Trajectory Optimization in a Cooperative Aerial Reconnaissance Model

Petr Stodola [1,*], Jan Drozd [2], Jan Nohel [1], Jan Hodický [3] and Dalibor Procházka [4]

[1] Department of Intelligence Support, University of Defence, 662 10 Brno, Czech Republic; jan.nohel@unob.cz
[2] Department of Tactics, University of Defence, 662 10 Brno, Czech Republic; jan.drozd@unob.cz
[3] Department of Aircraft Technology, University of Defence, 662 10 Brno, Czech Republic; jan.hodicky@unob.cz
[4] Centre for Security and Military Strategic Studies, University of Defence, 662 10 Brno, Czech Republic; dalibor.prochazka@unob.cz
[*] Correspondence: petr.stodola@unob.cz

Abstract: In recent years, the use of modern technology in military operations has become standard practice. Unmanned systems play an important role in operations such as reconnaissance and surveillance. This article examines a model for planning aerial reconnaissance using a fleet of mutually cooperating unmanned aerial vehicles to increase the effectiveness of the task. The model deploys a number of waypoints such that, when every waypoint is visited by any vehicle in the fleet, the area of interest is fully explored. The deployment of waypoints must meet the conditions arising from the technical parameters of the sensory systems used and tactical requirements of the task at hand. This paper proposes an improvement of the model by optimizing the number and position of waypoints deployed in the area of interest, the effect of which is to improve the trajectories of individual unmanned systems, and thus increase the efficiency of the operation. To achieve this optimization, a modified simulated annealing algorithm is proposed. The improvement of the model is verified by several experiments. Two sets of benchmark problems were designed: (a) benchmark problems for verifying the proposed algorithm for optimizing waypoints, and (b) benchmark problems based on typical reconnaissance scenarios in the real environment to prove the increased effectiveness of the reconnaissance operation. Moreover, an experiment in the SteelBeast simulation system was also conducted.

1. Introduction

The use of unmanned aerial vehicles (UAVs) for monitoring, surveillance, and/or reconnaissance operations both in the military and civilian domains has been a rising trend in recent years. Although this trend is relatively new, some of its features are already clear: the technology behind these systems is becoming increasingly mature; the effectiveness of these operations can be rapidly increased by using multiple UAVs that cooperate with one another; and route planning for individual systems is critical in these applications, as it determines the quality of operational performance.

In the military, reconnaissance and surveillance operations serve to increase situation awareness on the battlefield. For planning and executing these operations, Command, Control, Communication, Computer, Intelligence, Surveillance, and Reconnaissance (C4ISR) systems are used. Moreover, these

systems often contain decision support features that support a commander in their decision-making processes.

1.1. Motivation

At the University of Defence, Czech Republic, the Tactical Decision Support System (TDSS) is being developed. The objective of this system is to support tactical level commanders of the Czech Army in their decision-making process [1]. A number of models of military tactics are included in the TDSS. If one of the models is compatible with the task of the commander, they can use this system for planning; TDSS provides possible variants and options to fulfil the task, including the second-order effects. Thus, the task can be conducted effectively in the optimal or near-optimal manner (according to the selected criterion). More information about this topic can be found in the literature [1–8].

TDSS contains several models used for planning reconnaissance or surveillance tasks via both ground and aerial unmanned systems; for example, cooperative aerial reconnaissance, cooperative ground reconnaissance, cooperative aerial surveillance, and surveillance via a number of unmanned ground sensors (UGS).

Reconnaissance of the area of interest is one of the most frequent tasks of a tactical level commander. Effective reconnaissance can be critical; planning fast and complete aerial reconnaissance using a fleet of UAVs can significantly contribute to the success of the operation at hand. The original aerial reconnaissance model implemented in TDSS has been extended by findings and results from this article, thus increasing its efficiency.

1.2. Contribution and Organization of the Article

Cooperative Aerial Reconnaissance (CAR) is the model proposed by the authors in the past. This original model transforms a reconnaissance operation into a problem of planning the trajectories of available UAVs in a graph created from waypoints deployed in the area of interest to be explored. Section 3 revises the basic approach and principles of the CAR model published within the previous research of the authors. Also, the original algorithm used for the waypoints' deployment appears in this section in order to compare the old approach with the new one.

The contribution of this article is in the substantial improvement of the CAR model. The main idea behind this improvement consists of reducing the number of waypoints that need to be deployed, with the quality worsening or the requirements of the whole operation being undermining without this reduction. This is done by finding the minimal number of waypoints necessary and optimizing their positions. Section 4 is the key part of this article; it introduces the novel approach, formulates the optimization problem, and presents the proposed algorithms for (a) optimizing the number of waypoints and (b) optimizing the positions of waypoints. Section 5 sets out the experiments conducted in order to verify the proposed algorithms.

2. Literature Review

Issues around aerial reconnaissance have been discussed in many scientific publications. Paucar et al. [9] deal with the use of UAVs for surveillance and reconnaissance operations in military areas and analyse the benefits of their model in the Ecuadorian Armed Forces. Chen et al. [10] consider a reconnaissance task assignment problem for multiple UAVs with different sensor capacities; the authors propose a modified multi-objective symbiotic organisms search (MOSOS) algorithm to solve the problem. Wang et al. [11] model UAV reconnaissance mission planning as an uncertain multi-objective orienteering problem; to solve this problem, the authors propose a discreet multi-objective bat algorithm with a local search strategy.

Many publications exist that examine route planning for UAVs in various applications. Vivaldini et al. [12] propose a framework for efficient visual data acquisition using UAVs; this combines perception, environment representation, and route planning in the task of disease classification. Liu et al. [13] present a UAV route planning problem for aerial photography under interval uncertainties.

The authors of [14] consider the offline route planning of UAVs for a coverage search mission in a river region. The military topic of battlefield situation awareness is covered in the work of [15]; here, the authors propose a hybrid algorithm combining A* and ant colony optimization algorithms for real-time path planning.

Similarly, many publications deal with methods for UAV route optimization. A great overview of methods for trajectory optimization is given in the work of [16]; the methods considered in this paper include linear programming, dynamic programming, genetic algorithms, and neural networks. Different strategies for patrolling tasks conducted by multiple agents on a network defined as a graph are presented in the work of [17]. Turker et al. [18], meanwhile, use simulated annealing for route planning of multi-UAV systems on a graph.

The authors of [19] present a persistent intelligence, surveillance, and reconnaissance routing problem, which includes collecting data from a set of specified task locations and delivering those data to a control station. Zhen et al. [20] consider a vehicle routing problem in which UAVs monitor a set of areas with different accuracy requirements. The problem of how to deploy multiple UAVs most efficiently was considered in the work of [21], in an application where UAVs act as wireless base stations that provide coverage for ground users.

Popular methods often used in applications of reconnaissance, surveillance, or search for an object are based on coverage path planning. An extensive survey of such methods is provided by Cabreira et al. [22]; their article aims at analyzing the existing literature related to the different approaches, including both simple geometric flight patterns and more complex grid-based solutions considering full and partial information about the area of interest. The problem of searching for a single lost target by a single autonomous airborne sensor system using the Bayesian approach was examined in the work of [23]; a stationary or a drifting object at sea was considered as a target. The similar problem of locating a mobile, non-adversarial target in an indoor environment using multiple robotic platforms was investigated by Hollinger et al. [24]; they refer to the problem as multi-robot efficient search path planning (MESPP).

