Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

UAV search-and-rescue planning using an adaptive memetic algorithm*

Libin HONG[1], Yue WANG[2], Yichen DU[2], Xin CHEN[1], Yujun ZHENG[†‡1]

[1]*School of Information Science and Technology, Hangzhou Normal University, Hangzhou 311121, China*

[2]*College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China*

[†]E-mail: yujun.zheng@computer.org

Received Nov. 13, 2020; Revision accepted May 16, 2021; Crosschecked Sept. 1, 2021

Abstract: The use of unmanned aerial vehicles (UAVs) is becoming more commonplace in search-and-rescue tasks, but UAV search planning can be very complex due to limited response time, large search area, and multiple candidate search modes. In this paper, we present a UAV search planning problem where the search area is divided into a set of subareas and each subarea has a prior probability that the target is present in it. The problem aims to determine the search sequence of the subareas and the search mode for each subarea to maximize the probability of finding the target. We propose an adaptive memetic algorithm that combines a genetic algorithm with a set of local search procedures and dynamically determines which procedure to apply based on the past performance of the procedures measured in fitness improvement and diversity improvement during problem-solving. Computational experiments show that the proposed algorithm exhibits competitive performance compared to a set of state-of-the-art global search heuristics, non-adaptive memetic algorithms, and adaptive memetic algorithms on a wide set of problem instances.

Key words: Memetic algorithm; Self-adaptive; Unmanned aerial vehicle (UAV); Search-and-rescue
https://doi.org/10.1631/FITEE.2000632 **CLC number:** TP399

1 Introduction

As one of the most challenging humanitarian missions, search-and-rescue operations include finding and providing assistance to victims lost in complex terrains such as mountains, forests, and rivers. Such missions can greatly benefit from the use of unmanned aerial vehicles (UAVs) to survey the environment by providing imagery, video, and other sensory data for finding evidence about target location (Murphy et al., 2008; Waharte and Trigoni, 2010), especially in regions that are inaccessible and dangerous to humans. With the rapid improvement of functionality, flexibility, and autonomy of UAVs, their use in these missions is becoming more commonplace, as demonstrated by recent humanitarian operations in disasters including the 2015 Tianjin Port explosion, the 2016 Kaohsiung earthquake, and Syrian conflict.

In most search-and-rescue missions, the search area is large and the time is very limited. If there is no prior knowledge about the target location, then the only strategy is to exhaustively and uniformly search the area. However, in practice, there is usually some information available to indicate that the search effort should not be evenly distributed over the entire area. In typical search theory formulations, the search area is usually divided into a set of subareas, each being assigned a probability of target existence (Kamrani and Ayani, 2009). The

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (Nos. 61872123 and 61473263) and the Zhejiang Provincial Natural Science Foundation, China (No. LR20F030002)

ORCID: Libin HONG, https://orcid.org/0000-0003-2579-0523; Yujun ZHENG, https://orcid.org/0000-0002-6095-6325

aim of search planning is to appropriately allocate the search time and other resources so that subareas with higher probabilities are prioritized, i.e., to search more thoroughly and earlier.

There are different problem formulations and solution methods for UAV search planning in the literature, and most of them adopt one or more of the following assumptions:

1. The search area is mapped into a grid (i.e., each subarea is a grid cell), and thus choices for the next step of movement can be limited to four directions (North, East, South, and West). This is not suitable in many cases where subareas may be separated by terrains that are inaccessible to the victims or regions occupied by rescue teams.

2. The UAV flies at a fixed altitude (or a constant height over the ground). This may be reasonable for traditional fixed-wing UAVs but inadequate for today's UAVs that continue to become more lightweight and agile (Waharte et al., 2010).

3. The path is symmetric; i.e., the time for moving from one subarea to another is the same as that in the opposite direction. This assumption is also based on the previous assumption and can greatly simplify the planning task. However, it is ineffective if the UAV searches two subareas at different altitudes, where the time for the UAV to fly from a higher altitude to a lower altitude can be less than that in the opposite direction.

4. The UAV can always measure the state of an entire subarea; that is, the posterior probability that the target is found by the UAV searching the subarea containing the target is assumed to be 100%. However, ensuring that every sign can be sensed by the UAV may be impossible or take an unacceptable amount of time, and in most cases we need to find a workable tradeoff between coverage and detection (Goodrich et al., 2008).

In this paper, we present a UAV search planning problem that does not make any of the above assumptions and is therefore much more practical. In our problem, the search area is divided into a set of subareas that are not necessarily adjacent to each other. Each subarea has a prior probability that the target is present in it, which usually depends on information about the target, such as the time and place last seen and environmental features (e.g., rivers or roads). If no prior information is known, then we can assume a uniform distribution (Waharte and Trigoni, 2010). The UAV can search each subarea at different altitudes: the higher the altitude, the larger the proportion covered and the less search time needed, but the less the precision of observation and the smaller the consequent probability of finding the target (Fig. 1). Thus, it is reasonable to perform more detailed search on subareas with a higher probability of target location. The aim of the problem is to work out the most effective search plan, including the search sequence of the subareas and the search altitude for each subarea, such that the probability of finding the target can be maximized and the target can be found as quickly as possible.

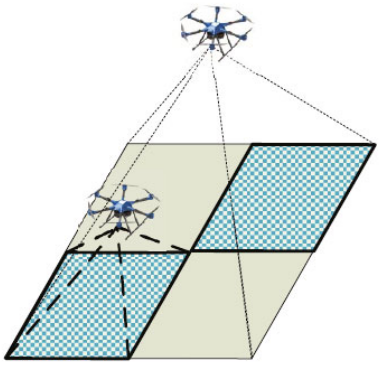

Fig. 1 Illustration of unmanned aerial vehicle (UAV) search at different altitudes

Because the considered problem is highly complex, it is impractical to use traditional exact algorithms (e.g., brute-force search or its variants) to solve large-size instances. In this paper, we propose an adaptive memetic algorithm, which defines a set of different local search (LS) procedures and adapts the choice of the procedures based on their historical performance measured in fitness improvement and diversity improvement during problem-solving. Computational experiments on a variety of problem instances show that the proposed algorithm exhibits competitive performance compared to a set of state-of-the-art global search heuristics, non-adaptive memetic algorithms, and adaptive memetic algorithms.

2 Related work

UAV path planning problems, from two-dimensional (2D) path planning to three-dimensional (3D) path planning, from path planning for a single UAV to coordinated path replanning

for multiple UAVs, have been extensively studied in the literature (Chandler et al., 2000; Nikolos et al., 2007; Tisdale et al., 2009; Dong et al., 2011; Ragi and Chong, 2013; Duan and Li, 2014; Zhang B and Duan, 2014). However, most of the problems focus on obstacle avoidance and target sensing/tracking, and the studies on UAV path planning for search-and-rescue tasks are relatively limited. Considering a problem of searching for a lost target at sea by a single UAV, Bourgault et al. (2006) used a one-step lookahead method that always chooses an area with the maximum probability of finding the target. This myopic planning often fails to maximize the overall probability of success, which is similar to the greedy search, one of the three search methods proposed by Hansen et al. (2007). The two other methods are the contour search which follows offset paths to conduct searches in a highly systematic fashion without leaving large holes or overlap, and the composite search which switches between the greedy search and the contour search. Zhou et al. (2020) studied cooperative UAV target tracking with a limited communication range by integrating UAV motion control, target state estimation, and network topology control. They demonstrated the stability and convergence properties of the coupled system under bounded noise. Lin L and Goodrich (2009) modeled a wilderness search-and-rescue problem as a combinatorial optimization problem with respect to probability accumulated in the 2D space, and developed a set of algorithms based on ideas including local hill climbing, convolution, and evolutionary algorithms. Both of the evolutionary algorithms proposed by them use the probability accumulated for each path as the fitness function; the path representation is encoded as a sequence of directions (North, East, South, and West) in the first algorithm and as a sequence of node positions in the second one. However, the problem assumption of a 100% target detection rate does not always hold in practice.

