function for evaluating nodes, thereby refining the path selection process. The hallmark of the A* algorithm lies in its heuristic function, a strategic mechanism that steers the search expansion directly towards the target. This not only amplifies the search efficiency but also guarantees the discovery of the shortest possible path. In essence, the A* algorithm synthesizes the core strengths of both Dijkstra’s and BFS algorithms, leveraging a cost function alongside a heuristic search strategy to strike a harmonious balance between search efficiency and the attainment of optimal pathfinding solutions. The cost function is shown below:

\[ f(n) = g(n) + h(n) \]  \hspace{1cm} (1)

where \( n \) is the current point, \( f(n) \) is the total generation value of the current point \( n \), \( g(n) \) denotes the actual generation value from the initial point to this point in position \( n \), and \( h(n) \) denotes the generation value from the current point \( n \) to the end point.

The cost function calculates the distance between two positions, \( h(n) \) is the specific embodiment of the heuristic information in the cost function, and so far the usual methods for calculating the heuristic are Manhattan distance, Chebyshev distance, Euclidean distance and so on. If the node is located at location \( (x_n, y_n) \) and the target point is located at location \( (x_{goal}, y_{goal}) \), then the Euclidean distance can be calculated using the following formula:

\[ h(n) = \sqrt{(x_{goal} - x_n)^2} + \sqrt{(y_{goal} - y_n)^2} \]  \hspace{1cm} (2)
The Manhattan distance function between \((x_n, y_n)\) and \((x_{goal}, y_{goal})\) can be expressed as follows:

\[
h(n) = |x_{goal} - x_n| + |y_{goal} - y_n|
\]  

(3)

The Chebyshev distance function between \((x_n, y_n)\) and \((x_{goal}, y_{goal})\) can be expressed as follows:

\[
h(n) = \max\{ |x_{goal} - x_n|, |y_{goal} - y_n| \}
\]  

(4)

In this study, we use the Euclidean distance function as a heuristic function to estimate the straight line distance between two coordinate points. Despite its effectiveness, the A* algorithm shows several limitations, particularly in environments dense with obstacles or in scenarios requiring long-distance pathfinding. In such cases, the algorithm’s performance can be significantly diminished due to an increased number of nodes to be searched, leading to longer search times and reduced efficiency. These limitations manifest as follows:

1. Increased search time in obstacle-dense environments—When navigating through areas with numerous obstacles, the A* algorithm may require additional time to circumnavigate these barriers, resulting in an extended search process;
2. Rapid growth in search nodes for longer paths—For extensive search paths, the exponential increase in potential nodes escalates the computational demand, prolonging the search duration and impacting the algorithm’s practical applicability.

To address these challenges, this paper introduces a bidirectional search strategy aimed at refining the A* algorithm. This innovative approach is designed to mitigate the identified limitations by enhancing the algorithm’s ability to efficiently manage scenarios characterized by numerous obstacles. The bidirectional search strategy not only accelerates the search process by concurrently expanding from the start and end points, but also reduces the search space, thereby diminishing the computational load and search time. This adjustment is particularly effective in complex environments, enabling the algorithm to maintain high levels of efficiency and accuracy in pathfinding tasks.

4.1.2. Design of the Bidirectional Alternating Search Strategy

The conventional A* algorithm, while effective in its primary objective, often falls prey to the inefficiency of extensively searching unnecessary nodes during the pathfinding process, thereby augmenting the time required to ascertain the optimal path. To mitigate this inefficiency, an innovative bidirectional search strategy has been adopted. This methodology entails conducting searches in both forward and reverse directions concurrently, with the forward search progressing from the starting point towards the target, and the reverse search moving from the target back to the starting point. A key feature of this refined approach is the utilization of dynamic target points within both the forward and reverse searches. Specifically, the current node in one search operation is designated as the temporary target for the opposing search, creating a dynamic and responsive search environment. This strategic adjustment not only bolsters computational efficiency by minimizing the exploration of non-essential nodes, but also substantially lowers the overall cost associated with the path planning task. The corresponding forward search and reverse search evaluation functions are as follows:

\[
f_1(n) = g_1(n) + h_1(n)
\]  

(5)

\[
f_2(n) = g_2(n) + h_2(n)
\]  

(6)

When the algorithm runs, the path endpoints \(N\) and \(G\) are used as the initial points for forward and backward searching. During forward routing, from point \(N\) to point \(G\), points \(N\) and \(G\) are used as the starting and ending points for forward routing, and the optimal node \(N_1\) is generated. At this point, the algorithm switches from forward path finding to reverse path finding, from point \(G\) to point \(N\). At this point, point \(G\) and point \(N_1\)
generated by the forward path finding alone are used as the starting and ending points for
the reverse search, and the optimal $G_1$ is generated again. After several repeated searches,
the optimal nodes generated in the forward and reverse directions will overlap, and the line
between this point and the starting and ending points is the optimal path. The flowchart of
the bidirectional search strategy of the A* algorithm is shown in Figure 5.

The flowchart of the bidirectional search strategy of the A* algorithm is shown in Figure 5.

As shown in Figure 5, the target point is changed every time during forward and reverse searching, which can cause the forward and reverse search paths to approach quickly. The synchronized bidirectional search ends when the extended node extendNode1 of the current node in the forward search encounters the current node currentNode2 of the current node in the reverse search, or the extended node extendNode2 of the current node in the reverse search encounters the current node currentNode1 of the current node in the forward search. The synchronized bidirectional search method greatly reduces the search time and improves the search efficiency during the search process, but it still has some drawbacks. For example, when one of the forward or reverse searches deviates from the optimal path, the other one will immediately deviate from it too.

4.2. Modified Potential Field Function of APF

4.2.1. Traditional APF Algorithm

Figure 6 shows the force analysis when the UGV enters the range of action of the
obstacle, where $F_{rep}$ indicates the obstacle repulsive force on the UGV, $F_{att}$ indicates the
gravitational force on the end point on the UGV, and $F_{total}$ indicates the combined force on
the UGV, that is, the direction in which the UGV is actually moving.
The Artificial Potential Field (APF) principle introduces a nuanced conceptual framework for navigating UGV, predicated on the interaction between repulsive forces emanating from obstacles and attractive forces directed towards the destination. This dual-force mechanism plays a pivotal role in steering the UGV away from potential hazards while simultaneously guiding it towards its intended goal. Potential fields are categorized into gravitational and repulsive potential fields. Gravitational potential energy comes from the allure of the destination, an attraction towards the goal, while repulsive potential energy comes from the distance and configuration of the obstacles, and is used to steer the UGV away from potential hazards. The essence of APF-based navigation lies in the delicate balance between these opposing forces. This balance ensures that the UGV is simultaneously attracted to the destination and repelled by nearby obstacles, thus bypassing them to reach the intended goal. The calculation formula is as follows:

\[
U_{\text{attr}}(q) = \frac{1}{{2}} \xi \rho^2(q, q_{\text{goal}})
\]  

(7)

where \( \xi \) is the scale factor and \( \rho(q, q_{\text{goal}}) \) denotes the difference between the current position of the UGV and the end point.

