


Article

Air Defense Interception Plan Generation Method Based on Modified A* Optimization Algorithm

Xiaocheng Song ^{1,*}, Zhi Li ¹, Shuting Feng ¹, You Li ², Liang Sun ³ and Jingjing Jiang ⁴ 

¹ Department of Beijing Institute of Electronic Engineering, China Aerospace Science and Industry Corporation Co., Ltd., Beijing 100143, China

² School of Aerospace Science and Technology, Xidian University, Xi'an 710126, China

³ School of Intelligence Science and Technology, University of Science and Technology Beijing, Beijing 100083, China

⁴ Department of Aeronautical and Automotive Engineering, Loughborough University, Loughborough LE11 3TU, UK

* Correspondence: sxchitman@126.com

Abstract: Aiming at the air defense task requirements for an enemy's large-scale aircraft attack, this paper presents a plan generation algorithm which can quickly give an interception scheme. The main contribution of this paper is the modification of the standard A* algorithm and its combination of the optimization algorithm and air-defence mission. Firstly, the enemy's attack weapon and our defense platform are modeled, and kinetic equations and interception efficiency functions are constructed, and the intercepted criteria are established. Then, the interception-cost mixed optimal function is established to clarify the system optimization objective. Secondly, aiming at the characteristics of strong time sensitivity of air defense interception, a modified A* optimization algorithm with fast convergence characteristics is used to solve the optimization problems, the standard A* algorithm is modified and the optimal air defense interception plan under the condition of mixed performance index is given. Finally, the proposed method is verified by numerical simulations.

Keywords: air defense interception; scheme generation; modified A* optimization; fast convergence rate



Citation: Song, X.; Li, Z.; Feng, S.; Li, Y.; Sun, L.; Jiang, J. Air Defense Interception Plan Generation Method Based on Modified A* Optimization Algorithm. *Electronics* **2023**, *12*, 719. <https://doi.org/10.3390/electronics12030719>

Academic Editor: Mahmut Reyhanoglu

Received: 23 November 2022

Revised: 19 January 2023

Accepted: 27 January 2023

Published: 1 February 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Modern air attack operations are completed by the cooperation of a variety of weapons, including cruise missiles, ballistic missiles, guided bombs, bombers, attack aircraft and helicopters. The design method of the interception plan of the traditional air defense weapons is difficult to adapt to all-weather, all-air-space, multi-level and saturated air attack modes. At present, the design method of an air defense weapon interception scheme has not yet formed a complete theoretical system. Most of the research is only focused on specific practical problems, and the optimization mathematical model is established according to certain assumptions. Hence, it is difficult to achieve the maximization of air defense efficiency and minimization of interception cost at the same time.

With the development of unmanned technology, artificial intelligence technology and big data technology, battlefield perception is becoming more and more ubiquitous, and cluster operations are becoming more and more autonomous and coordinated, and combat system are becoming more and more cloudy. Faced with more diversified and intelligent means of enemy surprise defense, passive defense is less favorable. At the same time, air defense and anti-missile operations involve a large number of elements, changing operational styles, which puts forward higher requirements. An air defense and antimissile command control system is the core of realizing an air defense and antimissile network and systematic operation. How to find the most effective and appropriate command and control method to maximize its effectiveness has become a key problem in the research process of air defense and antimissile command and control systems. The intelligent command and

control system can greatly improve the robustness and agility of the system, which is the inevitable direction of the development of the intelligent command and control system in the future.

Due to the continuous development of artificially intelligent (AI) technologies such as exploring big data, data mining and deep learning, more and more researchers are combining AI technology with aerospace defense, and use intelligent technology and big data to promote the upgrading of aerospace defense systems. Hughes [1], Liu [2] and Bai [3], in order to meet the urgent demand for real-time intelligent assignment of weapon targets in the highly dynamic environment of modern air defense operations, considering the number of weapons and interception capability, taking the maximum damage efficiency as the optimization target, established an optimization model for weapon target assignment, and then a method of intelligent allocation of air defense weapon targets based on a neural network was proposed. On the basis of depth analysis of the main factors affecting the air defense operations of missile mixed groups, Liu [4] and Wang [5] aimed at the combat effectiveness assessment of the ground-air missile mixed group, applying BP neural network technology to establish the air defense combat effectiveness evaluation index system and effectiveness evaluation model, and evaluated the combat effectiveness of the ground-air missile mixed group. In the literature, Wang [6], Chang [7] and James [8] solve the problem of insufficient speed of the optimal algorithm of allocation strategy in large-scale scenes through combining deep reinforcement learning with it and intelligently solving the problem of large-scale air defense task allocation. In the references, aiming at the problems of multiple and complex weapon target allocation constraints in current air defense operations, the traditional model cannot truly reflect the war process, and model credibility is low; a weapon target optimization allocation model based on multi-agent system theory is proposed and solved by using an improved and accelerated gradient descent algorithm. Wang [9], Liu [10] and Dong [11] have, respectively, studied the weapon assignment interception and air defense combat system by using neural network technology, but have not studied the timeliness of air defense interception or proposed an optimization scheme for intelligent algorithms. Although researchers [12–15] proposed a solution to the timeliness problem of air defense tasks in the reference, they only combined deep learning with an air defense allocation algorithm, and did not propose an optimization scheme for the algorithm to improve its fast convergence. In these references, although the improved AGD-distributed multi-intelligence system solves the problem of weapon distribution in air defense operations, the timeliness of air defense is not studied. Researchers [16–20] also paid attention to the combination of AI techniques and military missions, and some AI methods are developed for anti-missile, anti-ship, anti-aircraft missions. The researchers pointed out that AI techniques are key to future war and need much more attention.

In addition, AI technologies, such as deep learning and data mining, are not only applied to air defense, but also widely used in various fields, such as spacecraft fault diagnosis, drones, robots, navigation, image processing, network security and so on. It can be seen that AI technology has been widely used in various scientific and technological fields, and plays an important role in the development of various aspects of science and technology. In this study, an algorithm that can quickly give interception plans and plan generation is proposed to meet the air defense mission requirements of large-scale aircraft attacks. Its main advantages and characteristics could be concluded as follows: (1) the interception-cost mixed optimal function is established; (2) the system optimization objective is clarified. An A* optimization algorithm with fast convergence rate is proposed to solve the optimization problem. Finally, the proposed method is verified by numerical simulations.