Many publications focusing on monitoring and surveillance using UAVs assume that the ground area to be monitored is perfectly flat. This assumption is not realistic; uneven terrain or high objects (buildings, obstacles) may occlude same parts in the area to be explored. The analysis of this issue is examined by Geng et al. [25] on a problem of continuous surveillance by a group of UAVs in an urban area. The similar problem is considered by Jakob et al. [26]; they present their own approach to plan the aerial surveillance in urban areas via a fleet of fixed-wing UAVs. They propose and compare three planning methods that take into account sensor occlusions caused by high buildings and other obstacles in the target area. Savkin and Huang [27] present their approach to estimate the minimal number of UAVs needed to monitor very uneven target terrain.

Although UAV technology is a very topical issue and thus there are many scientific publications considering various problems from different perspectives, the authors of this article are not aware of any research that deals with the trajectory optimization of UAVs by means of waypoint deployment.

3. Cooperative Aerial Reconnaissance

The CAR model is designed for exploring an area of interest by a fleet of unmanned aerial systems (UASs). The objective of the model is to plan the routes of individual UASs so that the area of interest is explored in an optimal or near-optimal manner. The optimization criterion depends on the commander and their task at hand; in most cases, the requirement is to explore the area as fast as possible. The CAR model assumes that the target area is perfectly flat.

The model was mathematically formulated in the work of [28] in detail. The principle is to cover the area of interest evenly, using a number of waypoints so that no space is unexplored when each waypoint is visited during a reconnaissance operation by any UAS available in the fleet. Each waypoint needs to be visited just once. The CAR model plans the trajectories of all UASs according to the selected optimization criterion.

3.1. Example Situation

Figure 1 shows an example situation. There are 2 UASs (labelled A and B) available in the fleet deployed in the area of operations. The area of interest to be explored is covered by 27 waypoints (see the green dots in Figure 1a). In Figure 1b, the trajectories (blue lines) of individual UASs are shown, which represent the solution found by the CAR model; as can be seen, the waypoints are evenly distributed between available UASs.
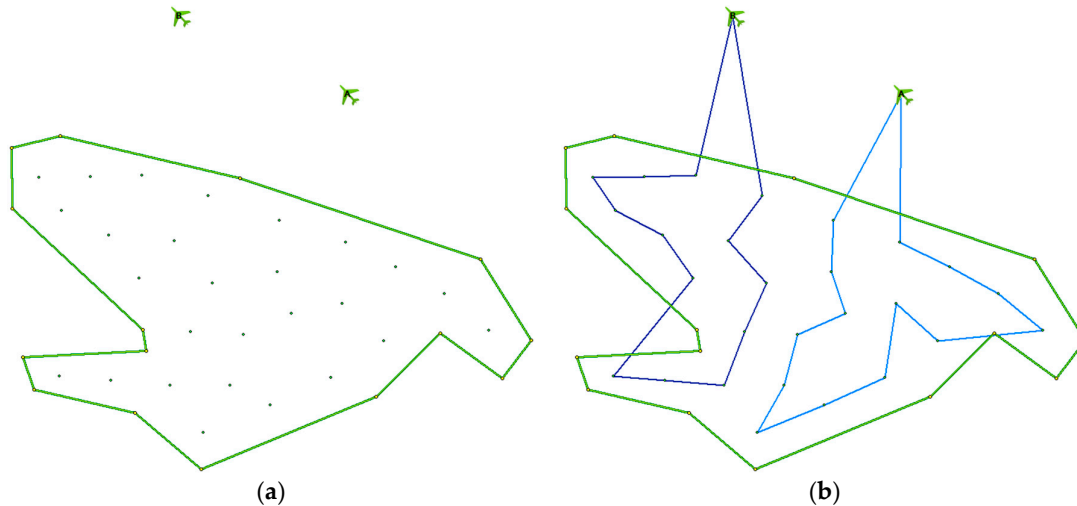
Figure 1. Example situation: (**a**) area of interest covered by waypoints; (**b**) trajectories for individual unmanned aerial systems (UASs).

The phase of the CAR model that plans the trajectories for UASs is similar to the well-known multi-depot vehicle routing problem (MDVRP). The difference consists in formulating the optimization criterion. Whereas the objective in the MDVRP is to minimize the sum of lengths of all routes, the objective in the CAR model is alterable according to the requirements of the commander; in most cases, it is to minimize the time needed to complete the longest route by the corresponding UAS.

The original CAR model was improved in the work of [29], where the trajectories were smoothed in order to achieve the following: (a) shorten their length and thus hasten the reconnaissance operation, and (b) be able to use fixed-wing UASs that cannot change direction so abruptly. The result of this improvement, applied to the previous example situation, is shown in Figure 2 in red.
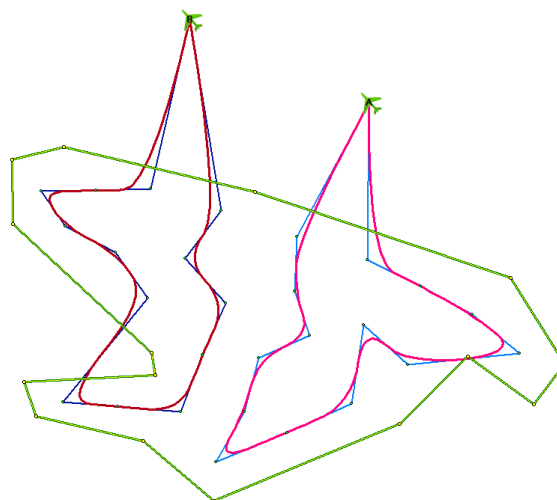

Figure 2. Improvement in the trajectories.

3.2. Requirements for the Deployment of Waypoints

The area of interest has to be covered by a number of waypoints so that, after reconnaissance, there is no space in the area left unexplored. This depends on the technical parameters of the sensory systems of UASs, as well as the tactical requirements of the commander. The former concerns the vertical and horizontal field of view and resolution of the camera, while the latter relates to the minimum and maximum permitted height of flight above ground level of UASs in the fleet.

The situation is presented in Figure 3. A UAS flies at the height H_{AGL} above ground level, which is the minimum height allowed. At this height, the camera of the UAS scans the green area (rectangle), the size of which is determined by the angular vertical and horizontal field of view (A_{VFOV}, A_{HFOV}). The resolution and other technical parameters of the camera must be adequate to be able to scan the area at the required level of detail from the maximum allowed height of flight.
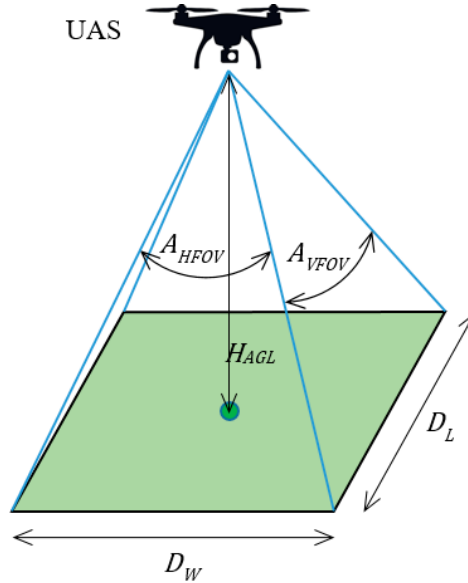
Figure 3. Parameters for deployment of waypoints.

The width and length of the scanned rectangle, D_W and D_L, respectively, are calculated according to Formula (1).

$$D_W = 2 \cdot H_{AGL} \cdot \tan\left(\frac{A_{HFOV}}{2}\right),$$

$$D_L = 2 \cdot H_{AGL} \cdot \tan\left(\frac{A_{VFOV}}{2}\right),$$

(1)

where D_W, D_L are the width and length, respectively, of the scanned area; H_{AGL} is the minimum flight height above the ground level; and A_{HFOV}, A_{VFOV} are the angular horizontal and vertical fields of view, respectively.

