symmetry

MDPI

Article

A Police Booth Planning Method Based on Wolf Pack Optimization Algorithm Using AAF and DGSS

Dongxing Wang [1,2], Zhishu Sun [1,*] and Fangbo Wu [1]

[1] Department of Computer and Information Security Management, Fujian Police College, Fuzhou 350007, China; wangdongxing85@163.com (D.W.); wufangbo1125@163.com (F.W.)
[2] School of Intelligence Science and Technology, University of Science and Technology Beijing, Beijing 100083, China
* Correspondence: zhishusun@fjpsc.edu.cn

Abstract: Efficient police booth deployment is vital for optimizing law enforcement and maintaining public safety. This paper tackles two key challenges in current solutions: insufficient coverage and redundant overlaps. First, this article introduces the Simultaneous Optimization for Max Coverage and Min Overlap model (SOOM-MCMO), which formalizes the dual objectives into a unified optimization framework. Second, to address limitations of traditional wolf pack algorithm, this article presents the Adaptive-Approaching Framework with Dynamic-Grid-Siege Wolf Pack Algorithm (AAF-DGS-WPOA). This technique dynamically adjusts searcher populations using spatial symmetry and hybrid optimization strategies, thereby enhancing coverage precision and reducing computational costs. Our proposed method (AAF-DGS-WPOA-based Police Booth Planning Method, PBPM-AAFDGS-WPOA) was evaluated on 20 public datasets as well as SOOM-MCMO. Results showed 15–30% coverage improvement and a 16.65% runtime reduction versus popular benchmarks like PSO, GA, and WDX_WPOA. The improved wolf pack algorithm also outperformed traditional approaches in coverage sufficiency, resource efficiency, and system responsiveness. This work advances practical methods for police booth planning, achieving higher social security outputs with lower resource investment.

Keywords: police booth planning; swarm intelligence; wolf pack optimization; public security

check for updates

Academic Editor: Theodore E. Simos

Received: 3 April 2025
Revised: 29 April 2025
Accepted: 29 April 2025
Published: 30 April 2025

Citation: Wang, D.; Sun, Z.; Wu, F. A Police Booth Planning Method Based on Wolf Pack Optimization Algorithm Using AAF and DGSS. *Symmetry* **2025**, *17*, 692. https://doi.org/10.3390/sym17050692

Copyright: © 2025 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

As the frontline facility for maintaining social order, the importance of police booths is self-evident. The planning and establishment of police booths constitute a crucial aspect of modern public security management, aiming to enhance law enforcement efficiency, swiftly respond to emergencies, and strengthen patrols and management in surrounding areas. A scientifically reasonable plan for police booths can significantly improve the utilization of police resources, effectively reduce crime rates, and enhance the public's sense of security [1].

Although police booths play a crucial role in social security management, there are still some issues that need to be addressed in the current planning process. Insufficient coverage is a major concern. Due to limited police presence in some areas, the distribution of police booths has not fully covered all high-risk areas, resulting in the emergence of public security blind spots, making these areas prone to becoming hotbeds for criminal activities. The high overlap rate of police booth distribution is also a significant issue that

cannot be overlooked. This overlap may lead to the waste of police force and the reduction in efficiency, thus affecting the effect of the overall public security management [2].

Many scholars have conducted extensive research on the optimization of site selection for public safety facilities such as police booths and fire stations, with the aim of improving facility coverage and service efficiency. Jiang et al. took the central urban area of Lanzhou as an example and proposed a multi-criteria spatial optimization model based on multi-source geographic spatial data. By deleting exclusion zones, evaluating potential crime risks, and using analytic hierarchy process, the layout of police booths was optimized, significantly reducing the overlap of service areas and improving coverage [3]. Wang [4] and Chen et al. [5] utilized point-of-interest (POI) data and multi-time traffic condition data to enable the newly configured fire station to respond to fire events in a shorter period of time. Yang et al. proposed a fire station layout method based on comprehensive forest fire risk assessment, which meets the demand with acceptable construction costs and good coverage by combining the minimization of facility point model and maximum coverage model [6]. De Domingo et al. utilized historical data and existing location information to significantly shorten response time through the Floyd–Warshall dynamic programming algorithm and k-means clustering technique [7]. KhoshAmooz et al. used fuzzy logic methods, combined with linear membership functions and gamma fuzzy operations, to determine the optimal location for new facilities [8]. Tian [9] and Guo et al. [10] explored optimization techniques for cross regional collaborative deployment and significantly improved the coverage of high-risk and medium- to low-risk areas through coverage maximization models and facility point minimization models. Shahparvari et al. utilized Geographic Information Systems (GIS) and location allocation models to improve emergency response speed [11]. Ming et al. proposed a distributed robust optimization model (DRM) for solving the uncertainty problem in fire station location selection. By minimizing the expected total cost in the worst-case scenario, the location, quantity, and demand allocation schemes were optimized [12]. Although these studies have made some progress in improving the coverage and response efficiency of public safety facilities, there are still some limitations, such as high algorithm complexity and long calculation time, which need to be further optimized and improved in practical applications.

In recent years, the wolf pack optimization algorithm (WPOA) has been widely studied and applied as a biomimetic swarm intelligence optimization method in multiple fields. Scholars have proposed various improvement strategies to address the bottlenecks of WPOA, such as insufficient local search capability, slow convergence speed, and poor performance in handling high-dimensional complex problems. For example, Yanchao et al. proposed an adaptive shrinkage grid search chaotic wolf optimization algorithm based on adaptive difference update, which significantly improves global optimization accuracy and convergence speed by adaptively adjusting the search step size [13]. Lu et al. proposed a weighted decision quantum wolf pack evolutionary algorithm based on fuzzy control, which enhances the local search ability and global convergence of the algorithm by optimizing the initial position selection and the search direction operator of the leading wolf [14]. Zhang [15] and Jiang et al. [16] achieved efficient path planning for unmanned aerial vehicles in complex environments by combining the wolf pack optimization algorithm with genetic algorithm or simulated annealing algorithm. Lai et al. proposed a discrete wolf pack algorithm that improves the effectiveness and global search capability of the algorithm by swapping individual code bits and introducing loop operations [17]. Sahu et al. proposed an improved wolf pack algorithm based on the fluorescent guidance mechanism in the artificial firefly swarm optimization algorithm. By improving the search mode and introducing a dynamic mutation strategy, it effectively solves the problem of 3D trajectory planning for unmanned aerial vehicles [18]. Zhu [19]

and Wang et al. [20] proposed a chaotic interference wolf pack algorithm for multimodal multi-objective optimization problems, which effectively prevents local optima in high-dimensional optimization problems. Chen et al. proposed the OGL-WPA algorithm based on opponent learning and Levy flight strategy, which improved the algorithm's global and local search capabilities [21]. Jin et al. combined the wolf pack algorithm with radial basis function neural networks and proposed the WCA-RBF neural network model, which effectively improved the accuracy of the algorithm [22]. These improvement measures not only enrich the research on swarm intelligence optimization algorithms in theory, but also demonstrate excellent performance in practical applications, providing new ideas and methods for solving complex optimization problems.

Therefore, based on the effective utilization of the wolf pack optimization algorithm, this paper proposes a police booth planning method based on an improved wolf pack optimization algorithm (AAF-DGSS-WPOA), which uses strategies of adaptive-approaching and dynamic-grid-siege to solve the planning problem of police booths (PBPM-AAF-DGSS-WPOA). The rest of this article is organized as follows: Section 2 reviews relevant work on swarm intelligence algorithms and their applications. Section 3 provides a detailed introduction to the proposed PBPM-AAF-DGSS-WPOA algorithm. Section 4 describes the experimental setup and benchmark results. Section 5 discusses statistical analysis and limitations and summarizes the conclusions.

2. Related Works

2.1. Swarm Intelligence Optimization

Swarm intelligence optimization algorithms are a category of computational intelligence technologies rooted in group behavior found in nature. They aim to discover optimal solutions by simulating the interactions among individual organisms. Common swarm intelligence optimization algorithms include particle swarm optimization (PSO) [23] and ant colony optimization (ACO) [24], among others. These algorithms have demonstrated exceptional performance in addressing diverse complex optimization challenges and are extensively employed in various fields, including engineering optimization, scheduling issues, and path planning. The strengths of swarm intelligence optimization algorithms lie in their simplicity, ease of implementation, high computational efficiency, and robust adaptability to problem models. These attributes have garnered increasing attention from researchers. However, there are still some shortcomings, such as parameter sensitivity and susceptibility to local optima, which require further improvement.

2.2. Wolf Pack Optimization Algorithm

Among the swarm intelligence optimization algorithms that have emerged in recent years, the WPOA stands as a typical representative. First proposed by Yang et al., this algorithm simulates the predation behavior and prey allocation of wolf packs, abstracting three intelligent behaviors: migration, summon-raid, and siege, as well as the "winner takes all" rule for selecting the leading wolf and the "survival of the strong" mechanism for updating wolf packs. With the continuous improvement of scholars, the performance of the algorithm has become more efficient, and the improvement of WDX-WPOA is relatively large [25]. The specific steps are as follows:

Initialization: The algorithm first needs to initialize a series of parameters, such as the total number of wolf packs numW, the dimension of the solution space D, the number of the wolfs that compete for the leader q, the maximum number of iterations T, the spatial scope of the solution [$range_{min}$, $range_{max}$], the initial value step_a for migration, the initial value step_b for summon determined by the spatial scope of the solution and the number

of wolf packs, the initial value step_c_max and step_c_min for siege. Then, the position of each wolf is determined by Equation (1).

$$\begin{cases} x_i = (x_{i1}, x_{i2}, \ldots, x_{id}, \ldots, x_{iD}) \ (i = 1, 2, \ldots, N; d = 1, 2, \ldots, D) \\ x_{id} = range_{min} + rand(0,1) \times (range_{max} - range_{min}) \ (i = 1, 2, ..., N; d = 1, 2, ..., D) \end{cases} \tag{1}$$

where x_i represents the number of each wolf, x_{id} represents the position of each wolf, and $rand(0,1)$ is a function that generates random numbers in the interval [0,1].