Another assumption often used in UAV search problems is that the altitude of the UAV remains unchanged, which is reasonable for traditional fixed-wing UAVs but inadequate for today's lightweight and agile UAVs (Waharte et al., 2010). Considering a problem of detecting a slow-moving ground target by a UAV, Al-Helal and Sprinkle (2010) derived closed-form expressions for calculating the altitude at which the UAV should fly to maximize the probability of detection, but the work was also based on the assumption that the UAV can measure the entire state of the search area. Waharte and Trigoni (2010) analyzed the performance of different search methods in three categories, including greedy heuristics, potential-based heuristics, and partially observable Markov decision process (POMDP) based heuristics by considering that UAVs can change the size of their observation areas by changing altitudes. The results showed that using POMDP-based heuristics to exploit the acquired or prior knowledge of the target location can speed up the search, but its computational cost could not be neglected.

Bertuccelli and How (2006) investigated the search for a moving target whose motion is poorly known. They proposed an algorithm that uses an approach similar to particle filtering to stochastically simulate the uncertain state transition matrix, but approximates the posterior distribution with an analytical, closed-form distribution. To control and plan the path of a UAV in tracking a helicopter performing search-and-rescue, Ryan and Hedrick (2005) proposed an algorithm consisting of four modes, named cut corner path, fixed curve path, convergent path, and outside loop path, each of which was formulated to provide safety and contiguous sensor coverage between the UAV and the helicopter. For search-and-identify missions, van Willigen et al. (2011) used particle swarm optimization to generate a predefined UAV path, which can be adapted at runtime based on newly acquired information. Ruan and Duan (2020) proposed a multi-objective social learning pigeon-inspired optimization algorithm for multi-UAV obstacle avoidance. They verified the effectiveness of the method by simulating the flight process of five UAVs in a complex obstacle environment. To solve a UAV search path planning problem, Wang Y et al. (2017) adapted five metaheuristic algorithms and then integrated them into a hyper-heuristic framework that exhibits better performance than any individual heuristic. However, the adaptive selection of the hyper-heuristic is at the individual metaheuristic level rather than at the evolutionary operator level; the hyper-heuristic also does not exploit LS procedures to improve solution accuracy. There are also some studies devoted to the cooperative search of multiple UAVs in search-and-rescue (Jin et al., 2006; Altshuler et al., 2008;

Waharte et al., 2010; López-Ortiz and Maftuleac, 2015; Du et al., 2019; Zheng et al., 2020, 2021), but they adopted assumptions similar to the above problems with a single UAV. Interested readers can refer to Zhang H et al. (2020) for a review of cooperative UAV path planning. Compared to the problems in the literature, our problem considers both the posterior probability of detection and changes of the flying altitude.

3 UAV search problem

First we formulate a problem that uses a UAV to search for a target (e.g., a victim) in a search area consisting of m subareas that are not necessarily adjacent (we will show that the formulation can be easily extended to the cases with multiple targets and/or multiple UAVs at the end of this section). The variables of the problem are shown in Table 1. Each subarea i ($1 \le i \le m$) is assigned a probability $p(i)$ of target existence, assuming that we have a prior probability distribution function that describes the initial belief of the target location. The (average) altitude of subarea i is denoted as $\alpha(i)$.

Ideally, the UAV can fly at any altitude below its highest flight altitude, but in practice we can limit the UAV flight height (over the subarea it searches) to one of K values, defined as $h(1), h(2), \cdots, h(K)$, and we say that the search mode is k if the UAV flies at $h(k)$ ($1 \le k \le K$). For example, the UAV can use a standard NTSC (National Television Systems Committee) video camera to detect a human shape at a height of about 60 m or detect unusual colors from clothing at a height of about 100 m (Goodrich et al., 2008). The expected time for the UAV to search through subarea i at $h(k)$ is estimated as $t_{\mathrm{s}}(i,k)$. Moreover, based on information including the camera's resolution and field of view, the subarea's size and terrain features, and the target's size and other observable features, we can derive a posterior probability $p_{\mathrm{s}}(i,k)$ that the target can be detected by the UAV using mode k on subarea i under the precondition that the target is presented in subarea i ($1 \le i \le m,\ 1 \le k \le K$).

Detection probability and time are two critical factors in search planning. In addition to $t_{\mathrm{s}}(i,k)$, we are given the time for the UAV to fly from subarea i at height $h(k_i)$ to subarea j at height $h(k_j)$, denoted by $t_{\mathrm{f}}(i,k_i,j,k_j)$ ($1 \le i \le m,\ 1 \le k \le K$). As a special case, we use $t_{\mathrm{f}}(0,0,i,k_i)$ to denote the time for the UAV to fly from the starting position (usually the position of the rescue team) to subarea i at height $h(k_i)$.

Table 1 Nomenclature

Input variable	Explanation
m	Number of subareas
$p(i)$	Probability of target existence in subarea i
$\alpha(i)$	The (average) altitude of subarea i
$d(i,j)$	Horizontal distance between two subareas i and j
K	Number of search modes
$h(k)$	Flight height of search mode k
$p_{\mathrm{s}}(i,k)$	Posterior probability that the target can be detected by the UAV using mode k on subarea i if it contains the target
$t_{\mathrm{s}}(i,k)$	Time for searching through subarea i with mode k
$t_{\mathrm{f}}(i,k_i,j,k_j)$	Time for flying from subarea i with mode k_i to subarea j with mode k_j
t^{U}	Upper limit of the completion time of the search
Decision variable	**Explanation**
$\boldsymbol{x}=(x_1,x_2,...,x_m)$	Sequence of the subareas to be searched
$\boldsymbol{y}=(y_1,y_2,...,y_m)$	Search modes used for the subareas $(x_1,x_2,...,x_m)$
Intermediate variable	**Explanation**
$t(x_i)$	End time of the search on subarea x_i

In practice, $t_{\mathrm{f}}(i,k_i,j,k_j)$ can be estimated based on the horizontal distance $d(i,j)$ between subareas i and j, the UAV's weight M, the maximum horizontal velocity $v_{\max}$, and the maximum thrust force $F_{\max}$. Let $H_i=\alpha(i)+h(k_i)$. Here, we need to consider two cases, $H_i \ge H_j$ and $H_i < H_j$. When $H_i \ge H_j$, as illustrated in Fig. 2a, the UAV's velocity on the path from subareas i to j can be estimated as

$$v(i,k_i,j,k_j)=\frac{F_{\max}+Mg\sin\theta}{F_{\max}}v_{\max}, \qquad (1)$$

where g is the gravitational acceleration, θ is the angle between the horizontal and the velocity vector, i.e., $\theta=\arctan(h(i,j)/d(i,j))$, and $h(i,j)=H_i-H_j$

