drones

MDPI

*Article*

# Multi-Objective Deployment of UAVs for Multi-Hop FANET: UAV-Assisted Emergency Vehicular Network

**Haoran Li [1], Xiaoyao Hao [2], Juan Wen [1,*], Fangyuan Liu [3,4] and Yiling Zhang [2]**

[1] State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an 710071, China; hrli@xidian.edu.cn
[2] Hangzhou Institute of Technology, Xidian University, Hangzhou 330100, China; 23011211020@stu.xidian.edu.cn (X.H.); 22011210946@stu.xidian.edu.cn (Y.Z.)
[3] School of Information Engineering, Xi'an University, Xi'an 710065, China; lfy@xafy.edu.cn
[4] School of Information Engineering, Xi'an Fanyi University, Xi'an 710105, China
* Correspondence: juanwen@xidian.edu.cn

**Abstract:** In the event of a sudden natural disaster, the damaged communication infrastructure cannot provide a necessary network service for vehicles. Unfortunately, this is the critical moment when the occupants of trapped vehicles need to urgently use the vehicular network's emergency service. How to efficiently connect the trapped vehicle to the base station is the challenge facing the emergency vehicular network. To address this challenge, this study proposes a UAV-assisted multi-objective and multi-hop ad hoc network (UMMVN) that can be used as an emergency vehicular network. Firstly, it presents an integrated design of a search system to find a trapped vehicle, the communication relay, and the networking, which significantly decreases the UAV's networking time cost. Secondly, it presents a multi-objective search for a trapped vehicle and navigates UAVs along multiple paths to different objectives. Thirdly, it presents an optimal branching node strategy that allows the adequate use of the overlapping paths to multiple targets, which decreases the networking cost within the limited communication and searching range. The numerical experiments illustrate that the UMMVN performs better than other state-of-the-art networking methods.

**Keywords:** UAV-assisted vehicular network; multi-hop ad hoc networks; multi-objective deployment

check for updates

**Citation:** Li, H.; Hao, X.; Wen, J.; Liu, F.; Zhang, Y. Multi-Objective Deployment of UAVs for Multi-Hop FANET: UAV-Assisted Emergency Vehicular Network. *Drones* **2024**, *8*, 262. https://doi.org/10.3390/drones8060262

Academic Editors: Dawei Wang and Ruonan Zhang

Received: 28 April 2024
Revised: 3 June 2024
Accepted: 4 June 2024
Published: 13 June 2024

**Copyright:** © 2024 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

With the wide deployment of the ICT infrastructure, the vehicular network has rapidly developed, enabling automatic driving, intelligent transportation, and in-car entertainment. The intelligent capability of the vehicular network is built on the foundation of a well-operating ICT infrastructure [1]. However, the smart vehicle will inevitably encounter a situation involving off-line infrastructure during natural disasters, such as earthquakes or tsunamis. In an emergency situation, reconnecting the vehicle to the uninfluenced infrastructure, such as an existing base station, is an urgent task. Therefore, there are four primary requirements for an emergency network: it must search for and locate trapped vehicles in a complex environment, ignore obstacles and format a stable network, widely cover the disaster region, and have a low relay node deployment cost.

Benefiting from rapid movement and a multifunctional load [2], the unmanned aerial vehicle (UAV) has a potential role to play as a relay node for an emergency vehicular network. Thus, the UAV-assisted emergency vehicular network has attracted considerable interest in industrial and academic fields in recent years. The UAV-assisted emergency vehicular network is essentially a multi-hop and multi-target ad hoc network [3]. UAVs play the role of relay nodes, which can reconnect many trapped vehicles to the existing base station. First, multiple UAVs must be arranged to search in different directions for a probable trapped vehicle; to simultaneously construct a gradually expanding ad hoc network with a searching process [4]; and, finally, to find the trapped vehicle and

immediately provide the vehicle with an accessible service. To satisfy the four primary requirements, the emergency network needs the following characteristics: connectivity, adaptability, security, energy efficiency, and scalability [5].

The UAV-assisted emergency vehicular network mainly depends on the performance of a UAV deployment algorithm [6]. The different design preferences of the algorithm induce discrepant deployment results. The energy efficiency deployment algorithm can dispatch fewer UAVs to cover more trapped vehicles and can also support fewer hop count paths for networking and rapidly backhauling a trapped vehicle's request message. The securable and connectable deployment algorithm not only prevents the UAV from colliding with an obstacle or other UAVs during the searching process, it also reduces the effect of the obstacle's interference on the backhauling process [7]. The adaptable and scalable deployment algorithm focuses on the intelligent sensing and navigation capability of the UAV and continuously adjusts the searching direction and navigation strategy when the UAV gains new information about the target or obstacle [8].

In the UAV-assisted emergency vehicular network, the UAV deployment problem is not a simple centralized placement problem with a global perspective [9]; it is a distributed implementation of individual searches and navigation in complex environments with obstacles, and it involves the formatting of an ad hoc network to connect the trapped vehicle and the existing base station. Furthermore, the UAV-assisted emergency vehicular network faces three major challenges, as follows:

- UAVs need to automatically plan a trajectory through the coarse-grained locations of multiple targets in complex environments with obstacles [10]. The trajectory planning method requires the avoidance of collisions with obstacles or other UAVs. It also has the capability to automatically adjust the trajectory by sensing obstacles or using target information.
- The ad hoc network is constructed according to the multi-objective requirements, including a lower arrangement cost, less link interference, and a lower hop count. Considering the requirements of a rapid connection and stable link, the multi-objective network optimum strategy needs to be inserted into the trajectory planning method, rather than implementing optimum adjustment after the network is built. It is undoubtedly going to enhance the difficulty if the UAVs must find a path and construct a link in an unknown environment with little target location information and no obstacle information.
- In the search for multiple targets, parts of multiple paths can overlap; thus, a portion of the relay nodes can be shared to decrease the arrangement costs. The problem transforms into one in which the branching node must be set so that it can decide when and where to separate the common trajectory into multiple paths to different targets [11].

To address the abovementioned challenge, this study proposes a UAV-assisted multi-objective and multi-hop ad hoc network for an emergency vehicular network (UMMVN). Firstly, an intelligent trajectory planning method is presented for searching for trapped vehicles, sensing obstacles, building a short-distance communication link, and navigating to an objective location. Secondly, a multi-objective deployment strategy for an emergency network is presented for guidance to optimum relay node locations. Thirdly, an optimal branching node strategy is presented that allows the adequate use of the overlapping paths to multiple targets. Specifically, the study's contributions are expressed as follows:

- An intelligent trajectory planning method is presented for searching for trapped vehicles. Each UAV navigates to the cross-point location between the boundary of the last UAV's maximum communication range and the line to the target. During the navigating process, the UAV implements collision-less movements while sensing the obstacle's location.
- A multi-objective deployment strategy is presented for the emergency network with a lower arrangement cost, less link interference, and a lower hop count. A multi-

objective particle swarm optimization algorithm (MOPSO) is applied to solve this deployment problem [12].

An optimal branching node strategy is presented that allows the adequate use of the overlapping paths to multiple targets. It sets a branching node to decide when to separate a common trajectory into multiple paths to different targets according to the actual situation.

The remainder of this paper is organized as follows. Section 2 introduces related work on the development of the UAV-assisted network's tasks and the optimal deployment method. Section 3 describes the problem formulation in detail. Section 4 describes the UMMVN and its algorithm. Section 5 describes the experiment and compares the proposed algorithm with other state-of-the-art algorithms. Section 6 concludes the study and presents future work.

## 2. Related Works

### *2.1. Search and Rescue Task in FANET*

From a communication perspective, FANET's mission can be classified into four categories: SAR (search and rescue), coverage, construction, and delivery and transportation [13]. In the SAR task, the UAVs need to search for mobile or stationary targets and give assistance to the trapped person once they are found. In the coverage task, the UAVs provide an area coverage service (mapping and monitoring) or network coverage (aerial base station and aerial relay node). In the construction task, the UAVs need to lift materials and then place them at the planned location. In the delivery and transportation task, the UAVs need to transport goods from one place to another. The focus of our UMMVN is the SAR task.

In general, the SAR task needs to follow the fundamental assumption in two ways. In the traffic model, the transmitted data are always of the coordination, sensing, and service types, such as 1 Mbps–2 Mbps images or a video stream, which require a 50–100 ms time delay. The geographical distribution of the generating traffic consists of small and medium-sized areas. In the mobility model, the paparazzi mobility model (PPRZM) is adopted to randomly search in a specified target area; the Gauss–Markov (GM) model is adopted to reach the trapped person in a disaster area; the random waypoint (RWP) model is adopted for scanning in a circular area.

Based on the application's aim, after searching for the objectives, the UAVs implement a high-altitude fire-fighting operation [14], which requires the capability to produce large-capacity video transmission with a low delay. UAVs localize the victims and evaluate the rescue difficulty level [15], which requires the capability to produce rapid image transmission with a low delay. UAVs implement an emergency communication accessing service to connect the victims and the base station [16], which requires a stable network state. Although different applications have different preferences, the common requirement is always the low delay. Therefore, our study regards the low delay as a major objective in the optimization of the network's performance.

### *2.2. The Deployment Optimization in SAR Task*

In the SAR task, the UAV deployment problem is a critical issue when deciding both the link constructing time and the transmission delay time [17]. Naturally, the UAV deployment problem is a non-convex problem in a complex environment. According to the specific partition criterion, the deployment problem can be divided into the following categories.

- Static or dynamic deployment. As the static deployment method [18] obtains the geographical location information around the trapped target in advance, it always adopts a centralized decision when arranging to send a suitable UAV to the relay placement area for optimal networking. Conversely, the dynamic deployment method [19] always solves the problems of an unknown environment, which include the unknown obstacle, the unknown trapped target, and even an uncertain searching region. Thus, it always adopts a distributed decision to make the UAV implement a series of actions that involve sensing, finding a pathway, searching for a target, and obstacle avoid-

ance. After receiving a round of information, the UAVs collaborate to make an action decision in the next round.

- Single or multiple targets. In the single-target SAR task, the deployment method only considers the shortest path to connect the existing base station to the trapped target without obstacles. However, in the multi-target SAR task, the deployment method simultaneously takes two factors into account. The first is the choice of multiplexed nodes. A higher number of multiplexed nodes means less placement cost and easier destructibility. The second one is related to the large search span in dynamic deployment. The temporary network searching process can guarantee the synchronization of each round of information; consequently, the search direction is adjusted in a timely manner to avoid meaningless migration.
- Single or multiple objectives. In single-objective optimization, the solving method is relatively easy to use when determining the optimal deployment solutions. In multi-objective optimization, the weight combination among the multiple objectives is critical for deciding the performance of the deployment solutions [20]. The manual setup easily transforms the solution into an experienced local optimum. The machine setup may search out multiple solutions, but most of the solutions are difficult to realize because the actual environment has complex and unpredictable constraints.

In our study, the UMMVN adopts dynamic, multi-target, and multi-objective deployment optimization to approximate the actual SAR environment.