Migration: During this process, each wolf will search for a target in its nearby space according to the set rules. Firstly, generate an adaptive grid with D dimensions to reflect the local domain space of the current position. Each grid includes $(2 * k + 1)^H$ points generated by Equation (2), where k represents the number of points taken in the same direction for each dimension, $step_a$ represents the migration step size, $x_{i\text{-}new}$ represents the new position of each wolf. Then, calculate the fitness value of each wolf based on the fitness function and continuously compare them. At the end of the migration, the wolf with the best fitness will successfully compete as the leading wolf.

$$\begin{cases} [k, x_{id-new}] = x_{id} + step_a * k \ (k = -K, \ldots, 0, \ldots, K; i = 1, 2, \ldots, N; d = 1, 2, \ldots D) \\ x_{i-new} = \{[k, x_{i1-new}], [k, x_{i2-new}], \ldots, [k, x_{iD-new}]\} \ (i = 1, 2, ..., N; k = -K, ..., 0, ..., K) \end{cases} \tag{2}$$

Summon-Raid: After producing the leading wolf, the wolf pack will summon other wolves to quickly advance and participate in hunting to increase their hunting power. In this step, WDX-WPOA follows the strategy of adaptive raiding to bring half of the wolves with poor fitness values around the leading wolf, as shown in Equation (3), while the other half follows the original strategy.

$$Population \ = \ Generation \ (1, D, ADS, ADS * (-1)) + x_{ld} \tag{3}$$

where *Population* is an array of newly generated wolf positions, *Generation* is a function that generates wolves based on given parameters, D represents the dimension of the solution space, x_{ld} represents the position of the leading wolf, and ADS is the size of the adaptive raid distribution, calculated by Equation (4).

$$\begin{cases} ADS = (range_{max} - range_{min})/numW \\ OrderId = sort(pop_wolf) \end{cases} \tag{4}$$

where $range_{max}$ and $range_{min}$ represent the solution space range, $numW$ represents the number of wolf packs, *pop_wolf* is an array that records all wolf fitness values, and *sort* is a function that arranges the array from small to large. The remaining wolves approach the leading wolf by taking the median distance from the current position to the leading wolf or taking the opposite position. The specific formula is shown in Equation (5).

$$\begin{cases} x_{id-opposition} = 2 * x_{ld} - x_{id} \ (i = 1, 2 \ldots N; d = 1, 2 \ldots D) \\ x_{i-m_d} = \frac{x_{id} + x_{ld}}{2} \\ x_i = Bestfitness\left(\left[x_{i_1}, ..., x_{i_D}\right], ... \left[x_{i-m_1}, ..., x_{i-m_D}\right]\right) \ (i = 1, 2 \ldots Q; d = 1, 2 \ldots D) \end{cases} \tag{5}$$

where $x_{id\text{-}opposition}$ represents the position of the wolf on the opposite side, x_{ld} represents the position of the leading wolf, x_{id} represents the current position of the wolf, $x_{i\text{-}m_d}$ represents the midpoint position between the current wolf and the leading wolf, and *Bestfitness* represents a fitness function.

Siege: After the process of summon-raid, the wolf pack will be distributed around the leading wolf, and each wolf will move around according to the siege step size *step_c_max* and *step_c_min*. The specific equation is shown in Equation (6).

$$\begin{cases} [k, x_{id-new}] = x_{id} + step_c * k \ \ (k = -K, \ldots, 0, \ldots, K; i = 1, 2, \ldots, N; d = 1, 2, \ldots D) \\ x_{i-new} = \{[k, x_{i1-new}], [k, x_{i2-new}], \ldots, [k, x_{iD-new}]\} \ (i = 1, 2, \ldots, N; k = -K, \ldots, 0, \ldots, K) \end{cases} \quad (6)$$

where k represents the number of points taken in the same direction for each dimension, *step_c* represents the siege step size, and $x_{i\text{-}new}$ represents the new position of each wolf.

Regeneration: Using the "winner takes all" generation rule of the wolf pack and the "survival of the strong" wolf pack update mechanism, after each iteration, the strong performing wolves are retained in the wolf pack, the poor performing wolves are eliminated, and then a certain number of artificial wolves are randomly added through the set rules for the next iteration. Evaluate through the fitness function until the conditions are met or the maximum number of iterations is reached. Finally, the leading wolf is the global optimal solution we are looking for.

2.3. Benchmark Functions

In order to evaluate the optimization capability of the algorithm proposed in this article, we conducted comparative experiments with various existing optimization algorithms. In the experimental design, we selected a set of classic test functions and datasets, as shown in Table 1. These test functions and datasets aim to comprehensively evaluate the performance and advantages of our proposed algorithm from different perspectives. Firstly, we selected some test functions with multimodal features. These functions have strong interference and high complexity in the search space. This can help us test the algorithm's global search ability and robustness when facing complex situations. The characteristics of multimodal functions can reveal whether algorithms can effectively escape from local optima and ultimately find global optima during the optimization process. Secondly, in order to test the performance of the algorithm in terms of convergence speed and accuracy, we selected smooth, continuous, unimodal, and convex functions, which typically have characteristics similar to bowl shaped structures. Under such testing conditions, we can observe whether the algorithm quickly and accurately converges to the optimal solution, thereby evaluating its efficiency in handling simple optimization problems. Finally, to evaluate the performance of the algorithm in handling dynamic changes and local optimization, we also included some very smooth and relatively gentle functions. These functions can simulate some dynamic environments in practical applications and test the performance of algorithms in the face of slowly changing optimization objectives. Through these diverse testing functions, we can systematically measure the optimization performance of the proposed algorithm under different challenges.

Table 1. Benchmark Functions.

Order	Function	Expression	Dim	Range	Optim
1	Ackley	$F1 = -20exp\left(-0.2\sqrt{\frac{1}{2}\sum_{i=1}^{2} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + \exp(1)$	2	[−32.768,32.768]	Min f = 0
2	Bukin6	$F2 = 100\sqrt{\lvert x_2 - 0.01x_1^2\rvert} + 0.01\lvert x_1 + 10\rvert$	2	[−15,3]	Min f = 0
3	Drop-Wave	$F3 = -\left(1 + \cos\left(12\sqrt{x_1^2 + x_2^2}\right)\right) / 0.5\left(x_1^2 + x_2^2\right) + 2$	2	[−5.12,5.12]	Min f = −1
4	Eggholder	$F4 = -(x_2 + 47)\sin\left(\sqrt{\lvert x_2 + \frac{x_1}{2} + 47\rvert}\right) - x_1\sin\left(\sqrt{\lvert x_1 - (x_2 + 47)\rvert}\right)$	2	[−512,512]	Min f = −959.6407
5	Griewank	$F5 = \sum_{i=1}^{2}\frac{x_i^2}{4000} - \prod_{i=1}^{2} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	2	[−600,600]	Min f = 0

Table 1. *Cont.*

Order	Function	Expression	Dim	Range	Optim
6	Levy	$F6 = \sin^2(\pi w_1) + \sum_{i=1}^{2-1} (w_i - 1)^2 \left[1 + 10\sin^2(\pi w_i + 1)\right] + (w_2 - 1)^2 \left[1 + \sin^2(2\pi w_2)\right]$, where $w_i = 1 + \frac{x_i - 1}{4}$, for all $i = 1, 2$	2	[−10,10]	Min f = 0
7	Levy13	$F7 = sin^2(3\pi x_1) + \left(x_1 - 1\right)^2 \left[1 + sin^2(3\pi x_2)\right] + \left(x_2 - 1\right)^2 \left[1 + sin^2(2\pi x_2)\right]$	2	[−10,10]	Min f = 0
8	Cross-In-Tray	F8 = $-0.0001 * (\text{abs}(\sin(x_1)\sin(x_2) exp(\text{abs}(100 - \sqrt{x_1^2 + x_2^2}/\pi)))+1)^{0.1}$	2	[−10,10]	Min f = −2.06261
9	Schaffer2	$F9 = 0.5 + \frac{\sin^2\left(x_1^2 - x_2^2\right) - 0.5}{\left[1 + 0.001\left(x_1^2 + x_2^2\right)\right]^2}$	2	[−100,100]	Min f = 0
10	Bohachevsky1	$F10 = \mathbf{x}_1^2 + 2\mathbf{x}_2^2 - 0.3 * \cos(3\pi \mathbf{x}_1) - 0.4 * \cos(4\pi \mathbf{x}_2) + 0.7$	2	[−100,100]	Min f = 0
11	Perm0-d-β	$F11 = \sum_{i=1}^{2} \left(\sum_{j=1}^{2} (j + \beta) \left(x_j^i - 1/\text{j}^\text{i} \right) \right)^2$	2	[−2,2]	Min f = 0
12	Rotated Hyper-Ellipsoid	$F12 = \sum_{i=1}^{2} \sum_{j=1}^{i} x_j^2$	2	[−65.536,65.536]	Min f = 0
13	Sum Squares	$F13 = \sum_{i=1}^{2} i x_i^2$	2	[−10,10]	Min f = 0
14	Trid	$F14 = \sum_{i=1}^{2} (x_i - 1)^2 - \sum_{i=2}^{2} x_i x_{i-1}$	2	[−4,4]	Min f = −2
15	Booth	$F15 = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	2	[−10,10]	Min f = 0
16	Matyas	$F16 = 0.26\left(x_1^2 + x_2^2\right) - 0.48 x_1 x_2$	2	[−10,10]	Min f = 0
17	Easom	$F17 = -\cos(\text{x1}) * \cos(\text{x2}) * \exp\left[-(\text{x1} - \pi)^2 - (\text{x2} - \pi)^2\right]$	2	[−4,4]	Min f = −1
18	Eggcrate	$F18 = \mathbf{x}_1^2 + \mathbf{x}_2^2 + 25 * \left(\sin^2 \mathbf{x}_1 + \sin^2 \mathbf{x}_2\right)$	2	[−π,π]	Min f = 0
19	Bohachevsky3	$F19 = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1 + 4\pi x_2) + 0.3$	2	[−100,100]	Min f = 0
20	Bridge	$F20 = -\left(\frac{\sin\left(\sqrt{x_1^2 + x_2^2}\right)}{\sqrt{x_1^2 + x_2^2}} - 0.7129 + \exp\left(\frac{\cos(2\pi x_1) + \cos(2\pi x_2)}{2}\right) \right)$	2	[−10,10]	Min f = −3

2.4. Problem Formulation (SOOM-MCMO)

Police booth is a typical public service facility that plays an important role in enhancing the public's sense of security and deterring criminal activities. In general, without considering factors such as crowd density and public facilities, the goal of each region's layout is to fully cover urban areas and ensure that police forces can reach any emergency location within a limited time. At the same time, in order to reduce deployment costs and make the police force as balanced as possible, it is necessary to minimize unnecessary police booths and avoid the phenomenon of overly dense police booths. Therefore, this article constructs a single objective optimization model (SOOM-MCMO) with the objectives of maximizing coverage and minimizing overlap to reflect the problem of police booth planning.