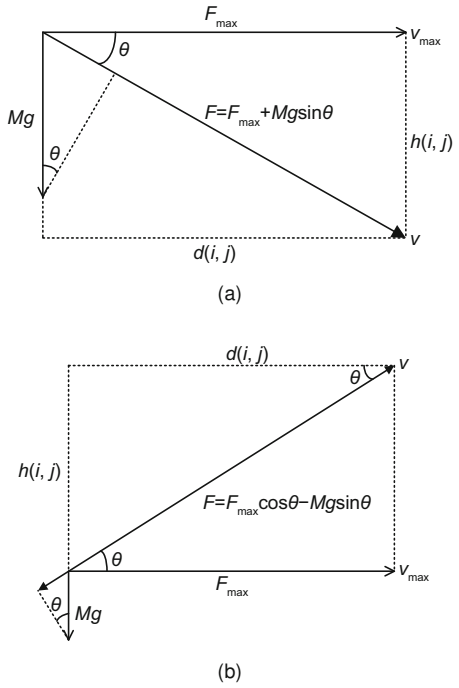

Fig. 2 Illustration of estimating flying time, for which we assume that the velocity is proportional to the thrust force along the direction of the velocity: (a) $H_i \geq H_j$, the UAV glides; (b) $H_i < H_j$, the UAV climbs

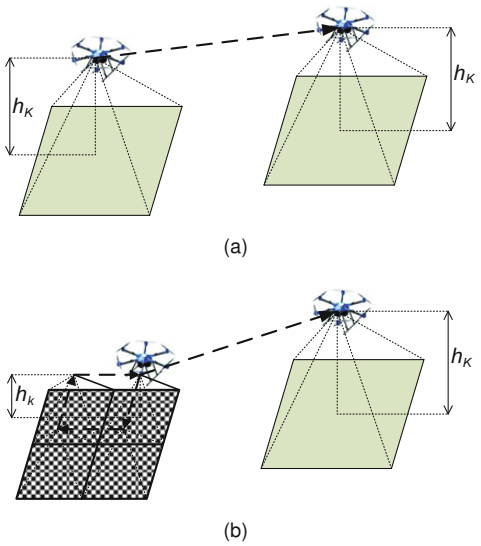

Fig. 3 The horizontal distance for the UAV to fly from one subarea to another: (a) the UAV uses search mode K on both subareas, and the horizontal distance is between the centers of the two subareas; (b) the UAV uses search mode $k < K$ on the first subarea, and thus the distance starts from the UAV's last position in the first subarea to the first position in the second subarea

is the vertical distance. Then the flying time is

$$t_{\mathrm{f}}(i,k_i,j,k_j)=\frac{\sqrt{d^2(i,j)+h^2(i,j)}F_{\max}}{(F_{\max}+Mg\sin\theta)v_{\max}}. \quad (2)$$

The case of $H_i < H_j$ is a bit more difficult because the UAV climbs. As illustrated in Fig. 2b, the velocity can be estimated as

$$v(i,k_i,j,k_j)=\frac{F_{\max}\cos\theta-Mg\sin\theta}{F_{\max}}v_{\max}, \quad (3)$$

and the flying time is

$$t_{\mathrm{f}}(i,k_i,j,k_j)=\frac{\sqrt{d^2(i,j)+h^2(i,j)}F_{\max}}{(F_{\max}\cos\theta-Mg\sin\theta)v_{\max}}. \quad (4)$$

It should be noted that Eqs. (2) and (4) are coarse estimations. More precisely, the horizontal distance may also depend on the UAV's search modes. For example, when sensing a subarea with the largest search mode K, the UAV usually stops at only one position, i.e., right above the center of the subarea, and thus the distance $d(i,j)$ is between the two centers (Fig. 3a). However, when using a more refined search mode $k < K$, the UAV may need to stop at several positions to sense the subarea, and thus $d(i,j)$ is between the last position of subarea i and the first position of subarea j (Fig. 3b). This can also be added to our problem formulation without much effort.

The decision variables of our problem can be divided into two parts:

1. The search sequence $\boldsymbol{x}=(x_1,...,x_i,...,x_m)$, where x_i denotes the i^{th} subarea to be searched ($1\leq i\leq m$);

2. The search modes $\boldsymbol{y}=(y_1,...,y_i,...,y_m)$, where y_i is the search mode used for subarea x_i ($1\leq i\leq m$).

The search on the first subarea x_1 starts at $t_{\mathrm{f}}(0,0,x_1,y_1)$ and ends at

$$t(x_1)=t_{\mathrm{f}}(0,0,x_1,y_1)+t_{\mathrm{s}}(x_1,y_1). \quad (5)$$

The search on subarea x_2 starts at $t(x_1)+t_{\mathrm{f}}(x_1,y_1,x_2,y_2)$ and ends at

$$\begin{aligned}t(x_2)=&t_{\mathrm{f}}(0,0,x_1,y_1)+t_{\mathrm{f}}(x_1,y_1,x_2,y_2)\\&+t_{\mathrm{s}}(x_1,y_1)+t_{\mathrm{s}}(x_2,y_2).\end{aligned} \quad (6)$$

By analogy, the end time of the search on

subarea x_i $(1 \le i \le m)$ is calculated as

$$t(x_i) = t_{\mathrm{f}}(0,0,x_1,y_1) + \sum_{i'=2}^{i} t_{\mathrm{f}}(x_{i'-1}, y_{i'-1}, x_{i'}, y_{i'}) + \sum_{i'=1}^{i} t_{\mathrm{s}}(x_{i'}, y_{i'}). \tag{7}$$

The objective is to maximize the total time-weighted probability of finding the target, and the problem is formulated as

$$\max \quad f(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{m} \frac{t^{\mathrm{U}} - t(x_i)}{t^{\mathrm{U}}} p_{\mathrm{s}}(x_i,y_i)p(x_i) \tag{8}$$

$$\text{s.t.} \quad \text{Eq. (7)},$$

$$t(x_m) \le t^{\mathrm{U}}, \tag{9}$$

$$\boldsymbol{x} \in \mathrm{perms}(1,2,\cdots,m), \tag{10}$$

$$1 \le y_i \le K, \quad i = 1,2,\cdots,m, \tag{11}$$

where t^{U} is the upper limit of the completion time of the operation, and $\mathrm{perms}(1,2,\cdots,m)$ denotes the set of all permutations of $\{1,2,\cdots,m\}$. The objective function (8) indicates that the higher the probability $p(x_i)$ that the target exists in a subarea, the smaller the value of $t(x_i)$ and the larger the expected values of $p_{\mathrm{s}}(x_i,y_i)$; i.e., the subarea is expected to be searched more quickly and with a more detailed mode. Constraint (9) indicates that the completion time of the search should not exceed the upper limit. Note that in practice, whenever the UAV finds the target in subarea x_i, the search stops and the remaining $(m-i)$ subareas will not be investigated.

It is not difficult to extend the above problem formulation to the case of searching multiple targets. Suppose that there are n targets; each subarea i is assigned with n probabilities $p(i,1),p(i,2),\ldots,p(i,n)$, where $p(i,j)$ denotes the probability that target j is in the subarea, and it is also assigned with $n \cdot K$ posterior probabilities, where each $p_{\mathrm{s}}(i,j,k)$ denotes the posterior probability of finding target j in subarea i with search mode k $(1 \le i \le m,\ 1 \le j \le n,\ 1 \le k \le K)$. In this case, the objective function (8) can be extended to maximize the sum of the (weighted) probabilities of finding all the targets:

$$\max f(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{m}\sum_{j=1}^{n} \frac{t^{\mathrm{U}} - t(x_i)}{t^{\mathrm{U}}} w_j p_{\mathrm{s}}(x_i,j,y_i) \cdot p(x_i,j), \tag{12}$$

where w_j is the importance weight of target j.