2. Mathematical Model

Suppose the enemy attack weapon platform cluster is defined as follows. $M = (target_1, target_2, \dots, target_m)$, and i th intercept target is launched from origin O

at the initial moment, the whole trajectory is in its orbit xOz plan and satisfies the parabolic form as shown in Equation (1).

$$\begin{cases} z_{T_i}(t) = ax_{T_i}^2(t) + bx_{T_i}(t) \\ [\dot{x}_{T_i}(t)]^2 + [\dot{z}_{T_i}(t)]^2 = v_{T_i}^2 \\ x_{T_i}(t_0) = 0, y_{T_i}(t_0) = 0, z_{T_i}(t_0) = 0 \end{cases} \quad (1)$$

Similarly, assume our interceptor weapon platform cluster is defined as follows. $N = (chaser_1, chaser_2, \dots, chaser_n)$, assume the initial position selected region Ω of j th interceptor is consist of excluding M disjoint disc regions in the xOy plane, that is

$$\begin{cases} \Omega_j \triangleq \bigcap_{i=1}^n \Omega_i \\ \Omega_j \triangleq \left\{ (x_j, y_j, z_j) \mid (x_j - x_n)^2 + (y_j - y_n)^2 \geq r_n^2 \right\} \end{cases} \quad (2)$$

All parameters of our interceptor in the above equation are known. It is also assumed that interception region satisfies

$$[x_i(t) - x_j(t)]^2 + [y_i(t) - y_j(t)]^2 + [z_i(t) - z_j(t)]^2 \leq \delta. \quad (3)$$

Taking into account the possible local orbital maneuvers, interference, etc., we assume that the probability of our interceptor successfully intercepting the enemy's i th aircraft is ρ_{ij} , after reaching the interception area. Meanwhile, the cost function of our launching i th interceptor is τ_i . Then, the probability function describing of our interceptor cluster intercepting the enemy's aircraft destruction is

$$J = \sum_{j=1}^{k_i} \prod_{i=1}^m (1 - l\rho_{ij}) \frac{\alpha_i}{\beta_j} \quad (4)$$

where α_i, β_j are the value function and cost function of i th enemy weapon and j th of interceptor, l is the possibility value parameter which describes the importance of interception possibility (larger l means the enemy weapon is more important and needs to be intercepted even with higher cost), and k_i meets

$$\sum_{i=1}^m k_i \leq n. \quad (5)$$

3. Problem Description

The object of this study can be summarized as follows. Find the mapping $f : N(j) \rightarrow M(i)$ of our interceptor for the enemy's aircraft under condition of Equations (1)–(3) and (5), and maximize the objective Equation (4).

However, it is worth noting that due to the specificity of the problem, this study has the following characteristics for its research question.

(1) Time sensitive: generally the time window from detecting enemy strike targets to enemy weapons hitting our target is usually in the order of minutes, which also involves the command chain of our decision–command–combat platform, so the time left for generation of air defense interception plans is extremely short.

(2) The large amount of data: the weapon, from the enemy's weapon platform to our interception platform, may be more than a hundred kinds, and the large amount of data computing further intensifies the urgency of the problem of time sensitivity.

(3) The algorithm must converge: an urgent combat threats scheme generation system cannot accept the results of algorithm dispersion, which would result in no scheme generation. In other words, the scheme generation algorithm must be convergent.

In summary, the problems faced in this study lead to the disadvantages of the traditional variational approach to solving optimization problems in terms of a large number of

operations and slow convergence rate, while the current emergence of big data and reinforcement learning methods have the disadvantages of time sensitivity, and the possibility of non-convergence, so it is necessary to find a new method to solve the problem of air defense interception scheme generation.

4. Modified A* Algorithm

4.1. Scheme of Modified A* Algorithm

A* algorithm, as a heuristic algorithm, synthetically examines the relationship between intermediate nodes and the edge values between the initial and final points. If the problem of path search has a solution, it can find the optimal path in the feasible solution, and avoid excessive unnecessary search loss; if the algorithm fails to obtain the feasible solution, it is known through the algorithm's traversal of all nodes in the grid that there is no feasible path between the initial and final nodes.

Its advantages are: (1) the algorithm structure and process are simple and reliable; (2) the number of nodes can be counted; (3) the convergence capability is strong. This algorithm starts from the center point and spreads around to find the optimal solution of the objective function, so the convergence capability is strong (even if there is no subsequent better result and the algorithm spreads, the initial value is the optimal output; hence, the system can guarantee the existence of the output).

The core advantage of this algorithm is that a heuristic function is set up to assign different weights to each node, which determines the direction of the path search and improves the search efficiency. The general form of this heuristic function is

$$h(n) = p(n) + f(n) \quad (6)$$

where $h(n)$ is the difference of the optimization function between the beginning and end of the path containing node n ; $p(n)$ is the difference of the optimization function between node n and the initial node; $f(n)$ is the difference of the optimization function between node n and the next code. Thus, if the shortest path is the goal, we only need a node that minimizes $h(n)$ as the priority search direction. In turn, the optimization problem is transformed into a system node-partitioning and node-finding problem.

Standard A* optimization algorithm has the advantage of simple structure and strong adaptability, but its disadvantage is its long and non-smooth path. In this study, the standard A* algorithm is modified from following aspects: (1) implement the sugar diffusion algorithm to deduce the computation of meshing grids; (2) implement the direction expand method to reduce the time to find the path; (3) clear the redundancy nodes by secondary redundancy method to simplify the path; (4) implement the adaptive smoothing algorithm to smooth the generated path.

Comparing with the standard A* optimization algorithm, the specific steps proposed in this study of the algorithm are follows.

Step 1. Set flag bit s for the initial node and place it in the Opened table.

Step 2. With s as the current node, place all the nodes adjacent to it into the Opened table, and calculate the corresponding heuristic function value $h(n)$, and set s point as the parent of each adjacent node. Finally, move the point s into the Closed table as the examined node.

Step 3. If Opened table is empty, the optimization search fails and there is no solution. Instead, the node corresponding to the smallest $h(n)$ is moved into the Closed table. For convenience, this part assumes that its parent node is b .

Step 4. If a is the target node and the path search is successful, output the path and end the program. If not, go to the next step.

Step 5. With a as current node, repeat the process of step 2 to step 3 to find the node c corresponding to the smallest $h(n)$ and set the parent node of c as a . Note that if the node adjacent to a is already in the Closed table, it is not examined.

It is worth noting that during the search process, the objective function of node b to c needs to be examined. If it is smaller than the actual objective function optimization result

to a , the parent node of c needs to be set to b , otherwise remains unchanged. In addition, for the optimal path search problem, if there is a feasible solution, the A* algorithm is solvable; if $f(n)$ is solvable in the heuristic function, the optimal path is solvable.

The process of the modified A* algorithm is given in Figure 1 and the orange brackets are the modifications made in this study.