Generally speaking, urban areas are abstracted as an irregular plane, but in order to facilitate computational processing and focus more on solving the main contradiction of the problem, we consider it as a two-dimensional regular plane. We assume that a maximum of N police stations can be used within a plane area of L km $\times$ L km, where N is a predetermined fixed value, and the police coverage of each police booth is a circle with a radius of R km. The position of each booth is represented by its coordinates (x_i, y_i) in a two-dimensional plane, where i = 1, 2, . . ., N. The positions of all booths can form a vector:

$$X = (x_1, y_1, x_2, y_2, \ldots, x_N, y_N) \tag{7}$$

And define (x_i, y_i) cannot exceed the boundary of the region, that is:

$$0 \le x_i \le L, 0 \le y_i \le Li \in \{1, 2, ..., n\} \tag{8}$$

where L represents the side length of the planar area. Furthermore, we define the indicator function $I(x, y)$ to indicate whether a point (x, y) is within the coverage range of any booth:

$$I(x,y) = \begin{cases} 1 & \text{if } \exists i \in \{1,2,\ldots,N\}, \sqrt{(x-x_i)^2+(y-y_i)^2} \leq R \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

The actual coverage area C can be expressed as:

$$C = \iint_{\Omega} I(x,y)dxdy \tag{10}$$

where Ω represents the entire $L \times L$ urban area. The total area P covered by multiple booths in the urban area is represented as:

$$P = \iint_{\Omega} (\sum_{i=1}^{N} I(x,y|x_i,y_i) - I(x,y))dxdy \tag{11}$$

where $\sum_{i=1}^{N} I(x,y|x_i,y_i) - I(x,y)$ represents the number of times point (x, y) is repeatedly covered. If a point is covered by multiple booths, this value is positive. If a point is not covered or only covered by one booth, this value is 0. In order to comprehensively consider the area of coverage and the number of booths, this article introduce a weight factor α, which is used to balance the relative importance of coverage area and overlap in the objective function [26]. The specific test function is as follows:

$$fitness(X) = \alpha C - (1-\alpha)P \tag{12}$$

where α represents a weight factor between [0,1], used to adjust the proportion of coverage area and overlap area in the objective function. In this problem, we pay more attention to the coverage of the area, so we set α to 0.8, which indicates that the punishment for uncovered areas is more important than the increase in coverage. Through this weight setting, the model focuses more on ensuring complete coverage of the region rather than simply minimizing coverage, ensuring that penalties for uncovered points dominate in fitness evaluation, thereby driving the algorithm to prioritize finding solutions that can provide maximum coverage. This fitness function combines the comprehensive performance between coverage and overlap. We can judge the quality of the site selection plan based on the value of the fitness function. The higher the fitness function value, the better the comprehensive performance of the plan.

Obviously, the above problem ignores factors such as actual geography and administrative limitations. (x_i, y_i) can take any value, so this is a continuous problem, just like the test function in Section 2.3. Therefore, we can use the test function in Section 2.3 to evaluate the improved wolf pack algorithm and then use it to optimize the location of police booths.

3. The Improved Method

Although WDX_WPOA has significant advantages in global search and convergence speed compared to traditional optimization algorithms, it also has some drawbacks, such as significantly longer iteration time in the application of the test function shown in Table 1. Therefore, to address this issue, this article designed the two following strategies to improve algorithm performance and then utilized the newly proposed algorithm to implement a more effective optimization site selection scheme for police booths.

3.1. Adaptive-Approaching Factor (AAF)

In optimization problems, how to efficiently generate a new set of solutions is crucial for algorithm performance. During the process of summon-raid, the traditional grid generation method involves generating new points by taking the midpoint from the current wolf to the optimal wolf. The specific equation is as follows:

$$Middle_Wolf = (Best_Wolf + Current_Wolf)/2 \tag{13}$$

where *Best_Wolf* represents the position of the leading wolf, *Current_Wolf* represents the position of the current wolf, and *Middle_Wolf* represents the position of the midpoint between the two. Although this method is very convenient in calculation, it often lacks enough flexibility and adaptability in dealing with dynamic optimization problems, which leads to insufficient exploration of search space in different iteration stages. To address this issue, we propose an improved mesh generation method by introducing an adaptive-approaching factor to better adapt to the dynamics and complexity of the problem. Specifically, the approaching factor changes dynamically according to the number of iterations, and the calculation equation is as follows:

$$App_Factor = \begin{cases} 0.5 + \frac{t}{T} \ (t \leq 300) \\ 0.7 \ (t > 300) \end{cases} \tag{14}$$

where *App_Factor* represents the adaptive-approaching factor; t represents the current iteration times; T represents the total iteration times, which is usually set to 600. When t is less than or equal to 300, that is, before iterating to half of the total number of iterations, the value of *App_Factor* gradually increases. In the later iteration, the value of *App_Factor* is fixed at 0.7. After the introduction of *App_Factor*, we no longer use the original method but use this factor to generate new points. The $Location_{Summon_Raid}$ of the new points is calculated by Equation (15).

$$Location_{Summon_Raid} = App_Factor * Best_wolf + (1 - App_Factor) * Current_Wolf \tag{15}$$

where *Best_Wolf* represents the position of the leading wolf, *Current_Wolf* represents the position of the current wolf, and *App_Factor* is the introduced factor, which is obtained from Equation (14). In this way, we can obtain the coordinates of a new point close to the optimal wolf.

In order to better demonstrate the superiority of this improvement point, we simulated the distribution of new generated points before and after the introduction of *App_Factor*. The original strategy is shown in Figure 1a,c, and the improved strategy is shown in Figure 1b,d.

By comparing these images, we can clearly see that the newly generated grid points after improvement are more dispersed. This optimization of grid point distribution can not only improve the search ability of the algorithm, but also significantly improve the convergence speed, thereby shortening the overall running time. In addition, as the number of iterations increases, the value of *App_Factor* will dynamically change, allowing the generated grid points to more flexibly reflect the relationship between the current state and the optimal solution.

The main purpose of introducing the adaptive-approaching factor (*App_Factor*) is to dynamically adjust the approaching speed of the current wolf to the leading wolf in the initial stage of the search. Specifically, the new grid point generation mechanism tends to focus on the target points, thus smoothly transitioning from the current point to the optimal point, which can significantly improve the exploration ability of the algorithm and

help it find the global optimal solution faster. In the later stages of the search, the value of *App_Factor* remains unchanged, which helps maintain the stability of the solution and the optimization ability of the algorithm, ensuring that the algorithm pays more attention to the known optimal solution and further refines the search near the current wolf to improve the accuracy and reliability of the final result. Through this adaptive strategy, the algorithm exhibits different exploration and mining behaviors in the early and later stages, not only retaining the efficiency of traditional grid generation methods, but also further improving its search accuracy and speed.