The gravitational potential field that envelops a UGV plays a crucial role in its navigation towards an endpoint, exhibiting a dynamic intensity that is inversely proportional to the UGV’s proximity to its destination. This field is most potent when the UGV is at its greatest distance from the endpoint, exerting a strong guiding force that propels the vehicle forward. As the UGV advances closer to its target, the gravitational potential field begins to wane, progressively reducing its influence on the vehicle’s trajectory. This diminution continues until the UGV reaches the endpoint, at which juncture the gravitational potential field dissipates entirely, signifying the cessation of its navigational pull.

The guidance system for UGV utilizes the negative gradient of the gravitational potential energy field to navigate. This mathematical calculation is pivotal in ascertaining both the magnitude and orientation of the gravitational force acting upon the UGV, guiding it along the most efficient trajectory towards its endpoint. As the vehicle progresses closer to its destination, the intensity of this gravitational force diminishes. This gradual reduction in force serves as an indicator of the diminishing need for directional guidance, ensuring that the UGV’s path to the endpoint becomes increasingly straightforward as it nears its goal, optimizing the navigation process by aligning with the path of least resistance. Thus, the gravitational force \( F_{\text{attr}} \) can be expressed as:

\[
F_{\text{attr}}(q) = -\nabla U_{\text{attr}}(q) = -\xi \rho(q, q_{\text{goal}})
\]  

(8)

Figure 6. Force diagram of a UGV in an artificial potential field.

As shown in Figure 5, the target point is changed every time during forward and reverse search encounters the current node of the current node in the search. The synchronized bidirectional search ends when the extended node extends the current node in the forward search. The synchronized bidirectional search method greatly reduces the search time.
We construct the repulsive potential field function based on the distance between the UGV and the obstacle, and the direction of the repulsive force vector obtained according to Equation (8) points to the UGV. The repulsive potential field $U_{rep}$ can be expressed as:

$$U_{rep}(q) = \begin{cases} \frac{1}{2} \eta \left( \frac{1}{\rho(q, q_{obs})} - \frac{1}{\rho_0} \right)^2, & \text{if } \rho(q, q_{obs}) \leq \rho_0 \\ 0, & \text{if } \rho(q, q_{obs}) \geq \rho_0 \end{cases}$$

(9)

where $\eta$ is the repulsion factor, $\rho(q, q_{obs})$ denotes the distance between the UGV and the obstacle, and $\rho_0$ denotes the radius of influence of the obstacle.

According to Equation (12), we can conclude the following: As a UGV navigates closer to an obstacle, it experiences a repulsive force emanating from the obstacle’s repulsive potential field. This force intensifies the nearer the UGV is to the obstacle, inversely diminishing as the distance between them increases. In areas devoid of obstacles, the UGV encounters no such repulsive forces. This interaction underscores the dynamic relationship between the UGV’s proximity to obstacles and the intensity of the repulsive force, highlighting a mechanism designed to prevent collisions by naturally guiding the UGV away from potential impediments in its path.

Similarly, the exact value of the repulsive force $F_{rep}$ is obtained by solving for the negative gradient of the repulsive potential field $U_{rep}$, whose repulsive force can be calculated as:

$$F_{rep}(q) = -\nabla U_{rep}(q)$$

$$= \begin{cases} K_{rep} \left( \frac{1}{\rho(q, q_{obs})} - \frac{1}{\rho_0} \right) \frac{q - q_{obs}}{||q - q_{obs}||}, & \text{if } \rho(q, q_{obs}) \geq \rho_0 \\ 0, & \text{if } \rho(q, q_{obs}) < \rho_0 \end{cases}$$

(10)

where $K_{rep}$ is the repulsive force scale factor, which acts in the same way as $\eta$ in the above equation.

The initial stance of a UGV within the navigational space is marked by a pronounced potential energy, a direct consequence of its substantial separation from the intended target. This scenario is further complicated by the presence of obstacles, which introduce a repulsive potential field around them, incrementally elevating the potential energy in their immediate surroundings. The UGV’s trajectory is subsequently molded by the intricate interplay between various potential fields—namely, the gravitational (attractive) force emanating from the endpoint and the repulsive forces generated by obstacles. The convergence of these forces creates a composite potential field, acting as a navigational beacon for the UGV. To achieve successful navigation, the UGV embarks on a journey from regions characterized by elevated potential energy—its starting position and the vicinity of obstacles—to zones where the potential energy is markedly lower. This strategic maneuvering enables the UGV to adeptly sidestep obstacles, thereby mitigating collision risks and facilitating a smoother progression towards its endpoint. The movement pattern of the UGV, governed by the gradient of potential energy across the navigational field, illustrates a purposeful transition from high-energy states to more energetically favorable locales, encapsulating the vehicle’s adeptness in negotiating complex terrains by leveraging the differential in potential energy to guide its path towards the destination. The combined potential field and the combined force are represented as follows:

$$U = U_{att} + U_{rep}$$

(11)

$$F = F_{att} + F_{rep}$$

(12)

4.2.2. Modified Potential Field Function

The Artificial Potential Field (APF) algorithm, celebrated for its high real-time performance and path-smoothing properties, is particularly adept at dynamic obstacle avoidance in localized path planning. It conceptualizes obstacles and targets within the environment as generating repulsive and attractive forces, respectively, forming a potential field through
which the path is planned. This method enables a dynamic feedback mechanism with the surrounding environment, showcasing significant adaptability. However, despite these strengths, the APF algorithm encounters several critical limitations that can severely impact its effectiveness, particularly when integrated with the A* algorithm for comprehensive path planning solutions. These limitations primarily include:

1. The local minimum problem—This issue arises when the UGV is influenced by equal and opposite forces, resulting in a net force of zero and causing the vehicle to become stuck in a local minimum without reaching the goal. This situation is depicted in Figure 7a, where the gravitational force towards the goal is counterbalanced by repulsive forces from nearby obstacles, leading to potential oscillations and stalling in the path planning process;

2. The target point unreachability problem—As illustrated in Figure 7b, when obstacles surround the target point, the UGV experiences a reduction in gravitational force as it approaches the goal, while facing increasing repulsive forces. This imbalance can prevent the UGV from completing its path to the target, culminating in the failure of the path planning task.