## 3. Problem Formulation

### *3.1. Problem Description*

The main aim of this study is to establish an optimal multi-hop ad hoc network that connects a base station to multiple targets via a sequence of UAV nodes. As illustrated in Figure 1, the location information of the base station and target points are known in advance. The UAVs play the role of intermediate routers for the relaying of messages. Each UAV has three different modules that can be used: a GPS module for identifying the current position and the target's position; a sensory system for detecting both obstacles and targets; and a network module for implementing peer-to-peer communication with adjacent UAVs within the communication range.

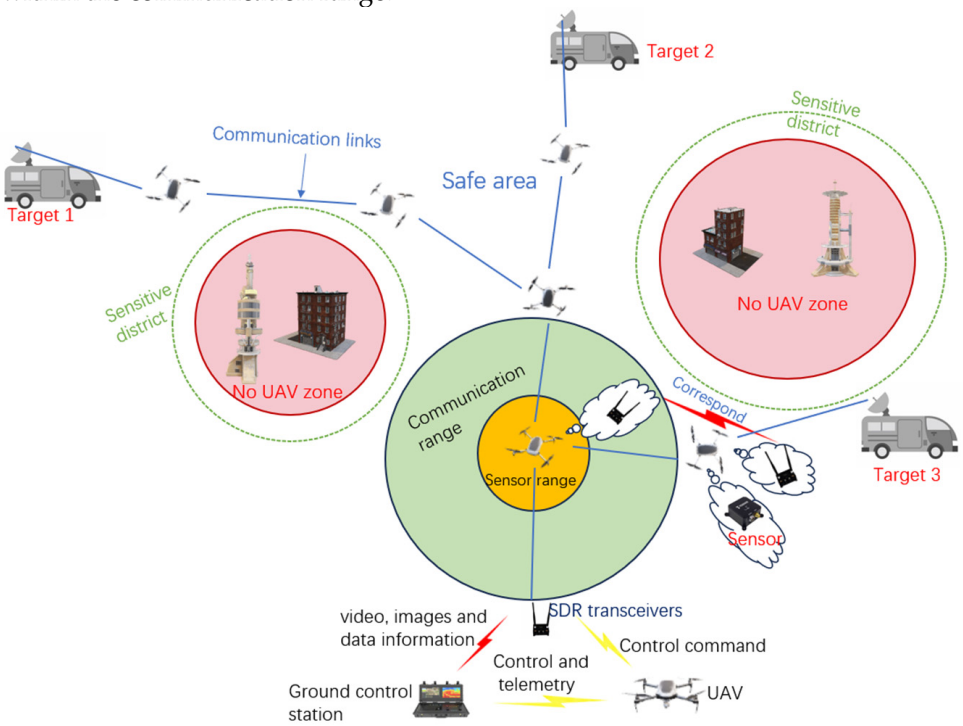



**Figure 1.** The multi-hop ad hoc network and UAV model.

As shown in Figure 1, the SDR transceiver sends and receives digital signals through wireless channels to achieve video, images, and data information. At the same time, it can also receive control instructions from the ground control station to the UAV and forward them to the UAV. The Ground control station is used to control and telemetry UAVs and can monitor their dynamics in real-time. Each UAV is equipped with a sensor for sensing obstacles and an SDR transceiver for building a communication platform and communicating with other communication devices. The orange area represents sensor range, which represents the maximum range that the UAV can sense. The green area represents communication range, which represents the maximum range that the UAV can communicate with other communication devices. A circular area composed of green dashed line and red solid line represents sensitive district, which represents the safe distance that the UAV needs to maintain from obstacles. The pink area represents no UAV zone represents obstacles, and safe area represents the area outside of sensitive area.

#### 3.1.1. UAV Model

As illustrated in Figure 2, the UAV is represented as a flying particle, whose position is $P = [x, y, z]^T$ and heading angle is $\theta$. The minimized collision area is a circle with particle location $P$ as the center and $S$ as the radius. $R_C$ and $R_S$ are, respectively, the communication and sensing ranges of the UAV; normally, $R_S$ is smaller than $R_C$, which means that the UAV sends several rounds of information to the communication boundary. Considering that the SAR task is built in a 3D world, the UAV has $V = \left[v_x, v_y, v_z\right]^T$ to reflect the velocity. Thus, in each time step $T$, the UAV's position and navigating direction are updated as follows:

$$\begin{aligned} P(t+T) &= P(t) + V(t)T \\ \theta(t+T) &= atan2\left(v_y, v_x\right) \end{aligned} \tag{1}$$

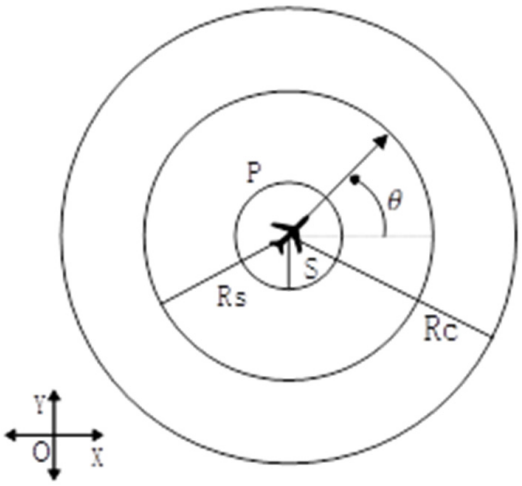



**Figure 2.** The UAV model with safe distance, sensing, and communication ranges.

#### 3.1.2. Obstacle Avoidance Model

As illustrated in Figure 3, the UAV is represented as a flying particle with position $P_i$, and its size is $R_u$. When both the UAV and the obstacle occur at height $z$, an obstacle K is represented as a circle with center $O_m$ and radius $R_m$. The UAV needs to maintain a safe distance $S$ to avoid obstacles. Let $d_m$ be the distance between $P_i$ and $O_m$. In the search process, $S$ is a primary factor in the trajectory planning.

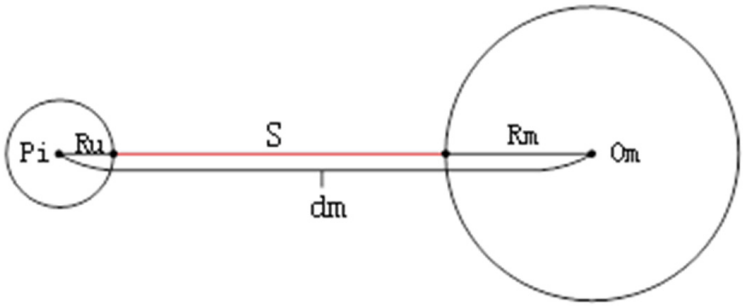



**Figure 3.** Obstacle avoidance model.

#### 3.1.3. Target Angle Model

As depicted in Figure 4, before implementing the SAR task, the targets' location and base station are known. To ensure that the UAV approximates the target at each round of the searching process, the searching direction depends on the target angle, which is defined as the angle between the vectors $\overrightarrow{P_iP_{inew}}$ and $\overrightarrow{P_iT_j}$. $\overrightarrow{P_iP_{inew}}$ is represented as the vector from the current position $P_i$ to the next search position $P_{inew}$. $\overrightarrow{P_iT_j}$ is represented as the vector from the current position $P_i$ to the j-th target $T_j$. The target angle is defined as follows:

$$\sigma_i = atan2\left(\left\|\overrightarrow{P_iP_{inew}} \times \overrightarrow{P_iT_j}\right\|, \left\|\overrightarrow{P_iP_{inew}} \cdot \overrightarrow{P_iT_j}\right\|\right) \tag{2}$$

where operators '$\times$' and '$\cdot$' are, respectively, the cross and dot products.

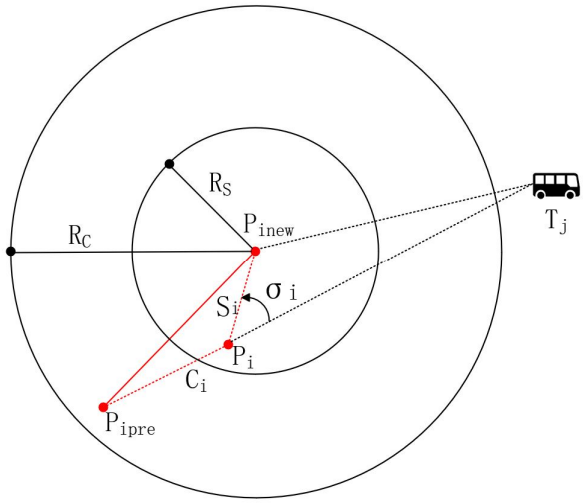



**Figure 4.** The target angle model.

### *3.2. Optimization Indexes*

This section describes the method used to find the optimal position for UAVs to establish a multi-hop ad hoc communication route from the base station to the targets. This approach involves modeling the UAVs and obstacles, formalizing them, finding a fitness function, and optimizing the indicator to obtain the optimal position. At each round, the UAV's search process is a multi-objective optimization process that searches the optimal relay location $P_{inew}$ under restricted conditions. The restricted conditions are listed as follows: (1) minimization of the target angle; (2) maintenance of the UAV's location within its neighbors' communication range; (3) relay search within the sensing range; and (4) avoidance of obstacles.

$$\min F(x) = \{F_1(x), F_2(x), F_3(x), F_4(x)\} \tag{3}$$

where $F_1$ is the minimization of the target angle; $F_2$ is the optimization of the communication links; $F_3$ is the optimization of the sensing links; and $F_4$ is the avoidance of obstacles.

(1) Minimizing the target angle $F_1$:

$$F_1 = atan2\left(\left\|\overrightarrow{P_iP_{inew}} \times \overrightarrow{P_iT_j}\right\|, \overrightarrow{P_iP_{inew}} \cdot \overrightarrow{P_iT_j}\right) \tag{4}$$

As illustrated in Figure 4 and shown in Formula (4), $P_{inew}$ is the critical variable which decides the next deployed UAV's location. The function $atan2(x,y)$ returns the azimuth between the inverse tangent of y/x. The next deployed UAV is deemed to have the smallest target angle, which approximates the line of sight between the base station and the target. The smaller target angle means that the path searched is closer to the target point. Meanwhile, the performance of the path optimization is better and the deployment cost is lower, which means $F_1$ is smaller.

(2) Optimizing communication links $F_2$:

$$F_2 = \begin{cases} |R_C - C_i| & if\ C_i \le R_C \\ \infty & otherwise \end{cases} \tag{5}$$

As illustrated in Figure 4 and shown in Formula (5), the independent variable in this formula is $C_i$, which is the distance between two UAVs, and the dependent variable is the cost of optimizing the communication links $F_2$, which is the communication range between the two UAVs. When $C_i \le R_C$, that is, the distance between two UAVs is less than the maximum communication range of the UAVs, it indicates that the two UAVs can communicate normally. At this time, the communication links $F_2$ is defined as $|R_C - C_i|$, which means that the larger the distance between the two UAVs, the lower the number of UAVs needed from the base station to the target points; the lower the cost, and the smaller the $F_2$. When $C_i > R_C$, the distance between the two UAVs is greater than the maximum communication range of the UAVs. At this time, the two UAVs cannot communicate and need to send another UAV for communication. Therefore, the communication links $F_2$ is $\infty$.