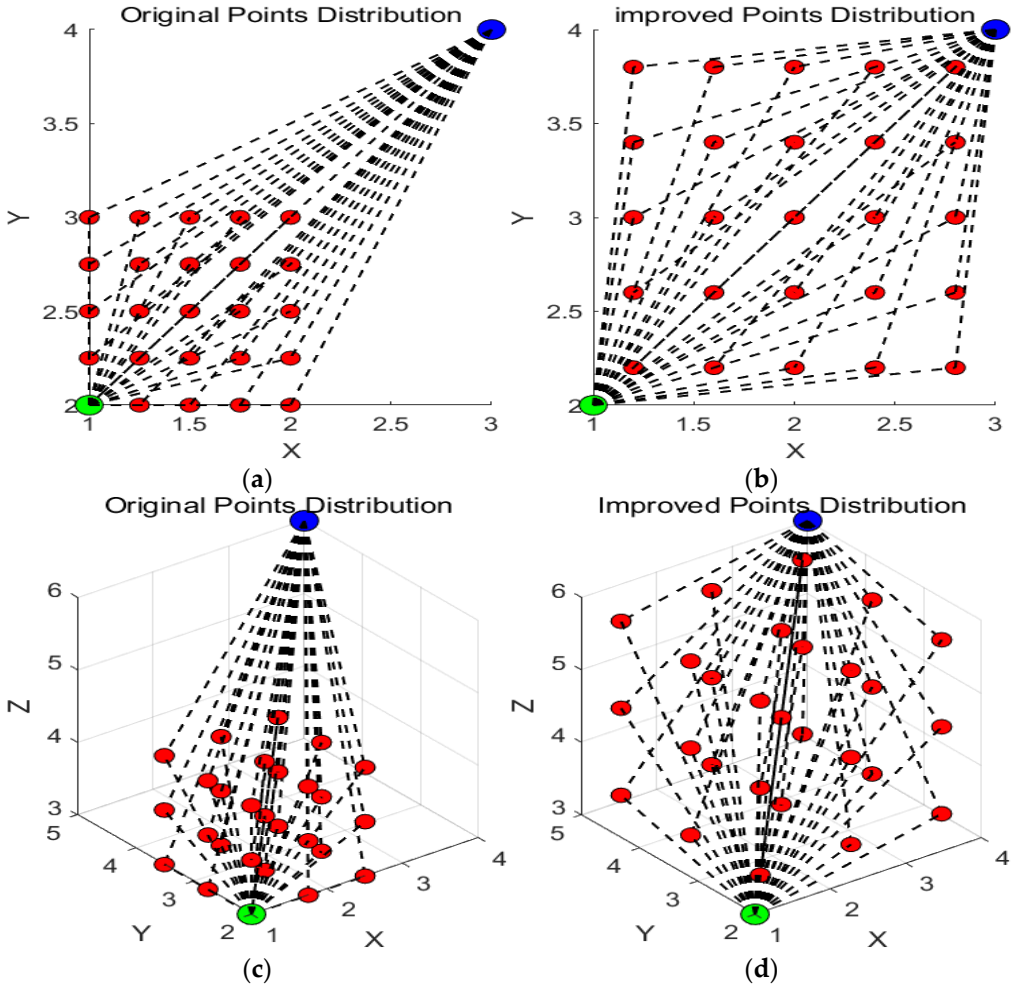


Figure 1. (**a**,**c**): about WDX-WPOA; (**b**,**d**): about AAF-DGSS-WPOA.

In order to further verify the performance of the algorithm, we compared the improved algorithm with WDX-WPOA and focused on evaluating their convergence speed under different test functions.

Figure 2 shows the three groups with obvious effects in the experiment, from which we can see that the image curves of each group of improved algorithms are smoother than other curves, and the convergence speed is faster. This shows that the introduction of *App_Factor* can not only improve the convergence speed of wolves but also enhance the algorithm's ability to find the global optimal solution to a certain extent.

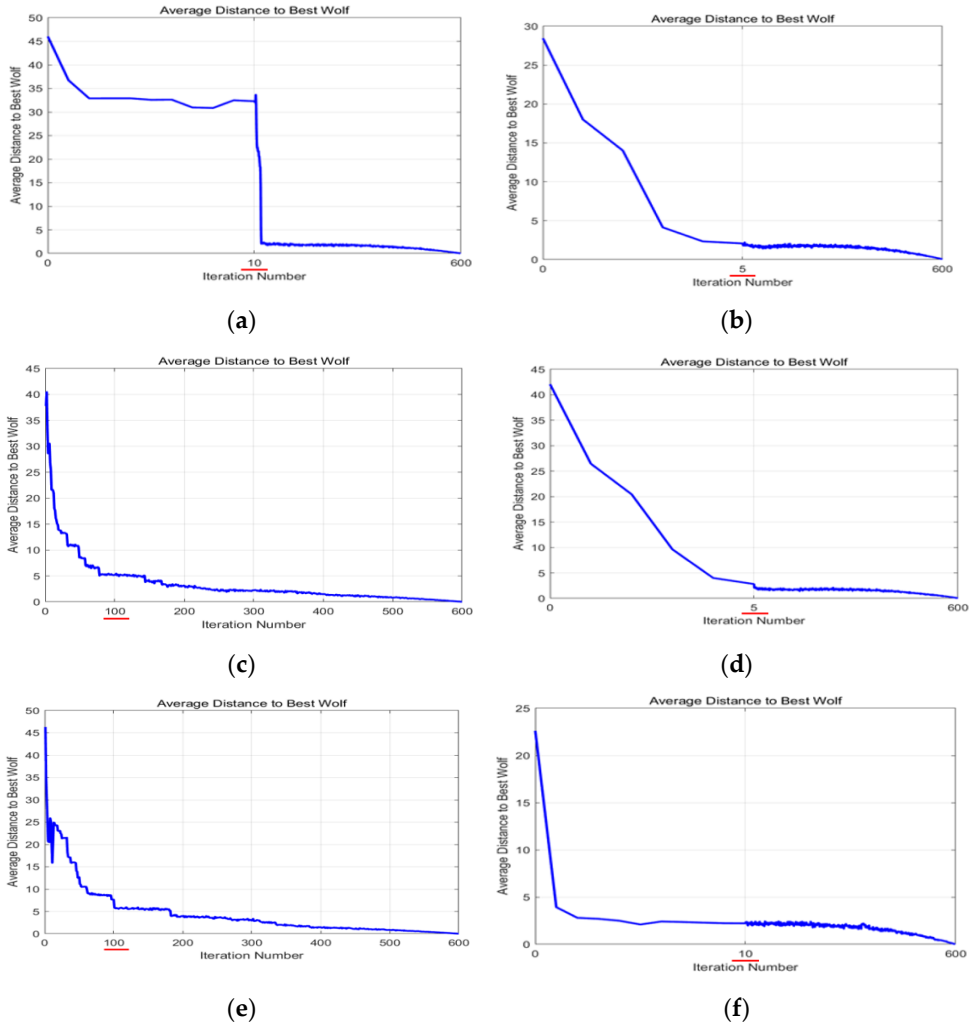


Figure 2. (**a**,**c**,**e**): about WDX-WPOA; (**b**,**d**,**f**): about AAF-DGSS-WPOA.

3.2. Dynamic-Grid-Siege Strategy (DGSS)

During the siege process of the WDX-WPOA algorithm, an adaptive grid with D dimensions is first generated to reflect the local neighborhood space of the current position, which includes $(2 \times \text{num} + 1)^{\text{D}}$ points generated by Equations (3) and (4), where num represents the number of points taken in the same direction for each dimension, usually set to 5. Among them, the step size of wolf pack siege decreases with the increase in the number of iterations, and its step size changes as shown in Equation (16).

$$step_c = step_c_min * (range_{max} - range_{min}) * exp(\left(log\left(\frac{step_c_min}{step_c_max}\right)\right) * \frac{t}{T}) \quad (16)$$

where $step_c_min$ represents the initial minimum value of the siege step size, $step_c_max$ represents the maximum value of the siege step size, $range_{max}$ represents the maximum value of the solution space, $range_{min}$ represents the minimum value of the solution space, t represents the current number of iterations, T represents the total number of iterations, and $step_c$ represents the siege step size in the actual calculation.

However, for simple prey, it only requires fewer wolf packs to besiege and capture; and the wolves tend to summon more wolves to besiege and gradually move closer to the prey to capture the prey when the prey is stronger. Based on the above principle, we abstracted the behavior. The number of iterations is the embodiment of the difficulty of finding the optimization of the fitness function, and the simple fitness function often needs

only a few iterations to find the optimal value (prey), while the complex fitness function often requires many iterations to catch the prey. Thus, as the number of iterations increases, we utilized spatial symmetry to boost wolves in each dimension, with the variation per dimension demonstrated in Equation (17).

$$new_num = num + round\left(\frac{3}{(1 + exp(0.01 * (60 - t)))}\right) - 2 \tag{17}$$

where *num* represents the number of wolves in each dimension at the beginning, t represents the current number of iterations, round is a function of rounding numbers, and *new_num* represents the number of wolves in each dimension during actual calculation.

In the original strategy, num is a fixed value set to 5, so the number of points in each direction is 11. After introducing the dynamic-grid-siege strategy, the value of *new_num* gradually increases from 3 to 6 with the increase in iterations, so the number of points in each direction changes from 9 to 13. We have drawn the distribution of points before and after the improvement. In order to make the image more intuitive, we have simplified it. Figure 3a,b is the image about WDX-WPOA, Figure 3c,d is the initial image of AAF-DGSS-WPOA, and Figure 3e,f is the final image of AAF-DGSS-WPOA.