To address the challenges of target inaccessibility and local optimization problems, the gravitational potential energy field function has been improved to ensure a more efficient path for the UGV. This improvement allows the gravitational force acting on the UGV to weaken as the distance between the UGV and the endpoint expands, and the rate of gravitational weakening is proportional to the distance between the UGV and the endpoint. This adjustment ensures that the navigation mechanism is more nuanced
and responsive, resulting in smoother transitions and more accurate target homing. The improved gravitational potential field and gravitational force equation are shown below:

\[
U_{att}(q) = \begin{cases} 
\frac{1}{2}K_{att}(q - q_{goal})^2, & \text{if } \rho(q, q_{goal}) \leq d \\
\frac{1}{2}K_{att}d^2 - \frac{1}{2}K_{att}d^2, & \text{if } \rho(q, q_{goal}) > d 
\end{cases}
\]

\[(13)\]

\[
F_{att}(q) = -\nabla U_{att}(q) = \begin{cases} 
K_{att}(q - q_{goal}), & \text{if } \rho(q, q_{goal}) \leq d \\
-dK_{att}\frac{q_{goal} - q}{\|q_{goal} - q\|}, & \text{if } \rho(q, q_{goal}) > d 
\end{cases}
\]

\[(14)\]

where \(K_{att}\) is the gravitational scale factor, \(q\) is the position of the UGV, \(q_{goal}\) is the target point position, and \(d\) is the distance factor.

As a UGV navigates towards its endpoint, encountering an obstacle in close proximity triggers a notable decrease in gravitational force and a sharp increase in repulsive force. This significant repulsive force can hinder the UGV from successfully reaching its destination. To mitigate this issue, the repulsion function has been optimized to ensure that while the repulsive force still intensifies as the UGV approaches the endpoint, it does so at a more controlled rate. This crucial adjustment effectively tempers the overwhelming repulsive force experienced near the endpoint, enabling a more balanced and feasible approach for the UGV to navigate around obstacles and reach its intended goal. The following is a detailed description of the repulsive potential field and the associated improvements made to the repulsive force:

\[
U_{rep}(q) = \begin{cases} 
\frac{1}{2}\eta\left(\frac{1}{\rho(q, q_{obs})} - \frac{1}{\rho_0}\right)^2 \rho(q, q_{obs}), & \text{if } \rho(q, q_{obs}) \leq \rho_0 \\
0, & \text{if } \rho(q, q_{obs}) \geq \rho_0 
\end{cases}
\]

\[(15)\]

\[
F_{rep}(q) = -\nabla U_{rep}(q) = \begin{cases} 
\eta\left(\frac{1}{\rho(q, q_{obs})} - \frac{1}{\rho_0}\right)^2 \rho(q, q_{goal}) - \rho(q, q_{obs}), & \text{if } \rho(q, q_{obs}) \leq \rho_0 \\
0, & \text{if } \rho(q, q_{obs}) > \rho_0 
\end{cases}
\]

\[(16)\]

\[
a_1 = \nabla \rho(q, q_{obs}) = \frac{q - q_{obs}}{\|q - q_{obs}\|}
\]

\[(17)\]

\[
a_2 = -\nabla \rho(q, q_{goal}) = -\frac{q - q_{goal}}{\|q - q_{goal}\|}
\]

\[(18)\]

where \(i\) is a positive number, \(q_{goal}\) is the position of the target point, and \(q_{obs}\) is the position of the obstacle.

As shown in Figure 8, we evaluated the MAPF algorithm and the traditional APF algorithm in a comparative study of path planning simulations using a 20 × 20 grid map, where the blue hexagons represent the starting node locations, the black grids represent the obstacles, the red pentagrams represent the end locations, and the red lines represent the final paths generated by the algorithm. We observed that the traditional APF algorithm frequently led to oscillation around obstacles and could trap the UGV in a local minimum, inhibiting its progress towards the intended target, as depicted in Figure 8a. Conversely, our MAPF algorithm, which features a modified potential field function and a distance factor, was able to plot a reasonable path within the same simulated environment. We found that the MAPF algorithm effectively addresses and resolves the issues associated with target unreachability and local minima that were prevalent in the traditional APF method. It significantly improves path smoothness and increases the robustness of the original algorithm, as illustrated in Figure 8b, thereby enabling safer and more effective UGV path planning.
The flow chart of the MAPF algorithm is depicted in Figure 9. The steps of the MAPF algorithm are shown below.

1. Initialize the starting parameters of UGV path planning;
2. Calculate the combined force on the UGV by Equations (14) and (16) and determine whether it falls into the local extreme value. If not, the unmanned vehicle moves according to the magnitude and direction of the combined force;
3. The UGV moves to the next position in the direction of the combined force;
4. Judge whether the current position is the target position. If yes, the planning task is completed. Otherwise, repeat the above steps.

4.3. Path Optimization

4.3.1. Floyd Algorithm

Floyd’s algorithm, colloquially termed here as the interpolation method, is ingeniously crafted to ascertain the shortest path connecting any two points within directed graphs [31], showcasing its robustness by accommodating scenarios where edge weights may assume negative values. At the heart of this pathfinding algorithm lies the path matrix—a cornerstone of its theoretical framework. This matrix is pivotal in delineating the algorithm’s strategic approach to path planning, serving as a comprehensive repository that captures the lengths of the shortest paths between all pairs of vertices within the graph. The algorithm iteratively updates the path matrix, refining and consolidating the information on the shortest paths by systematically considering all possible intermediate vertices that might lie on a path between any two vertices. Through this iterative refinement, the algorithm constructs a detailed matrix representation that encapsulates the optimal distances between every pair of nodes, effectively leveraging the matrix as a navigational guide to chart the most efficient course through the graph. The following is the theoretical idea of interpolation method path planning:

1. Solve a graph by building a weight matrix and updating it recursively from the weight matrix to the minimum path distance matrix between any two points;
2. Create a path of subsequent nodes (path matrix), counting the minimum distance between any points;
3. Each value in the weight matrix is updated recursively, and if Equation (19) holds, the element values in the path matrix are modified, as computed by Equation (20).

$$l(x, y) > l(x, z) + l(z, y)$$  \hspace{1cm} (19)
\[ l(x, y) = l(x, z) + l(z, y) \]  \hspace{1cm} (20)

Interpolation is a path planning algorithm that solves for the minimum distance between multiple sources, and it utilizes dynamic programming ideas to solve for the shortest path in a matrix with positive or negative edge weights. By using the interpolation method to optimize the path planned performed by the algorithm, it can effectively remove the points with too many transitions in the path, reduce the redundant transitions, and eliminate the redundant points in the same straight line, which can help reduce the number of points stored in the UGV when it is running, and at the same time, its motion path is effectively smoothed. The schematic diagram of the interpolation method is shown in Figure 10, where black rectangles indicate obstacles and dashed lines indicate optional paths for UGV.

![Flow chart of the MAPF algorithm](image-url)

**Figure 9.** The flow chart of the MAPF algorithm.

Denote the shortest distance from point \( x \) to \( y \) by \( l(x, y) \). From the Figure 10, we can infer that when there is an obstacle between \( x \) and \( y \) that cannot be passed, \( l(x, y) = +\infty \), and the path is \( x \rightarrow y \). Insert points \( a, b \) between points \( x, y \):

If \( l(x, a) + l(a, y) < l(x, y) \), then \( l(x, y) = l(x, a) + l(a, y) \), and the path is \( x \rightarrow a \rightarrow y \).