(3) Optimizing sensing links $F_3$:

$$F_3 = \begin{cases} |R_S - S_i| & if\ S_i \le R_S \\ \infty & otherwise \end{cases} \tag{6}$$

As illustrated in Figure 4 and shown in Formula (6), the independent variable in this formula is $S_i$, which is the range that the UAV uses to search for the best position, and the dependent variable is the cost of optimizing the sensing links $F_3$, which is generated by the UAV when searching for the best position. When $S_i \le R_S$, that is, when the distance the UAV uses to search for the optimal position is less than its sensing range, the sensing links $F_3$ is defined as $|R_S - S_i|$, which means the larger the range that the UAV uses to search for the next position, the fewer times it needs to search for the optimal position within the communication range, and the lower the cost incurred; that is, $F_3$ is smaller. When $S_i > R_S$, the distance needed to search for the best position exceeds the range that the UAV can sense, which is clearly impossible, indicating that the search mechanism has encountered a problem. At this point, the best position is inaccurate and needs to be searched for again to provide the UAV with the next best position, as the sensing links $F_3$ is $\infty$.

(4) Avoiding obstacles $F_4$:

$$F_4 = \begin{cases} 0 & if\ d_m > R_m + R_u + S \\ R_m + R_u + S - d_m & if\ R_m + R_u < d_m \le R_m + R_u + S \\ \infty & if\ d_m \le R_m + R_u \end{cases} \tag{7}$$

As illustrated in Figure 3 and shown in Formula (7), the UAV is represented as a flying particle with position $P_i$, and its size is $R_u$. The obstacle K is represented as a circle with center $O_m$ and radius $R_m$. The UAV needs to maintain a safe distance $S$ to avoid obstacles. Let $d_m$ be the distance between $P_i$ and $O_m$. The independent variable in this Formula (7) is $d_m$, which is the distance between the center of the UAV and the center of the obstacle. The dependent variable is obstacle avoidance $F_4$, which refers to the cost incurred by the UAV when maintaining a certain safe distance from the obstacle. When $d_m > R_m + R_u + S$, the UAV is far beyond the safe distance S, which is very safe. At this time, obstacle avoidance $F_1$ is 0, which means there is no cost, indicating that the UAV is not in danger of collision. When $R_m + R_u < d_m \leq R_m + R_u + S$, the UAV is within a safe distance $S$ and incurs costs, indicating that the UAV may be in danger. Therefore, obstacle avoidance $F_4$ is defined as $R_m + R_u + S - d_m$, meaning that the smaller the $d_m$, the higher the cost incurred and the more likely it is to be at risk of collision. When $d_m \leq R_m + R_u$, the UAV will definitely collide with obstacles, and the cost is infinite. Obstacle avoidance $F_4$ is $\infty$.

## 4. The UMMVN Method

### *4.1. The Framework of UMMVN*

In this study, the method for deploying UAVs for multiple targets is shown in Figure 5. Firstly, the UAV deployment is carried out for target 2. The UAVs are deployed in sequence, and each UAV uses the MOPSO algorithm to determine the next optimal position within its sensing range. A behavior-based controller is used to navigate the UAV to that position and to continue the search until it reaches the communication boundary. Then, the closest position within the communication boundary is selected as the final position of the UAV, and the process is repeated until it reaches the target. Afterwards, the optimal branching node strategy is used to select the branching point. The UAVs at targets 1 and 3 first fly to the branching node in sequence; then, they perform path optimization. The above process is repeated to deploy the UAVs until they reach targets 1 and 3.

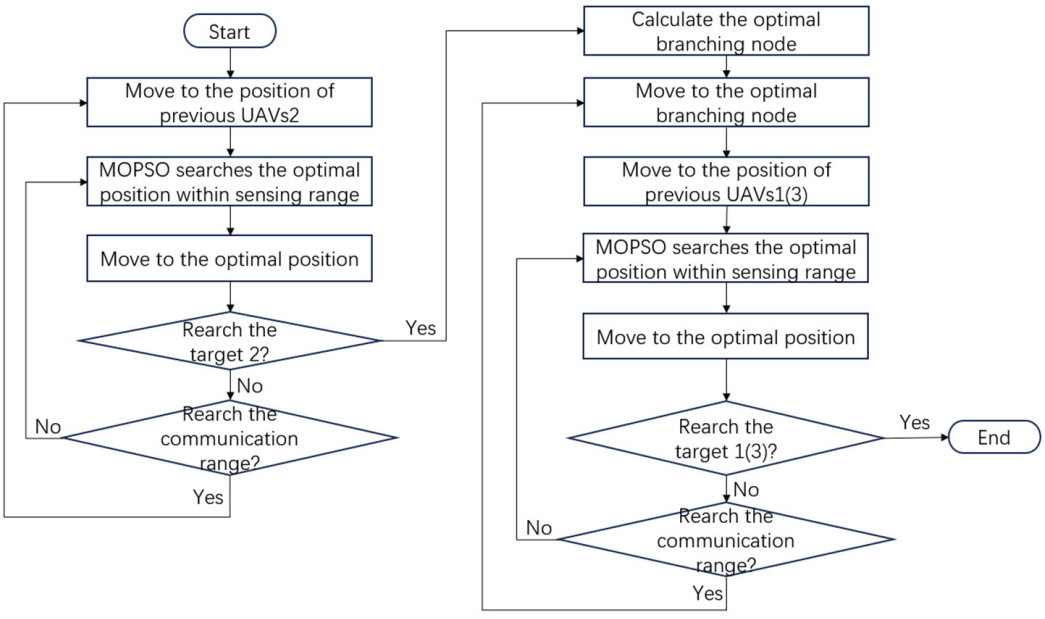



**Figure 5.** Deployment approach diagram.

The framework of the UMMVN mainly includes the trajectory planning method, relay deployment method, and multi-target searching method.

#### 4.1.1. MOPSO-Based Trajectory Planning Method

Multi-objective particle swarm optimization (MOPSO) is a non-convex optimization algorithm which imitates a bird swarm's flying pattern to iteratively search for the optimal solution. In the SAR task, the MOPSO is deeply embedded into the searching process of the UAVs, which means that each UAV is a particle, and the UAV's trajectory in the complex environment is equal to a particle's searching path in the searching region.

Algorithm 1 demonstrates how the MOPSO algorithm is used to implement a planning trajectory. Before implementing the experiment, it sets the value of the best solution $gbest$ in a population and the maximum number of iterations $IterMax$. Firstly, it generates a set of random particles, and each particle represents a possible position of the UAV. The obstacles in the complex environment can be recognized by the sensors carried by the UAV. Then, it calculates the multi-objective fitness value of each particle $i$ through Formula (4) and then uses the multi-objective density-based evaluating method to find the best solution of the current generation. Thirdly, it updates the best cost value $pbest_i$ and the best cost value $gbest$ of the population for particle $i$ and updates the velocity $v_i$ and position $x_i$ of particle $i$ according to Formulas (8) and (9). Fourthly, it repeats this process until the maximum iteration value is reached. Finally, the MOPSO chooses the location related to the final $gbest$ as the optimal location for the UAV.

$$v_i(t+1) = \omega v_i(t) + s_1 r_1(pbest_i(t) - x_i(t)) + s_2 r_2(gbest(t) - x_i(t)) \tag{8}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{9}$$

where $pbest_i$ is the best position of particle $i$; $gbest$ is the best position of the swarm; $s_1$ and $s_2$ are, respectively, the individual and global acceleration coefficients; and $r_1$ and $r_2$ are two random values in the range of (0,1).

**Algorithm 1.** MOPSO-based planning trajectory

**Input:** $gbest$, and $IterMax$
**Output**: Optimal position
1. Initialize: MOPSO initial population
2. Mapping obstacles in environment into restricted area in the searching region
3. **for** $k$ from 1 to $IterMax$ do
4. **for** Particle $i$ in solutions do
5. Calculate cost value $\{F_1, F_2, F_3, F_4\}$ of Particle $i$
6. Evaluate Particle $i$'s priority in terms of four objectives and density
7. Update $gbest$ and $pbest_i$
8. Update velocities $v_i$ and positions $x_i$
9. **end**
10. **end**

#### 4.1.2. Single-Target Relay Deployment Method

The single-target deployment method is a networking, sensing, and navigating integration method that is used to guide the UAV from the base station to the target using MOPSO. It includes four states: idle, movement, explore, and occupied. The details are illustrated as follows:

"Idle": The UAV is not working when it is waiting for any messages or instructions from the base station. Once the UAV receives the instructions from the previous UAV on the search, its state immediately transforms to "Movement".

"Movement": After receiving the instructions, the UAV follows the existing trajectory and moves to the positions of the previous UAVs. Then, it arrives at the optimal position found by the last UAV. After reaching the current optimal position, its state transforms to "Explore".

"Explore": After the UAV flies to the position of the previous UAV, the MOPSO algorithm is used to explore the next optimal position within its sensing range. Given that

the sensing distance is smaller than the communication distance, it iteratively implements this process until it arrives at the communication boundary. The position with the smallest distance to the communication boundary is selected as the final position of the UAV. When the UAV stays at that position, its state transforms to "Occupied."

"Occupied": After the UAV stays at the position, if the position is not that of the target, the UAV will broadcast its position to the idle UAV located at the base station, and then the idle UAV repeats the abovementioned four steps until it finds the target. If the position is that of the target, the multi-hop ad hoc communication network is constructed.

Figure 6 shows the deployment process of a single target relay, which is as follows:

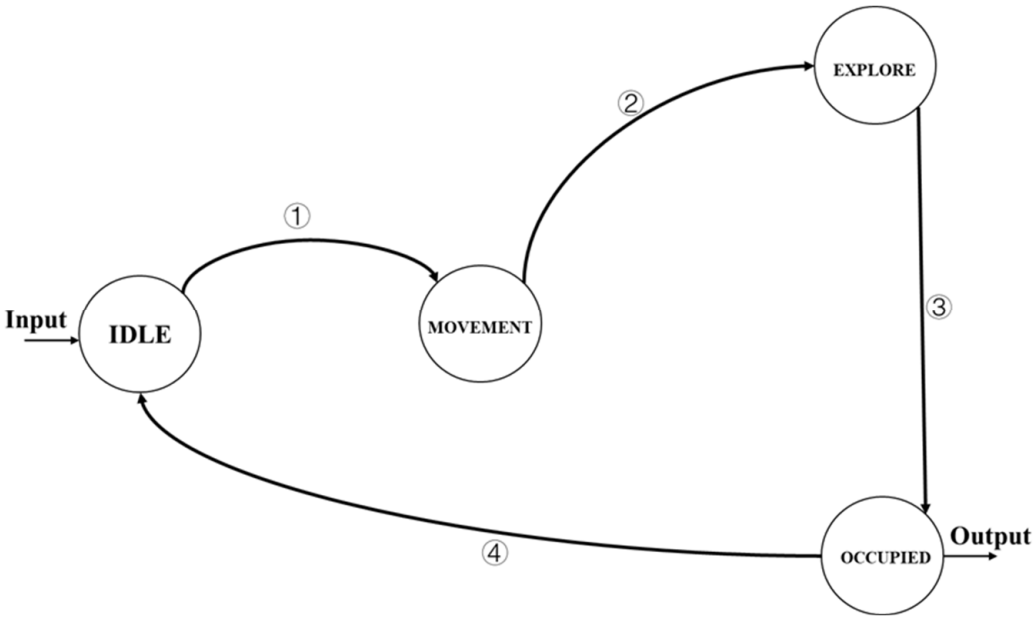



**Figure 6.** Single-target relay deployment.

① Receive instructions from the previous UAV on the search;

② UAV flies to the current optimal position found by the last UAV;

③ UAV uses MOPSO to explore and arrives at the optimal position within its sensing range;

④ If the position is not that of the target, the UAV will broadcast its position to the idle UAV located at the base station.