To ensure that no space is left unexplored after the reconnaissance operation, the maximum permitted distance D_{max} from whichever point lying inside the area of interest to its nearest waypoint must be calculated according to Formula (2). When calculating D_{max}, the smaller value of the width and length D_W and D_L is used. This is because the trajectories of the UASs (i.e., their pitch when located at individual waypoints) are not known at the time when the waypoints are deployed; therefore, the smaller (i.e., worse) value must be used.

$$D_{max} = \frac{\min(D_W, D_L)}{2},$$

(2)

where D_{max} is the maximum permitted distance from whichever point inside the area of interest to the nearest waypoint, and D_W, D_L are the width and length, respectively, of the scanned area.

3.3. Waypoint Deployment in the Original CAR Model

A simple algorithm was used in the original CAR model to deploy the waypoints. This algorithm is presented in Figure 4. The waypoints are inserted along individual edges of the polygon, forming the area of interest.

```
Original Waypoint Deployment
1. set  W = ∅
2. set  d_w = 0.8 · D_max
3. for each edge E_i of the polygon forming the area of interest
4.     set d_x = √0.75 · d_w
5.     calculate a set of points X at distance d_x from the edge E_i
       and with distance d_w between nearest neighbours in this set
6.     for each point X_j ∈ X
7.         if X_j is inside the area of interest and distance to the nearest
           waypoint in W is not smaller than d_w
8.             insert X_j into W
9.         set d_x = d_x + √3 · d_w
10.        go to point 5 when at least 1 waypoint has been inserted into W
11. return W
```

Figure 4. Original waypoint deployment algorithm.

The algorithm is fast and straightforward. However, it does not guarantee that the maximum distance requirement is ensured from all points in the area of interest. The result can be influenced by playing with distance d_w (see point 2 in Figure 4) and repeating the whole algorithm. It was empirically found that the D_{max} requirement is met in most cases when $d_w = 0.8 \cdot D_{max}$.

4. Trajectory Optimization

This section deals with improving the CAR model by optimizing the number and positions of waypoints in the area of interest.

4.1. Optimization Problem

The optimization problem consists of finding the minimum possible number of waypoints n, provided that condition (3) is not violated.

$$d_{max} \leq D_{max}, \tag{3}$$

where d_{max} is the longest existing distance from a point in the area of interest to the nearest waypoint, and D_{max} is the maximum permitted distance introduced in Section 3.2 (see Formula (2)).

The algorithm calculating the minimum number of waypoints n is shown in Figure 5. It is based on the principle of the half-interval search. Firstly, the limits n_{min} and n_{max} are determined; the lower limit n_{min} is set to 1 and the higher limit n_{max} is calculated according to Formula (4), which ensures that condition (3) is met with certainty. Then, the limits gradually approach one another until they meet, which means that the minimum possible number of waypoints is found.

$$n_{max} = \left\lceil \frac{w}{\sqrt{3} \cdot D_{max}} \right\rceil \cdot \left\lceil \frac{l}{\sqrt{3} \cdot D_{max}} \right\rceil, \tag{4}$$

where n_{max} is the higher limit of the number of waypoints; w and l are width and length, respectively, of the area of interest; and D_{max} is the maximum permitted distance (see Formula (2)).

```
Optimization of the Number of Waypoints
  1. set n_min = 1
  2. calculate n_max
  3. do
  4.    set n = ⌊(n_min + n_max)/2⌋
  5.    optimize positions of n waypoints and calculate d_max
  6.    if d_max > D_max
  7.       set n_min = n + 1
  8.    else
  9.       set n_max = n
 10. while n_min ≠ n_max
 11. return n and positions of waypoints
```

Figure 5. Algorithm for optimization of the number of waypoints.

4.2. Optimization of the Positions of Waypoints

The key part of the algorithm in Figure 5 is point 5, which is a stand-alone optimization problem. For every particular deployment of a given number of waypoints, the value of d_{max} can be calculated. The optimization criterion of this problem is expressed by Formula (5). The objective is to deploy the waypoints so that the value of d_{max} is minimal. This consists of finding the positions of n waypoints in a plane, that is, there are $2n$ independent variables.

$$\text{minimize}(d_{max}), \tag{5}$$

where d_{max} is the longest existing distance between a point in the area of interest and the nearest waypoint.

The mathematical formulation of the problem is as follows: let $P = \{P_1, P_2, \dots\}$ be the set of all points that are inside the polygon forming the area of interest; there are an infinite number of points in this set. Let $W = \{W_1, W_2, \dots, W_n\}$ be the set of n waypoints; and let $D = \{d_1, d_2, \dots\}$ be the set of real numbers that represent distances between each point $P_i \in P$ and the nearest waypoint $W_{(i)} \in W$, according to Formula (6).

$$d_i = \left|P_i - W_{(i)}\right| \quad \text{for all } P_i \in P, \tag{6}$$

where d_i is the distance between points P_i and $W_{(i)}$, $P_i \in P$ is a point lying inside the area of interest, and $W_{(i)} \in W$ is the nearest waypoint to point P_i.

The distance $d_{max} \in D$ is the largest value in set D; see Formula (7).

$$d_{max} = \max D = \max(d_1, d_2, \dots) \tag{7}$$

4.2.1 Calculation of the Longest Distance

This section deals with the principles of calculating the distance d_{max} formulated above. The value of d_{max} is a function of waypoints $W = \{W_1, W_2, \dots, W_n\}$, where $n \geq 1$ and the polygon of the area of interest $A = \{A_1, A_2, \dots, A_m\}$. This polygon is represented by the number of m points, where $m \geq 3$, thus it is a function of $m + n$ points in a plane (which are $2n + 2m$ variables); see Formula (8).

$$d_{max} = f(W, A) = f\left(W_{1x}, W_{1y}, \dots, W_{nx}, W_{ny}, A_{1x}, A_{1y}, \dots, A_{mx}, A_{my}\right) \tag{8}$$

The value of d_{max} is defined as the longest existing distance between a point $P_i \in P$ inside polygon A and its nearest waypoint $W_{(i)} \in W$. As there is an infinite number of points in set P, the longest distance cannot be determined by calculating all distances $d_i \in D$.

Instead, a subset $C \subset P$ containing a finite number of points has to be identified. These points $C = \{C_1, C_2, ..., C_k\}$ are candidates for becoming points from which the value of d_{max} is calculated using Formula (9).

$$d_{max} = \max(|C_1 - W_{(1)}|, |C_2 - W_{(2)}|, ..., |C_k - W_{(k)}|) \tag{9}$$

Points in subset C are identified as follows:

(a) Points that are common to three or more regions of a Voronoi diagram created on waypoints W and, at the same time, which lie inside polygon A.
(b) Points that are common to two regions of a Voronoi diagram created on waypoints W and the border of polygon A.
(c) Border points $A_i \in A$ of the polygon.

The principle is graphically shown in Figure 6 in an example with 10 waypoints. The green lines represent the polygon of the area of interest, the small yellow dots are waypoints, and the red lines border the regions of the Voronoi diagram. The blue, orange, and green circles show the points of subset C; the colours are used to distinguish between the three cases mentioned above. In this example, the farthest point F from the waypoints is marked in red.
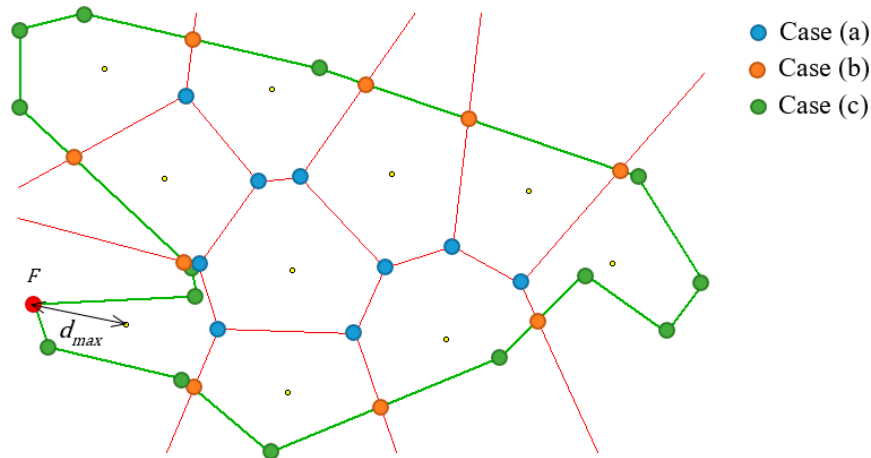

Figure 6. Principle of calculating the longest distance.