If \( l(x, b) + l(b, y) < l(x, a) + l(a, y) \), then \( l(x, y) = l(x, b) + l(b, y) \), and the path is \( x \rightarrow b \rightarrow y \).
The optimal path from \( x \) to \( y \) is \( x \rightarrow b \rightarrow y \). In this paper, the node deletion strategy is experimentally verified as shown in Figure 11, where the blue hexagram is the starting node position, the black grid represents the obstacle, the red pentagram represents the end position, and the red line represents the final path generated by the algorithm.

![Figure 10](image-url) Principle of path optimization by the interpolation method.

![Figure 11](image-url) Optimized path given by interpolation method. (a) Traditional A* algorithm path. (b) Optimized path yielded by interpolation method.

Table 1 presents a performance comparison between an interpolation method and the traditional A* algorithm for pathfinding scenarios, under identical obstacle conditions. The interpolation method demonstrates its superiority by producing a path with a shorter overall distance. Moreover, it significantly reduces the number of path inflection points and decreases the cumulative number of path turning angles, indicating a more direct and efficient route. Experimental outcomes strongly affirm the interpolation method’s effectiveness in optimizing pathfinding, showcasing its advantages over the conventional A* algorithm in terms of path length, smoothness, and navigational efficiency.
Table 1. Comparison of path optimization results using the interpolation method.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Path Length (m)</th>
<th>Average Computation Time (s)</th>
<th>Number of Path Turning Points</th>
<th>Cumulative Turning Angle (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A* Optimization</td>
<td>76.72</td>
<td>0.34</td>
<td>22</td>
<td>1260</td>
</tr>
<tr>
<td>Optimization path</td>
<td>73.43</td>
<td>0.96</td>
<td>12</td>
<td>990</td>
</tr>
</tbody>
</table>

4.3.2. Path Smoothing

The application of the interpolation method in path planning significantly minimizes the count of path nodes, yet it introduces challenges such as paths characterized by large turns and rough transitions, which detract from the ideal movement patterns for UGVs. This reduction in nodes highlights the ongoing need for further refinement to achieve smoother UGV navigation. To address this, B-Spline curves are recognized for their capacity to smooth out corners while maintaining the general trajectory of the paths. These curves are effectively employed in the study to enhance path smoothness, offering a tailored solution that bridges the gap between node reduction and the requirement for fluid, efficient UGV movement [32]. In this paper, three B-spline curves are used for path smoothing.

The B-spline curve is defined as:

$$ P(t) = \sum_{i=0}^{n} P_i F_{i,n}(t), \ t \in [0,1] $$

In Equation (21), $P_i$ is the equation for the $i$-th control point, $F_{i,n}(t)$ is the basis function of the $n$-th degree B-spline, and the formula for the $n$-th degree B-spline curve is:

$$ F_{i,n}(t) = \frac{1}{n!} \sum_{j=0}^{n-i} (-1)^j C_{n+1}^j (t + n - i - j) $$

$$ C_{n+1}^j = \frac{(n+1)!}{j!(n+1-j)!} $$

Figure 12 showcases the application of a B-spline curve for path smoothing, with key features highlighted for clarity. A blue hexagram marks the starting node, while obstacles are represented by a black grid. The journey’s end is denoted by a red pentagram, with the algorithm’s final path traced by a red line, illustrating a smooth trajectory from start to finish.

![Figure 12](image-url)

**Figure 12.** The smooth path of the A* algorithm. (a) Before path smoothing. (b) After path smoothing.
As shown in Table 2, the optimization of the path using three B-spline curves reduces the path length, and the path is smoother in cases with the same obstacle. The experimental results verify the effectiveness of the path smoothing method.

Table 2. Comparison of path optimization results after path smoothing.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Path Length (m)</th>
<th>Average Computation Time (s)</th>
<th>Number of Path Turning Points</th>
<th>Cumulative Turning Angle (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A* before path smoothing</td>
<td>73.43</td>
<td>0.96</td>
<td>12</td>
<td>990</td>
</tr>
<tr>
<td>A* after path smoothing</td>
<td>69.59</td>
<td>0.98</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

4.4. BA*-MAPF Algorithm

To enhance the traditional A* algorithm, which is slow and searches excessively, a bidirectional search strategy is introduced. This approach dynamically sets target points in both forward and reverse searches, significantly boosting efficiency. On another front, the APF algorithm is refined to address its propensity for local optimums and difficulty encountered in endpoint approach due to excessive repulsive forces. Adjustments include modifying the gravitational potential energy field function with a distance factor and refining the repulsive force field function for better obstacle management. In comparing global versus local path planning, the A* algorithm shines in global path planning but struggles with dynamic obstacles, while the APF algorithm is adept at local path planning and navigating around moving obstacles, albeit sometimes taking longer routes. To address the problem of difficult obstacle avoidance for UGVs in complex dynamic environments, we propose the BA*-MAPF algorithm, which not only guarantees to find the globally optimal path, but also effectively handles the case of dynamic obstacles. In the process of combining the A* algorithm and the APF algorithm, the following two issues need to be considered: Firstly, we should consider using the A* algorithm for the initial bootstrap planning when the UGV has not entered the radius of action of the obstacle, and secondly, on the basis of this path, we should determine whether there are any reserved nodes of the A* algorithm in the area of action of the obstacle, and continue to carry out the deletion if there are any. The model of the BA*-APF algorithm is shown in Figure 13:

![Figure 13. Model diagram of BA*-APF algorithm.](image-url)
As shown in Figure 13, \((q_1, \ldots, q_i, \ldots, q_j)\) is the path node of the A* algorithm and \(r\) is the radius of obstacle action, where \(q_1, q_3, q_4, q_5\) and \(q_7\) are the reserved nodes for improving A* algorithm.

Assuming that the UGV at time \(t\) is located at point \(q_1\), when the UGV advances to \(q_3\), it enters into the range of action of the APF algorithm. At this time, since points \(q_4\) and \(q_5\) are in the range of action of the potential field, they are discarded, and the next node \(q_7\) is set as the target point. The flow chat of the BA*-MAPF algorithm is shown in Figure 13.

First, the BA* algorithm searches for paths in the global environment and plans the overall path using the acquired environmental information. In the initial stage, the UGV follows the global path planned by the BA* algorithm. At the same time, the UGV detects the surrounding environment in real time using its own configured sensing devices. If a dynamic obstacle is found within the detection range, and when the retained nodes are within the range of the dynamic obstacle, the path nodes that are within the range of the obstacle are considered for deletion, and then obstacle avoidance is carried out using the MAPF algorithm, which models the obstacle as a repulsive potential field, and the end point as a gravitational potential field. Under the action of the artificial potential field, the UGV will search for the path to the end point. When the UGV leaves the range of the obstacle and reaches the next target node, the algorithm returns to the BA* algorithm and finally finds the path with the lowest cost. The flow of the BA*-MAPF algorithm is shown in Figure 14.