When there are multiple UAVs, the problem can be divided into two stages: the first stage is to allocate the subareas to the UAVs, and the second stage is to determine the search sequence and search modes for each UAV based on the above problem formulation (Eqs. (8)–(11)).

4 An adaptive memetic algorithm for the problem

From the above formulation, we can see that the problem contains a subproblem of determining the search sequence $\boldsymbol{x}$, which is an extended version of the NP-hard traveling salesman problem (TSP). Our problem is much more complex than the TSP in two aspects: (1) its objective function is much more computationally expensive; (2) for each subsolution $\boldsymbol{x}$, we need to solve an integer programming problem to determine the search modes $\boldsymbol{y}$. We have extended a branch-and-bound algorithm originally proposed for the TSP (Volgenant and Jonker, 1982) for the presented UAV search planning problem, but found that, within a time limit of 30 min, the exact algorithm can solve only small instances with up to 30 subareas and two search modes. Nevertheless, in emergency search-and-rescue operations the size can be much larger and the solution time is usually limited to several minutes. Therefore, it is necessary to use much more efficient heuristic methods.

Based on the memetic algorithm framework we propose here a heuristic method that combines a population-based global search with different LS procedures (Chak and Feng, 1995; Moscato and Cotta, 2003; Krasnogor and Smith, 2005) and has been shown to be efficient in solving a variety of optimization problems (Qi et al., 2015; Zheng et al., 2015; Cabassi and Locatelli, 2016; Lai and Hao, 2016; Wang XP and Tang, 2017; Zhang YZ et al., 2017). In particular, our method consists of multiple LS procedures (also called multi-memes in the field of memetic algorithms (Ong et al., 2006; Özcan et al., 2016; Sheng et al., 2016)) and uses an adaptive approach to decide which procedure to apply according to their past performance measured in two aspects, fitness and diversity. The pseudocode of our algorithm for the problem is presented in Algorithm 1, where $\widehat{r}$, $\widehat{\mu}$, and g^{L} are control parameters, rand() produces a random number uniformly distributed in [0,1], and $f_{\max}$ and $f_{\min}$ are the maximum and

minimum objective values among the population, respectively.

The key contribution of our algorithm is twofold: (1) we propose six LS procedures that are efficient and specific to the problem; (2) we design a simple yet effective memetic strategy for hybridization and adaptive selection among the LS procedures. Results show that the proposed algorithm outperforms several state-of-the-art metaheuristics and memetic algorithms on test instances.

Algorithm 1 Proposed adaptive memetic algorithm for the UAV search planning problem

Randomly initialize a population P of solutions to the problem
while termination condition is not satisfied **do**
  Create an empty population P' for the next generation
  for each solution $\boldsymbol{z}=(\boldsymbol{x},\boldsymbol{y})\in P$ **do**
    if rand() $<\widehat{r}\cdot\frac{f(\boldsymbol{z})-f_{\min}}{f_{\max}-f_{\min}}$ **then**
      if $\boldsymbol{z}$ has been unchanged for g^{L} generations **then**
        Replace $\boldsymbol{z}$ with a new randomly generated solution
      end if
      Add $\boldsymbol{z}$ to P'
    else
      Select another solution $\boldsymbol{z}'\in P$ with a probability proportional to $f(\boldsymbol{z}')$
      Use crossover to generate an offspring $\boldsymbol{z}^{\mathrm{c}}$ from $\boldsymbol{z}$ and $\boldsymbol{z}'$
      if rand() $<\widehat{\mu}\cdot\frac{f_{\max}-f(\boldsymbol{z}^{\mathrm{c}})}{f_{\max}-f_{\min}}$ **then**
        Mutate $\boldsymbol{z}^{\mathrm{c}}$
      else
        Select a meme with a probability proportional to its suitability
        Apply the meme on $\boldsymbol{z}^{\mathrm{c}}$ and then update the fitness of the meme
      end if
      Add $\boldsymbol{z}^{\mathrm{c}}$ to P'
    end if
  end for
  $P\leftarrow P'$
end while
return the best solution found so far

4.1 Global search procedure

Our algorithm first initializes a population P of solutions. At each generation, each solution $\boldsymbol{z}=(\boldsymbol{x},\boldsymbol{y})$ has a probability $\left(\widehat{r}\cdot\frac{f(\boldsymbol{z})-f_{\min}}{f_{\max}-f_{\min}}\right)$ of directly entering the population for the next generation; the larger the objective value $f(\boldsymbol{z})$, the higher the probability. Otherwise, a crossover operation is conducted on $\boldsymbol{z}$ and another solution $\boldsymbol{z}'=(\boldsymbol{x}',\boldsymbol{y}')$ selected with a probability proportional to its objective value to generate an offspring $\boldsymbol{z}^{\mathrm{c}}=(\boldsymbol{x}^{\mathrm{c}},\boldsymbol{y}^{\mathrm{c}})$. The crossover is based on the position-based crossover (PBX) (Syswerda, 1991), which is chosen by testing a number of well-known permutation-based crossover operators. The crossover operation is performed as follows:

1. Randomly select a subset of n positions in $\boldsymbol{x}$, and copy the components at these positions to the corresponding positions in $\boldsymbol{x}^{\mathrm{c}}$ (where n is a random integer uniformly distributed in $[1,m/2]$);

2. Fill the other positions of $\boldsymbol{x}^{\mathrm{c}}$ with the remaining components in the same order as in $\boldsymbol{x}'$;

3. For each x_i^{c} inherited from $\boldsymbol{x}$, set the corresponding y_i^{c} as a random integer in the range of $[\min(y_i,y_i'),\max(y_i,y_i')]$;

4. For each x_i^{c} inherited from $\boldsymbol{x}'$, set the corresponding $y_i^{\mathrm{c}}=y_i'$.

In this way, the offspring $\boldsymbol{z}^{\mathrm{c}}$ inherits more features from the second (and probably fitter) parent $\boldsymbol{z}'$. Fig. 4 illustrates such a crossover operation, where $x_2=3$, $x_4=4$, and $x_6=5$ of the first parent $\boldsymbol{z}$ are directly copied to the offspring, their search modes (probably) get close to the modes of the second parent $\boldsymbol{z}'$, and the remaining components of the offspring are set in the same order as in $\boldsymbol{z}'$.

Each newly generated offspring has probability $\widehat{\mu}\cdot\frac{f_{\max}-f(\boldsymbol{z}^{\mathrm{c}})}{f_{\max}-f_{\min}}$ of being mutated; the smaller the objective value, the higher the probability. The mutation operation is performed as follows:

1. Reverse a randomly selected subsequence of $\boldsymbol{x}^{\mathrm{c}}$ while keeping the search mode of each subarea;

2. Randomly set each y_i^{c} to another value in $[1,K]$ with a probability of 0.5.

If an offspring is not mutated, it will be improved by a dynamically selected LS procedure.

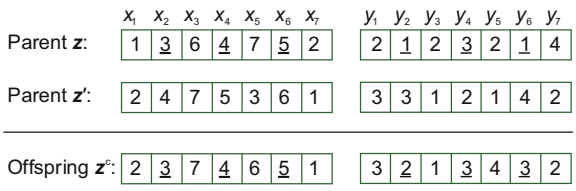

Fig. 4 Illustration of the crossover operation for the considered problem

4.2 Local search procedures

We propose six LS procedures to improve a solution $\boldsymbol{z}=(\boldsymbol{x},\boldsymbol{y})$, among which four procedures are conducted on search sequence $\boldsymbol{x}$ and the other two are conducted on search modes $\boldsymbol{y}$.