The computational complexity of the proposed principle is mainly influenced by the construction of the Voronoi diagram. This can be done using Fortune's algorithm with complexity $O(n \cdot \log n)$, where n is the number of waypoints [30]. The result of this algorithm is a list of edges forming Voronoi regions.

Once the Voronoi diagram has been constructed, the computational complexity of calculating d_{max} for the three cases mentioned above is as follows:

(a) $O(n \cdot m)$. The points of subset C are the vertices of the edges of the Voronoi diagram that lie inside the polygon. The number of these vertices is linearly dependent on the number of waypoints n. The test to see whether they are inside the polygon can be done with complexity $O(m)$, where m is the number of border points of the polygon. One advantageous feature of Fortune's algorithm is that each edge is connected with the two waypoints lying in the corresponding regions. These waypoints are the nearest for both vertices and the distance to them can be calculated immediately.
(b) $O(n \cdot m)$. The points of subset C are created as intersections of the edges of the Voronoi diagram and edges of the polygon. The number of edges of the former is linearly dependent on the number of waypoints n. The number of edges of the latter is identical to the number of points m of the polygon. Again, as in the previous case, the pair of nearest waypoints for each intersection is known because it is connected with the corresponding edge of the Voronoi diagram.

(c)　$O(n \cdot m)$. The points of subset C are created from m points of the polygon. For each of these points, the nearest waypoint has to be found with complexity $O(n)$.
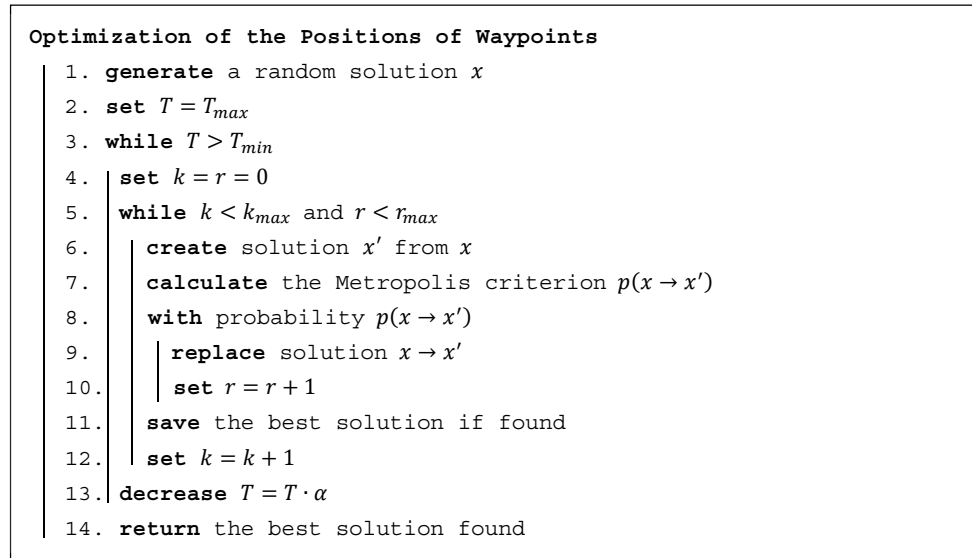
As can be seen, the computational complexity for each of the three cases—(a), (b), and (c)—is the same: $O(n \cdot m)$. Thus, the total computational complexity of calculating the value of d_{max} for n waypoints and the polygon with m points is given in Formula (10).

$$O(n \cdot \log n + n \cdot m) \tag{10}$$

4.2.2 Optimization Method

For a solution to this problem, an adapted simulation annealing algorithm is used. This algorithm is a generic probabilistic method and has proven to be very successful when used by the authors in similar position optimization problems; see the works of [31,32].

This algorithm is presented in Figure 7 using pseudocode. A solution x represents a particular deployment of n waypoints, that is, it is a vector of $2n$ independent variables, where each pair corresponds to a position of one waypoint in a plane: $x = (x_1, x_2, \ldots, x_{2n})$.

```
Optimization of the Positions of Waypoints
 1. generate a random solution x
 2. set  T = T_max
 3. while  T > T_min
 4.   set  k = r = 0
 5.   while  k < k_max  and  r < r_max
 6.     create solution x' from x
 7.     calculate the Metropolis criterion p(x → x')
 8.     with probability p(x → x')
 9.       replace solution x → x'
10.       set  r = r + 1
11.     save the best solution if found
12.     set  k = k + 1
13.   decrease  T = T · α
14.   return the best solution found
```

Figure 7. Simulated annealing for the problem of optimizing waypoint positions.

The key phase of the algorithm is the creation (transformation) of a new solution $x' = (x'_1, x'_2, \ldots, x'_{2n})$ from the current solution x in point 6. This is done by adding random value(s) with normal distribution to one or more variables of vector x, according to Formula (11).

$$x'_i = \begin{cases} x_i + \text{RandN}(\mu, \sigma^2) & \text{for selected variables } x_i \in x \\ x_i & \text{otherwise} \end{cases},$$

$$\mu = 0, \tag{11}$$

$$\sigma = \frac{(T - T_{min}) \cdot R_{max}}{T_{max} - T_{min}},$$

where RandN is a random number generator with a normal distribution $N(\mu, \sigma^2)$, T is the current temperature in interval $\langle T_{min}, T_{max} \rangle$, and R_{max} is the maximum range in which variable x_i should change—this value is connected with the size (width or length) of the area of interest $R_{max} = size/3$.

The decision on selecting which variables to change in Formula (11) depends on the variant of the algorithm. Usually, one of these variants is used:
- Select all variables from x.
- Select variables from x with probability p.

- Select just one variable randomly from x. This variant is used for this optimization problem as it was empirically proven to be the variant with the most successful results.

When the new solution x' is created, this solution replaces the original solution x with probability $p(x \rightarrow x')$, calculated according to the Metropolis criterion in Formula (12). The Metropolis criterion states that if the new solution is better than the original, the probability of its replacement is 1; otherwise, this probability depends on the difference in their qualities and the current temperature.

$$p(x \rightarrow x') = \begin{cases} 1 & \text{for } d'_{max} \leq d_{max} \\ e^{-\frac{d'_{max} - d_{max}}{T}} & \text{otherwise} \end{cases}, \tag{12}$$

where d'_{max} is the longest distance for the new solution x' and d_{max} is the longest distance for the original solution x.

4.2.3 Improvement of the Optimization Method

The optimization method introduced in Section 4.2.2 can be further improved for this particular problem. When evaluating the quality of a solution, that is, calculating the value of d_{max} for the given variables of vector x, the position of the furthest point F in the polygon of the area of interest in relation to the waypoints is determined as a by-product (see the red circle in Figure 6 as an example).

Seeing as the aim in resolving the problem is to minimize the distance d_{max} (see Formula (5)), it is likely that a better solution can be created by moving the waypoints (see Formula (11)) that are closer to point F than the others, especially for the later phases of the optimization, when the current temperature is heading towards the minimal temperature T_{min} and changes are smaller.