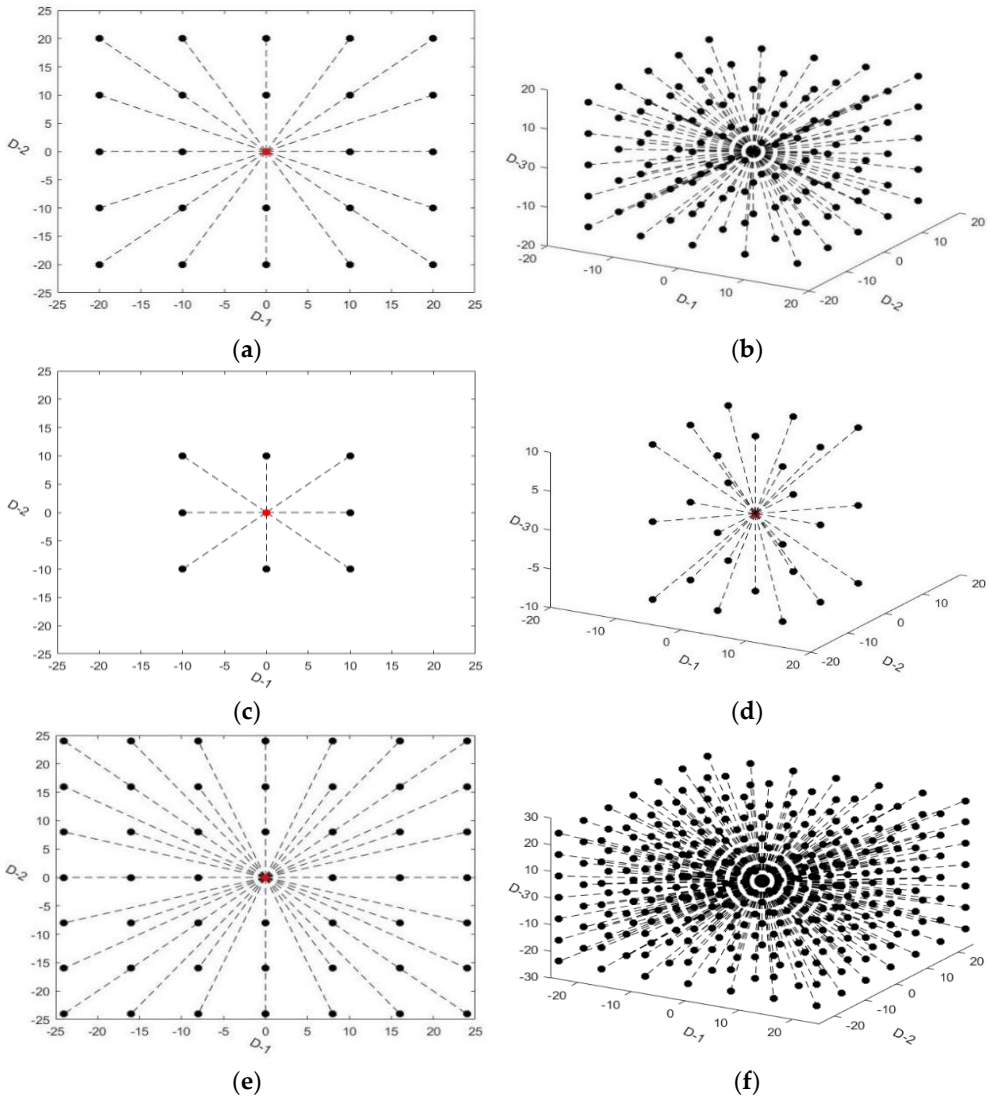


Figure 3. (**a**,**b**): about WDX-WPOA; (**c**–**f**): about AAF-DGSS-WPOA.

For the simple fitness function, only a small number of iterations are required, so the above improved method reduces a large number of unnecessary search points to save computing resources. For complex fitness functions, which require a large number of iterations, the above improved method adds more and more densely distributed search points in the later iterations to enhance the calculation accuracy of the algorithm. Therefore, the above improved method not only improves the optimization performance of complex fitness functions, but also reduces the time required for simple fitness functions.

3.3. Steps of AAF-DGSS-WPOA

Based on the introduction of adaptive-approaching factor and dynamic grip siege strategy, WDX-WPOA has been improved and AAF-DGSS-WPOA has been proposed. The following are the steps for implementing a new improved algorithm, as shown in the Figure 4.

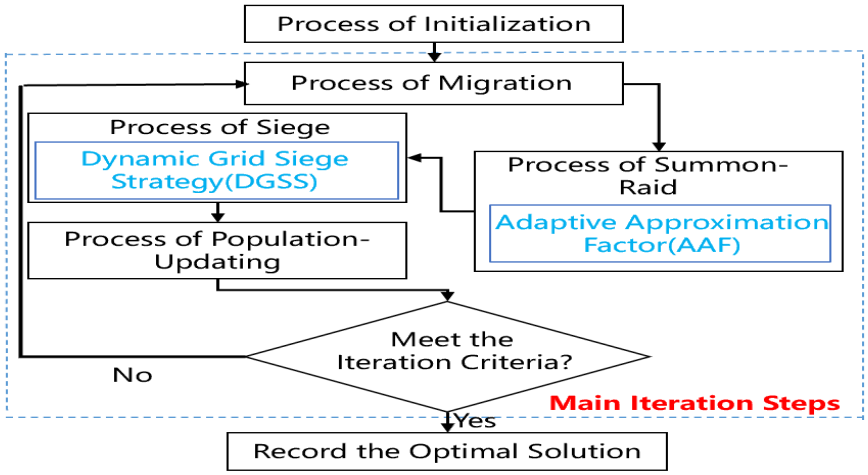


Figure 4. Flowchart about AAF-DGSS-WPOA.

Except for the introduction of AAF during the summon_raid process and the use of DGSS during the siege process, all other steps are consistent with WDX-WPOA, as detailed in Sections 3.1 and 3.2.

3.4. Steps of PBPM-AAF-DGSS-WPOA

Based on the improved algorithm (AAF-DGSS-WPOA), the specific steps of location planning for police booths are shown in Figure 5.

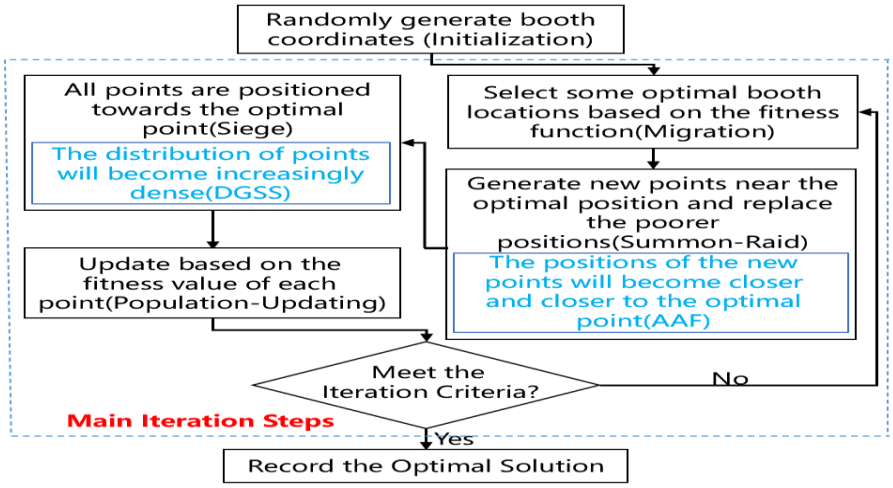


Figure 5. Flowchart about PBPM-AAF-DGSS-WPOA.

4. Mathematical Experiment

4.1. Experimental Designment

To evaluate the optimization performance of the improved algorithm, we conducted comparative experiments using three optimization algorithms: classical genetic algorithm (GA), particle swarm optimization (PSO), and WDX-WPOA. Twenty sets of test functions and datasets were used in the experiment, as shown in Table 1.

All numerical experiments were conducted on a computer equipped with the Windows 11 Home 23H2 operating system, 12th Gen Intel (R) Core (TM) i7-12700H 2.30 GHz processor, and 16 GB of memory, with an integrated development environment of MATLAB-2024a. The GA experiment was implemented by calling the toolbox in MATLAB 2017a. The PSO experiment was implemented using MATLAB's "PSOt" toolbox. The WDX-WPOA experiment was implemented according to the steps in Section 2.2, and AAF-DGSS-WPOA was ultimately implemented using MATLAB-2024a. The specific initial parameters of each algorithm are shown in Table 2.

Table 2. Initial Parameters.

Order	Algorithm Name	Configure Ration
1	GA	Crossover probability is 0.8, the mutation probability is 0.01, the max iteration T = 600.
2	PSO	Inertia weight is 0.5, the Cognitive coefficient is 1.5, the Social coefficient is 1.5, the max iteration T = 600.
3	WDX-WPOA	Initial value of search step size step_a0 = 1.5; the initial max value of siege step size step_c_max = 1×10^{6} and the minimum value of siege step size step_c_min = 1×10^{-40}; the max iteration T = 600; the amount of the wolf population N = 50.
4	AAF-DGSS-WPOA	Initial value of search step size step_a0 = 1.5; the initial max value of siege step size step_c_max = 1×10^{6} and the minimum value of siege step size step_c_min = 1×10^{-40}; the max iteration T = 600;the amount of the wolf population N = 50.

4.2. Experimental Results and Corresponding Analysis

To verify the excellent performance of AAF-DGSS-WPOA, this article performed thirty calculations on each test function using four different algorithms, and recorded the optimal value, worst value, mean value, standard deviation, average iteration times and average iteration time for optimization. The specific results are shown in Table 3.

Table 3. Raw Data of Experiments.

Function	Algorithm	Optimal Value	Worst Value	Average Value	Standard Deviation	Average Iteration	Average Time
1 Ackley min f = 0	GA	7.92×10^{-6}	0.00012803	0.000051735	6.95×10^{-10}	176.7	0.23365
	PSO	1.71×10^{-5}	0.00057189	0.00011362	6.89×10^{-9}	600	0.046053
	WDX-WPOA	0	0	0	0	38.3333	0.053561
	AAF-DGSS-WPOA	0	0	0	0	32.4	0.035122
2 Bukin6 min f = 0	GA	0.6912	11.2453	3.7961	2.6443	600	0.013
	PSO	0.0012517	0.13	0.058051	0.041452	600	0.013559
	WDX-WPOA	0.030632	1.1	0.67996	0.39957	600	0.99632
	AAF-DGSS-WPOA	0.014982	1.1	0.51109	0.36422	600	0.65582
3 Drop-Wave min f = −1	GA	−0.99992	−0.78573	−0.93986	0.04808	600	0.012028
	PSO	−1	−0.93625	−0.98512	0.026965	218.6333	0.004777
	WDX-WPOA	−1	−1	−1	0	49.5	0.065078
	AAF-DGSS-WPOA	−1	−1	−1	0	47.8	0.058533
4 Eggholder min f = −959.6407	GA	−959.6387	−629.6112	−876.5954	78.8508	600	0.012111
	PSO	−959.6407	−718.1675	−926.7076	53.5701	600	0.012546
	WDX-WPOA	−959.6407	−935.338	−947.2171	11.904	600	1.0656
	AAF-DGSS-WPOA	−959.6404	−935.3379	−948.744	11.7553	600	0.65137
5 Griewank min f = 0	GA	0.004788	0.31789	0.075813	0.063402	600	0.013661
	PSO	0	0.019719	0.0026303	0.0045421	339.3	0.0085504
	WDX-WPOA	0	0	0	0	17.5667	0.025562
	AAF-DGSS-WPOA	0	0	0	0	13.5	0.019246