#### 4.1.3. Multi-Target Searching Method

The multi-target searching method is a multi-target UAV deployment plan, as illustrated in Algorithm 2. Firstly, single-target relay deployment is used to deploy the UAV to target 2. After deployment, the optimal fork point strategy is used to calculate the two points with the smallest distance from the base station. The UAVs are sent to target 1 and target 3 as the two optimal fork points. Then, a UAV is dispatched to fly to the optimal fork point, and the UAVs are deployed to target 1. The process is repeated until target 1 is reached. The process for deploying UAVs to target 3 is similar to that used for target 1.

Taking three target points as an example, the drones are divided into three categories. UAVs1 represents the drones deployed from the starting point to target 1, UAVs2 represents the drones deployed from the starting point to target 2, and UAVs3 represents the drones deployed from the starting point to target 3. First, the distance between the starting point and target 1 is calculated, initialized to m1, and the distance between the starting point and target 3 is calculated, initialized to m2. Then, the Single target relay deployment method is used to deploy UAVs2 to target 2. At this point, record the positions of each UAV and calculate the positions of the UAVs closest to targets 1 and 3. Select the value with the smallest distance and assign it to m1 and m2, respectively. Record the position of the UAV with the

smallest distance from targets 1 and 2 and assign it to P1 and P2. Use P1 and P2 as the optimal bifurcation points for targets 1 and 3. Afterwards, path optimization is performed on target 1 and target 3. For target 1, UAVs1 first flies to the position of P1, and then the Single target relay deployment method is used to deploy UAVs1 on target 1. Similarly, the deployment of drones at target 3 is the same. When all three targets are deployed, the deployment of multi-target and multi hop self-organizing network drones is completed.

**Algorithm 2.** Multi-objective Multi-hop Ad-hoc Communication Route Deployment Strategy

```
Initialization
m1: Calculate the distance between the start and the target 1
m2: Calculate the distance between the start and the target 3
minimum_distance1 = m1
minimum_distance2 = m2
Single objective multi-hop ad-hoc communication route deployment of UAVs2 to target 2
Record the position of the UAVs2
for i = 1:end
    d1: Calculate the distance between the UAVs2(i) and target 1
    d2: Calculate the distance between the UAVs2(i) and target 3
        if d1 < minimum_distance1
            minimum_distance1 = d1
            P1 = the position of the UAVs2(i)
        end
        if minimum_distance1==m1
            P1 = start
        end
        if d2 < minimum_distance2
           minimum_distance2 = d2
            P2 = the position of the UAVs2(i)
        end
        if minimum_distance2==m2
            P2 = start
        end
end
While not reach target 1
    UAVs1(j) moves to P1
    Single objective multi-hop ad-hoc communication route deployment of
    UAVs1 to target 1
end
While not reach target 3
    UAVs3(k) moves to P2
    Single objective multi-hop ad-hoc communication route deployment of
    UAVs3 to target 3
end
```

### *4.2. Optimal Branching Node Strategy*

This study proposes an optimal branching node strategy for multi-target UAV deployment to improve the resilience, flexibility, and reliability of the networking and to reduce the networking costs.

The principle of this strategy is as follows.

Firstly, UAV deployment was carried out for target 2, with a total of three UAVs deployed from the base station to target 2 (excluding the target-connected vehicles). After deployment, the Euclidean distances between the base station, the three UAVs, and the connected vehicles located at target 2, target 1, and target 3 were calculated. The reason for choosing the Euclidean distance is that it is the true distance between two points in n-dimensional space, reflecting the real situation, and its calculation results are more realistic

and accurate. The method used to calculate Euclidean distance is shown in Formula (10) (three-dimensional space).

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \tag{10}$$

where $[x_1, y_1, z_1]^T$ and $[x_2, y_2, z_2]^T$ are the coordinate position of any two points; $d$ is the Euclidean distance between the two points.

Secondly, when comparing the calculation results, it is necessary to select the point with the smallest Euclidean distance as the branching node. The reason is that due to the minimum distance between the branching node and the target location, it has the least number of deployed UAVs and the shortest search path, which greatly reduces the search time for the optimal location and the deployment time of the UAVs. Therefore, this branching node is optimal. As shown in Figure 7, among the five points, the closest point to target 1 is point B. Therefore, the position of point B is recorded, and point B is used as the branching node for target 1. Similarly, the closest point to target 2 is point A. Therefore, the position of point A is recorded, and point A is used as the branching node for target 2. The point with the smallest distance is selected as the branching node based on the distance between each UAV and the other two target points.

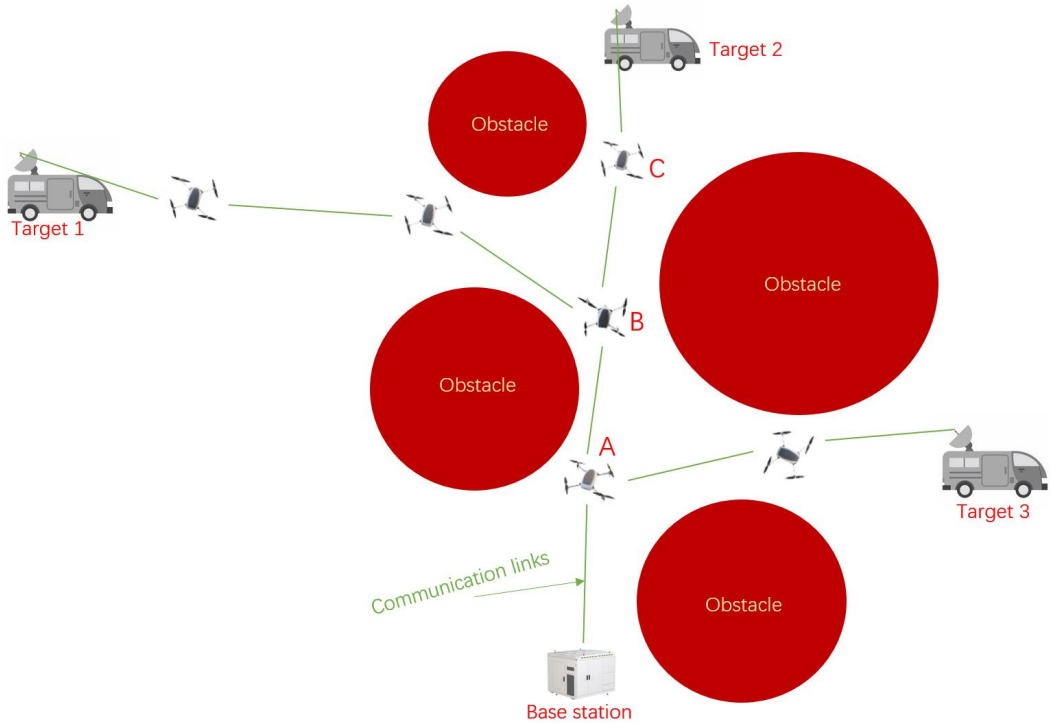



**Figure 7.** Optimal branching node strategy.

Finally, for targets 1 and 3, the UAVs are dispatched to fly to their respective branching nodes, and then the MOPSO algorithm is used to conduct a relay search within the UAV communication range to find the optimal position of the UAV. This process is repeated until the target is reached and the deployment of the UAV is completed.

The selection of branching nodes in this branching node strategy can be changed according to the target and starting point, with high flexibility. At the same time, the use of this branching node strategy for UAV deployment adopts distributed networking, which improves the resilience and reliability of the network.

### *4.3. Searching Algorithm*

#### 4.3.1. MOPSO Algorithm

MOPSO algorithm is a swarm intelligence based optimization algorithm that seeks the optimal solution to a problem by simulating the foraging behavior of social animals such as birds and fish. The following is the basic architecture of the MOPSO algorithm:

Initialization: Randomly generate a set of particles in the search space as the initial population. Each particle represents a potential solution to the problem and has a position (representing the specific value of the solution) and a velocity (representing the direction and velocity of the solution's change).

Fitness evaluation: For each particle in the population, use a fitness function to calculate its fitness value. The specific form of the fitness function depends on the specific problem to be solved.

Update individual and global optima: Each particle remembers the best position they have ever found (individual optimal solution), that is, the individual optimal position (pBest) and the individual optimal fitness value. At the same time, the entire population will also remember the best position (global optimal solution) that all particles have found, namely the global optimal position (*gBest*) and the global optimal fitness value.

Update speed and position: In each iteration, particles update their speed and position based on their individual optimal position and the global optimal position of the population.

Iteration process: Repeat the fitness evaluation, update individual and global optima, update speed and position until the termination conditions are met.

Termination and output: When the termination condition is met, the algorithm stops iterating and outputs the global optimal position as the optimal solution to the problem.

#### 4.3.2. NSGA2 Algorithm

The NSGA2 algorithm is a search algorithm that simulates the process of biological evolution. It simulates Darwin's natural selection theory through selection, crossover, and mutation operations. In genetic algorithms, a solution is a chromosome, and a population is a set of chromosomes. The overall architecture of the NSGA2 algorithm is as follows:

Initialize: Create an initial population.

Selection: Select the optimal or highly adaptable individual from the current population to generate the next generation.

Cross over: Selected individuals exchange information in some way to generate new solutions.

Mutation: In order to maintain the diversity of the population, algorithms will randomly change some features of new individuals.

Calculate individual fitness: The fitness of each individual (i.e., the quality of the solution) is determined by a fitness function.

Termination: When the termination conditions are met, select the individual with the highest fitness as the optimal solution.

#### 4.3.3. Algorithm Comparison

The MOPSO algorithm and NSGA2 algorithm are both evolutionary algorithms that seek the optimal solution through population search. They are all metaheuristic algorithms used to solve optimization problems, capable of handling complex optimization problems and searching for global optimal solutions in multidimensional spaces. The main difference between them lies in the optimization and iteration methods.