4.2.1 LS1: multiple swaps in a search sequence

The first procedure generates $k_{\max}$ neighboring solutions (where $k_{\max}$ is a control parameter), each of which is obtained by swapping two randomly selected components x_i and x_j and then swapping the corresponding y_i and y_j. The best neighboring solution, if better than the original $\boldsymbol{z}$, will replace $\boldsymbol{z}$ in the population.

4.2.2 LS2: consecutive swaps in a search sequence

The second procedure is performed as follows:

1. Let $k=1, \boldsymbol{z}'=\boldsymbol{z}$.

2. Swap two randomly selected components x'_i and x'_j and then swap the corresponding y'_i and y'_j. If $\boldsymbol{z}'$ is better than the original $\boldsymbol{z}$, use $\boldsymbol{z}'$ to replace $\boldsymbol{z}$ and stop the procedure.

3. Set $k=k+1$. If $k<k_{\max}$, then go to step 2; otherwise, stop the procedure.

4.2.3 LS3: multiple insertions in a search sequence

The third procedure generates $k_{\max}$ neighboring solutions, each of which is obtained by randomly picking a component x_i and re-inserting it into another position while keeping its search mode. The best neighboring solution, if better than the original $\boldsymbol{z}$, will replace $\boldsymbol{z}$ in the population.

4.2.4 LS4: consecutive insertions in a search sequence

The fourth procedure is performed as follows:

1. Let $k=1, \boldsymbol{z}'=\boldsymbol{z}$.

2. Randomly pick a component x_i and re-insert it into another position. If $\boldsymbol{z}'$ is better than the original $\boldsymbol{z}$, use $\boldsymbol{z}'$ to replace $\boldsymbol{z}$ and stop the procedure.

3. Set $k=k+1$. If $k<k_{\max}$, then go to step 2; otherwise, stop the procedure.

4.2.5 LS5: multiple nearest search modes

The fifth procedure generates $k_{\max}$ neighboring solutions, each of which is obtained by setting a randomly selected component $y_i=y_i\pm 1$. The best neighboring solution, if better than the original $\boldsymbol{z}$, will replace $\boldsymbol{z}$ in the population.

4.2.6 LS6: best search mode of a subarea

The sixth procedure generates $(K-1)$ neighboring solutions by setting a randomly selected component y_i to the other $(K-1)$ search modes. The best neighboring solution, if better than the original $\boldsymbol{z}$, will replace $\boldsymbol{z}$ in the population.

4.3 Adaptive selection of the local search procedures

To adaptively select the most appropriate LS procedures during problem-solving, we assign each LS procedure with a suitability index S_i $(1\le i\le 6)$, such that the selection probability of a procedure is proportional to its index. Initially, all S_i values are set to the population size $|P|$. After the first LP generations (where LP is a parameter called the learning period), at each generation we update the values according to the application effects of the procedures in the previous LP generations. The effects are evaluated based on two aspects, fitness improvement and diversity improvement. The fitness improvement of the i^{th} LS procedure over the LP generations is calculated as

$$\mathrm{FI}_i=\frac{2\sum_{\boldsymbol{z}\in Z_i}f(\boldsymbol{z})}{f(\boldsymbol{z}_c^*)+f(\boldsymbol{z}_{c-\mathrm{LP}}^*)}+n_{\mathrm{FI}_i}^*, \tag{13}$$

where Z_i is the set of solutions found by the procedure, c denotes the current generation number, $\boldsymbol{z}_c^*$ is the best known solution at the current generation, $\boldsymbol{z}_{c-\mathrm{LP}}^*$ is the best known solution at the $(c-\mathrm{LP})^{\text{th}}$ generation, and $n_{\mathrm{FI}_i}^*$ is the number of times that the procedure finds a new best known solution.

The diversity improvement of the i^{th} LS procedure over the LP generations is calculated as

$$\mathrm{DI}_i=\frac{\sum_{\boldsymbol{z}\in Z_i}\sum_{k=1}^{K_N}D(\boldsymbol{z},\boldsymbol{z}_k)}{D_{\mathrm{avg}}}, \tag{14}$$

where $D(\boldsymbol{z},\boldsymbol{z}_k)$ is the Euclidean distance between solution $\boldsymbol{z}$ and its k-nearest neighbor $\boldsymbol{z}_k$, K_N is a control parameter typically set to three, and D_{avg} is the average distance among $|P|$ solutions uniformly distributed in the whole search space.

At each generation after the first LP generations, each suitability index is updated as

$$S_i=S_i+\frac{w_{\mathrm{F}}\mathrm{FI}_i+w_{\mathrm{D}}\mathrm{DI}_i}{\kappa_i n_i}, \tag{15}$$

where w_{F} and w_{D} are two control parameters for adjusting the weights of the two improvement metrics, n_i is the number of selections (invocations) of the i^{th} LS procedure during the previous LP generations, and κ_i is a coefficient related to the computational cost of an invocation of the procedure. Because each invocation of LS1, LS3, or LS5 generates $k_{\max}$ neighbors, each operation of LS2 or LS4 generates at most $k_{\max}$ neighbors but will stop whenever a neighbor better than the original one is found, and each operation of LS6 generates $(K-1)$ neighbors, we set the value of κ_i to 1 for LS1, LS3, and LS5, to 0.5 for LS2 and LS4, and to $(K-1)/k_{\max}$ for LS6. Moreover, to facilitate global exploration in early generations and emphasize local exploitation in later generations, we set w_{F} to linearly increase from 0.4 to 0.9 with generation, and set $w_{\mathrm{D}} = 1-w_{\mathrm{F}}$.

5 Computational experiments

5.1 Experimental setting

We use a test set of 18 problem instances, including 12 instances for single-target search and 6 instances for multi-target search. Table 2 presents the basic information of the instances. All the instances are constructed based on real-world terrains; even for some instances with similar sizes, their geographic and geomorphic conditions can be very different and hence their difficulties are not similar. Among the 18 instances, instance 15 is designed to search for missing boats and bodies in a vast lake, instances 3, 7, and 12 to search for missing tourist in wide plains, and all other instances to search for lost targets in mountains or mountain forests.

To validate the performance of the proposed adaptive memetic algorithm, on the test instances, we compare it with 15 other algorithms, which can be divided into four groups. The first group is a benchmark algorithm (B&B) that uses a branch-and-bound method (Volgenant and Jonker, 1982) to find the exact optimal search sequence and uses another branch-and-bound method (Tomlin, 1971) to determine the optimal search mode for each search sequence. The second group has four algorithms, including a genetic algorithm (GA) that uses the same crossover and mutation operations as our algorithm but does not use LS, and the following three state-of-the-art metaheuristic algorithms for UAV path planning:

Table 2 Summary of the test instances of the component selection problem

Instance	m	K	n	Area (km^2)	$\bar{d}$ (km)	$\bar{h}$ (m)
1	20	4	1	11.6	0.9	135
2	20	5	1	11.6	0.9	135
3	27	4	1	9.5	0.7	266
4	32	4	1	13.3	1.3	30
5	39	6	1	19.6	1.5	48
6	46	5	1	17.9	1.3	26
7	56	5	1	22.1	1.2	41
8	56	6	1	22.1	1.2	41
9	60	4	1	35.2	1.8	103
10	60	6	1	35.2	1.8	103
11	88	5	1	39.0	1.5	59
12	96	4	1	39.7	1.2	36
13	27	4	2	9.5	0.7	266
14	39	6	2	19.6	1.5	48
15	56	5	2	22.1	1.2	41
16	39	6	3	11.6	0.9	135
17	56	5	3	19.6	1.5	48
18	60	4	3	35.2	1.8	103

m: number of subareas; K: number of potential search modes; n: number of targets; Area: total search area; $\bar{d}$: average horizontal distance between a pair of subareas; $\bar{h}$: average vertical distance between a pair of subareas

1. a spherical vector-based particle swarm optimization (SPSO) that searches the configuration space of the UAV via the correspondence between the particle position and the speed, turn angle, and climb/dive angle of the UAV (Phung and Ha, 2021);

2. a quantum-behaved pigeon-inspired optimization (QPIO) algorithm that uses an adaptive factor parameter and gradually decreasing pigeon population (Hu et al., 2019);

3. a knee-guided differential evolution (KNDE) algorithm that uses a knee solution based on the minimal Manhattan distance to guide the search direction of the algorithm (Yu et al., 2021).

The third group consists of six memetic algorithms that combine GA with one of the six LS procedures (denoted as MA1–MA6). The fourth group consists of four algorithms, each using the same crossover, mutation, and six LS procedures as our algorithm, but employing different hyper-heuristics for adaptive selection among the local search procedures:

1. a basic hyper-heuristic that uses roulette selection, where the selection probability of each LS procedure is proportional to its number of times in finding new best solutions (Zheng et al., 2015);

2. a backtracking search (BS) hyper-heuristic algorithm, where a BS algorithm is employed as the

high-level strategy to manipulate low-level LS procedures in a trial population (Lin J, 2019);

3. a multi-local-search hyper-heuristic that uses a multi-armed bandit (MAB) to select one among LS procedures based on reward and punishment calculated according to the solutions produced (Turky et al., 2020);

4. a hyper-heuristic based learning method that calculates the selection probability of each LS procedure according to a set of values including the relative improvement, learning coefficient, and probabilities of the procedures (Burcin et al., 2021).

The above four memetic schemes are denoted as MA-S, MA-BS, MA-B, and MA-L, respectively. Our adaptive memetic algorithm based on fitness and diversity performance metrics is denoted as MA-FD. We tune the parameters of metaheuristic and memetic algorithms using an evolutionary optimization approach based on the framework by Eiben and Smit (2011). That is, for each algorithm, we consider obtaining its optimal parameter setting as an optimization problem, the objective function of which is to maximize the average performance of the algorithm on the whole test set. We employ three fast evolutionary algorithms, including self-adaptive differential evolution (DE) (Qin et al., 2009), ecogeography-based optimization (Zheng et al., 2014), and water wave optimization (Zheng, 2015), to optimize the parameters, and choose the best setting among the results of the three algorithms. As a result, the parameters of MA-FD are as follows: $\widehat{r}=0.88$, $\widehat{\mu}=0.45$, $g^{\mathrm{L}}=6$, $k_{\max}=m/3$, LP=10, and $|P|=30$. Some automatic parameter-tuning tools, such as ParamILS (Hutter et al., 2009) or irace (López-Ibáñez et al., 2016), which automatically invoke a family of algorithms and evaluate their results, can be used for the same purpose.

The experimental environment is a computer with Intel i7-6700 CPU and 8 GB memory. All the algorithms are implemented with Visual C# 2015. The B&B algorithm is used to generate a benchmark solution denoted by $\boldsymbol{z}^*$ for each test instance within a maximum running time of 12 h. All the remaining metaheuristic and memetic algorithms stop when the CPU time reaches 600 s to ensure a fair comparison. Each algorithm is run 30 times on each instance.

5.2 Experimental results

Tables 3 and 4 present the maximum (max), minimum (min), mean, median, and standard deviation (std) of the resulting objective function values on the first 12 (single-target) instances and the last 6 (multi-target) instances, respectively, where the best mean and median value(s) among the 15 metaheuristic algorithms on each instance are shown in bold. We also conduct nonparametric Wilcoxon rank sum tests to compare MA-FD and each other comparative algorithm, and those mean values with superscript "†" indicate that the result of the corresponding algorithm is statistically significantly different from that of MA-FD (at a confidence level of 95%).

The first three instances are of relatively small size and simple, and the median values of the 15 metaheuristic algorithms are all the same as those of the benchmark (optimal) solutions. However, GA cannot always obtain the optimal solutions on the three instances (as denoted by its mean and minimum values), which indicates that its global search cannot always guarantee reaching the global optimum without the help of a local search procedure. QPIO, MA1, and MA2 cannot guarantee the optimal solution on instance 3, indicating that QPIO is less robust than SPSO and KNDE, and that LS1 and LS2 are less robust than the other LS procedures on this instance.

On all remaining instances, none of the metaheuristic and memetic algorithms can guarantee the optimal solutions. However, on instance 4, the median values of all five adaptive memetic algorithms reach the exact optimal objective function value; on instance 5, the median values of MA-L and MA-FD reach the exact optimal value. In general, among the first four metaheuristic algorithms, the overall performance of GA is the worst, and that of KNDE is the best; SPSO and QPIO exhibit similar overall performance to MA1–MA6, but their performances vary from instance to instance. The mean and median values of SPSO are similar to those of QPIO on relatively small- and medium-size instances, but become larger than those of QPIO on large-size instances. On most instances, SPSO obtains better mean and median values than some non-adaptive memetic algorithms but worse values than others. MA1–MA6 using different LS procedures exhibit

Table 3 Comparative results of the algorithms on single-target instances

Scheme	Instance 1					Instance 2					Instance 3				
	Max	Min	Mean	Median	Std	Max	Min	Mean	Median	Std	Max	Min	Mean	Median	Std
B&B	0.409	–	–	–	–	0.362	–	–	–	–	0.383	–	–	–	–
GA	0.409	0.385	†0.406	**0.409**	0.004	0.362	0.353	†0.360	**0.362**	0.003	0.383	0.375	†0.380	**0.383**	0.002
SPSO	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.383	**0.383**	**0.383**	0.001
QPIO	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.379	†0.382	**0.383**	0.001
KNDE	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.383	**0.383**	**0.383**	0.000
MA1	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.372	†0.380	**0.383**	0.003
MA2	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.375	†0.382	**0.383**	0.002
MA3	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.383	**0.383**	**0.383**	0.000
MA4	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.383	**0.383**	**0.383**	0.000
MA5	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.383	**0.383**	**0.383**	0.000
MA6	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.383	**0.383**	**0.383**	0.000
MA-S	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.383	**0.383**	**0.383**	0.000
MA-BS	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.383	**0.383**	**0.383**	0.000
MA-B	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.383	**0.383**	**0.383**	0.000
MA-L	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.383	**0.383**	**0.383**	0.000
MA-FD	0.409	0.409	**0.409**	**0.409**	0.000	0.362	0.362	**0.362**	**0.362**	0.000	0.383	0.383	**0.383**	**0.383**	0.000