Table 3. *Cont.*

Function	Algorithm	Optimal Value	Worst Value	Average Value	Standard Deviation	Average Iteration	Average Time
6 Levy min f = 0	GA	0.00024335	1.1263	0.12324	0.21281	600	0.038618
	PSO	1.50×10^{-32}	1.50×10^{-32}	1.50×10^{-32}	1.09×10^{-47}	600	0.039204
	WDX-WPOA	0	2.68×10^{-11}	9.79×10^{-13}	4.81×10^{-12}	569.7667	0.87339
	AAF-DGSS-WPOA	0	0.39478	0.013159	0.070865	479.1333	0.7371
7 Levy13 min f = 0	GA	0.011247	2.2797	0.22303	0.76184	600	0.01238
	PSO	0.00010961	−0.97283	−0.97283	3.33×10^{-16}	600	0.012789
	WDX-WPOA	0	0	0	0	27.0667	0.038007
	AAF-DGSS-WPOA	0	0	0	0	21.7	0.03573
8 CROSS-IN-TRAY min f = −2.06261	GA	−2.0556	−1.6954	−1.8635	0.2652	14.3	0.008943
	PSO	−2.0556	−0.96923	−1.5411	0.29142	600	0.016291
	WDX-WPOA	−2.0626	−2.0626	−2.0626	0	4.0333	0.0097716
	AAF-DGSS-WPOA	−2.0626	−2.0626	−2.0626	0	3.785	0.0090403
9 Schaffer2 min f = 0	GA	1.03×10^{-6}	0.042464	0.010477	0.0093243	600	0.01321
	PSO	0	0	0	0	66.9667	0.0019346
	WDX-WPOA	0	0	0	0	11.7333	0.016255
	AAF-DGSS-WPOA	0	0	0	0	9.7	0.012018
10 Bohachevsky1 min f = 0	GA	0.011268	0.91934	0.48134	0.25563	600	0.011984
	PSO	0	0	0	0	78.1667	0.0017231
	WDX-WPOA	0	0	0	0	14.5	0.020572
	AAF-DGSS-WPOA	0	0	0	0	12.3	0.017201
11 Perm0-d-β min f = 0	GA	0.011057	388.7314	26.2837	71.8592	600	0.01213
	PSO	0	0	0	0	175.3667	0.0052171
	WDX-WPOA	0	0	0	0	26.4	0.030897
	AAF-DGSS-WPOA	0	0	0	0	21.7667	0.024111
12 Rotated Hyper-Ellipsoi min f = 0	GA	0.00039244	0.12985	0.034819	0.03493	600	0.012687
	PSO	1.96×10^{-134}	1.02×10^{-129}	8.56×10^{-131}	2.28×10^{-130}	600	0.013828
	WDX-WPOA	0	0	0	0	26.3	0.029679
	AAF-DGSS-WPOA	0	0	0	0	22.7	0.023062
13 Sum Squares min f = 0	GA	2.42×10^{-6}	0.0025094	0.00051005	0.00048937	600	0.012084
	PSO	5.91×10^{-137}	2.18×10^{-132}	3.76×10^{-133}	5.82×10^{-133}	600	0.013233
	WDX-WPOA	0	0	0	0	26.2667	0.029123
	AAF-DGSS-WPOA	0	0	0	0	21.1667	0.023601
14 Trid min f = −2	GA	−0.037736	−1.9991	−1.8925	0.11219	600	0.01241
	PSO	−2	−2	−2	0	600	0.014356
	WDX-WPOA	−2	−2	−2	0	12.8	0.013702
	AAF-DGSS-WPOA	−2	−2	−2	0	9.1667	0.012218
15 Booth min f = 0	GA	4.93×10^{-12}	4.92×10^{-9}	8.73×10^{-10}	1.11×10^{-18}	74.79	0.088336
	PSO	5.62×10^{-23}	8.78×10^{-17}	5.13×10^{-18}	2.36×10^{-34}	600	0.02997
	WDX-WPOA	0	0	0	0	25.1333	0.027151
	AAF-DGSS-WPOA	0	0	0	0	20.4	0.023447
16 Matyas min f = 0	GA	9.11×10^{-6}	0.042161	0.010059	0.010711	600	0.01241
	PSO	1.76×10^{-120}	2.71×10^{-116}	2.87×10^{-117}	5.48×10^{-117}	600	0.013081
	WDX-WPOA	0	0	0	0	25.5	0.030189
	AAF-DGSS-WPOA	0	0	0	0	20.3333	0.02357
17 Easom min f = −1	GA	−1	0	−0.75001	0.18749	72.91	0.084762
	PSO	−1	-6.30×10^{-61}	−0.90001	0.089988	593.02	0.033852
	WDX-WPOA	−1	−1	−1	0	13.6667	0.018649
	AAF-DGSS-WPOA	−1	−1	−1	0	11.7333	0.014768
18 Eggcrate min f = 0	GA	1.13×10^{-11}	3.20×10^{-8}	4.13×10^{-9}	4.66×10^{-17}	74.59	0.085997
	PSO	6.23×10^{-24}	1.42×10^{-8}	1.42×10^{-10}	1.99×10^{-18}	597.56	0.030566
	WDX-WPOA	0	0	0	0	26.2	0.032477
	AAF-DGSS-WPOA	0	0	0	0	21.0667	0.026856
19 Bohachevsky3 min f = 0	GA	0.012263	1.3156	0.49003	0.33209	600	0.023037
	PSO	0	0	0	0	78.2	0.014223
	WDX-WPOA	0	0	0	0	14.4	0.019403
	AAF-DGSS-WPOA	0	0	0	0	12.1333	0.017448
20 Bridge min f = -3	GA	−2.0588	−2.0077	−2.0175	0.0098833	600	0.017844
	PSO	−3.0054	−2.0487	−2.748	0.39821	600	0.019433
	WDX-WPOA	−3.0054	−3.0054	−3.0054	4.44×10^{-16}	600	0.73772
	AAF-DGSS-WPOA	−3.0054	−3.0054	−3.0054	4.44×10^{-16}	600	0.63927

From Table 3, we can see that the overall performance of the improved algorithm is superior to the other three optimization algorithms. Firstly, in terms of optimization performance, AAF-DGSS-WPOA always manages to successfully find the theoretical optimal value among multiple test functions. For example, in functions containing multiple peaks and complex search spaces (such as Ackley, Drop-Wave, GRIEWANK, etc.), AAF-DGSS-WPOA has reached the optimal solution, while GA and PSO, although obtaining solutions

close to the optimal value in certain functions, often exhibit biases in complex multi-modal problems. This indicates that the improved algorithm performs well in handling complex optimization problems and has better optimization accuracy.

In addition, in terms of stability, the standard deviation of AAF-DGSS-WPOA is usually zero or extremely small (such as Drop-Wave, Cross-In-Tray, Trid and other functions), demonstrating extremely high stability. This means that in multiple runs, AAF-DGSS-WPOA maintains consistent performance, while other algorithms (especially GA and PSO) have significantly higher standard deviations than AAF-DGSS-WPOA on certain complex functions, showing some volatility.

Moreover, in terms of convergence speed, AAF-DGSS-WPOA only requires fewer average iterations and average iteration times on most functions. For example, in the LEVY and PERM functions, AAF-DGSS-WPOA has significantly fewer average iterations than other algorithms, demonstrating its ability to converge quickly. Although in some functions (such as EGGHOLDER, Bukin6, etc.) the number of iterations did not increase, there were improvements in other aspects such as mean and standard deviation, especially in terms of significant improvement in average iteration time. Therefore, AAF-DGSS-WPOA has good convergence speed.

Finally, in terms of overall performance, the robust performance of AAF-DGSS-WPOA in multiple dimensions, especially its outstanding performance on complex problems, proves the potential application of this algorithm in the field of optimization. In contrast, GA and PSO can achieve similar results on some simple functions, but in complex environments, they usually require more computing resources and iteration times, and their performance is not good enough.

We focused on comparing the difference in average iteration time between AAF-DGSS-WPOA and WDX-WPOA, and the results are shown in Table 4 and Figure 6. It can be seen from this that, without affecting the global optimization accuracy, the improvement rate of iteration time is always positive, indicating that the improved algorithm takes less time. It is worth mentioning that the improvement rates of functions Ackley, Bukin6, and EGGHOLDER are all above 30%, while the improvement rates of most other functions are also around 20%, with the worst improvement rate reaching 5.99%. In summary, through the analysis of experimental data results, we can find that the improved algorithm has higher optimization accuracy, better stability, and faster convergence speed.

Table 4. Time-Spent Comparison between AAF-DGSS-WPOA and WDX-WPOA.

Function \ Algorithm	WDX-WPOA	AAF-DGSS-WPOA	Improvement Rate
F1	0.053561	0.035122	34.43%
F2	0.99632	0.65582	34.18%
F3	0.065078	0.058533	10.06%
F4	1.0656	0.65137	38.87%
F5	0.025562	0.019246	24.71%
F6	0.87339	0.7371	15.60%
F7	0.038007	0.03573	5.99%
F8	0.0097716	0.0090403	7.48%
F9	0.016255	0.012018	26.07%
F10	0.020572	0.017201	16.39%
F11	0.030897	0.024111	21.96%

Table 4. *Cont.*

Function \ Algorithm	WDX-WPOA	AAF-DGSS-WPOA	Improvement Rate
F12	0.029679	0.023062	22.30%
F13	0.029123	0.023601	18.96%
F14	0.013702	0.012218	10.83%
F15	0.027151	0.023447	13.64%
F16	0.030189	0.02357	21.93%
F17	0.018649	0.014768	20.81%
F18	0.032477	0.026856	17.31%
F19	0.019403	0.017448	10.08%
F20	0.73772	0.63927	13.35%