(1) Optimization method:

The MOPSO algorithm is a group collaboration based algorithm that searches for the optimal solution by simulating the behavior of bird or fish schools. In the MOPSO algorithm, each individual (particle) updates its position and velocity iteratively, and individuals can communicate and cooperate with each other.

The NSGA2 algorithm is a biological evolution based algorithm that searches for optimal solutions by simulating natural genetic mechanisms. In the NSGA2 algorithm, individuals in the population are updated through operations such as selection, crossover, and mutation, gradually evolving into better individuals.

(2) Iteration method:

The iterative approach of the MOPSO algorithm is to search for the optimal solution by updating the position and velocity of each particle. During the iteration process, each particle will calculate its next position based on its current position and velocity, and update its own optimal position and global optimal position.

The iterative approach of the NSGA2 algorithm is to update individuals in the population through genetic operations (selection, crossover, and mutation). During the iteration process, individuals with high fitness are selected based on the fitness function to participate in crossover and mutation operations, gradually optimizing the population.

Compared to the NSGA2 algorithm, the MOPSO algorithm has fewer parameters, relatively simple parameter settings, and does not require complex adjustments to parameters such as crossover probability and mutation probability. Therefore, the MOPSO algorithm is simpler and easier to implement. The MOPSO algorithm usually has faster convergence speed, stronger exploration ability in search space, better global search ability, easier avoidance of local optima, and can find the optimal solution faster. The MOPSO algorithm has better adaptability to high-dimensional optimization problems and nonlinear problems, and can better handle complex optimization problems.

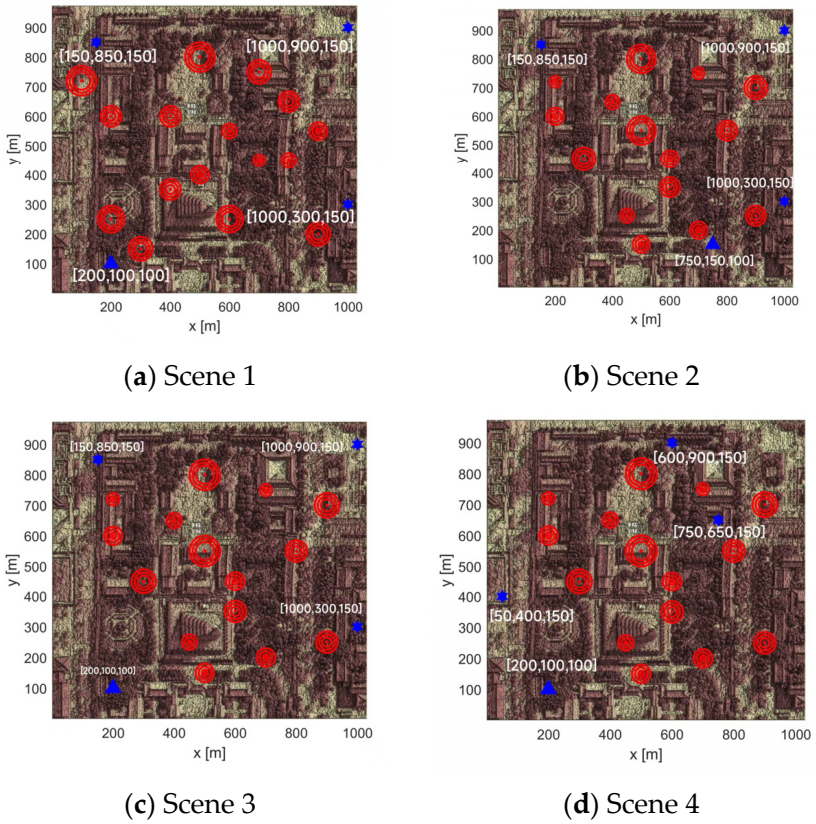
## 5. Experiment

### *5.1. Experiment Setup*

In this experiment, MATLAB R2023a software was used for the simulation experiments, in which the required scene was generated using a digital elevation model (DEM) map and attaching the obstacles. This scenario included a base station, three targets, and 15 or 16 different obstacle area ranges. This simulation experiment mainly generated four scenarios, among which the last three scenarios were compared with the first scenario, which comprised different base station positions, target positions, and obstacle distributions and regional ranges. The parameters of the UAV itself were set, with a sensing range of $R_S$ = 60 m, a communication range of $R_C$ = 360 m, a UAV radius of $R_u$ = 1 m, and a safe distance of $S$ = 10 m between the UAV and the other obstacles. At the same time, this experimental simulation was mainly divided into two parts. Firstly, the MOPSO algorithm found the optimal location for the deployment of the unmanned aerial vehicles. Therefore, the parameter settings in the MOPSO algorithm were as follows: the maximum number of iterations $maxIter$ = 120, the MOPSO population size was 80, the inertia coefficient $\omega$ = 0.8, and the damping ratio was 0.99. The second part involved navigating the UAV to the optimal position, which included three behaviors: navigation to the target point, avoidance of obstacles, and avoidance of other UAVs. Therefore, the parameter settings of the behavior-based controller were as follows: the two parameters for navigation to the target point were set to $a_d$ = 30, $b_d$ = 20; the two parameters for the avoidance of obstacles were set to $a_o$ = 30, $b_o$ = 15; and the two parameters for the avoidance of UAVs were set to $a_u$ = 30, $b_u$ = 20.

The parameter settings of the four scenes are as follows:

As shown in Figure 8, the blue triangles depict the base stations, the blue hexagons signify the target points, while the red circles indicate the obstacles. Scene 1 and scene 3 have different obstacles, scene 2 and scene 3 have different base station positions, and scene 4 and scene 3 have different target positions.




(**a**) Scene 1 (**b**) Scene 2

(**c**) Scene 3 (**d**) Scene 4

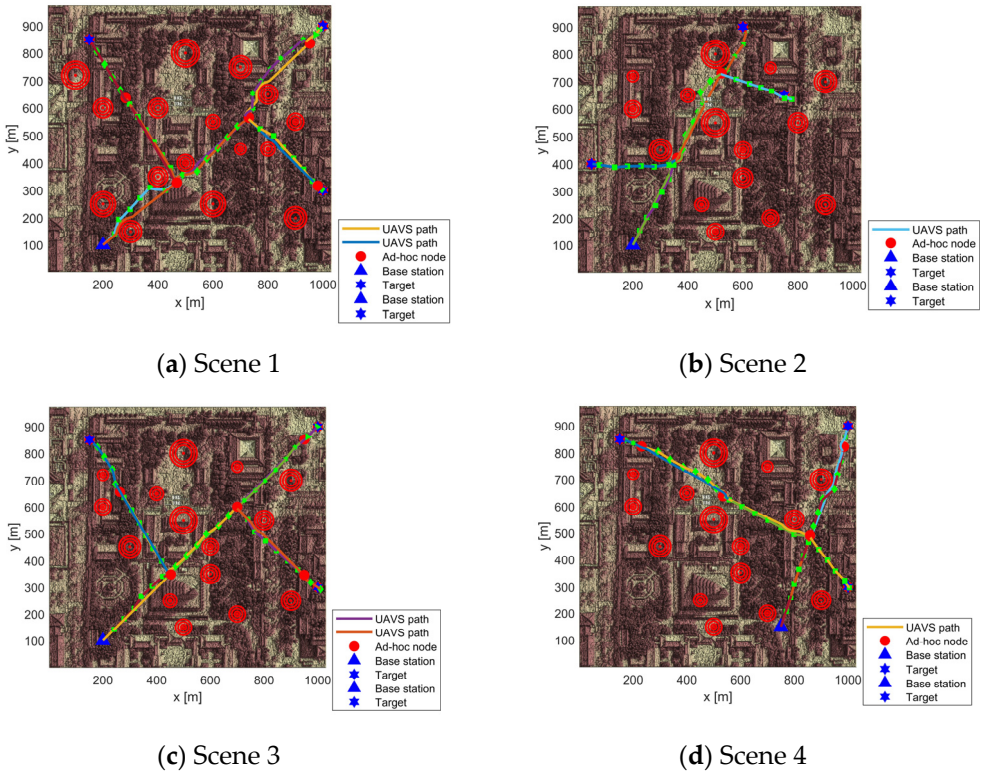
**Figure 8.** Scene parameter settings.

### *5.2. Experiments Results*

Firstly, regarding parameter settings, a safety distance of 10 m was set between the UAV and the obstacles. Within the allowable range of error, this safety distance was sufficient to ensure that the UAV did not collide with the obstacles. Secondly, the MOPSO algorithm set the number of iterations to 120, which ensured that the particles had a sufficiently large search space and were more likely to find the optimal solution and achieve good optimization results. The inertia coefficient set by the MOPSO algorithm was tested multiple times and was finally selected as 0.8, which balanced the global and local optimal positions, making the experimental results more ideal. In the MOPSO algorithm, the damping ratio was continuously adjusted based on the experimental results and multiple tests, and the final decision was to set the damping ratio to 0.99, so that the system vibration speed tended to the equilibrium position as quickly as possible without generating oscillations.

Figures 9 and 10 show the deployment path of the UAVs from a two-dimensional perspective and the deployment effect of the UAVs in three-dimensional space, respectively; Table 1 shows the data for this simulation; Table 2 shows the convergence rate of the MOPSO-based optimal position searching algorithm.