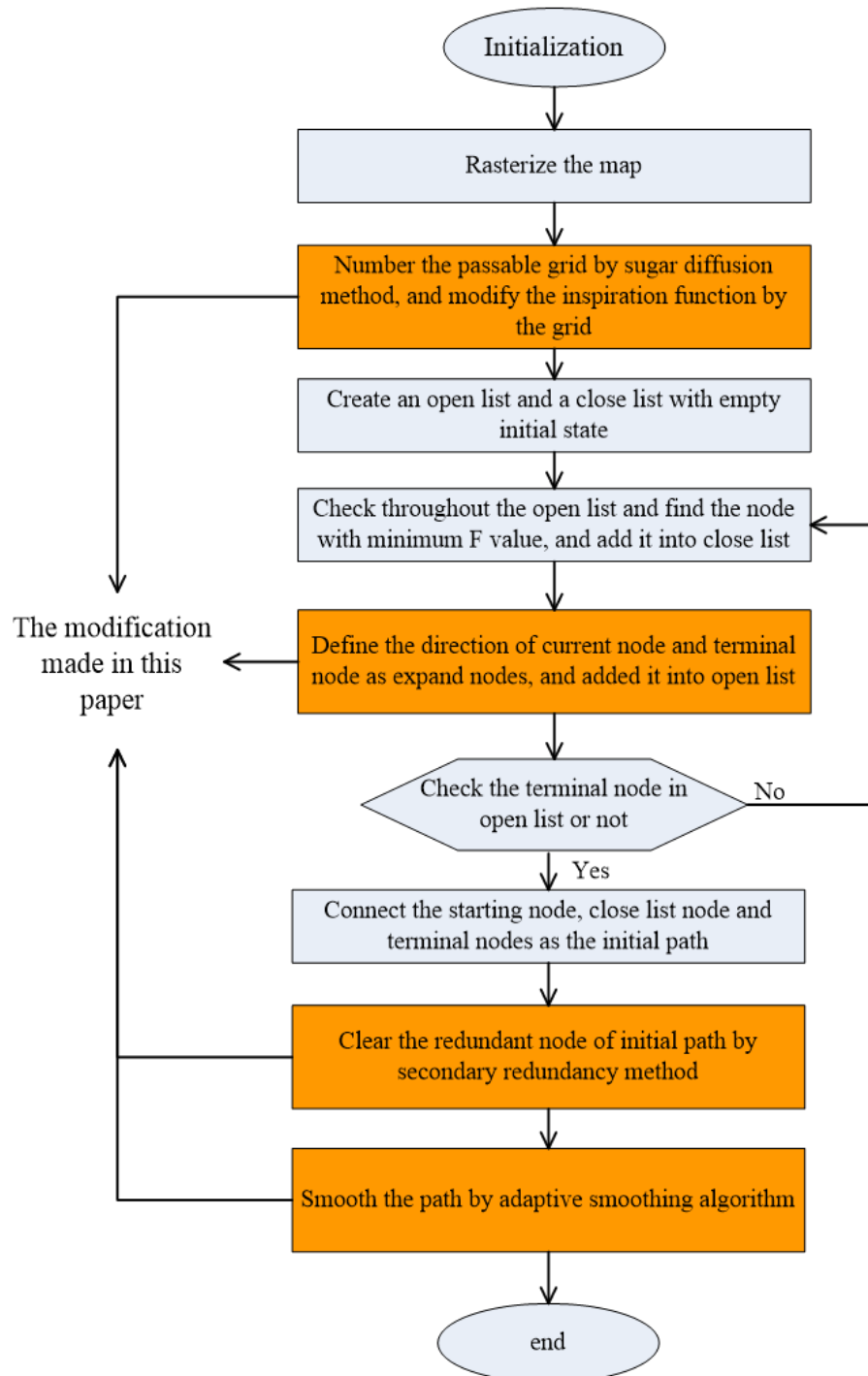


Figure 1. The flow-process chart of modified A* optimization algorithm.

4.2. Sugar Diffusion Method

The A* algorithm tends to consume a lot of time in path planning when the map is more complex, as shown in Figure 2.

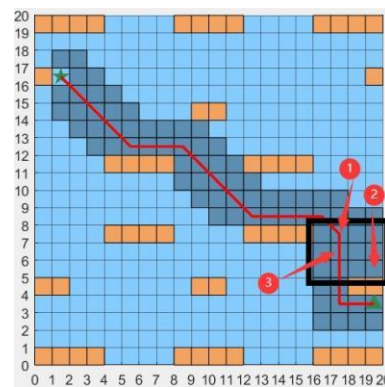


Figure 2. Schematic diagram of complex environment.

The blue boxes in Figure 2 represent the passable areas, the orange boxes represent the obstacles, and the gray boxes represent the boxes that the A* algorithm has searched during the path planning. In order to find the correct path nodes, we need to backtrack the whole algorithm and recalculate the $f(n)$ values of different raster, thus the A* algorithm takes a lot of time trying to jump out of the black box to find the grid labeled 3.

In order to solve this problem, this study adopts the sugar diffusion method to preprocess the map, number the map raster, and then improve the A* algorithm heuristic function according to the raster number, which can quickly jump out of a complex environment and reduce the computational time of the algorithm. The sugar diffusion method was firstly proposed to solve the problem of “gyration structure” of the map in a special case of the ant colony algorithm. It is assumed that there is a strongly smelling food at a certain point of the map, and the smell of the food will be radiated to the surrounding area in layers, with the strongest smell in the grids where the food is located and the lighter smell in the grids which are farther away from the food. The odor concentration of the grids on the same layer is the same, and all the grids with odor radiation are marked with the odor element value in turn.

The biggest advantages of the sugar diffusion method are twofold. Firstly, all the passable areas of the whole grid map can be marked with the odor element value, and only the obstacle grids or the passable areas completely surrounded by the obstacle grids will not be marked; secondly, the sugar diffusion method has a single tendency. Therefore, the end point can be set as the “food center”.

The sugar diffusion method can also choose 4-neighborhood diffusion and 8-neighborhood diffusion when odor diffusion is performed, and the 8-neighborhood diffusion method is chosen in this study. The diffusion process of the sugar diffusion method is shown below.

(1) Creates a collection of marked odor raster P and the initial state of the collection P is empty.

(2) Consider the endpoint as the raster where the food is located, and mark the odorant value as 1, and add the endpoint to the set P .

(3) Take all the grids with the smallest odorant value from the set P and denote the value of the odorant as t , then 8-neighborhood expansion is performed with these grids as the center. If the neighboring grid is an obstacle, the grid should be ignored; if the neighboring grid is not tagged with odor value, its odor value is marked $t + 1$ and is added to the set P ; if the neighboring grid is already in the set P , the grid should be ignored.

(4) After all the grids with the odor element value t are expanded, if there are still grids in the map that are not marked as passable areas except for the obstacle grids and the passable areas completely surrounded by the obstacle grids, then go back to step 3 and continue the next expansion step until the whole map is marked.

4.3. Direction Expand Method

The traditional A* algorithm searches in eight directions, as shown in Figure 3, where the yellow dots represent the robot’s retaining wall location and the numbers 1 to 8 represent the eight search directions of the A* algorithm.

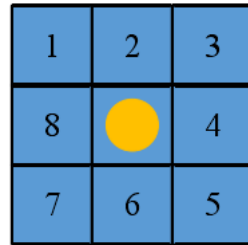


Figure 3. Search directions of traditional A* algorithm.

Through the study, it is known that each expansion only searches a limited number of directions. It can determine the next node with the same 8 directions searched, then the directions that do not need to search can be called useless directions. Such a diagram is shown in Figure 4.

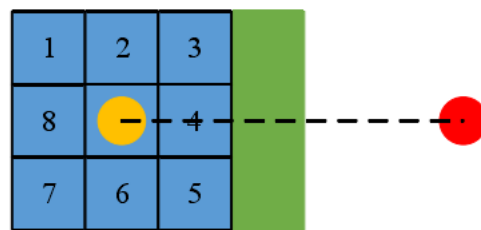


Figure 4. Diagram of useless direction.