The algorithm was modified to use this feature as follows: the probability of each variable being selected is calculated according to Formula (13); then, one variable is selected for transformation, not randomly, but based on these probabilities using the roulette wheel principle.

$$p(x_{2i-1}) = p(x_{2i}) = \frac{(d_{W_i})^{-1}}{2 \cdot \Sigma_i (d_{W_i})^{-1}} \quad \text{for each } W_i \in W, \tag{13}$$

where $p(x_j)$ is the probability of variable $x_j \in x$ being selected, $j = 1, 2, \dots, 2n$; and d_{W_i} is the distance between waypoint $W_i \in W$ and point F.

5. Experiments and Results

This section presents experiments and results of the proposed algorithm on two sets of benchmark problems.

5.1. Benchmark Problems for Verification

The first set of benchmark problems was designed to verify the quality of the proposed probabilistic algorithm. Polygons forming the area of interest for individual problems are created by combining a number of adjacent hexagons; the problems vary from one another according to the number of hexagons combined. The number of waypoints corresponds to the number of hexagons.

The reason for applying these benchmark problems is that it is easy to determine the optimal solution for each, which can then be compared with the solution found by the optimization method. The optimal solution is the solution where waypoints are located at the centres of hexagons. An example of a problem with 17 waypoints (optimally deployed) is shown in Figure 8. The green line demarcates the resulting polygon, while the red lines represent a Voronoi diagram.
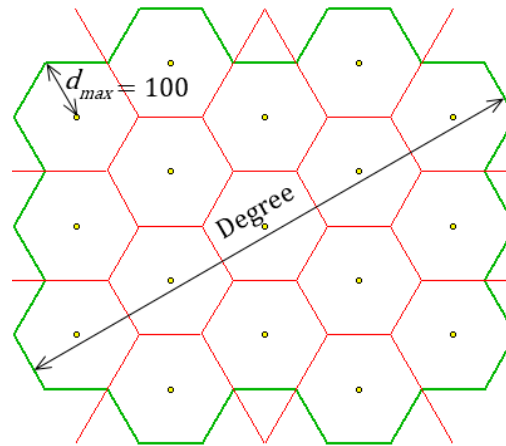
Figure 8. Example of a benchmark problem (v03).

Table 1 shows the basic parameters of the benchmark problems. Degree represents the number of hexagons in diagonals of the polygon (see Figure 8).

Table 1. Set of benchmark problems for verification.

Benchmark Problem	Degree	Number of Waypoints	Number of Variables per Solution Vector
v01	1	1	2
v02	3	7	14
v03	5	17	34
v04	7	31	62
v05	9	49	98
v06	11	71	142

The proposed optimization algorithms (original and modified versions—see Section 4.2.2 and 4.2.3, respectively) were executed 500 times on each benchmark problem. The parameters of the algorithm (see Figure 6) were set as follows: $T_{max} = 100$, $T_{min} = 10^{-6}$, $\alpha = 0.95$, $k = 2000$, $r = 200$. The experiments were executed on a computer with the following parameters: Intel Core i7-7700 CPU @ 3600 GHz, 32 GB RAM.

Table 2 shows the results achieved by the original simulated annealing algorithm (SA-original) and its modified version (SA-modified). The table records the best solution found, mean and standard deviation for each benchmark problem, as well as the average runtime of a single execution of the algorithm. As can be seen, the modified version is comparable to the original version for simpler problems (v01, v02, v03); however, although the original algorithm is almost twice as fast for v05 and v06 problems, the improvement in solutions of the modified version is clear. The slower runtime of the modified version for more complex problems is caused by calculating the probabilities for selecting variables with computational complexity $O(n)$ in the solution transformation process (see point 6 in Figure 6), whereas the same process is done with complexity $O(1)$ in the original version.

Table 2. Results for benchmark problems for verification.

Problem	SA-Original				SA-Modified			
	Best	Mean	Stdev	Runtime	Best	Mean	Stdev	Runtime
v01	100.000	100.000	0.000	0.5 sec	100.000	100.000	0.000	0.5 s
v02	100.000	100.004	0.025	5 sec	100.000	100.005	0.033	5 s
v03	100.001	100.064	0.098	12 sec	100.001	100.010	0.022	14 s
v04	100.005	100.884	0.854	25 sec	100.003	100.090	0.230	34 s
v05	100.555	104.904	2.868	38 sec	100.018	100.668	1.594	74 s
v06	102.417	111.278	3.497	53 sec	100.114	103.318	2.377	98 s

Table 3 compares the results. The optimal solution for each benchmark problem is 100 (distance from the centre of any hexagon to its circumscribed circle). The best solutions found by the original and modified algorithms are compared with the optimal solution in the form of percent error. The last column of Table 3 shows the improvement of the modified algorithm compared with the original version. The average percent error for the original and modified algorithms compared with the optimal solution over all six benchmark problems is 0.496% and 0.023%, respectively. The modified version outperforms the original algorithm by 0.473% on average; the results achieved are very close to the optimal solution even for a problem with 71 waypoints, that is, for 142 independent variables.

Table 3. Comparison of results for benchmark problems for verification.

Problem	Optimal Solution	SA-Original		SA-Modified		Algorithm Improvement
		Best	Error	Best	Error	
v01	100	100.000	0.000%	100.000	0.000%	0.000%
v02	100	100.000	0.000%	100.000	0.000%	0.000%
v03	100	100.001	0.001%	100.001	0.001%	0.000%
v04	100	100.005	0.005%	100.003	0.003%	0.002%
v05	100	100.555	0.555%	100.018	0.018%	0.537%
v06	100	102.417	2.417%	100.114	0.114%	2.300%

5.2. Benchmark Problems of Typical Reconnaissance Scenarios

The second set of benchmark problems enables a comparison between the new approach to waypoint deployment according to the algorithm in Figure 5 and the original approach in Figure 4. The most important criterion is the reduction in the number of waypoints, the effect of which is to shorten the overall time for a reconnaissance operation.

Table 4 shows the parameters of the benchmark problems, created based on typical reconnaissance situations in the real environment. The table records the basic parameters of the polygon, along with the number of points forming the area of interest and the number of UAVs in the fleet available for the reconnaissance operation. The last column of Table 4 details the maximum permitted distance given by the technical parameters of the particular UAVs and tactical requirements according to Formula (2).

Table 4. Set of benchmark problems for reconnaissance. UAVs—unmanned aerial vehicles.

Problem	Area of Interest				Number of UAVs	D_{max}
	Area	Width	Length	Points		
r01	0.517 km²	1.21 km	0.78 km	15	2	120 m
r02	0.679 km²	0.95 km	1.14 km	5	2	100 m
r03	2.640 km²	2.20 km	1.20 km	4	1	150 m
r04	2.085 km²	1.60 km	2.10 km	7	4	150 m
r05	1.675 km²	2.15 km	1.25 km	9	4	100 m
r06	2.757 km²	3.75 km	3.04 km	25	5	120 m

Table 5 shows the optimization of the number of waypoints deployed for the benchmark problems using the algorithm presented Figure 5 compared with the number of waypoints determined using the original algorithm mentioned in Figure 4. The values of the longest distance (d_{max}), that is, the solutions, do not exceed the maximum permitted limit (D_{max}) in any of the benchmark problems. The last column of Table 5 records the number of waypoints saved by as a result of optimization. On average, the reduction in the number of waypoints is 38.9%.

The computational times of the waypoints' deployment optimization are also recorded in Table 5. The entire process of planning the reconnaissance operation will be extended by the extra time this process takes, as the computational times of the original waypoints deployment are negligible.