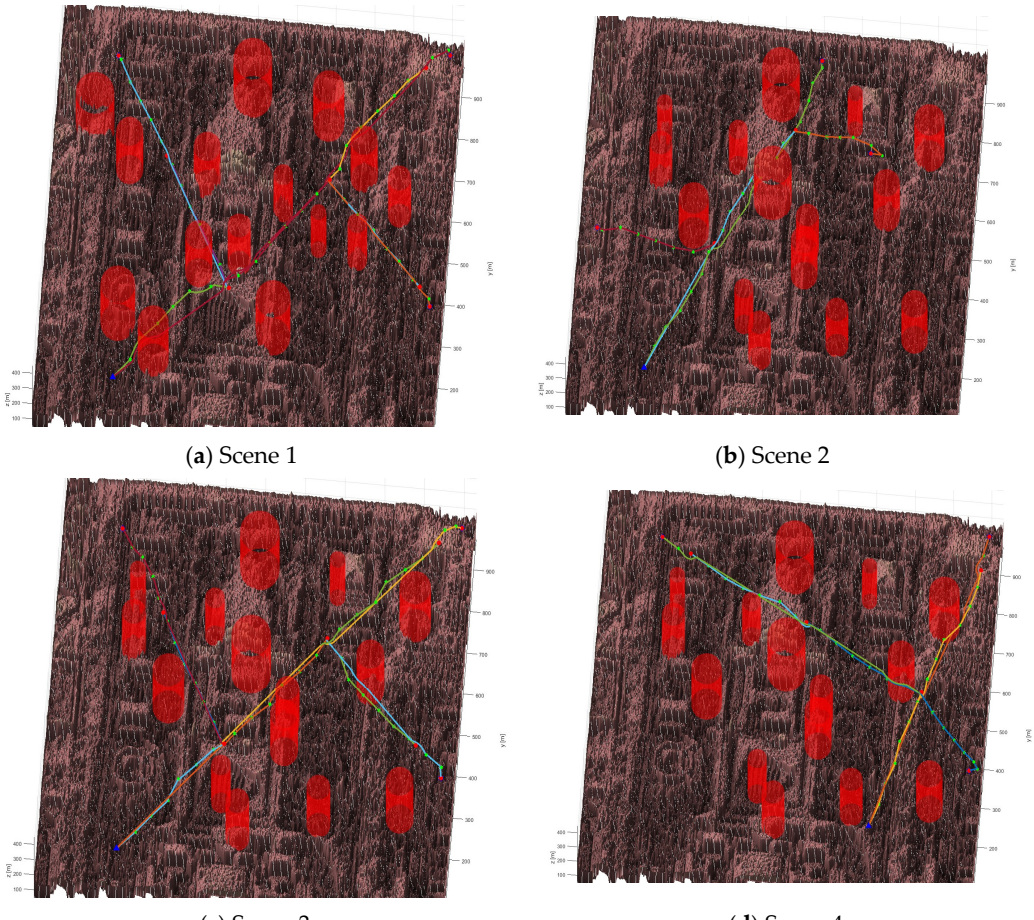
As shown in Figure 9, the green icon represents the best position searched by the MOPSO algorithm within the sensing range after each iteration, and different colored lines represent the flight routes of different drones. The meanings of the other icons are the same as those of the icons in Figure 8. Figure 9 shows the UAV deployment paths. The obstacles in scene 1 and scene 3 are different, the target positions in scene 2 and scene 3 are different, and the base station positions in scene 4 and scene 3 are different. From these four result graphs, it can be seen that the base station position, target position, and obstacles in the scene can be changed. No matter how the scene changes, the UAV can avoid obstacles and find the best flight route and deployment position, and the effect is very successful. This shows the flexibility and stability of the system.




(**a**) Scene 1 (**b**) Scene 2

(**c**) Scene 3 (**d**) Scene 4

**Figure 9.** UAV deployment paths.



(**a**) Scene 1 (**b**) Scene 2

(**c**) Scene 3 (**d**) Scene 4

**Figure 10.** UAV deployment 3D effect.

**Table 1.** Performance results of the UAV deployment.

| Scene | UAVs | | Link Length (m) | Actual Total Link Length (m) | Ideal Link Length (m) | Temporary Goal Number | Time Consuming (s) | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | Searching | Searching Time/ Temporary Goal (s) |
| 1 | uavs1 | 1 | 355.2071 | 583.8348 | 588.9528 | 7 | 5.4969 | 0.7853 |
| | | 2 | 228.6277 | | | 4 | 3.2112 | 0.8028 |
| | uavs2 | 1 | 354.8361 | 1140.155 | 1134.6347 | 7 | 5.6935 | 0.8134 |
| | | 2 | 355.3513 | | | 8 | 7.0538 | 0.8817 |
| | | 3 | 355.2691 | | | 7 | 5.4157 | 0.7737 |
| | | 4 | 74.6985 | | | 2 | 1.5258 | 0.7629 |
| | uavs3 | 1 | 355.5759 | 415.2266 | 400.7387 | 7 | 5.4158 | 0.7737 |
| | | 2 | 59.6508 | | | 1 | 0.74432 | 0.7443 |
| 2 | uavs1 | 1 | 308.0387 | 308.0387 | 304.7666 | 6 | 4.5579 | 0.7597 |
| | uavs2 | 1 | 354.9888 | 897.6647 | 894.4725 | 7 | 5.5376 | 0.7911 |
| | | 2 | 355.2532 | | | 8 | 7.2867 | 0.9108 |
| | | 3 | 187.4227 | | | 4 | 3.6366 | 0.9092 |
| | uavs3 | 1 | 243.4712 | 243.4712 | 231.849 | 5 | 4.0102 | 0.8020 |
| 3 | uavs1 | 1 | 355.3137 | 566.4317 | 570.6765 | 7 | 5.8619 | 0.8374 |
| | | 2 | 211.118 | | | 4 | 3.4564 | 0.8641 |
| | uavs2 | 1 | 355.2151 | 1167.4998 | 1143.1955 | 7 | 5.6761 | 0.8109 |
| | | 2 | 355.1901 | | | 7 | 5.8394 | 0.8342 |
| | | 3 | 355.1351 | | | 7 | 5.653 | 0.8076 |
| | | 4 | 101.9595 | | | 2 | 1.6899 | 0.8450 |
| | uavs3 | 1 | 355.272 | 438.801 | 449.6769 | 7 | 5.7628 | 0.8233 |
| | | 2 | 83.529 | | | 2 | 1.7585 | 0.8793 |
| 4 | uavs1 | 1 | 354.2717 | 814.7755 | 807.2607 | 7 | 5.8517 | 0.8360 |
| | | 2 | 358.7073 | | | 8 | 7.2191 | 0.9024 |
| | | 3 | 101.7965 | | | 2 | 1.6275 | 0.8138 |
| | uavs2 | 1 | 355.1346 | 791.1892 | 790.7522 | 7 | 5.7114 | 0.8159 |
| | | 2 | 355.3802 | | | 7 | 5.859 | 0.8370 |
| | | 3 | 80.6744 | | | 1 | 0.83004 | 0.8300 |
| | uavs3 | 1 | 227.6825 | 227.6825 | 232.0162 | 6 | 4.9498 | 0.8250 |

As shown in Figure 10, the red cylindrical icon represents the obstacles. The meanings of the other icons are the same as those of the icons in Figure 9. Figure 10 shows the deployment effect of the UAVs in three-dimensional space. From the perspective of three-dimensional space, the deployment effect of the drones shows successful avoidance of the obstacles, and the best flight route and deployment position can be found; this is almost identical to the effect of the two-dimensional drone deployment paths in Figure 9.

Table 1 shows the performance results of the UAV deployment:

(1) MOPSO search optimal time: The sensing range of the UAV was 60 m, and the flight speed was set to 20 m/s. Therefore, it took 3 s for the UAV to fly from the optimal point to the latest searched optimal point. From the perspective of iteration and search time, its search time was within 1 s, ensuring real-time performance.

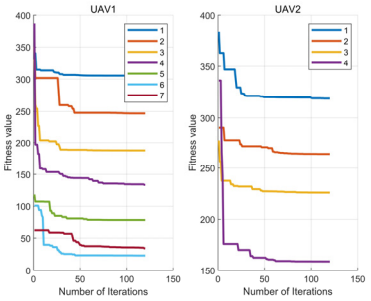
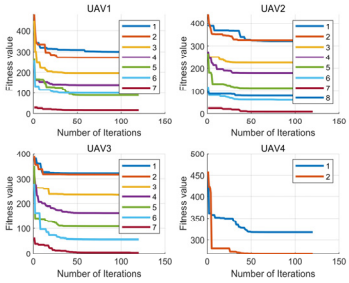
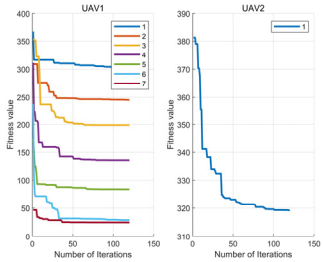
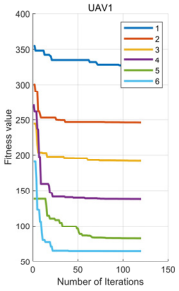
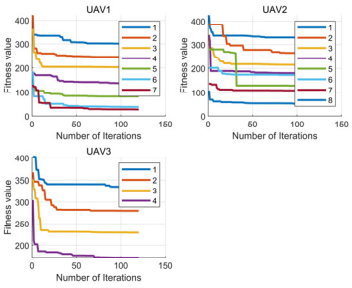
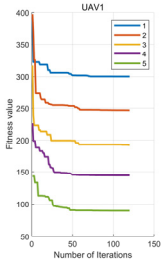
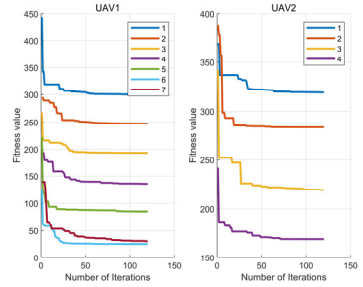
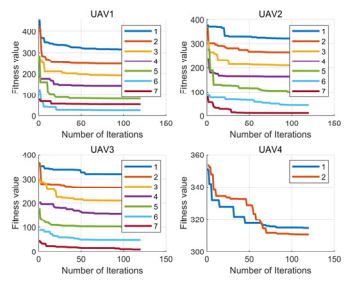
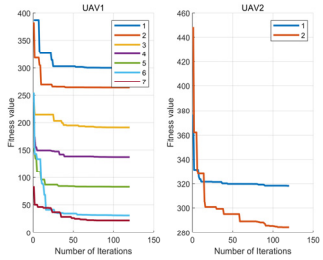
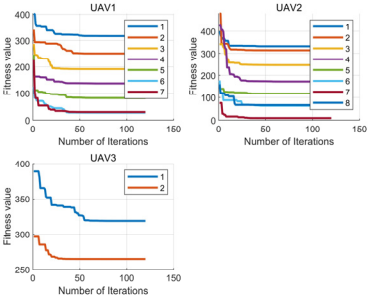
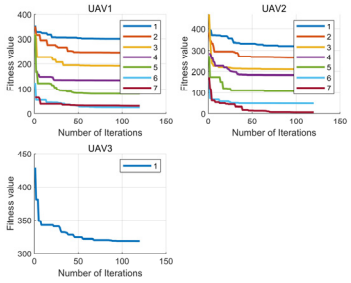
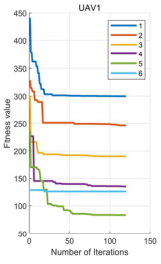
(2) Link length between UAVs: According to the data, the distance between every two UAVs was less than 360 m, indicating that they could communicate normally. The average intermediate link length was around 355 m, which was close to 360 m. Therefore, fewer UAVs were required, which reduced the costs.

(3) Total link length: According to the data, the difference between the total ideal link length and the actual link length was relatively small. Within an acceptable range, the reason for the error was due to the presence of obstacles. However, it was found in some of the data that the actual link length was smaller than the ideal link length.

This was because the target point reached by the UAV during the flight deviated from the actual target point, resulting in this situation.

Table 2 shows the convergence rate of the MOPSO-based optimal position searching algorithm. After multiple experiments and constant adjustments to the parameters, the final iteration number was determined to be 120. As shown in the result graph in Table 2, the MOPSO algorithm had the fastest iteration speed from 0 to 50, and the iteration speed from 50 to 100 was very slow with almost no change, indicating that the optimal position was reached. Only the last UAV's final iteration experienced shaking, which was due to the fact that the optimal position should have been located near to the target when it was about to arrive. However, during the 100 to 120 iterations, all the drones tended to flatten out, indicating that the searched position was the optimal position and would not change again.