---

**Figure 14.** The flow chart of the BA*-MAPF algorithm.
The pseudocode for the BA*-MAPF Algorithm (Algorithm 1) is as follows.

**Algorithm 1** BA*-MAPF algorithm

**Input:** EnvironmentMap, StartPoint, TargetPoint, DynamicObstacles

**Output:** Path

**Process:**
1. Initialize EnvironmentMap, StartPoint, TargetPoint, OpenListForward, OpenListBackward, ClosedListForward, ClosedListBackward
2. Initialize Path as an empty list, DynamicObstacles with properties
3. Path <- AStar (StartPoint, TargetPoint, OpenListForward, ClosedListForward)
4. ReversePath <- AStar (TargetPoint, StartPoint, OpenListBackward, ClosedListBackward)
5. While Path is not complete:
   7. If UGV encounters DynamicObstacles:
      8. ObstacleInfo <- Get information of the encountered DynamicObstacle
      9. LocalPath <- APF (CurrentPosition, ObstacleInfo, TargetPoint)
     10. Path.Integrate (LocalPath)
   11. Else:
      12. UGV moves to next position in Path
     13. If UGV’s CurrentPosition reaches TargetPoint:
      14. Output “Path planning completed” and break loop
     15. If Path is blocked:
      16. Path <- Replan AStar (CurrentPosition, TargetPoint, OpenListForward, ClosedListForward)
     17. Return Path

5. Simulation Experiments and Analysis

In our simulation experiments, we evaluated various algorithms—GA, PSO, GWO, A*, APF, and our newly proposed BA*-MAPF—for use in the static path planning of a UGV. Our initialization of the GA algorithm involved random seeds, roulette wheel selection, and single-point crossover, with the specific parameter settings detailed in Table 3 [33]. The PSO algorithm’s optimization process includes adjustments in inertia weights, individual learning factors, and social learning factors, with these parameters elaborated in Table 4 [34]. Inspired by the social structure of grey wolves, our GWO algorithm seeks global optima under the guidance of \( \alpha \), \( \beta \), and \( \delta \) wolves, as outlined in Table 5 [35]. Additionally, the APF algorithm we implemented calculates gravitational and repulsive forces to ascertain the UGV’s path, with its parameters enumerated in Table 6 [36]. Then, dynamic path planning for UGVs is performed using the APF algorithm and the BA*-MAPF algorithm, and experimental comparisons are made. Finally, the proposed BA*-MAPF algorithm is simulated in urban and 3D environments. The experimental system is configured with Intel(R) Core(TM) i5-7400U CPU@ 2.30 GHz 2.40 GHz, and the experimental environment is Matlab R2019a.

Table 3. The parameter settings of genetic algorithm (GA).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population size</td>
<td>100</td>
</tr>
<tr>
<td>Number of generations</td>
<td>50</td>
</tr>
<tr>
<td>Crossover probability</td>
<td>0.85</td>
</tr>
<tr>
<td>Mutation probability</td>
<td>0.1</td>
</tr>
</tbody>
</table>
Table 4. The parameter settings of the particle swarm optimization algorithm (PSO).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population size</td>
<td>100</td>
</tr>
<tr>
<td>Number of generations</td>
<td>50</td>
</tr>
<tr>
<td>Inertia weight</td>
<td>0.85</td>
</tr>
<tr>
<td>Cognitive coefficient</td>
<td>1</td>
</tr>
<tr>
<td>Social coefficient</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 5. The parameter settings of the grey wolf optimizer algorithm (GWO).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population size</td>
<td>100</td>
</tr>
<tr>
<td>Number of generations</td>
<td>50</td>
</tr>
<tr>
<td>$\alpha$ wolf coefficient</td>
<td>0.5</td>
</tr>
<tr>
<td>$\beta$ wolf influence</td>
<td>0.3</td>
</tr>
<tr>
<td>$\delta$ wolf influence</td>
<td>0.1</td>
</tr>
<tr>
<td>Inertia weight</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Table 6. The parameter settings of the artificial potential field algorithm (APF).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of generations</td>
<td>50</td>
</tr>
<tr>
<td>Attraction coefficient</td>
<td>50</td>
</tr>
<tr>
<td>Repulsion coefficient</td>
<td>0.5</td>
</tr>
<tr>
<td>Influence distance</td>
<td>0.3</td>
</tr>
<tr>
<td>Step size</td>
<td>0.9</td>
</tr>
</tbody>
</table>

5.1. Path Planning in Static Obstacle Environments

We conducted simulation experiments for the BA*-MAPF algorithm and other comparative algorithms, performing 20 trials across three distinct experimental scenarios to ensure the accuracy of our results. These experimental environments were meticulously designed using the raster map method outlined in Section 3. We created three different scenarios in a $20 \times 20$ environment with a blue hexagram marking the starting point, a red pentagram marking the end point, a red line representing the final path generated by the algorithm, and various shapes of obstacles, as shown in Figure 15. This approach allowed us to comprehensively evaluate the performance and reliability of our BA*-MAPF algorithm against other algorithms under controlled, yet varied, conditions.

![Figure 15](image-url)
We used the GA algorithm, PSO algorithm, GWO algorithm, APF algorithm, A* algorithm and BA*-MAPF algorithm for the simulation experiments of UGV path planning. Figure 16 demonstrates the simulation results in the Z-obstacle environment, while Figure 17 presents the results of the simulation experiments in the E-obstacle environment, and finally, Figure 18 shows the results of the simulation experiments in the combined obstacle environment.

Figure 15. Experimental scenario. (a) Z-type obstacle. (b) E-type obstacle. (c) Comprehensive obstacle.

Figure 16. The path planning results of different algorithms on a Z-type obstacle map. (a) GA. (b) PSO. (c) GWO. (d) APF. (e) A*. (f) BA*-MAPF.

The simulation results in the Z-obstacle scenario are shown in Figure 16. Based on the path planning results, the simulation experiments of the GA algorithm, PSO algorithm, GWO algorithm, A* algorithm, and BA*-MAPF algorithm plan the paths from the start to the end. Among these, the GA algorithm and PSO algorithm plan intricate routes, while the GA algorithm, PSO algorithm, GWO algorithm and A* algorithm plan more complicated and twisted routes, and the simulation experiments of the APF algorithm do not plan routes. Among them, the A* algorithm has a considerably reduced path length, planning time, number of turning points and cumulative turning angle compared to the GA algorithm, PSO algorithm and GWO algorithm. And the simulation experiments of the BA*-MAPF algorithm plan shorter path distances and smoother paths than the A* algorithm. The specific experimental results are shown in Table 7.
We meticulously documented the simulation outcomes for the E-type obstacle scenario in Figure 17, where algorithms like GA, PSO, GWO, APF, A* and BA*-MAPF were tested for their ability to navigate from start to finish. While each algorithm succeeded in planning a route, paths generated by some, notably the GA algorithm, exhibited significant complexity, and the APF algorithm encountered a local optimum that resulted in a dead-end. In comparison, the A* algorithm demonstrated superior efficiency, yielding shorter path lengths, reduced planning times, fewer turning points, and lower cumulative turning angles than GA, PSO, and GWO. Remarkably, our BA*-MAPF algorithm outperformed the A* algorithm by producing even shorter and smoother paths, a fact substantiated by the detailed comparative analysis in Table 8.