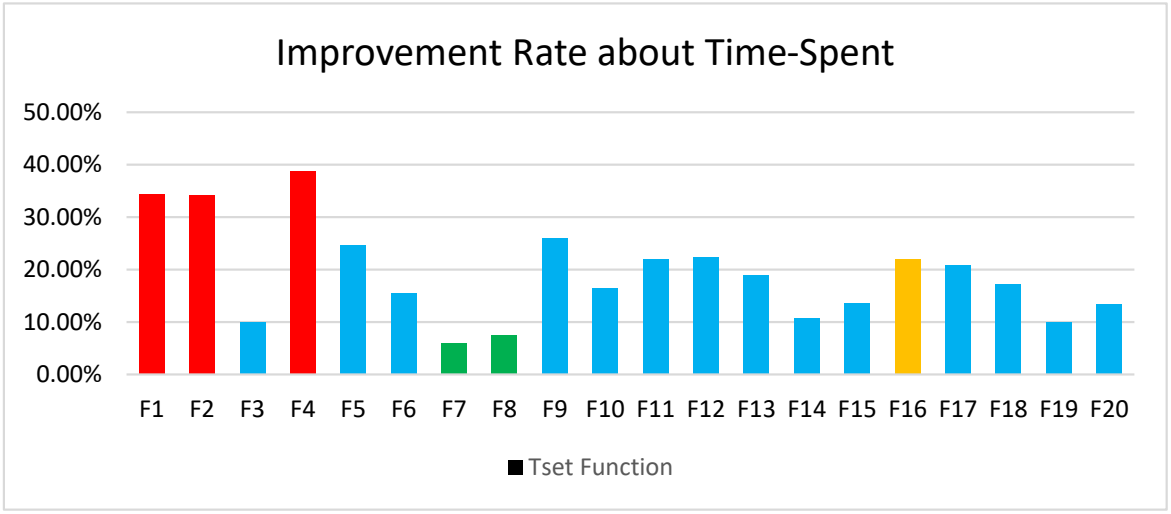


Figure 6. Histogram of improvement rate by comparing the time spent by AAF-DGSS-WPOA and WDX-WPOA on all 20 test functions in this paper. (F1: Ackley, F2: Bukin6, F3: Drop-Wave, F4: Eggholder, F5: Griewank, F6: Levy, F7: Levy13, F8: Cross-In-Tray, F9: Schaffer2, F10: Bohachevsky1, F11: Perm0-d-β, F12: Rotated Hyper-Ellipsoid, F13: Sum Squares, F14: Trid, F15: Booth, F16: Matyas, F17: Easom, F18: Eggcrate, F19: Bohachevsky3, F20: Bridge).

4.3. Experiment and Analysis of Optimal Location of Police Booths

We assume that the urban area is 50 km × 50 km, the number of police booths is 50, and the coverage radius of each police booth is 5 km. Based on AAF-DGSS-WPOA, the location planning steps for police booths are shown in Figure 5. Through the steps, we can obtain the distribution of police booths in urban areas, as shown in Figure 7d. In addition, we also used the GA, PSO, and WDX-WPOA algorithms to solve this problem, and the results are shown in Figure 7a–c.

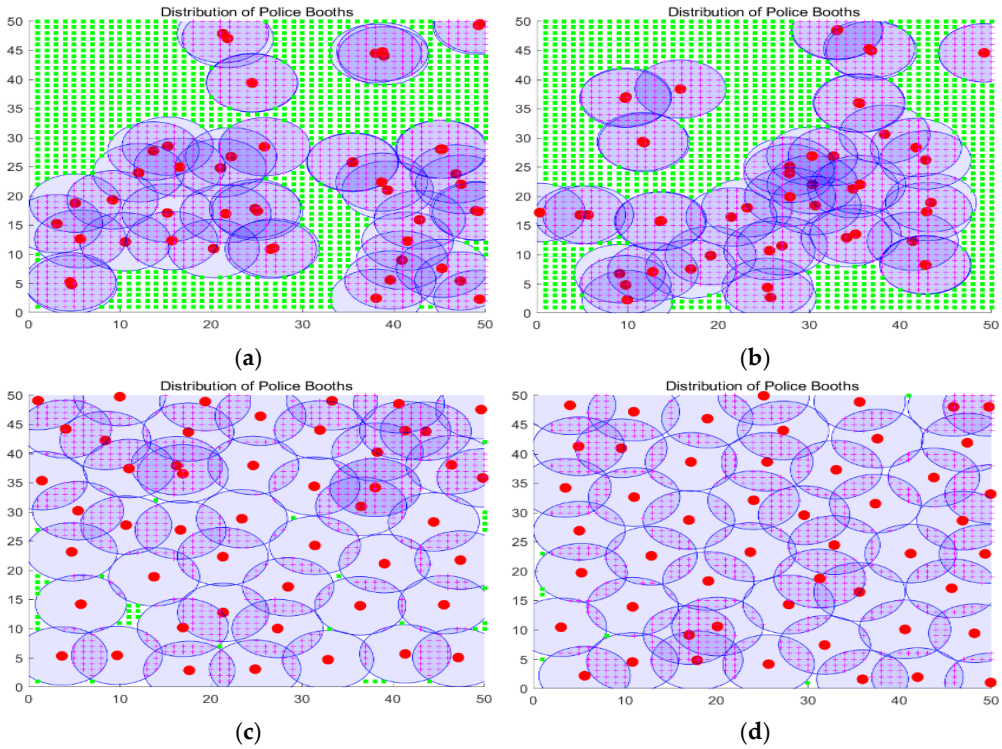


Figure 7. (**a**) about GA; (**b**) about PSO; (**c**) about WDX-WPOA; (**d**) about AAF-DGSS-WPOA.

In the above images, the red dots indicate the location of the booths, the green area represents the uncovered area, the purple area represents the covered area, and the deeper purple area represents the overlapping area. From these images, we can see that compared to GA and PSO, the wolf pack algorithm has more coverage areas and fewer overlapping areas in solving the problem of police booth location planning, resulting in better performance. To further compare the performance of different algorithms in solving this problem more intuitively, we tested the optimal value, average value, average coverage rate, average overlap rate, and average iteration time. The results are shown in Table 5.

Table 5. Raw Data of police booth planning.

Result / Algorithm	Optimal Value	Average Value	Average Coverage Rate	Average Overlap Rate	Average Time
GA	1297.2	1152.712	79.48%	61.78%	273.0579
PSO	1096.2	926.284	67.21%	65.39%	2598.784
WDX-WPOA	1476.9	1382.266	95.05%	37.35%	17,496.85
AAF-DGSS-WPOA CCCCWCWOACWOA	1563.7	1476.831	97.54%	36.76%	14,584.33

From both the optimal and average values, the fitness value of AAF-DGSS-WPOA is the highest. From the perspective of average coverage and average overlap, AAF-DGSS-WPOA also performs the best. Overall, AAF-DGSS-WPOA has good optimization performance.

From the perspective of running time, although the wolf pack algorithm has a much longer running time than GA and PSO in solving this problem, it has better performance in location planning, with higher coverage and lower overlap, and the time is also acceptable, after all, there is no perfect thing in the world. Moreover, it is gratifying that, as shown in Table 5 and Figure 8, the running time of AAF-DGSS-WPOA is shorter than that of

WDX-WPOA. From Figure 8, it can be seen that the new algorithm has improved the running time by about 16.65%, while also ensuring good optimization performance.

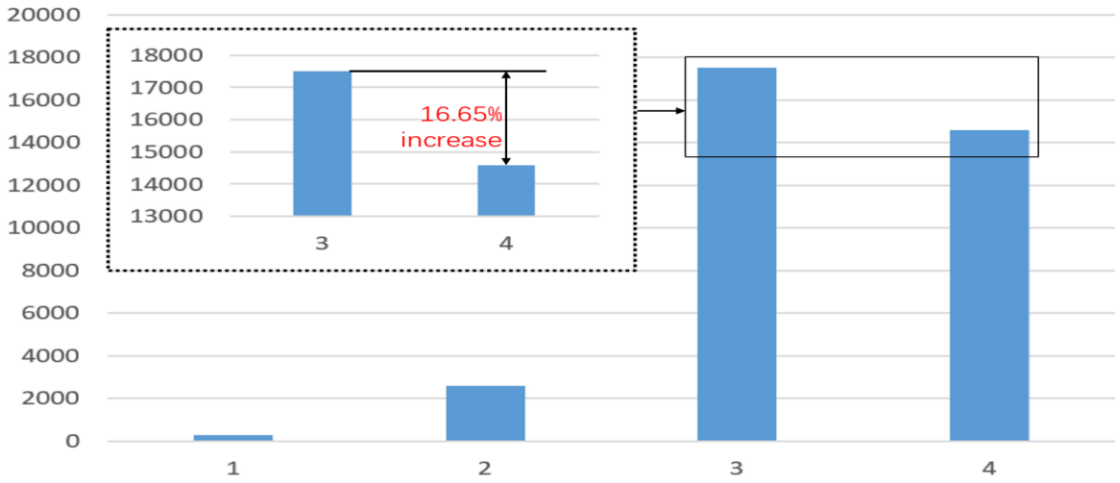


Figure 8. Time-Spent Comparison of police booths planning.

In summary, AAF-DGSS-WPOA has shown good overall performance, with better optimization capabilities and faster convergence speed.

5. Conclusions

This article proposes an optimized location selection method (PBPM-AAF-DGSS-WPOA) for police stations based on an improved wolf pack optimization algorithm to address the problems of insufficient coverage and high overlap rate in current planning. Firstly, this article provides a detailed analysis of the planning problem of police booths and constructs a single objective model with the optimization objectives of maximizing coverage and minimizing overlap (SOOM-MCMO). Secondly, in order to address the shortcomings of traditional wolf pack algorithms in search efficiency and global optimization capabilities, this paper has made two improvements to them. One is to introduce an adaptive-approaching factor on the basis of the original summon-raid strategy. Before reaching half of the total iteration times, the wolf pack will gradually approach from the current point to the target point, enhancing the exploratory ability of the wolf pack. After exceeding half, it will maintain a fixed proportion of approach to pay more attention to the influence of the leading wolf; the second is to propose dynamic-grid-siege strategy, which gradually increases the number of wolves in each dimension as the number of iterations increases. Finally, using an improved algorithm to implement site selection planning for police booths, a better solution was obtained. The experimental results show that the overall performance of AAF-DGSS-WPOA proposed in this paper is significantly better than classical swarm intelligence optimization algorithms such as GA, PSO, and WDX-WPOA on 20 public datasets and the objective function proposed in this paper. Specifically, this method performs well in terms of the optimality of the solution results and the efficiency of the running time, and it can effectively solve the optimization layout problem of police booths, providing scientific and reasonable planning suggestions for police departments. Meanwhile, the research findings of this article also have broad applicability and promotional value. This optimization method is not only applicable to the layout of police booths but can also be applied to the layout optimization of urban emergency facilities, the configuration optimization of public service facilities, and other fields, providing effective scientific basis for relevant decision-makers.