The red dot in Figure 4 represents the end point of this path planning, and the green rectangle is the obstacle. The current location of the robot is in the same horizontal line with the end point. When searching using the traditional A* algorithm, by comparing the $f(n)$ values of 8 directions, the $f(n)$ value of direction 4 is the smallest, and the next node of this extension is determined to be the node where direction 3 is located. The analysis shows that from the current path node, only search {3,4,5} in three directions can be derived from the next node for the direction 3 node, and at this time there is no need to search the remaining directions, which we call useless directions.

For this characteristic, we propose a directional expansion improvement to the traditional A* algorithm, which divides the 8 search directions into different sets of directions, selects a specific set of directions according to the relative position of the current path node and the end point at each search, and carries out direction expansion according to the directions in the set, ignoring the expansion directions outside the set. This method can achieve the purpose of reducing the search range and computational effort of the algorithm.

The specific step of directional expansion is that the coordinate difference between the current path node and the end point is found according to Equation (7).

$$\begin{cases} \Delta x = x_{goal} - x_n \\ \Delta y = y_{goal} - y_n \end{cases} \quad (7)$$

where (x_{goal}, y_{goal}) denotes the end point coordinates, and (x_n, y_n) denotes the current path node coordinates.

The magnitudes of Δx and Δy values are compared separately into 8 cases, and the 8 search directions are divided into 8 different sets of extended directions, and the

corresponding sets of search directions are selected according to the different magnitude cases of Δx and Δy in Equation (8).

$$\text{Expansion direction} = \begin{cases} \{2, 3, 4, 5, 6\}, & \Delta x > 0 \text{ and } |\Delta x| > |\Delta y| \\ \{8, 1, 2, 3, 4\}, & \Delta y > 0 \text{ and } |\Delta y| > |\Delta x| \\ \{2, 1, 8, 7, 6\}, & \Delta x < 0 \text{ and } |\Delta x| > |\Delta y| \\ \{8, 7, 6, 5, 4\}, & \Delta y < 0 \text{ and } |\Delta y| > |\Delta x| \\ \{3, 4, 5\}, & \Delta x > 0 \text{ and } \Delta y = 0 \\ \{1, 2, 3\}, & \Delta y > 0 \text{ and } \Delta x = 0 \\ \{1, 8, 7\}, & \Delta x < 0 \text{ and } \Delta y = 0 \\ \{7, 6, 5\}, & \Delta y < 0 \text{ and } \Delta x = 0 \end{cases} \quad (8)$$

The numbers in the set of 8 search directions correspond to the different search directions in Figure 3. In this way, the number of directions can be reduced from the traditional 8 to 3 or 5 for each search, thus reducing the search range and computational effort of the algorithm. In addition, to avoid the failure of directional expansion in extreme map environments, the traditional 8-neighborhood expand method is used to determine the next node at a node when the next node cannot be searched by the direction expand method.

4.4. Secondary Redundancy Clear Algorithm

Path redundant node means that the path length can be reduced after the node is deleted, and the optimized path will not cause safety problems to cross obstacles, and the redundant node in the path planned by the traditional A* algorithm is caused by its search mechanism.

In this study, we propose a secondary redundancy clear algorithm with a safe distance limit to reduce the path length by removing redundant nodes on the premise that the distance between the path and the obstacle is sufficient for the actual movement of the robot. The secondary redundancy clear algorithm is improved on the basis of the common redundant point removal algorithm, and the algorithm flow of the common redundant point removal algorithm is as follows.

(1) Obtain the coordinates of all nodes on the path planned by the traditional A* algorithm and put them into the set $\{M_i | i = 1, 2, \dots, n\}$ in which node M_1 denotes the starting point and node M_n denotes the end point.

(2) Create a set W to hold critical nodes that cannot be removed, and the initial state of W includes only the starting point M_1 and ending point M_n .

(3) Judgment from the starting point M_1 , firstly connect the nodes M_1 and M_3 to get the straight line M_1M_3 , as shown in Figure 5, and judge whether the straight line M_1M_3 passes through the obstacle, if the straight line M_1M_3 does not pass through the obstacle, then prove that the node M_2 is a redundant node, and delete the node M_2 from the set $\{M_i\}$, then connect the straight line M_1M_4 for judgment until the line between M_1 and some intermediate node M_m passes through the obstacle, then prove that the node M_{m-1} is a key node. Put M_{m-1} into the set W , then start from the node M_{m-1} , connect the straight line $M_{m-1}M_{m+1}$ and continue to judge backward until it connects to the end point M_n to finish all judgments.

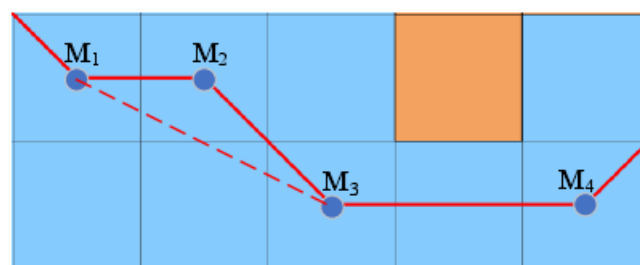


Figure 5. Schematic diagram of redundant point removal algorithm.

(4) At this point, the nodes in the set W are all the retained key nodes, and the optimization of the paths can be completed by connecting these nodes in turn.

However, the common redundant point removal algorithm determines whether a node is redundant by directly detecting whether the two-point line crosses the obstacle. If it does not, the intermediate node is removed. However, there are special cases when the two-point line may produce a tangent to an obstacle, as shown in Figure 6.

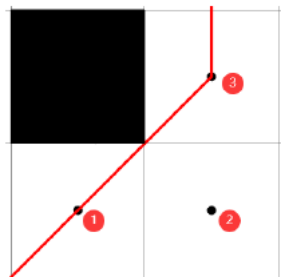


Figure 6. Special case diagram of redundant point removal algorithm.

The black dots 1, 2 and 3, in Figure 6, are the original path nodes determined by the traditional A* algorithm. When using the common redundancy removal algorithm, the line 13 is connected to determine whether it passes through the obstacle, and since the line 13 is tangent to the vertex of the obstacle, it does not pass through the obstacle, so the black dot 2 is determined as a redundant node, and the optimized path has dot 2 removed. Therefore, the path optimized by the common redundant point removal algorithm is not suitable for the actual robot walking.

To deal with such issue, this study sets the safety distance between the redundant points and the obstacles on the basis of the common redundant points removal algorithm: each time after connecting the straight line to two points, according to Equation (9), the connected line is divided into k equal parts, and the distance between the k equal parts and the center point of all obstacles is judged in turn whether it is less than the preset safety distance, as shown in Figure 7. Only if the distance between all the equipartition points and the center point of the obstacle is greater than the preset safety distance, then the nodes between the two points constituting the straight line are considered as redundant nodes and can be removed; when there is any equipartition point whose distance to the center point of the obstacle is less than the preset safety distance, then the redundant point removal operation cannot be performed. In this study, the preset safety distance is the length of the side of a grid.

$$k = \text{ceil}(\text{norm}(M_{i+2} - M_i) * k_1) \tag{9}$$

where $\text{norm}()$ function is used to find the Euclidean norm; $\text{ceil}()$ function is used to find the nearest integer not less than the number in parentheses; k_1 is any positive integer. The larger the value of k_1 , the more equal points are inserted in each straight line, and the higher the accuracy of judging whether the path is safe, but the computational load of the overall algorithm will be greatly increased. Through experimental comparisons, $k_1 = 5$ here, has the best effect.