<table>
<thead>
<tr>
<th>Experimental Scenario</th>
<th>Algorithm</th>
<th>Path Length (m)</th>
<th>Average Computation Time (s)</th>
<th>Number of Path Turning Points</th>
<th>Cumulative Turning Angle (°)</th>
<th>Smoothness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z-type obstacle</td>
<td>GA</td>
<td>109.9</td>
<td>10.67</td>
<td>29</td>
<td>2048</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>PSO</td>
<td>83.1</td>
<td>10.40</td>
<td>21</td>
<td>1262</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>GWO</td>
<td>84.5</td>
<td>10.04</td>
<td>23</td>
<td>1390</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>APF</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>A*</td>
<td>76.7</td>
<td>0.03</td>
<td>22</td>
<td>1260</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>BA*-MAPF</td>
<td>69.6</td>
<td>0.39</td>
<td>0</td>
<td>0</td>
<td>YES</td>
</tr>
</tbody>
</table>

Figure 17. The path planning results of different algorithms on an E-type obstacle map. (a) GA. (b) PSO. (c) GWO. (d) APF. (e) A*. (f) BA*-MAPF.

Table 7. Comparison of the results of the six algorithms with a Z-type obstacle.
Table 8. Comparison of the results of the six algorithms for an E-type obstacle.

<table>
<thead>
<tr>
<th>Experimental Scenario</th>
<th>Algorithm</th>
<th>Path Length (m)</th>
<th>Average Computation Time (s)</th>
<th>Number of Path Turning Points</th>
<th>Cumulative Turning Angle (°)</th>
<th>Smoothness</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-type obstacle</td>
<td>GA</td>
<td>63.4</td>
<td>10.55</td>
<td>16</td>
<td>1310</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>PSO</td>
<td>40.1</td>
<td>10.41</td>
<td>7</td>
<td>430</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>GWO</td>
<td>39.6</td>
<td>10.46</td>
<td>7</td>
<td>461</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>APF</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td>A*</td>
<td>36.5</td>
<td>0.02</td>
<td>5</td>
<td>270</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>BA*-MAPF</td>
<td>34.0</td>
<td>0.42</td>
<td>0</td>
<td>0</td>
<td>YES</td>
</tr>
</tbody>
</table>

We detail the simulation outcomes for the integrated obstacle scenario in Figure 18, examining the performances of GA, PSO, GWO, APF, A*, and BA*-MAPF algorithms in path planning from start to finish across winding routes. The path plotted by the GA algorithm emerged as particularly intricate. However, the paths generated by the A* algorithm showcase clear superiority in terms of path length, planning time, number of turning points, and cumulative turning angles compared to those by GA, PSO, and GWO. Notably, our BA*-MAPF algorithm excelled further, crafting paths that were not only shorter but also smoother than those produced by the A* algorithm, a superiority comprehensively documented in Table 9. This underscores our BA*-MAPF algorithm’s enhanced efficiency and effectiveness in navigating complex environments.
Table 9. Comparison of the results of the six algorithms for comprehensive obstacle.

<table>
<thead>
<tr>
<th>Experimental Scenario</th>
<th>Algorithm</th>
<th>Path Length (m)</th>
<th>Average Computation Time (s)</th>
<th>Number of Path Turning Points</th>
<th>Cumulative Turning Angle (°)</th>
<th>Smoothness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comprehensive obstacle</td>
<td>GA</td>
<td>54.9</td>
<td>9.37</td>
<td>22</td>
<td>1631</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>PSO</td>
<td>38.7</td>
<td>9.58</td>
<td>11</td>
<td>657</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>GWO</td>
<td>39.5</td>
<td>9.67</td>
<td>11</td>
<td>700</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>APF</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
</tr>
<tr>
<td></td>
<td>A*</td>
<td>32.7</td>
<td>0.02</td>
<td>8</td>
<td>405</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>BA*-MAPF</td>
<td>30.7</td>
<td>0.41</td>
<td>0</td>
<td>0</td>
<td>YES</td>
</tr>
</tbody>
</table>

5.2. Algorithm Performance Comparison

We compared and analyzed experimental results across various scenarios, meticulously documenting our findings. The average path length is depicted in Figure 19a, providing a clear visual representation of the efficiency of different path planning algorithms. Similarly, average computation time, a critical factor in evaluating algorithm performance, is illustrated in Figure 19b. We also quantified the navigational complexity through the average number of inflection points, as shown in Figure 19c, and the average cumulative number of turning angles, presented in Figure 19d. These figures collectively offer a comprehensive overview of our comparative analysis, highlighting the distinctions and efficiencies of each algorithm under consideration.

Figure 19. Comparative analysis of experimental results. (a) Average path length. (b) Average computation time. (c) Average number of path turning points. (d) Average cumulative turning angle.
In our analysis presented through histogram (a) in Figure 19, we observed that the BA*-APF algorithm consistently achieved shorter average path lengths compared to the GA, PSO, GWO, APF, and A* algorithms across three distinct simulation scenarios. Furthermore, histogram (b) highlights that, within all experimental setups, the path planning time for the BA*-APF algorithm was slightly longer than that of the A* algorithm, yet significantly shorter than those recorded for the GA, PSO, and GWO algorithms. This demonstrates the BA*-APF algorithm’s superior problem-solving efficiency, indicating its effectiveness in optimizing path planning outcomes while maintaining a balance between speed and path optimization. Histogram (c)(d) shows that after path smoothing, the overall path of the BA*-APF algorithm no longer has inflection points and has a lower number of turning angles, which greatly improves the motion efficiency of the UGV traveling at a uniform speed and reduces energy consumption, meaning such a path is more suitable for UGV travel.

To further compare the performance of the BA*-MAPF algorithm with those of the GA, PSO, GWO, APF, and A* algorithms, we also performed hypothesis tests as shown in Tables 10–12. The hypothesis test is a $t$-test and the null hypothesis is defined as $H_0$: The other algorithms are statistically superior to the BA*-MAPF algorithm. We chose the significance level $\alpha = 0.05$, and the degree of freedom was 19. If $t$ is greater than 1.729, the null hypothesis $H_0$ can be rejected, which means that the BA*-MAPF algorithm performs statistically better than the other algorithms. The BA*-MAPF algorithm and the A* algorithm plan the same path in each iteration, which means that the path length, the number of path turning points, and the number of turn-cum-turn angles do not fluctuate between the BA*-MAPF algorithm and the A* algorithm. Therefore, there is no need for hypothesis testing on the path length, the number of path turning points or the number of cumulative turning angles between the BA*-MAPF algorithm and the A* algorithm. Also, the APF algorithm cannot reach the end point in any of the three scenarios, so there is no need for hypothesis testing either. From Tables 10–12, we can see that the path length of the BA*-MAPF algorithm is the shortest among the above algorithms, and the number of path turning points and the number of turn-cum-turn angles of the BA*-MAPF algorithm are also the smallest among the above algorithms.