**Table 2.** Convergence rate of the MOPSO-based optimal position searching algorithm.

| Scene | Target 1 | Target 2 | Target 3 |
|---|---|---|---|
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| 4 |  |  |  |

From the experimental results, firstly, the MOPSO algorithm took much less time to search for the optimal position than the time taken by the UAV to fly to that point, ensuring a real-time search for the optimal position while flying. Secondly, the distance between the two UAVs was within the communication sensing range, which fully met the communication requirements and was very close to the communication range. Therefore, in terms of the number of UAV deployments, it can greatly save UAV costs. Thirdly, as shown in Figures 8 and 9, the selection of the branching node was consistent with the ideal branching node and was more in line with the actual situation. At the same time, the branching node could vary according to the position of the targets, obstacles, and base station. The proposed branching node strategy was very flexible in its application.

We also analyzed the advantages and disadvantages of the branching nodes, as follows:

Advantages: As shown in Figure 7, when the distance from the base station to target 3 differs significantly from the distance from A to target 3, three UAVs are needed from the base station to target 3, while only two UAVs are needed from A to target 3. Moreover, the link length between the two is significantly different, which reduces costs and time. Therefore, the further the distance between target 1 and target 3 and target 2, the better the choice of the branching node.

Disadvantages: As shown in Figure 7, when the distance between A and target 1 and between B and target 1 is small and the required number of UAVs for both is the same, and there is not much difference in the search link, the application of the optimal branching node strategy is not very good. Therefore, when selecting point A or B to become the branching node, certain constraints need to be added in order to choose a better branching node.

### *5.3. Comparative Analysis of Algorithm Performance*

More importantly, in order to search for the optimal location for drone deployment, we adopted the MOPSO algorithm. At present, other algorithms that can solve this problem include the NSGA2 algorithm. Therefore, we compared the MOPSO algorithm with the NSGA2 algorithm. Based on the characteristics of the two algorithms, in order to ensure fairness, we continuously adjust the experimental parameters to ensure that the difference between the indicator values and the ideal indicator values in the two algorithms is within a certain range, in order to find the optimal solution between the two algorithms separately, and the difference between the optimal solutions of the two algorithms is within a certain range, in order to ensure that the search time is the best indicator value for comparing the two algorithms.

Therefore, we controlled the position error of the two algorithms to be very small, which is measured by the position RMSE. At the same time, we also controlled the angle error of the two algorithms to be very small, which is measured by the angle RMSE. After controlling the position and direction of the two algorithms within a certain error range, the performance of the algorithms in emergency rescue is measured by comparing the search time for the optimal position between the two algorithms.

#### 5.3.1. Indicator Values

RMSE (Root Mean Squared Error) is a statistical measure commonly used to measure prediction model errors. It represents the square root of the average square of the difference between the predicted value and the actual observed value.

The calculation formula for RMSE is as follows:

$$\mathrm{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(Y_i - \hat{Y}_i\right)^2} \tag{11}$$

where $Y_i$ is the actual value of the $i$-th observation, $\hat{Y}_i$ is the predicted value of the $i$-th observation, and $m$ is the total number of observations.

For the RMSE($P$), its calculation formula is as follows:

$$\mathrm{RMSE}(P) = \sqrt{\frac{1}{3}\left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (z_i - \hat{z}_i)^2\right]} \tag{12}$$

where $(x_i, y_i, z_i)$ is actual position, $(\hat{x}_i, \hat{y}_i, \hat{z}_i)$ is ideal position, RMSE$(P)$ is position RMSE.

For the RMSE($A$), its calculation formula is as follows:

$$\mathrm{RMSE}(A) = \sqrt{(a_i - \hat{a}_i)^2} \tag{13}$$

where $a_i$ is actual angle, $\hat{a}_i$ is ideal angle, RMSE$(A)$ is angle RMSE.

For the RMSE(*average*), its calculation formula is as follows:

$$\mathrm{RMSE}(average) = \frac{1}{n}\sum_{j=1}^{n}\mathrm{RMSE}(j) \tag{14}$$

where RMSE$(j)$ is the $j$-th RMSE, $n$ is the total number of RMSEs, and RMSE $(average)$ is the average RMSE.

5.3.2. Performance Analysis

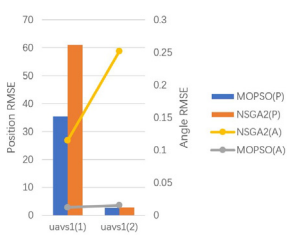
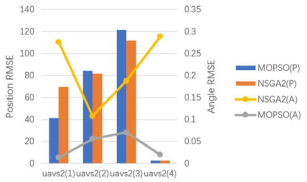
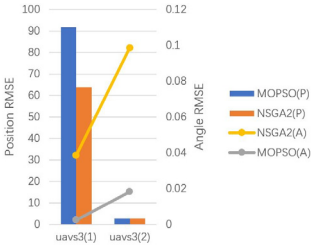
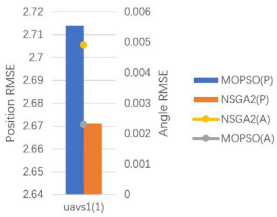
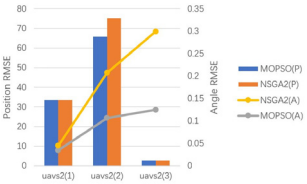
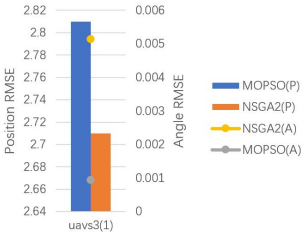
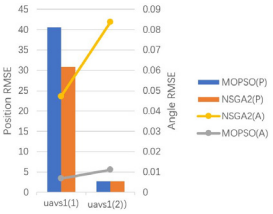
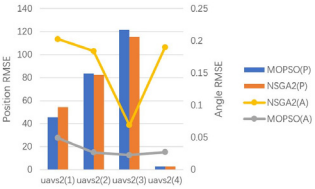
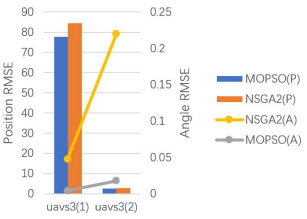
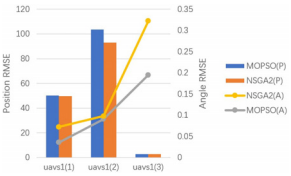
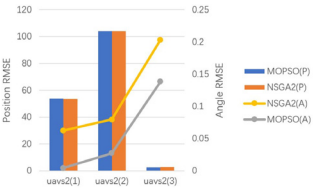
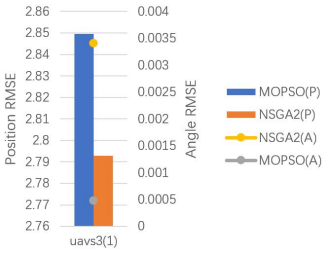
Table 3 shows the comparison of RMSE between MOPSO algorithm and NSGA2 al-gorithm:

As shown in Table 3, uavs1 represents the drone deployed from the base station to target point 1, uavs1 (1) represents the first drone deployed from the base station to target point 1. The bar chart represents the position RMSE of the MOPSO algorithm and NSGA2 algorithm, and the line chart represents the angle RMSE of the two algorithms. The left vertical axis represents the position RMSE, while the right vertical axis represents the angle RMSE. The horizontal axis represents different types of drones. The position RMSE of the MOPSO algorithm is denoted as MOPSO (P), and the angle RMSE is denoted as MOPSO (A). The NSGA2 algorithm follows a similar representation.

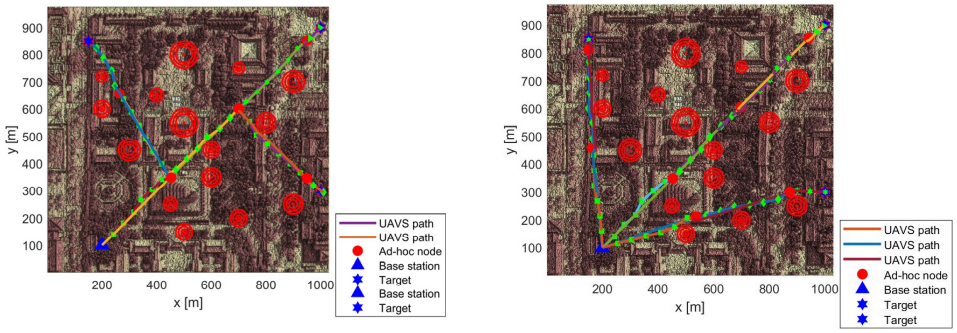
As illustrated in Table 3, the RMSE of drone positions deployed from the base station to various target points varies between the MOPSO and NSGA2 algorithms in different scenarios. Based on the position RMSE evaluation of both MOPSO and NSGA2 algorithms, the performance of the two algorithms appears to be comparable. Across Scenarios 1 and 2, the MOPSO algorithm demonstrates a lower average RMSE, whereas in Scenarios 3 and 4, the NSGA2 algorithm exhibits a lower average RMSE. The average RMSE difference between the two algorithms is minimal, indicating that they both provide similar optimal position results. However, in terms of angle RMSE, the MOPSO algorithm consistently outperforms the NSGA2 algorithm by achieving lower angle RMSE values across various scenarios. According to the average angle RMSE data for both algorithms, the MOPSO algorithm's angle RMSE is consistently lower across all four scenarios, with a significant difference between the two. This indicates that the MOPSO algorithm's search process is more stable, leading to a more consistent flight path for the drone towards the optimal position and enabling it to identify a superior route to reach its designated optimal deployment position.

By comparing the search times for the optimal position within the drone's sensing range, as presented in Table 4, it is evident that the MOPSO algorithm requires significantly less time than the NSGA2 algorithm. This demonstrates that the MOPSO algorithm incurs lower time costs and is thus better suited for emergency rescue operations.