Certainly, every algorithm has inherent trade-offs, and while AAF-DGSS-WPOA demonstrates superior optimization performance for police booth deployment, its computational efficiency remains a limitation compared to traditional GA and PSO approaches. Notably, it achieves a 16.65% reduction in computational overhead relative to the baseline WDX-WPOA model under identical test conditions, suggesting its potential applicability despite the time constraints. These performance characteristics indicate that the proposed method should primarily be considered for mid-scale applications where solution quality is prioritized over real-time computation.

Regarding the real-world applicability, this study indeed focuses primarily on theoretical optimization under simplified conditions. The geographical factors—such as terrain complexity, road network configurations, and urban population distribution patterns—were not fully incorporated into our modeling framework. This is particularly significant as densely populated metropolitan areas with fragmented political jurisdictions require different optimization strategies compared to homogeneous regions. For instance, administrative boundaries between municipalities often pose challenges in resource allocation that were not addressed in our current model.

Future research directions clearly lie in integrating spatial analysis tools (e.g., GIS mapping) with our algorithm to account for neighborhood-specific characteristics, including population density variations impacting patrol demand (urban vs. suburban areas), natural and man-made geographical barriers affecting response times, and infrastructure considerations like traffic flows and emergency route networks. Such enhancements would not only improve scalability to large urban environments but also ensure compliance with operational constraints in real-world police department deployments. By addressing these practical considerations, the algorithm could potentially serve as a decision support tool for policymakers while maintaining acceptable performance levels through parallel computing optimization.

Such enhancements would not only improve scalability to large urban environments but also ensure compliance with operational constraints in real-world police department deployments. By addressing these practical considerations, the algorithm could potentially serve as a decision support tool for policymakers while maintaining acceptable performance levels through parallel computing optimization.

In the following work, we will continue to improve the above shortcomings, shorten the running time, and consider more practical factors to better apply it to larger urban scales and more complex practical scenarios.

Author Contributions: Conceptualization, D.W.; methodology, F.W.; software, D.W., Z.S. and F.W.; Formal analysis, D.W.; writing—original draft, F.W.; writing—review and editing, D.W.; funding acquisition, Z.S. All authors have read and agreed to the published version of the manuscript.

Funding: This study was funded by Fujian Province's 2024 Science and Technology Driven Police Research Special Project (No. 2024Y0065).

Data Availability Statement: The datasets used and analyzed during the current study available from the corresponding author on reasonable request.

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

Abbreviations

Order	Full-Name	Abbreviation
1	Simultaneous Optimization for Max Coverage and Min Overlap model	SOOM-MCMO
2	Adaptive-Approaching Framework with Dynamic-Grid-Siege Wolf Pack Algorithm	AAF-DGS-WPOA
3	AAF-DGS-WPOA-based Police Booth Planning Method	PBPM-AAFDGS-WPOA
4	Genetic Algorithm	GA
5	Particle Swarm Optimization	PSO
6	Mentioned in reference [25]	WDX-WPOA
7	Adaptive-Approaching Factor	AAF
8	Dynamic-Grid-Siege Strategy	DGSS

References

1. Fondevila, G.; Vilalta-Perdomo, C.; Galindo Pérez, M.C.; Cafferata, F.G. Crime deterrent effect of police stations. *Appl. Geogr.* **2021**, *134*, 102518.
2. Stassen, R.; Ceccato, V. Police Accessibility in Sweden: An Analysis of the Spatial Arrangement of Police Services. *Polic. A J. Policy Pract.* **2021**, *15*, 896–911.
3. Jiang, Y.; Guo, B.; Yan, Z. Multi-Criterion Spatial Optimization of Future Police Stations Based on Urban Expansion and Criminal Behavior Characteristics. *ISPRS Int. J. Geo-Inf.* **2022**, *11*, 384. [CrossRef]
4. Wang, W.; Xu, Z.; Sun, D.; Lan, T. Spatial Optimization of Mega-City Fire Stations Based on Multi-Source Geospatial Data: A Case Study in Beijing. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 282. [CrossRef]
5. Chen, M.; Wang, K.; Yuan, Y.; Yang, C. A POIs Based Method for Location Optimization of Urban Fire Station: A Case Study in Zhengzhou City. *Fire* **2023**, *6*, 58. [CrossRef]
6. Yang, J.; Guo, K.; Dai, Y.; Tian, S.; Wang, W.; Jiang, Z.; Dai, Z. Spatial layout siting method for fire stations based on comprehensive forest fire risk distribution. *Case Stud. Therm. Eng.* **2023**, *49*, 103243.
7. De Domingo, M.; Ortigosa, N.; Sevilla, J.; Roger, S. Cluster-Based Relocation of Stations for Efficient Forest Fire Management in the Province of Valencia (Spain). *Sensors* **2021**, *21*, 797. [CrossRef] [PubMed]
8. KhoshAmooz, G. A new fuzzy location-based approach for fire station site selection in Tehran. *Appl. Geomat.* **2024**, *16*, 987–1001.
9. Tian, F.; Lei, J.; Zheng, X.; Yin, Y. Integrating Space Syntax and Location-Allocation Model for Fire Station Location Planning in a China Mega City. *Fire* **2023**, *6*, 64. [CrossRef]
10. Guo, K.; Wang, W.; Tian, S.; Yang, J.; Jiang, Z.; Dai, Z. Research on Optimization Technology of Cross-Regional Synergistic Deployment of Fire Stations Based on Fire Risk. *Sustainability* **2022**, *14*, 15725. [CrossRef]
11. Shahparvari, S.; Fadaki, M.; Chhetri, P. Spatial accessibility of fire stations for enhancing operational response in Melbourne. *Fire Saf. J.* **2020**, *117*, 103149.
12. Ming, J.; Richard, J.-P.P.; Qin, R.; Zhu, J. Distributionally robust optimization for fire station location under uncertainties. *Sci. Rep.* **2022**, *12*, 5394.
13. Yanchao, S.; Chen, W.; Wu, Y. A New Wolves Intelligent Optimization Algorithm. In Proceedings of the 2021 IEEE/ACIS 19th International Conference on Computer and Information Science (ICIS), Shanghai, China, 23–25 June 2021; pp. 143–147.
14. Lu, N.; Ma, L. Quantum Wolf Pack Evolutionary Algorithm of Weight Decision-Making Based on Fuzzy Control. *Chin. J. Electron.* **2022**, *31*, 635–646.
15. Zhang, H.; Lv, X.; Ma, C.; Cui, L. An Elite Wolf Pack Algorithm Based on the Probability Threshold for a Multi-UAV Cooperative Reconnaissance Mission. *Drones* **2024**, *8*, 513. [CrossRef]
16. Jiang, H.; Yu, Q.; Han, D.; Chen, Y.; Li, Z. A path planning method for unmanned aerial vehicle based on improved wolf pack algorithm. *Concurr. Comput. Pract. Exp.* **2024**, *36*, e8095.
17. Lai, R.; Gao, B.; Lin, W. Solving No-Wait Flow Shop Scheduling Problem Based on Discrete Wolf Pack Algorithm. *Sci. Program.* **2021**, *2021*, 4731012.
18. Sahu, K.K.; Nayak, S.C.; Behera, H.S. Extreme Learning With Metaheuristic Optimization for Exchange Rate Forecasting. *Int. J. Swarm Intell. Res.* **2022**, *13*, 1–25.
19. Zhu, Q.; Wu, H.; Li, N.; Hu, J. A Chaotic Disturbance Wolf Pack Algorithm for Solving Ultrahigh-Dimensional Complex Functions. *Complexity* **2021**, *2021*, 6676934.

20. Wang, H.; Wang, D. A Wolf Pack Optimization Algorithm Using RASGS and GBA for Multi-Modal Multi-Objective Problems. *Symmetry* **2022**, *14*, 2568. [CrossRef]
21. Chen, X.; Cheng, F.; Liu, C.; Cheng, L.; Mao, Y. An improved Wolf pack algorithm for optimization problems: Design and Evaluation. *PLoS ONE* **2021**, *16*, e0254239.
22. Jin, Z. Application of WCA-RBF Neural Network in Fault Diagnosis of Analog Circuits. *Int. Trans. Electr. Energy Syst.* **2023**, *2023*, 8812152.
23. Zhu, J.; Liu, J. A Simple and Scalable Structure of Particle Swarm Optimization Based on Linear System Theory. 2022. Available online: https://www.researchsquare.com/article/rs-2141043/v1 (accessed on 2 April 2025).
24. Chompookham, T.; Phiphitphatphaisit, S.; Okafor, E.; Surinta, O. Robust Model Selection for Plant Leaf Image Recognition Based on Evolutionary Ant Colony Optimization With Learning Rate Schedule. *IEEE Access* **2024**, *12*, 132369–132389.
25. Wang, D.; Qian, X.; Liu, K.; Ban, X.; Guan, X. An Adaptive Distributed Size Wolf Pack Optimization Algorithm Using Strategy of Jumping for Raid (September 2018). *IEEE Access* **2018**, *6*, 65260–65274.
26. Kim, J.; Lee, W.; Jung, J.; Choi, J.; Kim, E.; Kim, J. Weighted Localized Clustering: A Coverage-Aware Reader Collision Arbitration Protocol in RFID Networks. In Proceedings of the International Conference on Embedded Software and Systems (ICESS 2005), Xi'an, China, 16–18 December 2005.

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.