<table>
<thead>
<tr>
<th>Experimental Scenario</th>
<th>$H_0$</th>
<th>Test Statistic</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z-type obstacle</td>
<td>$\mu_{BA^*\text{-}MAPF} \geq \mu_{GA}$</td>
<td>$t = 41.05$</td>
<td>$H_0$: rejected (BA*-MAPF performs better than GA)</td>
</tr>
<tr>
<td></td>
<td>$\mu_{BA^*\text{-}MAPF} \geq \mu_{PSO}$</td>
<td>$t = 16.57$</td>
<td>$H_0$: rejected (BA*-MAPF performs better than PSO)</td>
</tr>
<tr>
<td></td>
<td>$\mu_{BA^*\text{-}MAPF} \geq \mu_{GWO}$</td>
<td>$t = 12.76$</td>
<td>$H_0$: rejected (BA*-MAPF performs better than GWO)</td>
</tr>
<tr>
<td></td>
<td>$\mu_{BA^*\text{-}MAPF} \geq \mu_{GA}$</td>
<td>$t = 23.55$</td>
<td>$H_0$: rejected (BA*-MAPF performs better than GA)</td>
</tr>
<tr>
<td>E-type obstacle</td>
<td>$\mu_{BA^*\text{-}MAPF} \geq \mu_{PSO}$</td>
<td>$t = 3.31$</td>
<td>$H_0$: rejected (BA*-MAPF performs better than PSO)</td>
</tr>
<tr>
<td></td>
<td>$\mu_{BA^*\text{-}MAPF} \geq \mu_{GWO}$</td>
<td>$t = 3.84$</td>
<td>$H_0$: rejected (BA*-MAPF performs better than GWO)</td>
</tr>
<tr>
<td></td>
<td>$\mu_{BA^*\text{-}MAPF} \geq \mu_{GA}$</td>
<td>$t = 18.55$</td>
<td>$H_0$: rejected (BA*-MAPF performs better than GA)</td>
</tr>
<tr>
<td>Comprehensive obstacle</td>
<td>$\mu_{BA^*\text{-}MAPF} \geq \mu_{PSO}$</td>
<td>$t = 7.55$</td>
<td>$H_0$: rejected (BA*-MAPF performs better than PSO)</td>
</tr>
<tr>
<td></td>
<td>$\mu_{BA^*\text{-}MAPF} \geq \mu_{GWO}$</td>
<td>$t = 9.58$</td>
<td>$H_0$: rejected (BA*-MAPF performs better than GWO)</td>
</tr>
</tbody>
</table>
Table 11. Results of hypothesis t-test on the number of turning points of algorithmic paths.

<table>
<thead>
<tr>
<th>Experimental Scenario</th>
<th>H₀</th>
<th>Test Statistic</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z-type obstacle</td>
<td>( \mu_{BA*-MAPF} \geq \mu_{GA} )</td>
<td>( t = 35.19 )</td>
<td>H₀: rejected (BA*-MAPF performs better than GA)</td>
</tr>
<tr>
<td></td>
<td>( \mu_{BA*-MAPF} \geq \mu_{PSO} )</td>
<td>( t = 46.63 )</td>
<td>H₀: rejected (BA*-MAPF performs better than PSO)</td>
</tr>
<tr>
<td></td>
<td>( \mu_{BA*-MAPF} \geq \mu_{GWO} )</td>
<td>( t = 37.8 )</td>
<td>H₀: rejected (BA*-MAPF performs better than GWO)</td>
</tr>
<tr>
<td></td>
<td>( \mu_{BA*-MAPF} \geq \mu_{GA} )</td>
<td>( t = 23.39 )</td>
<td>H₀: rejected (BA*-MAPF performs better than GA)</td>
</tr>
</tbody>
</table>

Table 12. Results of the hypothesis t-test for the cumulative number of revolutions of the algorithm.

<table>
<thead>
<tr>
<th>Experimental Scenario</th>
<th>H₀</th>
<th>Test Statistic</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z-type obstacle</td>
<td>( \mu_{BA*-MAPF} \geq \mu_{GA} )</td>
<td>( t = 38.93 )</td>
<td>H₀: rejected (BA*-MAPF performs better than GA)</td>
</tr>
<tr>
<td></td>
<td>( \mu_{BA*-MAPF} \geq \mu_{PSO} )</td>
<td>( t = 41.34 )</td>
<td>H₀: rejected (BA*-MAPF performs better than PSO)</td>
</tr>
<tr>
<td></td>
<td>( \mu_{BA*-MAPF} \geq \mu_{GWO} )</td>
<td>( t = 27.85 )</td>
<td>H₀: rejected (BA*-MAPF performs better than GWO)</td>
</tr>
<tr>
<td></td>
<td>( \mu_{BA*-MAPF} \geq \mu_{GA} )</td>
<td>( t = 25.48 )</td>
<td>H₀: rejected (BA*-MAPF performs better than GA)</td>
</tr>
</tbody>
</table>

5.3. Multi-Dynamic Obstacle Simulation

We used the APF algorithm and BA*-MAPF algorithm to conduct dynamic obstacle avoidance path planning simulation experiments for UGV. Where the environment size is \( 20 \times 20 \), the starting point (blue hexagram) and the end point (red pentagram) are set up, the blue squares are dynamic obstacles, the black squares are static obstacles and the red line represents the final path generated by the algorithm. The simulation results of path planning with a combination of dynamic and static obstacles are shown in Figure 20.

According to the experimental results of path planning for dynamic obstacles, both the APF algorithm and the BA*-MAPF algorithm successfully planned paths for the UGV from the starting point to the end point. However, during the path planning process, the UGV under the APF algorithm will collide with obstacles, and the obstacle avoidance effect is relatively poor, while the UGV using the BA*-MAPF algorithm successfully avoids every moving obstacle, and the obstacle avoidance effect is significantly improved. The
comparison results of these experimental scenarios fully demonstrate the superiority of the BA*-MAPF algorithm, proving its robustness in facing dynamic situations. Detailed comparisons of the experimental data can be found in Table 13.

According to the experimental results of path planning for dynamic obstacles, both the APF algorithm and the BA*-MAPF algorithm can avoid static and dynamic obstacles and plan optimal routes in urban environments. In this part of the experiment, we conducted experiments on grid maps obtained from the real-world benchmark proposed by N. Sturtevant [37]. In this part of the experiment, we used a localized London city map for the simulation, where the environment size is 50 × 50, the starting point is a blue hexagram, the goal point is a red pentagram, the red line represents the final path generated by the algorithm and dynamic obstacles (green squares) and static obstacles (black squares) are set up. The experimental results are shown in Figure 21.