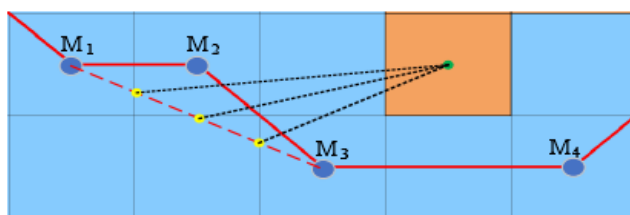


Figure 7. Diagram of safety distance.

The yellow dots in Figure 7 are the equipartition points of the line M_1M_3 , and the green dot is the center point of the obstacle. From Figure 7, we can see that since the distance between the three aliquots and the center point of the obstacle is greater than the edge length of a raster, the line M_1M_3 satisfies the safety distance condition, so the node M_2 is a redundant node that can be removed.

4.5. Adaptive Path Smoothing Algorithm

B spline curve is a common curve-fitting method, which is a generalization of the Bezier curve. Due to the defects of the Bezier curve such as inflexibility, poor control, and difficulty in making local modifications, Gordon et al. extended it and proposed the B spline curve, which retains all the advantages of the Bezier curve, while overcoming its drawbacks.

The B spline curve can be expressed by

$$P(t) = \sum_{i=0}^k P_i \cdot F_{i,k}(t) \tag{10}$$

where P_i is the position phase measure of each feature node and $F_{i,k}(t)$ is the k -order spline piecewise mixing function and $F_{i,k}(t)$ is expressed as shown in

$$F_{i,k}(t) = \frac{1}{k!} \sum_{m=0}^{k-i} (-1)^m C_{k+1}^m (t+k-m-i)^k \tag{11}$$

The full name of these curve segments is k times B-sample curve ($k = 0, 1, \dots, m$). The magnitude of k value reflects the smoothness of the curve. If k increases, then smoothness of the curve increases, but the computational effort also increases.

When $k = 1$, the primary B-sample curve at this point is a straight line determined by the starting point and the ending point, and its mathematical expression is given by Equations (10) and (11) as

$$P(t) = (1 - t)P_0 + tP_1 \tag{12}$$

When $k = 2$, an example graph of a quadratic B spline curve is shown in Figure 8, which consists of four B spline curves. Each individual curve is controlled by three adjacent vertices. The quadratic B-sample curve given by (13) is indicated.

$$P(t) = \frac{1}{2}(t - 1)^2P_0 + \frac{1}{2}(-2t^2 + 2t + 1)P_1 + \frac{1}{2}t^2P_2 \tag{13}$$

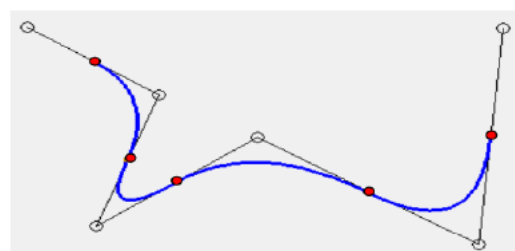


Figure 8. Second-order B-spline curve.

When $k = 3$, an example plot of a cubic B-spline is shown in Figure 9, which consists of 3 B-splines. Each individual curve is controlled by four adjacent vertices. Quadratic B-splines can be expressed using the Equation (14).

$$P(t) = \frac{1}{6}(1 - t)^3P_0 + \frac{1}{6}(3t^3 - 6t^2 + 4)P_1 + \frac{1}{6}(-3t^3 + 3t^2 + 3t + 1)P_2 + \frac{1}{6}t^3P_3. \tag{14}$$

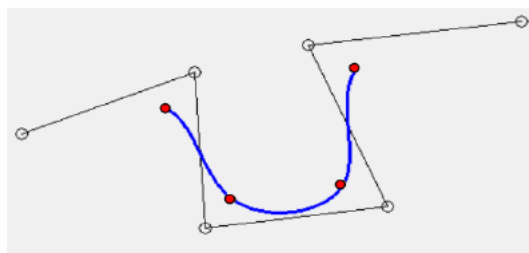


Figure 9. Triple B spline curve.

Considering the path optimization effect and the calculation amount of the algorithm, the cubic B-spline curve is selected for curve fitting by experimental comparisons, but because the distribution of the path nodes used to fit is relatively discrete, if the cubic B-spline curve is directly used for fitting, the path may pass through the obstacle due to the excessive amplitude of the fitting curve, so this study proposes an adaptive path smoothing algorithm based on the cubic B-spline for smoothing.

Firstly, the path nodes optimized by the secondary redundancy removal algorithm are obtained, and local endpoints are added near each path node except for the start and end points in an adaptive manner. The local endpoints are added as shown in

$$B_n = \begin{cases} A_n - \sum_{i=1}^n \frac{i}{len} \cdot (A_n - A_{n-1}) \\ A_n + \sum_{i=1}^n \frac{i}{len} \cdot (A_{n+1} - A_n) \end{cases} \tag{15}$$

where B_n is the added local endpoints; A_{n-1} , A_n and A_{n+1} are the coordinates of each path node, and len is the map edge length. Equation (15) can be used to add local endpoints with different spacing according to different distances between two nodes in maps of different sizes, as shown in Figure 10, where the black dots are the path nodes and the blue dots are the added local endpoints. It can be found from Figure 10 that since the length of $A_n A_{n+1}$ is greater than $A_{n-1} A_n$, the local endpoint spacing on $A_n A_{n+1}$ is greater than the local endpoint spacing on $A_{n-1} A_n$.

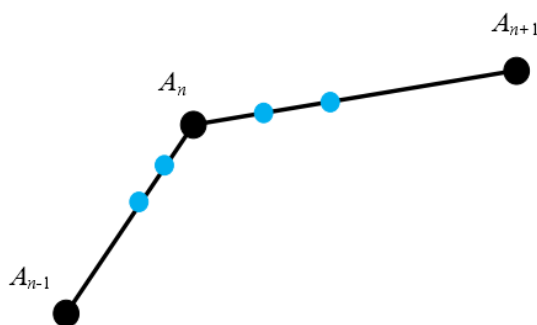


Figure 10. Add local endpoints.

By using both the path nodes and the added local endpoints as the feature nodes for curve fitting in Equation (12), and then fitting the curve, we can reduce the amplitude of the fitted curve and avoid obstacles while ensuring the smoothness of the curve.