Table 5. Optimization of the number of waypoints for reconnaissance benchmark problems.

Problem	Original Deployment		Optimized Deployment			Reduction in Waypoints
	Waypoints	d_{max}	Waypoints	d_{max}	Runtime	
r01	27	119.712	18	118.295	12 sec	9 (33.3%)
r02	59	99.162	31	99.058	16 sec	28 (47.5%)
r03	66	141.421	52	149.668	29 sec	14 (21.2 %)
r04	82	149.771	42	148.767	25 sec	40 (48.8%)
r05	124	99.556	75	99.552	50 sec	49 (39.5%)
r06	166	119.832	94	119.632	103 sec	72 (43.4%)

Table 6 compares reconnaissance operations based on the benchmark problems when planned using the original model and the new model in which the number of waypoints and their deployment has been optimized. The optimization criterion was the duration of the operation, that is, the time taken to conduct the whole operation. Table 6 also records the total distance covered by all UASs; however, distance was not the optimization criterion. The last column sets out the reduction in operational durations; the new model managed to shorten reconnaissance operations by 10.3% on average. Also, the total distance decreased on average by 11.2%, the effect of which is lower fuel consumption and the possibility to explore more extensive areas.

Table 6. Reconnaissance operations planned using the original and new models.

Problem	Original Deployment		Optimized Deployment		Reduction in Waypoints
	Duration	Distance	Duration	Distance	
r01	6:55	5.00 km	6:37	4.69 km	4.3 %
r02	12:57	7.77 km	10:39	6.31 km	17.8%
r03	22:42	13.62 km	21:32	12.93 km	5.1%
r04	6:35	15.65 km	5:40	13.34 km	13.9%
r05	6:20	15.17 km	5:39	13.48 km	10.8 %
r06	10:26	31.28 km	9:25	27.84 km	9.7 %

To illustrate these findings, Figure 9 shows the benchmark problem r04. Figure 9a describes the result of the original version (82 waypoints). Figure 9b, on the other hand, is the result of the new version (42 waypoints). In Figure 9c, the scenario is shown in the planning software TDSS [1,33], including the visualized terrain and environment.

(**a**) (**b**) (**c**)
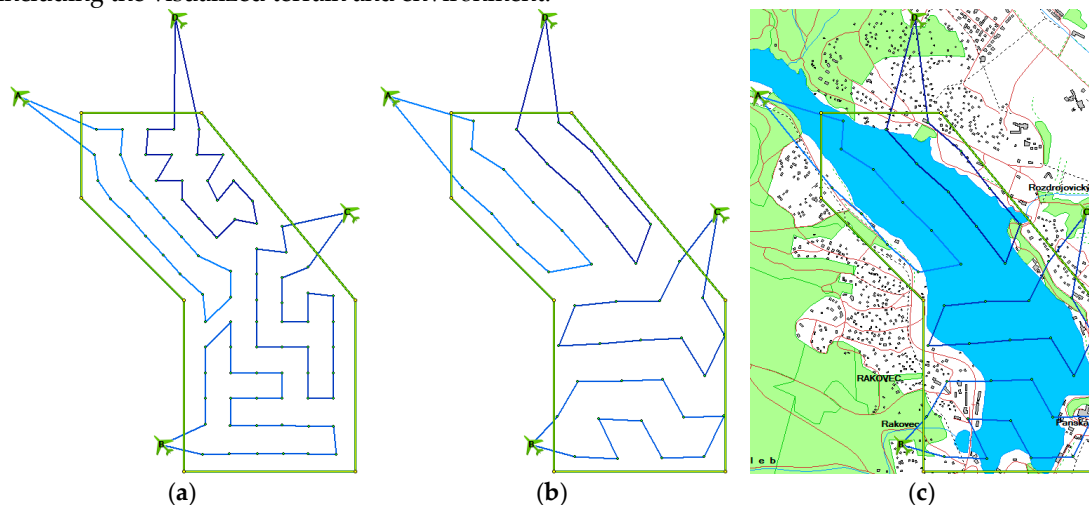
Figure 9. Benchmark problem r04: (**a**) original model; (**b**) new model; (**c**) Tactical Decision Support System (TDSS).

To prove the effectiveness of the proposed approach for the reconnaissance operations, the results were compared to the lawnmower pattern, which is a common approach when planning UAV

trajectories in problems such as reconnaissance or search for a target (Cabreira et al., 2019). The advantage of the lawnmower pattern is that the distance between adjacent flights of UAVs (in rows or columns) could be set to $2 \cdot D_{max}$; this is because the trajectory pattern is known in advance (the UAVs do not change their heading in a single column or row).

Table 7 compares the reconnaissance operations planned using the new model and the lawnmower pattern. The total distance covered by all UASs is also recorded, but again, as in the previous case, distance was not the optimization criterion. The difference in the last column of Table 7 shows that the new model outperforms the lawnmower pattern. The difference varies for individual benchmark problems from 7% to almost 60%. This is connected with the quality with which it was possible to divide the area of interest into portions for individual UAVs.

Table 7. Reconnaissance operations planned using the new model and the lawnmower pattern.

Problem	New Model		Lawnmower Pattern		Difference
	Duration	Distance	Duration	Distance	
r01	6:37	4.69 km	7:05	4.85 km	7.1%
r02	10:39	6.31 km	15:33	7:59 km	46.0%
r03	21:32	12.93 km	22:45	13.65 km	5.7%
r04	5:40	13.34 km	7:57	13.75 km	40.3%
r05	5:39	13.48 km	6:20	13.57 km	12.1%
r06	9:25	27.84 km	14:52	33.90 km	57.9%

The strength of the new approach proposed in this article is in the balanced and even division of the area of interest into portions for individual UAVs. This even division is almost never possible in the case of the lawnmower pattern, as the whole rows or columns must be visited successively in a straight line. This is illustrated in Figure 10 on the benchmark problem r04, where four UAVs are available. The values C_A, C_B, C_C, C_D show the time the corresponding UAV needs to finish its route. The solution is given by the maximum value ($C = \max(C_A, C_B, C_C, C_D,)$). As can be seen, individual routes are nicely balanced in the case of the new model, as opposed to the lawnmower pattern.


(a) (b)
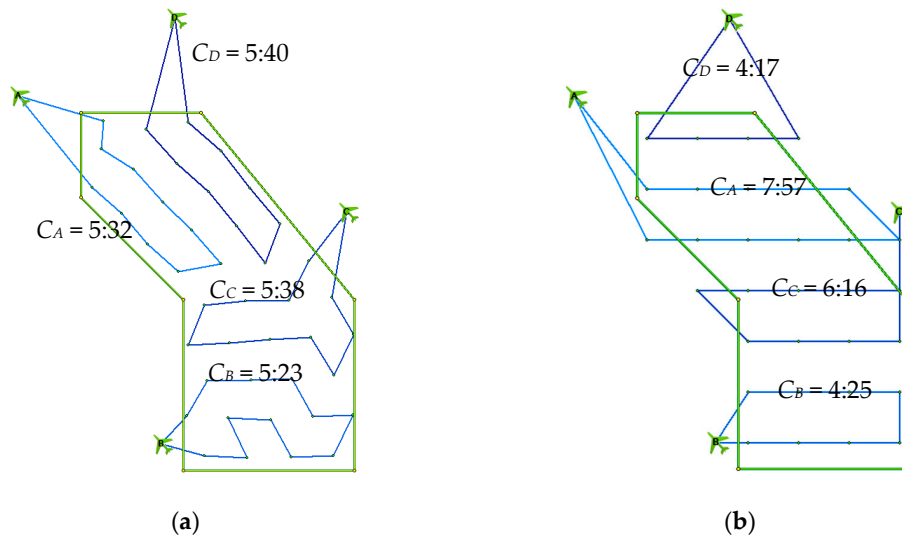
Figure 10. Comparison of the new model with the lawnmower pattern for benchmark problem r04: (**a**) new model; (**b**) lawnmower pattern.