![Figure 20](image-url) **Figure 20.** Path planning with dynamic obstacles. (a) APF. (b) BA*-MAPF.

<table>
<thead>
<tr>
<th>Experimental Scenario</th>
<th>Algorithm</th>
<th>Path Length (m)</th>
<th>Average Computation Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic and static obstacle map</td>
<td>APF</td>
<td>UGV collision</td>
<td>26.86</td>
</tr>
<tr>
<td></td>
<td>BA*-MAPF</td>
<td>32.88</td>
<td>30.95</td>
</tr>
</tbody>
</table>

5.4. Dynamic Obstacle Avoidance in Urban Environments

UGV are not only required to avoid static and dynamic obstacles while traveling, but also to validate their simulation in complex urban environments. In order to verify that the BA*-MAPF algorithm can avoid static and dynamic obstacles and plan optimal routes in urban environments, we conducted experiments on grid maps obtained from the real-world benchmark proposed by N. Sturtevant [37]. In this part of the experiment, we used a localized London city map for the simulation, where the environment size is 50 × 50, the starting point is a blue hexagram, the goal point is a red pentagram, the red line represents the final path generated by the algorithm and dynamic obstacles (green squares) and static obstacles (black squares) are set up. The experimental results are shown in Figure 21.

![Figure 21](image-url) **Figure 21.** Cont.
Figure 21. Path planning in urban environments. (a) t = 29 s. (b) t = 33.78 s. (c) t = 51.19 s. (d) t = 59.07 s. (e) t = 85.22 s. (f) t = 109.02 s.

From the path planning results in Figure 21, it can be seen that the UGV successfully performed dynamic obstacle avoidance three times throughout the path planning process, and each time it was able to effectively avoid moving obstacles. At the same time, the UGV was also able to search the global path from the starting point to the end point with the help of the BA*-MAPF algorithm, and always stayed away from the obstacles present in the environment, ensuring the safety of the path.

5.5. Path Planning in 3D Space

In order to better simulate the real situation and confirm the applicability of the BA*-MAPF algorithm in complex environments, it is necessary to carry out simulations in a 3D environment for verification. In this part of the experiment, we use 3D environment modeling for simulation, with an environment size of 100 × 100 × 100, a blue hexagram as the starting point, a green circle as the target point, and a red curve representing the final path generated by the algorithm. The colored grid points represent randomly generated 3D obstacles, with darker colors indicating lower elevation and lighter colors indicating higher elevation. If the UGV hits an obstacle, the experiment fails. The experimental results are shown in Figure 22.

Utilizing the BA*-MAPF algorithm, our UGV demonstrates exceptional navigation capabilities through 3D mountain terrains, as evidenced by the path planning results in Figure 22, effectively avoiding obstacles and securing a route from start to end. This algorithm enables the UGV to conduct comprehensive global path searches, consistently yielding shorter and more efficient paths. Our simulation experiments substantiate the BA*-MAPF algorithm’s effectiveness and adaptability in complex environments, showcasing its potential to significantly enhance UGV navigational performance in challenging terrains.
Utilizing the BA*-MAPF algorithm, our UGV demonstrates exceptional navigation capabilities through 3D mountain terrains, as evidenced by the path planning results in Figure 22, effectively avoiding obstacles and securing a route from start to end. This algorithm enables the UGV to conduct comprehensive global path searches, consistently yielding shorter and more efficient paths. Our simulation experiments substantiate the BA*-MAPF algorithm's effectiveness and adaptability in complex environments, showcasing its potential to significantly enhance UGV navigational performance in challenging terrains.

6. Conclusions

The BA*-MAPF algorithm resolves the limitations of the A* algorithm, including its high computational demand, generation of suboptimal paths, and issues with path smoothness, continuity, and redundancy, thereby significantly boosting overall efficiency. By incorporating the MAPF component, it effectively navigates around the traditional pitfalls of the APF algorithm, such as susceptibility to local optima and challenges in reaching endpoints, enhancing the accuracy of path planning. The introduction of a bidirectional alternating search strategy markedly increases the efficiency of the path search process. Furthermore, by employing interpolation to remove redundant path nodes, the algorithm not only shortens the path length but also reduces the required computation time, streamlining the pathfinding process for optimized navigational outcomes. Smoother paths are obtained by utilizing cubic B-spline interpolation curves. The potential field function is modified and a distance factor is introduced to improve the local optimality and the target unreachability problem that tends to occur in the APF algorithm. Path planning in dynamic obstacle environments is thus realized. Finally, simulation experiments were conducted in three static scenes and an urban dynamic environment. Our simulation results demonstrate
that the BA*-MAPF algorithm surpasses traditional algorithms such as GA, PSO, GWO, APF, and A* in key aspects of planning, including time, path length, number of turning points, and corners. The paths generated by the BA*-MAPF algorithm are notably more suited to UGV navigation in complex environments, showcasing its superior capability to efficiently and effectively chart courses that are optimal for UGV movement, thereby highlighting its significant advancements in path planning technology.

However, many application scenarios of UGVs are located in complex 3D dynamic environments where unexpected situations may arise. Therefore, we plan to design more comprehensive collision avoidance algorithms to solve the path planning problem of UGVs in these complex 3D dynamic environments in our future research.

**Author Contributions:** Conceptualization, X.M.; methodology, X.M.; software, X.M.; formal analysis, X.M.; investigation, X.M.; resources, X.F.; data curation, X.M.; writing—original draft, X.M.; writing—review and editing, X.M. and X.F.; visualization, X.M.; supervision, X.F.; project administration, X.F.; funding acquisition, X.F. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** Data sharing is not applicable to this article as no new data were created or analyzed in this study.

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

References


7. Zhao, S.; Hwang, S.H. Complete coverage path planning scheme for autonomous navigation ROS-based robots. *ICT Express* 2024, 10, 83–89. [CrossRef]


13. Zhang, G.; Han, J.; Li, J.; Zhang, X. APF-based intelligent navigation approach for USV in presence of mixed potential directions: Guidance and control design. *Ocean Eng.* 2022, 260, 119172. [CrossRef]


28. Gu, Q.; Zhen, R.; Liu, J.; Li, C. An improved RRT algorithm based on prior AIS information and DP compression for ship path planning. *Ocean Eng.* 2023, 279, 114595. [CrossRef]

29. Li, X.; Yu, S. Three-dimensional path planning for AUVs in ocean currents environment based on an improved compression factor particle swarm optimization algorithm. *Ocean Eng.* 2023, 280, 114610. [CrossRef]


34. Song, B.; Wang, Z.; Zou, L. An improved PSO algorithm for smooth path planning of mobile robots using continuous high-degree Bezier curve. *Appl. Soft Comput.* 2021, 100, 106960. [CrossRef]


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.