5. Interception Constraints

Once the theoretical optimization approach is given, there is one more step (the interceptability criterion of our interception platform for enemy strike weapons) to be performed before the optimization process can be carried out. Generally, without considering the enemy weapon maneuver (if the enemy’s weapon has end maneuver capability, it is considered that the interception probability of our weapon is reduced), the conditions for our

weapon to intercept the enemy weapon mainly depend on the time criterion, the speed criterion and the overload criterion. Our weapon can be considered to have the ability to intercept the enemy weapon if the above three conditions are satisfied. Such three conditions are introduced as follows.

(1) Time criterion. The physical meaning of this criterion is that our vehicle is able to approach the enemy weapon close enough and intercept it before it hits our target. We only consider the motion of the enemy vehicle in the xOz plane in this study, and its altitude is 0 when it hits our target. Considering the most extreme case under this condition we have

$$\begin{cases} x_{T_i}(t) = -b/a, |\dot{x}_{T_i}| \leq v_{T_i} \\ T_{threshold} = -b/av_{T_i} \end{cases} \quad (16)$$

Hence, our vehicle needs to approach the enemy vehicle within the threshold time. That is,

$$|d_{ij} - \delta|/v_{C_j} \leq T_{threshold} = -b/av_{T_i} \quad (17)$$

where v_{C_j} is the speed of our j th vehicle and d_{ij} is the initial distance between our j th interception platform and the enemy i th weapon defined as

$$d_{ij} = \sqrt{(x_j - x_{T_i}(t_0))^2 + (y_j - y_{T_i}(t_0))^2 + (z_j - z_{T_i}(t_0))^2}. \quad (18)$$

Thus, if Equation (17) is satisfied, the time constraint is satisfied.

(2) Speed criterion. Speed criterion refers to whether our vehicle has the ability to intercept enemy weapons at the end of interception. The current guidance methods, such as proportional guidance method, parallel approach method, etc., require our vehicle's speed to be larger than that of the enemy vehicle, so that the enemy weapon can be tracked at the end. This means our vehicle should meet the following condition

$$v_{C_j} \cos \eta_{C_j} - v_{T_i} \cos \eta_{T_i} \geq 0 \quad (19)$$

where η_{C_j}, η_{T_i} are the leading angles of the j th and i th vehicles of enemy and our side. Considering the worst condition, Equation (19) can be further simplified to

$$v_{C_j} \cos \eta_{sup} - v_{T_i} \geq 0 \quad (20)$$

with η_{sup} denoting the maximum forward angle of our vehicle.

(3) Overload criterion. In the guidance end, the line-of-sight guidance method, the parallel approach method and the most mainstream proportional guidance method require our vehicle to have a strong overload capacity to realize the end of maneuver. In other words, considering the requirements of the normal overload, we have

$$r \frac{dq}{dt} = v_{C_j} \sin \eta_{C_j} - v_{T_i} \sin \eta_{T_i} \quad (21)$$

In the most extreme cases, we have

$$r \frac{dq}{dt} \geq v_{C_j} \sin \eta_{low} - v_{T_i} \quad (22)$$

Thus, our overload needs to meet

$$g_j \geq v_{C_j} \sin \eta_{low} - v_{T_i} \quad (23)$$

This is the empirically determined minimum lead angle for our vehicle.

To sum up, the criteria (17), (20) and (23) need to be satisfied so that our vehicle is qualified to intercept and can enter the pool of selected weapons. Otherwise, it will be removed from the pool.

6. Plan Generation Process

Based on the optimization method and mathematical models in previous sections, the specific steps to generate the air defense interception plan are as follows.

- (1) Enter the data of the enemy and our weapons.
- (2) Intercept ability judgement based on enemy and our weapon parameters, and screening of our interceptor platform weapon pool with intercept ability for each enemy weapon.
- (3) Grid division. Construct a two-dimensional space with the vertical axis coordinates as our weapon number and the horizontal axis coordinates as the enemy weapon number. If our interceptor weapon j is assigned to intercept enemy weapon i , its grid coordinates are (i, j) .
- (4) Determine the initial value of the plan. In this study, the initial value is the single interceptor weapon with the highest probability of interception for us against enemy weapons.
- (5) Calculate the objective function and use it from the starting node to the surrounding nodes to perform A* optimize.
- (6) Determine whether the objective function is optimized; if it is, reconfigure the weapon and configure the weapon pool, update the node; if it is not, go directly to the next node.
- (7) Node ergodicity judgement. If the exploration completes the square grid region consisting of region $(0, 0)$ $(i, 0)$ $(0, j)$ (i, j) , then the node search for optimization is completed and the result is output; if not, then the search for optimization continues from the current node.

The process to generate air defense interception plan is shown as Figure 11.

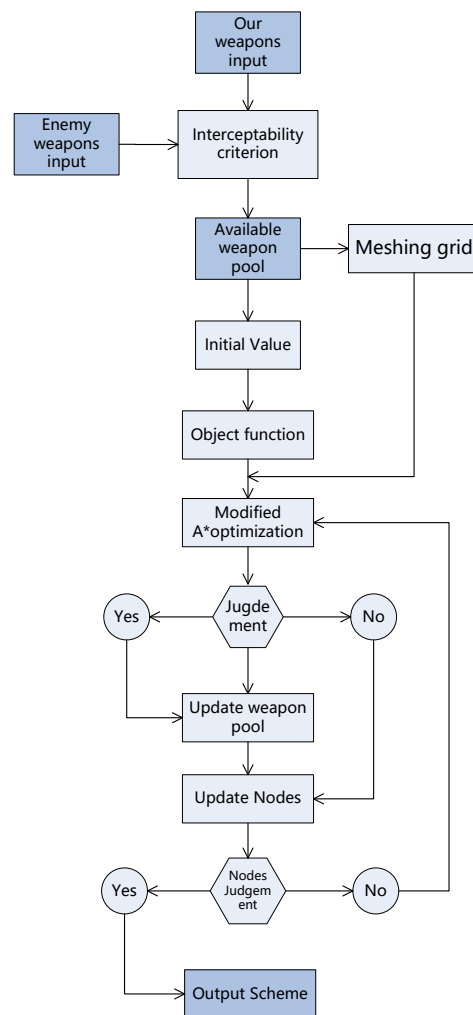


Figure 11. The flow chart of modified A* optimization algorithm.

7. Numerical Simulations

In order to demonstrate the superiority of the proposed method in this study, the standard A* optimization algorithm, ant optimization algorithm and sequence allocation algorithm are compared.

Firstly, set interception scene parameters. Assume there are 10 enemy missiles and their initial states are as Table 1.

Table 1. Parameters of enemy weapons.

	Position (km)	Maximum Velocity (km)	Value	Overload
1	(−200, 0, 0)	0.8	1	2 g
2	(−180, 0, 0)	0.8	1	2 g
3	(−400, 0, 0)	1	0.8	1 g
4	(−250, 0, 0)	1	0.8	1 g
5	(−300, 0, 0)	1.2	1	1.5 g
6	(−350, 0, 0)	1.2	1	1.5 g
7	(−100, 0, 5)	1	0.7	1 g
8	(−80, 0, 5)	1	0.7	1 g
9	(−100, 0, −5)	2	0.6	2 g
10	(−100, 0, −5)	2	0.6	2 g

Set our interception weapons parameters as Table 2.