The problem of the balanced division of the area of interest is even more apparent in the case of more complex and structured areas of interest and a larger number of UAVs used (e.g., the polygon bounding the area of interest of benchmark problem r06 has 25 points and 5 UAVs are available; the reconnaissance operation is then almost 60% longer when using the lawnmower pattern).

5.3. Simulation Experiment

The experiment based on benchmark problem r01 (see Table 4) was conducted using the Steel Beast Pro simulation system (https://www.esimgames.com). The reason for the experiment was to use simulation in order to prove the validity of the proposed model under various conditions. As UAVs, Lockheed Martin Quad Indago drones were used.

Figure 11 shows routes planned via the original (Figure 11a) and new version (Figure 11b). The model estimates that the operation would be completed, that is, at the moment when both drones had returned to their initial positions, in 6:55 min in the case of the original version (27 waypoints), and in 6:37 min in the case of the new version (18 waypoints). The length of the routes of UAV A is 2.54 km (original version) and 2.26 km (new version), and for UAV B, it is 2.46 km (original version) and 2.43 km (new version); the total distance travelled by both UAVs is 5.00 km (original version) and 4.69 km (new version), respectively.

| (**a**) | (**b**) |
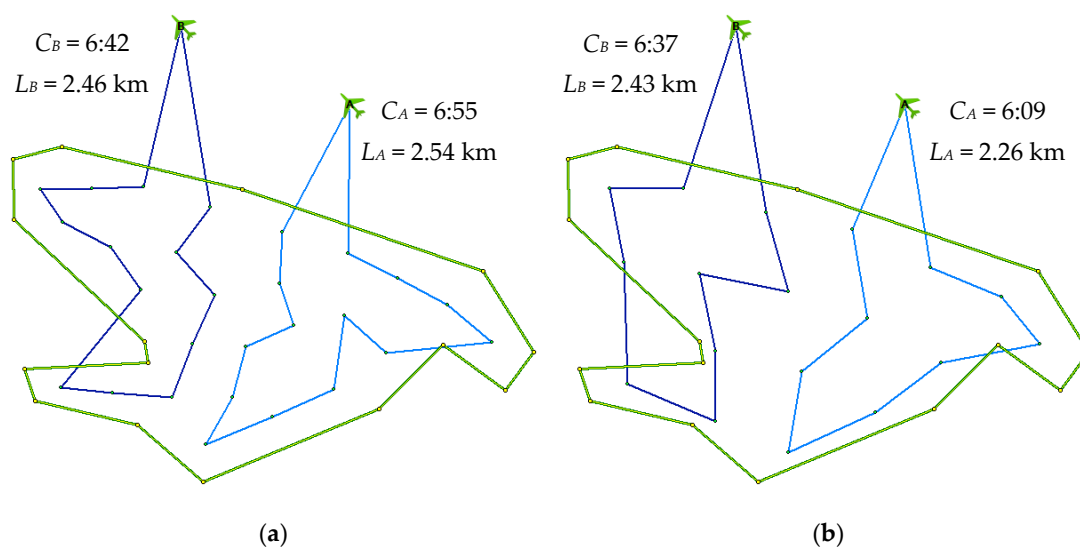
Figure 11. Experiment for simulation (r01): (**a**) original model; (**b**) new model.

Table 8 shows the results of the simulations. Overall, three separate experiments (e01, e02, e03) were conducted under various meteorological conditions. In each experiment, five simulations (s01, s02, s03, s04, s05) were carried out, and the average duration of the operation was calculated. In two cases, the simulation was not successful as a result of drone failure. The experiments show the influence of wind speed and precipitation on the flight duration. The wind extends the duration of the operation (compare e01 and e02); furthermore, the rain of medium intensity has a noticeable effect on the duration (e03 takes longer than e02 even if the wind speed is smaller).

Table 8. Results of the simulation experiment.

	e01 Sun No Wind		e02 Overcast Fresh Breeze (30 km/h)		e03 Medium Rain Moderate Breeze (20 km/h)	
Experiment						
Simulation	**Original**	**New**	**Original**	**New**	**Original**	**New**
s01	7:25	7:11	9:15	8:25	10:04	9:22
s02	7:11	6:54	8:05	8:14	9:48	8:47
s03	6:54	6:12	8:55	8:54	Failure	9:38
s04	7:14	6:55	8:45	9:01	9:34	8:26
s05	6:47	7:02	Failure	8:27	9:47	9:11
Average	7:06	6:50	8:45	8:36	9:48	9:04

Experiment e01 was carried out under ideal weather conditions (sunny, no wind), and thus can be compared to the results estimated by the model (r01)—see Table 9. The time estimated by the model was very close to the average duration of the simulations (difference of about 3%). Moreover, the simulation confirmed the improvement in shortening the duration of the operation when using the new model (about 4% in this case).

Table 9. Comparison of estimated and simulation results.

Experiment	Original Model	New Model	Improvement
e01	7:06	6:50	3.61%
r01	6:55	6:37	4.34%
Difference	2.63%	3.36%	0.72%

6. Conclusions

In this paper, the problem of optimizing the number and positions of waypoints in a cooperative aerial reconnaissance model using a number of unmanned aerial systems was examined. As a solution, a modified simulated annealing algorithm was proposed. The improvement of the new algorithm, compared with the principle used in the original CAR model, was verified in a series of experiments.

The experiments in the first set were designed to evaluate the proposed probabilistic algorithm. In these experiments, the optimal deployments of waypoints are already known and thus can be compared with the results achieved by the algorithm. The average difference between the optimal solution and the solution found by the simulated annealing was only 0.023%; in the most complex problem, with 71 waypoints (142 independent variables), the difference from the optimal solution was 0.114%.

The second set of experiments was based on typical reconnaissance scenarios, and their purpose was to evaluate the increase in effectiveness of the reconnaissance operation. The new model managed to shorten the time needed to conduct operations by an average of 10%. Also, the comparison of the results with the lawnmower pattern proved the effectiveness of the proposed method. Furthermore, one of these experiments was performed in the SteelBeast simulation system in order to verify the model under various meteorological conditions. The simulations confirmed the results estimated by the model.

In the CAR model, the collision avoidance of cooperating UAVs is not taken into consideration. It is supposed that each UAV has its own internal collision avoidance system or other precautions are taken (e.g., different height of flight for each vehicle). Moreover, trajectories of individual vehicles should not cross if planned optimally.

The future work of the authors will focus on the problem of persistent surveillance by a number of unmanned aerial vehicles. In this task, similar principles of the optimization of waypoints can be used. Also, the possibility of extending the model to avoid the potential collisions will be investigated. The model could integrate some principles of the traffic collision avoidance systems (TCAS), for example, see the work of [34]. Furthermore, the experiments will be conducted to verify the performance of the model when planning the reconnaissance of a very uneven terrain or in an urban area, where some parts of the area of interest may be occluded.

Author Contributions: conceptualization, methodology, software, writing, P.S.; validation, visualization, J.D. and J.N.; formal analysis, investigation, supervision, J.H. and D.P.

References

1.   Stodola, P. ; Mazal, J. Tactical Decision Support System to Aid Commanders in their Decision-Making. *Modell. Simul. Autonom. Syst*. **2016**, *9991*, 396–406.

2. Stodola, P.; Nohel, J.; Mazal, J. Model of Optimal Maneuver used in Tactical Decision Support System. In Proceedings of the 2016 21st International Conference on Methods and Models in Automation and Robotics, Miedzyzdroje, Poland, 29 August–1 September 2016.