Scheme	Instance 4					Instance 5					Instance 6				
	Max	Min	Mean	Median	Std	Max	Min	Mean	Median	Std	Max	Min	Mean	Median	Std
B&B	0.325	–	–	–	–	0.289	–	–	–	–	0.242	–	–	–	–
GA	0.325	0.313	†0.320	0.320	0.003	0.285	0.263	†0.276	0.278	0.005	0.236	0.225	†0.232	0.230	0.003
SPSO	0.325	0.316	†0.321	0.322	0.001	0.289	0.270	†0.283	0.282	0.003	0.237	0.227	†0.233	0.234	0.002
QPIO	0.325	0.316	†0.323	0.322	0.002	0.285	0.268	†0.281	0.282	0.004	0.237	0.225	†0.232	0.233	0.003
KNDE	0.325	0.321	†0.323	0.324	0.001	0.289	0.273	†0.282	0.282	0.003	0.237	0.230	†0.235	0.234	0.002
MA1	0.325	0.313	†0.322	0.322	0.003	0.285	0.268	†0.279	0.278	0.006	0.237	0.225	†0.232	0.233	0.003
MA2	0.325	0.316	†0.322	0.321	0.003	0.285	0.270	†0.280	0.282	0.004	0.237	0.227	†0.233	0.233	0.002
MA3	0.325	0.316	0.323	0.322	0.002	0.285	0.275	†0.283	0.282	0.003	0.236	0.221	†0.231	0.232	0.003
MA4	0.325	0.316	†0.322	0.322	0.002	0.285	0.273	†0.281	0.280	0.004	0.237	0.230	0.236	0.236	0.002
MA5	0.325	0.313	†0.320	0.320	0.003	0.285	0.268	†0.278	0.278	0.005	0.237	0.227	0.234	0.236	0.002
MA6	0.325	0.313	†0.321	0.320	0.003	0.285	0.270	†0.280	0.282	0.004	0.237	0.227	0.235	0.236	0.002
MA-S	0.325	0.320	0.324	**0.325**	0.002	0.289	0.275	†0.283	0.284	0.003	0.242	0.233	0.237	0.237	0.002
MA-BS	0.325	0.321	0.324	**0.325**	0.001	0.289	0.275	†0.284	0.284	0.004	0.242	0.233	0.238	**0.238**	0.002
MA-B	0.325	0.321	0.324	**0.325**	0.001	0.289	0.282	†0.286	0.285	0.002	0.242	0.236	**0.239**	**0.238**	0.001
MA-L	0.325	0.324	**0.325**	**0.325**	0.000	0.289	0.285	0.288	**0.289**	0.002	0.242	0.236	0.238	**0.238**	0.001
MA-FD	0.325	0.323	0.325	**0.325**	0.000	0.289	0.285	**0.289**	**0.289**	0.002	0.242	0.236	**0.239**	**0.238**	0.001

Scheme	Instance 7					Instance 8					Instance 9				
	Max	Min	Mean	Median	Std	Max	Min	Mean	Median	Std	Max	Min	Mean	Median	Std
B&B	0.217	–	–	–	–	0.205	–	–	–	–	0.133	–	–	–	–
GA	0.203	0.176	†0.192	0.190	0.006	0.185	0.159	†0.172	0.174	0.006	0.118	0.093	†0.107	0.109	0.008
SPSO	0.206	0.184	†0.195	0.199	0.006	0.188	0.161	†0.177	0.181	0.006	0.121	0.100	†0.111	0.111	0.007
QPIO	0.206	0.179	†0.195	0.195	0.008	0.188	0.161	†0.176	0.176	0.009	0.118	0.098	†0.108	0.109	0.009
KNDE	0.210	0.188	†0.198	0.198	0.005	0.190	0.161	†0.179	0.181	0.005	0.124	0.105	†0.115	0.115	0.006
MA1	0.210	0.190	†0.199	0.198	0.005	0.188	0.169	†0.179	0.181	0.005	0.121	0.098	†0.109	0.111	0.007
MA2	0.210	0.190	†0.202	0.203	0.005	0.192	0.169	†0.181	0.181	0.007	0.121	0.098	†0.112	0.115	0.006
MA3	0.203	0.183	†0.195	0.193	0.004	0.185	0.169	†0.177	0.179	0.004	0.121	0.103	†0.113	0.115	0.005
MA4	0.206	0.193	†0.201	0.199	0.004	0.188	0.169	†0.184	0.188	0.005	0.124	0.102	†0.112	0.115	0.007
MA5	0.206	0.188	†0.197	0.197	0.005	0.188	0.161	†0.178	0.181	0.007	0.121	0.098	†0.112	0.115	0.007
MA6	0.203	0.176	†0.192	0.193	0.007	0.185	0.159	†0.175	0.178	0.008	0.118	0.098	†0.109	0.109	0.005
MA-S	0.213	0.197	0.206	0.206	0.005	0.196	0.177	0.188	0.188	0.005	0.124	0.111	†0.117	0.118	0.004
MA-BS	0.213	0.199	0.208	0.206	0.004	0.198	0.180	0.190	0.188	0.005	0.124	0.105	†0.116	0.116	0.006
MA-B	0.217	0.203	**0.210**	**0.211**	0.004	0.198	0.177	0.188	**0.190**	0.007	0.124	0.114	0.118	**0.121**	0.003
MA-L	0.217	0.199	0.210	0.208	0.005	0.202	0.180	0.188	0.188	0.006	0.124	0.111	0.118	0.118	0.004
MA-FD	0.217	0.197	0.209	0.208	0.004	0.198	0.180	**0.190**	**0.190**	0.004	0.124	0.118	**0.119**	**0.121**	0.002

Scheme	Instance 10					Instance 11					Instance 12				
	Max	Min	Mean	Median	Std	Max	Min	Mean	Median	Std	Max	Min	Mean	Median	Std
B&B	0.106	–	–	–	–	0.058	–	–	–	–	0.030	–	–	–	–
GA	0.088	0.065	†0.076	0.077	0.007	0.063	0.033	†0.049	0.049	0.008	0.052	0.029	†0.040	0.038	0.006
SPSO	0.095	0.068	†0.081	0.080	0.008	0.068	0.037	†0.056	0.058	0.008	0.059	0.035	†0.047	0.049	0.008
QPIO	0.088	0.065	†0.077	0.079	0.011	0.063	0.035	†0.050	0.052	0.012	0.056	0.030	†0.044	0.045	0.009
KNDE	0.095	0.069	†0.083	0.084	0.009	0.068	0.039	†0.055	0.056	0.009	0.059	0.038	†0.050	0.051	0.006
MA1	0.095	0.072	†0.084	0.086	0.005	0.065	0.033	†0.050	0.056	0.009	0.059	0.032	†0.046	0.045	0.009
MA2	0.095	0.072	†0.084	0.086	0.005	0.068	0.036	†0.053	0.051	0.010	0.059	0.038	†0.051	0.056	0.005
MA3	0.088	0.069	†0.084	0.085	0.004	0.068	0.039	†0.055	0.056	0.007	0.058	0.029	†0.043	0.040	0.009
MA4	0.095	0.072	0.087	0.088	0.005	0.068	0.046	†0.058	0.061	0.007	0.059	0.038	†0.049	0.051	0.008
MA5	0.093	0.077	†0.083	0.086	0.005	0.068	0.037	†0.054	0.056	0.008	0.058	0.029	†0.045	0.047	0.010
MA6	0.088	0.068	†0.079	0.081	0.005	0.063	0.035	†0.051	0.047	0.008	0.056	0.030	†0.043	0.042	0.008
MA-S	0.095	0.077	0.087	**0.089**	0.006	0.072	0.056	0.064	0.064	0.004	0.063	0.053	0.058	0.058	0.003
MA-BS	0.095	0.075	†0.086	0.086	0.007	0.068	0.058	†0.062	0.061	0.003	0.059	0.051	0.056	0.056	0.003
MA-B	0.095	0.079	0.088	**0.089**	0.006	0.072	0.058	0.063	0.063	0.003	0.066	0.052	**0.060**	**0.060**	0.004
MA-L	0.096	0.077	**0.089**	0.088	0.006	0.076	0.058	**0.067**	**0.066**	0.004	0.063	0.051	0.058	0.058	0.002
MA-FD	0.095	0.077	0.088	0.088	0.005	0.076	0.058	0.064	**0.066**	0.004	0.066	0.051	0.059	0.058	0.003