Table 2. Parameters of interception weapons.

	Position (km)	Maximum Velocity (km)	Cost	Overload	Possibility to Intercept
1	(10, 0, 0)	0.8	1	2 g	0.8
2	(10, 0, 0)	0.8	1	2 g	0.8
3	(400, 0, 0)	1	0.8	1 g	0.9
4	(250, 0, 0)	1	0.8	1 g	0.9
5	(300, 0, 0)	1.2	1	1.5 g	0.7
6	(350, 0, 0)	1.2	1	1.5 g	0.7
7	(100, 0, 5)	1	0.7	1 g	0.8
8	(80, 0, 5)	1	0.7	1 g	0.8
9	(100, 0, 10)	2	0.6	2 g	0.8
10	(100, 0, 10)	2	0.6	2 g	0.7
11	(100, 0, 10)	1.5	1	1.5 g	0.7
12	(50, 50, 0)	1.5	1	1.5 g	0.8
13	(60, −60, 5)	2	1.5	2 g	0.9
14	(5, 5, 0)	2	1.5	2 g	0.9
15	(20, 20, 10)	1.5	1.5	1.5 g	0.7
16	(100, −50, 0)	2	1	1 g	0.6
17	(50, −100, 0)	2	1	1 g	0.8
18	(20, −20, 5)	1.5	1.2	1.2 g	0.8
19	(−20, 20, 5)	1.5	1.2	1.2 g	0.8
20	(0, 0, 0)	2	1.5	2 g	0.8

Noting that the possibility value parameter l partly determines the algorithm performance, hence, the simulation is made under 10 cases, i.e., $l = 0.1, 0.2, \dots, 1$, and the simulation results are given as follows.

Remark: Noting that there are many enemy weapons and interception weapons (total amount is 30 and the amount involves interception is more than 25), it is difficult to demonstrate all the interception process and, noting that the focus of this paper is the scheme generation algorithm, it is also meaningless to demonstrate all the interception process, hence, in this paper only the result of air-defence is demonstrated. Moreover, the X-axis of following figures is the possibility value parameter l (since the simulation is conducted by different possibility value parameter which describes the interception resolution strength), and its Y-axis is the value of the figure title (such as Figure 1 its Y-axis is the value of optimizations function).

Based on Figure 12, it could be found that the modified A* algorithm has the best object function under most conditions, which means the proposed has the best optimization for the plan generation algorithm. Based on Figure 13, it could be found that the proposed method could intercept all incoming enemy weapons for four conditions which performs best among all methods. In addition, it could be found that if the number of intercepted weapons is the core concern, we need to set larger possibility parameter l . Based on Figure 14, it could be found that the cost of our weapons is relatively low comparing with other methods (only bit larger than sequence allocation algorithm). Based on Figure 15, it could be found that the plan generation time for the proposed method is also relatively low among all methods. The performance of all methods could be concluded as Table 3.

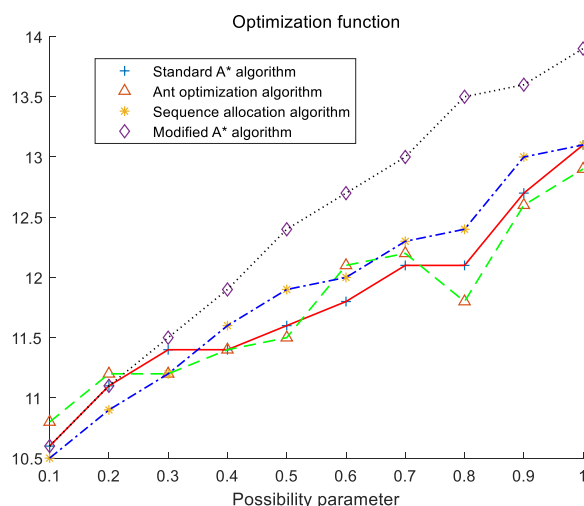


Figure 12. Curve of optimization function of different algorithms.

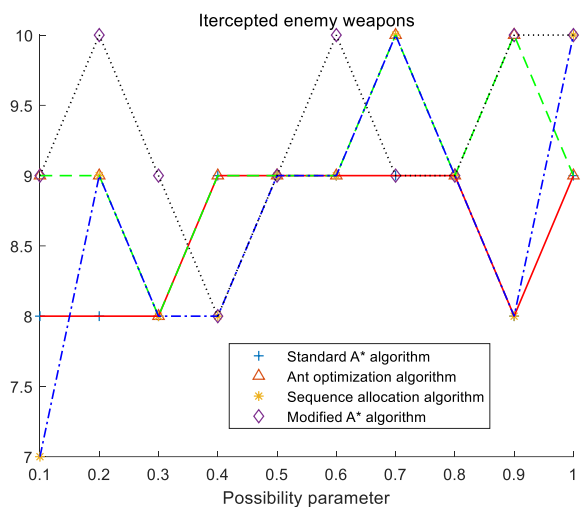


Figure 13. Curve of intercepted enemy weapons of different algorithms.

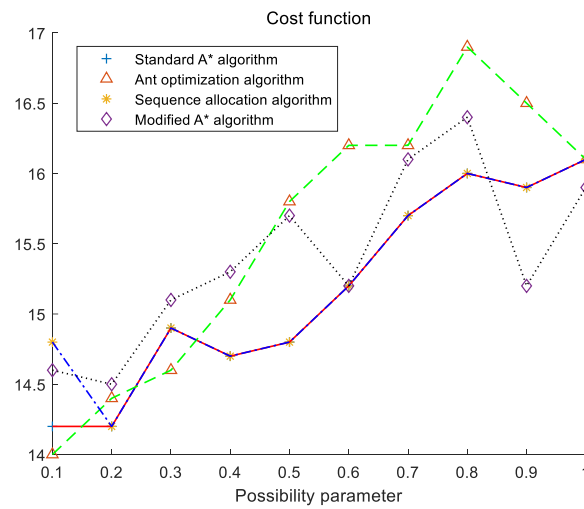


Figure 14. Curve of cost function of different algorithms.

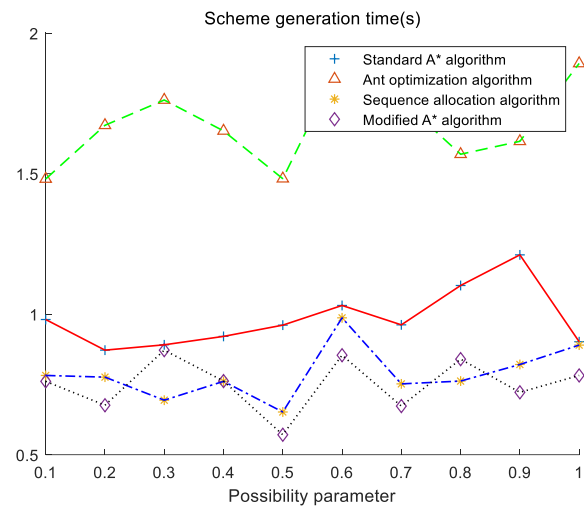


Figure 15. Curve of scheme generation time of different algorithms.