3. Pokonieczny, K.; Rybanský, M. Method of developing the maps of passability for unmanned ground vehicles. In Proceedings of the 9th IGRSM International Conference and Exhibition on Geospatial & Remote Sensing, Kuala Lumpur, Malaysia, 24–25 April 2018.

4. Blaha, M.; Potužák, L.; Šilinger, K. Linear and Angular Issues in Perspective Artillery Fire Control System. In Proceedings of the 22nd International Conference on Circuits, Systems, Communications and Computers, Majorca, Spain, 14–17 July 2018.

5. Rybanský, M. Trafficability Analysis through Vegetation. In Proceedings of International Conference on Military Technologies, Brno, Czech Republic, 31 May–2 June 2017.

6. Blaha, M.; Šilinger, K. Application support for topographical-geodetic issues for tactical and technical control of artillery fire. *Int. J. Circuits Syst. Signal Proc*. **2018**, *12*, 48–57.

7. Švorc, L.; Haššo, M.; Sarakhman, O.; Kianičková, K.; Stanković, D.M.; Otřísal, P. A progressive electrochemical sensor for food quality control: Reliable determination of theobromine in chocolate products using a miniaturized boron-doped diamond electrode. *Microchem. J*. **2018**, *142*, 297–304.

8. Petrea, N.; Ginghina, R.; Pretorian, A.; Petre, R.; Barsan, G.; Otřísal, P.; Mosteanu, D. Experimental Survey Regarding the Dangerous Chemical Compounds from Military Polygons that Affect the Military Health and the Environment. *Rev. Chim*. **2018**, *69*, 1640–1644.

9. Paucar, C.; Morales, L.; Pinto, K.; Sánchez, M.; Rodríguez, R.; Gutierrez, M.; Palacios, L. Use of Drones for Surveillance and Reconnaissance of Military Areas. In Proceedings of the Developments and Advances in Defense and Security, Salinas, Peninsula de Santa Elena, Ecuador, 18-20 April 2018.

10. Chen, H.-X.; Ying Nan, Y.; Yang,Y. Multi-UAV Reconnaissance Task Assignment for Heterogeneous Targets Based on Modified Symbiotic Organisms Search Algorithm. *Sensors* **2019**, *19*, 734.

11. Wang, J.; Guo, J.; Zheng, M.; Wang, Z.; Li, Z. Uncertain multiobjective orienteering problem and its application to UAV reconnaissance mission planning. *J. Intell. Fuzzy Syst*. **2018**. *34*, 2287–2299.

12. Vivaldini, K.C.T.; Martinelli,T.H.; Guizilini, V.C.; Souza, J. R.; Oliveira, M. D.; Ramos, F.T., Wolf, D.F. UAV route planning for active disease classification. *Auton. Robots*, **2018**.

13. Liu, H.; Lin, M.; Deng, L. UAV route planning for aerial photography under interval uncertainties. *Optik*, **2016**, *127*, 9695–9700.

14. Yao, P.; Xie, Z.; Ren, P. Optimal UAV Route Planning for Coverage Search of Stationary Target in River. *IEEE Trans. Control Syst. Technol*. **2019**, *27*, 822–829.

15. Dong, J.; Wu, G.; Yang, T.; Jiang, Z. Battlefield situation awareness and networking based on agent distributed computing. *Phys. Commun*. **2018**, *33*, 178–186.

16. Geiger, B. Unmanned Aerial Vehicle Trajectory Planning with Direct Methods, PhD thesis, Pennsylvania State University, University Park, PA, August 2009.

17. Chevaleyre, Y. Theoretical Analysis of the Multi-Agent Patrolling Problem. In Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology, Beijing, China, 24 September 2004.

18. Turker, T.; Yilmaz, G.; Sahingoz, O.K. GPU-Accelerated Flight Route Planning for Multi-UAV Systems Using Simulated Annealing. *Artif. Intell. Method. Syst. Appl*. **2016**, *9883*, 279–288.

19. Manyam, S.G.; Rasmussen, S., Casbeer, D.W.; Kalyanam, K., Manickam, S. Multi-UAV Routing for Persistent Intelligence Surveillance & Reconnaissance Missions. In Proceedings of the International Conference on Unmanned Aircraft Systems, Miami, FL, USA, 13–16 June 2017.

20. Zhen, L.; Lia, M.; Laport, G.; Wang, W. A vehicle routing problem arising in unmanned aerial monitoring. *Comput. Oper. Res*. **2019**, *105*, 1–11.

21. Mozaffari, M.; Saad, W.; Bennis, M.; Debbah, M. Efficient Deployment of Multiple Unmanned Aerial Vehicles for Optimal Wireless Coverage. *IEEE Commun. Lett*. **2016**, *20*, 1647–1650.

22. Cabreira, T.M.; Brisolara, L.B.; Ferreira, P.R. Survey on Coverage Path Planning with Unmanned Aerial Vehicles. *Drones*. **2019**, 3, 4.

23. Bourgault, F.; Furukawa, T.; Durrant-Whyte, H.F. Optimal Search for a Lost Target in a Bayesian World. *Field Serv. Rob*. **2006**, *24*, 209–222.

24. Hollinger, G.; Singh, S.; Djugash, J.; Kehagias, A. Efficient Multi-robot Search for a Moving Target. Int. *J. Rob. Res*. **2009**, *28,* 201–219.

25. Geng,L.; Zhang, Y.; Wang, J.; Fuh, J.Y.; Teo, S. Mission planning of autonomous UAVs for urban surveillance with evolutionary algorithms. In Proceedings of the IEEE International Conference on Control and Automation (ICCA), Hangzhou, China, 12–14 June 2013.

26. Jakob, M.; Semsch, E.; Pavlíček, D.; Pěchouček, M. Occlusion-aware multi-UAV surveillance of multiple urban areas. In Proceedings of the Workshop on Agents in Traffic and Transportation, Toronto, Canada, 11 May 2010, pp. 59–66.

27. Savkin, A.V.; Huang, H. Proactive Deployment of Aerial Drones for Coverage over Very Uneven Terrains: A Version of the 3D Art Gallery Problem, *Sensors* **2019**, *19* 1438.

28. Stodola, P.; Kozůbek, J., Drozd, J. Using Unmanned Aerial Systems in Military Operations for Autonomous Reconnaissance. In Proceedings of the Modelling and Simulation for Autonomous Systems. Prague, Czech Republic, 17–19 October 2018, pp. 514–529.

29. Stodola, P. Improvement in the Model of Cooperative Aerial Reconnaissance Used in the Tactical Decision Support System. *J. Defense Model, Simul.* **2017**, *14*, 483–492.

30. Fortune, S. A sweepline algorithm for Voronoi diagrams. *Algorithmica* **1987**, *2*, 153–174.

31. Stodola, P. Optimization Methods for Position of Unattended Ground Monitoring Systems. In Proceedings of the International Conference on Transport Means. Kaunas, Lithuania, 19–20 October 2006, pp. 79–182.

32. Stodola, P.; Mazal, J. Model of Optimal Cooperative Reconnaissance and its Solution using Metaheuristic Methods. *Defence Sci. J.* **2017**, *67*, 529–1535.

33. Stodola, P.; Mazal, J. Architecture of the Advanced Command and Control System. In Proceedings of the International Conference on Military Technologies. Brno, Czech Republic, 31 May–2 June 2017, pp. 340–343.

34. Tang, J.; Zhua, F.; Piera, M.A. A causal encounter model of traffic collision avoidance system operations for safety assessment and advisory optimization in high-density airspace. *Transport. Res. C-Emer*, **2018**, *96*, 347–365.