**Table 3.** Comparison of RMSE between MOPSO algorithm and NSGA2 algorithm.

| Scene | Uavs 1 | Uavs 2 | Uavs 3 | Average RMSE |
|---|---|---|---|---|
| 1 |  |  |  | Algorithm / Position / Angle<br>MOPSO 47.7753 0.02627<br>NSGA2 49.5446 0.10987 |
| 2 |  |  |  | Algorithm / Position / Angle<br>MOPSO 21.4970 0.05391<br>NSGA2 23.3936 0.05844 |
| 3 |  |  |  | Algorithm / Position / Angle<br>MOPSO 47.1425 0.02075<br>NSGA2 47.0254 0.10987 |
| 4 |  |  |  | Algorithm / Position / Angle<br>MOPSO 45.7855 0.07024<br>NSGA2 44.1710 0.04986 |

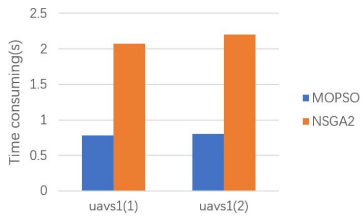
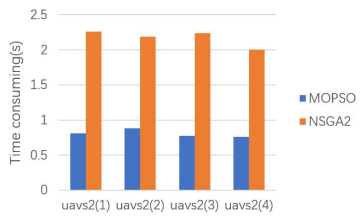
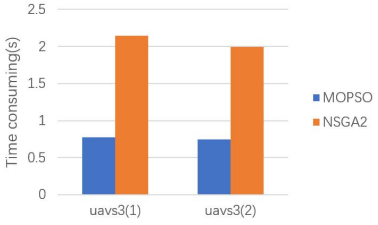
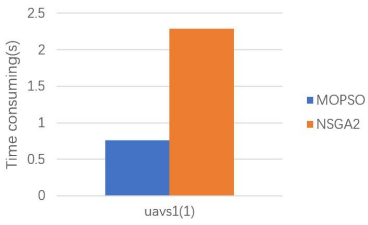
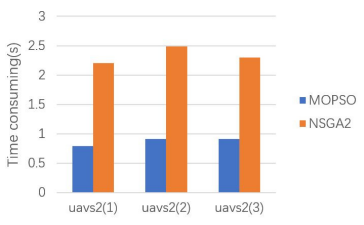
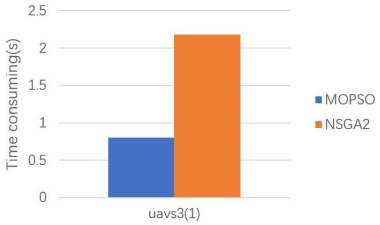
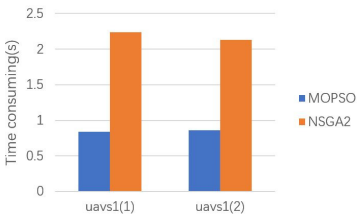
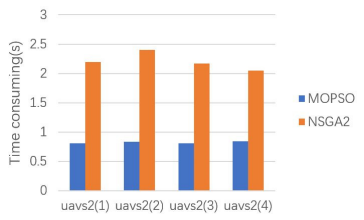
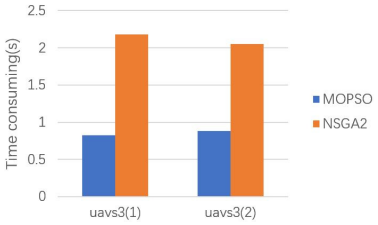
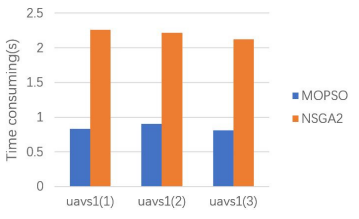
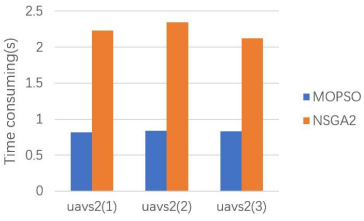
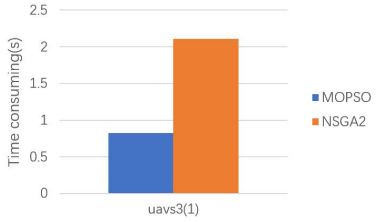
Moreover, for the deployment of unmanned aerial vehicle networks, we propose the optimal branching node strategy. We understand that one of the most advanced methods to solve the problem of networking between base stations and multiple targets is to convert multiple base stations and targets into a single base station and single target form for networking construction. Therefore, we will compare the two methods. To ensure fairness, we chose to deploy drones in the same scenario. Through experiments, as shown in Figure 11, we found that the proposed optimal branching node strategy requires fewer drones to be deployed.




(**a**) Optimal branching node strategy (**b**) Other strategy

**Figure 11.** UAVs deployment paths.

**Table 4.** Search time for optimal position within sensing range.

| Scene | Uavs 1 | Uavs 2 | Uavs 3 |
|---|---|---|---|
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| 4 |  |  |  |

Therefore, through experimental comparison, we found that our proposed UAV-assisted multi-objective and multi-hop ad hoc network not only ensures a short search time for the MOPSO algorithm, but also ensures a small number of drones to be deployed.

## 6. Conclusions

This study proposes a UAV-assisted multi-objective and multi-hop ad hoc network that can be used as an emergency vehicular network (UMMVN). Firstly, it presents an integrated design of a search system to find a trapped vehicle, the communication relay, and the networking, which significantly decreases the UAV's networking time cost. Secondly, it presents a simultaneous multi-objective search for a trapped vehicle and navigates the UAVs along multiple paths to different objectives. Thirdly, it presents an optimal branching node strategy that allows the adequate use of the overlapping paths to multiple targets, which decreases the networking cost within the limited communication and searching range. The numerical experiments illustrate that the UMMVN performs better than other state-of-the-art networking methods.

**Author Contributions:** Conceptualization, H.L.; methodology, X.H.; software, Y.Z.; formal analysis, X.H.; original draft preparation, X.H.; writing—review and editing, H.L.; visualization, J.W.; project administration, F.L.; funding acquisition, F.L. 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 No. 62371368, 62101410), the Foundation of Shaanxi Province (Grant No. QCYRCXM-2022-228).

**Data Availability Statement:** The data presented in this study are available on request from the corresponding author due to juanwen@xidian.edu.cn.

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

## References


1. Prakash, J.; Murali, L.; Manikandan, N.; Nagaprasad, N.; Ramaswamy, K. A vehicular network based intelligent transport system for smart cities using machine learning algorithms. *Sci. Rep.* **2024**, *14*, 468. [CrossRef] [PubMed]
2. Hayat, S.; Yanmaz, E.; Muzaffar, R. Survey on Unmanned Aerial Vehicle Networks for Civil Applications: A Communications Viewpoint. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 2624–2661. [CrossRef]
3. Thuy, N.D.T.; Bui, D.N.; Phung, M.D.; Duy, H.P. Deployment of UAVs for Optimal Multihop Ad-hoc Networks Using Particle Swarm Optimization and Behavior-based Control. In Proceedings of the 2022 11th International Conference on Control, Automation and Information Sciences (ICCAIS), Hanoi, Vietnam, 21–24 November 2022; pp. 304–309. [CrossRef]
4. Lee, H.; Zhu, Y.; Wapenski, D.; Wang, X.; Zhang, Q.; Palacharla, P.; Ikeuchi, T. A Stochastic Process based Routing Algorithm for Wireless Ad Hoc Networks. In Proceedings of the 2019 International Conference on Computing, Networking and Communications (ICNC), Honolulu, HI, USA, 18–21 February 2019; pp. 1018–1023. [CrossRef]
5. Wang, D.; Wu, M.; Chakraborty, C.; Min, L.; He, Y.; Guduri, M. Covert Communications in Air-ground Integrated Urban Sensing Networks Enhanced by Federated Learning. *IEEE Sens. J.* **2023**, *24*, 5636–5643. [CrossRef]
6. Wang, H.; Zhao, H.; Zhou, L.; Ma, D.; Wei, J. Deployment algorithm for minimum unmanned aerial vehicles towards optimal coverage and interconnections. In Proceedings of the 2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW), Barcelona, Spain, 15–18 April 2018; pp. 72–277. [CrossRef]
7. Yu, P.; Ding, Y.; Li, Z.; Tian, J.; Zhang, J.; Liu, Y.; Li, W.; Qiu, X. Energy-Efficient Coverage and Capacity Enhancement with Intelligent UAV-BSs Deployment in 6G Edge Networks. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 7664–7675. [CrossRef]
8. Chao, D. A survey of UAV-based edge intelligent computing. *Chin. J. Intell. Sci. Technol.* **2020**, *2*, 227–239. [CrossRef]
9. Masroor, R.; Naeem, M.; Ejaz, W. Efficient deployment of UAVs for disaster management: A multi-criterion optimization approach. *Comput. Commun.* **2021**, *177*, 185–194. [CrossRef]
10. Wang, D.; Wu, M.; Wei, Z.; Yu, K.; Min, L.; Mumtaz, S. Uplink Secrecy Performance of RIS-based RF/FSO Three-Dimension Heterogeneous Networks. *IEEE Trans. Wirel. Commun.* **2023**, *23*, 1798–1809. [CrossRef]
11. Liang, J.; Huang, X.; Xu, Q.; Liu, Y.; Zhang, J.; Huang, J. A Novel UAV-Assisted Multi-Mobility Channel Model for Urban Transportation Emergency Communications. *Electronics* **2023**, *12*, 3015. [CrossRef]
12. Pandey, K.K.; Kumbhar, C.; Parhi, D.R.; Mathivanan, S.K.; Jayagopal, P.; Haque, A. Trajectory Planning and Collision Control of a Mobile Robot: A Penalty-Based PSO Approach. *Math. Probl. Eng.* **2023**, *2023*, 1040461. [CrossRef]
13. Khan, M.A.; Qureshi, I.M.; Khanzada, F. A Hybrid Communication Scheme for Efficient and Low-Cost Deployment of Future Flying Ad-Hoc Network (FANET). *Drones* **2019**, *3*, 16. [CrossRef]
14. Liu, W.; Hu, K.; Li, Y.; Liu, Z. A review of prediction methods for moving target trajectories. *Chin. J. Intell. Sci. Technol.* **2021**, *3*, 149–160.
15. Keerthinathan, P.; Amarasingam, N.; Hamilton, G.; Gonzalez, F. Exploring unmanned aerial systems operations in wildfire management: Data types, processing algorithms and navigation. *Int. J. Remote Sens.* **2023**, *44*, 5628–5685. [CrossRef]
16. Wang, X.; Shi, S.; Xue, J.; Wu, C. Controller placement in software defined FANET. *Wirel. Networks* **2023**, *3*, 1–14. [CrossRef]
17. Chriki, A.; Touati, H.; Snoussi, H.; Kamoun, F. FANET: Communication, mobility models and security issues. *Comput. Netw.* **2019**, *163*, 106877. [CrossRef]
18. Lu, H.; Yang, Y.; Tao, R.; Chen, Y. Coverage Path Planning for SAR-UAV in Search Area Coverage Tasks Based on Deep Reinforcement Learning. In Proceedings of the 2022 IEEE International Conference on Unmanned Systems (ICUS), Guangzhou, China, 28–30 October 2022; pp. 248–253. [CrossRef]
19. Karimi-Bidhendi, S.; Guo, J.; Jafarkhani, H. Energy-Efficient Deployment in Static and Mobile Heterogeneous Multi-Hop Wireless Sensor Networks. *IEEE Trans. Wirel. Commun.* **2022**, *21*, 4973–4988. [CrossRef]
20. Wang, D.; He, T.; Lou, Y.; Pang, L.; He, Y.; Chen, H.-H. Double-edge Computation Offloading for Secure Integrated Space-air-aqua Networks. *IEEE Internet Things J.* **2023**, *10*, 15581–15593. [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.