Table 3. Performance comparison of different algorithms.

	Optimization Performance	Cost	Scheme Generation Time	Complexity
Standard A* algorithm	Medium (average 12.1)	Medium (average 15.3)	Medium (average 0.62 s)	Medium
Ant optimization algorithm	Medium (average 12.4)	Low (average 13.2)	High (average 1.69 s)	Complex
Sequence optimization	Low (average 11.6)	High (average 16.7)	Low (average 0.71 s)	Simple
Modified A* algorithm	Good (average 13.3)	Low (average 13.9)	Medium (average 0.75 s)	Medium

Based on the simulation results, it could be found that the proposed method has the best optimization performance and the lowest cost. Meanwhile, the plan generation time is largely reduced compared with the standard A* algorithm and the structure is relatively simple. This demonstrates the effectiveness and superiority of the proposed method in this study.

8. Conclusions

In this study, the standard A* algorithm is modified for air defense interception plan generation. The proposed methods and the simulation results demonstrate the following conclusions. (1) By meshing the grid properly, the convergence rate of optimization algorithm is improved, and the sugar diffusion method is an effective method for meshing grid. The proposed method has better convergence rate (improved more than 30%) by implementing this method and it is insightful for other optimization algorithms. (2) The optimization direction is another key issue for optimization methods, and in this study, the direction expansion and secondary redundancy clear method could help the optimization process. This property is proved by numerical simulation results. (3) The air defense interception mission is very sensitive to reaction time, which corresponds to system convergence rate and algorithm complexity. The structure of the proposed method is still relatively complex and this still needs further improvements in future research; (4) Comparing with standard optimization methods under air-defence background, the proposed modified A* algorithm has better optimization performance (optimization function and cost function) and lower convergence rate compared with the standard A* algorithm, hence, it could be concluded that the proposed method is an improvement, compared to the standard A* algorithm.

In addition, it is also worth noting that although a optimization method is proposed in this paper, a hardware-in-the-loop (HIL) simulation, or half hardware-in-the-loop simulation, is not conducted. The reason is the high cost of hardware-in-the-loop simulation of anti-missile experiments, and how to conduct the HIL simulation is the next focus of future work.

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

Funding: This research was funded by National Natural Science Foundation of China grant numbers 61903289, 62003375, 62103452, 62073102 and 61903025 and the Fundamental Research Funds for the Central Universities grant number FRF-IDRY-20-013.

Data Availability Statement: Not applicable.

Acknowledgments: The authors greatly appreciate the above financial support. The authors would also like to thank the associate editor and reviewers for their valuable comments and constructive suggestions that helped to improve the paper significantly.

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

References

1. Hughes, T.L.; McFarland, M.B. Integrated missile guidance law and autopilot design using linear optimal control. In Proceedings of the AIAA Guidance, Navigation, and Control Conference, AIAA, Denver, CO, USA, 14–17 August 2000.
2. Liu, J.; Wang, Y.; Nie, C. Some problems about anti-TBM disposition. *Syst. Eng. Electron.* **2001**, *3*, 66–68.
3. Bai, H.; Zhang, D.; Wang, Y. Effective deployment area of the surface to air missile weapon system intercepting tactical ballistic missile. *Mod. Def. Technol.* **2003**, *6*, 4–27.
4. Liu, J. Optimization of anti-TBM disposition. *Syst. Eng. Theory Pract.* **2002**, *2*, 123–126.
5. Liu, M.; Li, W.; Wang, Y.; Liu, Y. Optimization of the regional air defense disposition based on genetic algorithms. *Syst. Eng. Electron.* **2003**, *2*, 191–193.
6. Wang, J.; Lou, S.; Wang, Y.; Wu, F. Quantitative analysis of deployment distance between fire units based on the composite disposition of the air defense missiles. *Syst. Eng. Theory Pract.* **2006**, *2*, 263–265.
7. Chang, C.; Lin, D.; Qi, Z.; Wang, H. Study on the optimal terminal guidance law with interception and impact angle. *Trans. Beijing Inst. Technol.* **2009**, *29*, 233–236, 239.
8. McGrew, J.S.; How, J.P.; Williams, B. Air-combat strategy using approximate dynamic programming. *J. Guid. Control. Dyn.* **2010**, *33*, 1641–1654. [[CrossRef](#)]
9. Wang, J.; Gao, X.; Shu, P. Operational deployment model of ADM anti-BM in terminal phase. *Mod. Def. Technol.* **2011**, *39*, 22–28 + 34.

10. Liu, H.; Liu, M. Study of intercept dynamics modeling method based on optimal control. *New Technol. New Process* **2012**, *5*, 15–18.
11. Dong, F.; Lei, H.; Li, J.; Shao, L.; Hu, X. Design of integrated adaptive optimal sliding-mode guidance and control for interceptor. *J. Astronaut.* **2013**, *34*, 1456–1461.
12. Wang, C.; Liu, M.; Song, J. Optimal deploy of terminal segment missile defense weapon. *Mod. Def. Technol.* **2015**, *43*, 87–92.
13. Liu, Y.; Zhang, Z.; Huang, Z. Deployment of midcourse anti-missile weapon based on intercepting depth. *J. Command. Control.* **2017**, *3*, 8.
14. Yang, H.; Wang, D.; Lv, R. An optimal head-on guidance law for integrated guidance and control of underwater interceptor. *J. Unmanned Undersea Syst.* **2018**, *26*, 228–233.
15. Yang, Q.; Zhang, J.; Shi, G. Modeling of UAV path planning based on IMM under POMDP framework. *J. Syst. Eng. Electron.* **2019**, *30*, 115–124. [[CrossRef](#)]
16. Li, H.C.; Zhao, Q.S.; Sun, J.B.; Xia, B.Y.; Ding, J.Y. Modeling and Parameter Optimization of Anti-Missile System Combat Network. In Proceedings of the 2021 7th International Conference on Computing and Artificial Intelligence, Tianjin, China, 23–26 April 2021; pp. 314–320.
17. Zhang, J.Q.; Li, K.; Li, Y.; Li, J.L. An Approach to Target Reselection for Anti-Ship Missile against Centroid Jamming with Accurate Tracking Information. *Int. J. Pattern Recognit. Artif. Intell.* **2021**, *35*, 2150032. [[CrossRef](#)]
18. Liu, G.; An, Z.B.; Lao, S.Y.; Li, W. Firepower distribution method of anti-ship missile based on coupled path planning. *J. Syst. Eng. Electron.* **2022**, *4*, 1010–1024. [[CrossRef](#)]
19. Witold, B. Tuning of a Linear-Quadratic Stabilization System for an Anti-Aircraft Missile. *Aerospace* **2021**, *8*, 48. [[CrossRef](#)]
20. Huang, Q.; Zhang, Y.S.; Zhang, B.Z.; OuYang, S.J.; Qi, X.G.; Liu, X.X.; Zhai, D.B. Emerging SEM equipment system combat capability assessment method. *Procedia Comput. Sci.* **2021**, *183*, 545–550. [[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.