Table 4 Comparative results of the algorithms on multi-target instances

Scheme	Instance 13					Instance 14					Instance 15				
	Max	Min	Mean	Median	Std	Max	Min	Mean	Median	Std	Max	Min	Mean	Median	Std
B&B	0.309	–	–	–	–	0.232	–	–	–	–	0.049	–	–	–	–
GA	0.308	0.277	†0.292	0.289	0.007	0.220	0.190	†0.207	0.210	0.008	0.063	0.042	†0.052	0.055	0.007
SPSO	0.308	0.286	†0.296	0.297	0.007	0.222	0.197	†0.211	0.212	0.008	0.070	0.049	†0.063	0.063	0.006
QPIO	0.309	0.280	†0.295	0.293	0.010	0.220	0.192	†0.208	0.210	0.007	0.067	0.046	†0.058	0.060	0.006
KNDE	0.309	0.286	†0.297	0.298	0.006	0.224	0.197	†0.212	0.212	0.006	0.070	0.049	†0.061	0.063	0.007
MA1	0.308	0.280	†0.295	0.297	0.009	0.222	0.203	†0.214	0.216	0.005	0.070	0.049	†0.060	0.060	0.007
MA2	0.308	0.280	†0.294	0.297	0.008	0.222	0.203	†0.214	0.218	0.006	0.070	0.049	†0.061	0.062	0.007
MA3	0.308	0.286	†0.298	0.303	0.007	0.222	0.200	†0.212	0.214	0.007	0.067	0.046	†0.057	0.058	0.005
MA4	0.308	0.285	†0.300	0.306	0.006	0.222	0.201	†0.214	0.219	0.006	0.067	0.046	†0.060	0.063	0.005
MA5	0.306	0.285	†0.295	0.295	0.007	0.220	0.197	†0.212	0.216	0.007	0.063	0.039	†0.057	0.058	0.007
MA6	0.306	0.286	†0.297	0.295	0.006	0.220	0.197	†0.210	0.212	0.008	0.063	0.045	†0.055	0.057	0.005
MA-S	0.309	0.295	0.303	0.306	0.003	0.226	0.210	†0.219	0.220	0.005	0.073	0.056	†0.064	0.065	0.005
MA-BS	0.309	0.295	0.303	**0.306**	0.005	0.226	0.207	†0.218	0.220	0.006	0.073	0.056	†0.066	0.065	0.007
MA-B	0.309	0.303	**0.305**	**0.306**	0.002	0.232	0.207	0.221	0.222	0.005	0.073	0.058	0.066	0.067	0.004
MA-L	0.309	0.297	0.304	**0.306**	0.003	0.232	0.212	**0.223**	**0.224**	0.004	0.077	0.056	0.067	**0.069**	0.005
MA-FD	0.309	0.298	0.305	**0.306**	0.003	0.232	0.210	0.222	**0.224**	0.004	0.077	0.056	**0.068**	**0.069**	0.005

Scheme	Instance 16					Instance 17					Instance 18				
	Max	Min	Mean	Median	Std	Max	Min	Mean	Median	Std	Max	Min	Mean	Median	Std
B&B	0.206	–	–	–	–	0.017	–	–	–	–	0.006	–	–	–	–
GA	0.200	0.179	†0.190	0.192	0.006	0.046	0.032	†0.038	0.037	0.003	0.014	0.011	†0.012	0.012	0.001
SPSO	0.208	0.183	†0.198	0.200	0.008	0.051	0.037	†0.044	0.045	0.003	0.015	0.012	†0.014	0.014	0.001
QPIO	0.205	0.179	†0.192	0.192	0.008	0.047	0.034	†0.040	0.040	0.005	0.014	0.012	†0.013	0.013	0.001
KNDE	0.208	0.183	†0.198	0.199	0.005	0.051	0.037	†0.045	0.046	0.004	0.016	0.012	†0.014	0.014	0.001
MA1	0.203	0.181	†0.194	0.197	0.007	0.046	0.034	†0.040	0.040	0.003	0.014	0.011	†0.013	0.013	0.001
MA2	0.206	0.181	†0.195	0.199	0.007	0.047	0.034	†0.042	0.043	0.004	0.014	0.012	†0.013	0.013	0.001
MA3	0.208	0.186	†0.199	0.203	0.006	0.046	0.034	†0.041	0.043	0.003	0.015	0.012	†0.013	0.013	0.001
MA4	0.208	0.183	†0.197	0.200	0.006	0.047	0.036	†0.042	0.042	0.003	0.016	0.012	†0.014	0.014	0.001
MA5	0.210	0.197	†0.203	0.203	0.003	0.048	0.037	†0.044	0.046	0.004	0.015	0.012	†0.014	0.014	0.001
MA6	0.210	0.192	†0.203	0.206	0.005	0.046	0.033	†0.040	0.040	0.004	0.015	0.012	†0.014	0.013	0.001
MA-S	0.213	0.197	†0.206	0.206	0.005	0.056	0.043	0.050	0.049	0.003	0.017	0.012	0.015	0.015	0.001
MA-BS	0.213	0.197	†0.206	0.205	0.005	0.053	0.045	0.049	0.050	0.002	0.018	0.012	0.016	0.015	0.002
MA-B	0.219	0.203	0.210	0.211	0.004	0.056	0.049	0.053	0.053	0.004	0.021	0.012	**0.017**	**0.017**	0.002
MA-L	0.215	0.206	0.211	0.211	0.002	0.060	0.049	**0.055**	**0.054**	0.003	0.019	0.012	0.015	0.016	0.002
MA-FD	0.219	0.203	**0.211**	**0.213**	0.003	0.060	0.049	0.054	**0.054**	0.002	0.019	0.012	0.016	0.016	0.001

different performances. For example, the median values of MA1 are better than those of MA3 on 7 instances (instances 6–8, 10, 12, 14, and 15), worse than those of MA3 on 5 instances (instances 5, 9, 13, 16, and 17), and the same as those of MA3 on the remaining instances. In general, among the six non-adaptive memetic algorithms in the third group, the algorithms using consecutive local search operations generally perform better than those using multiple local search operations; i.e., MA2 generally performs better than MA1, and MA4 generally performs better than MA3. Moreover, memetic algorithms conducting local search on search sequences are typically better than those conducting local search on search modes (e.g., MA2 outperforms MA5 and MA6) on single-target instances, but the opposite is true on multi-target instances, because the local search on UAV search modes plays a more important role when the proportion of the search-mode sub-solution space to the whole solution space becomes larger. MA5 and MA6 have their own advantages in different situations. In general, none of the six memetic algorithms with fixed memes can be superior to others on all the instances.

By hybridization and adaptive selection among different memes, the last five adaptive memetic algorithms exhibit remarkably better performance than the pure metaheuristics and metaheuristics combined with a single LS procedure. Except the first three instances, no algorithm in the second

and third groups can obtain the best mean or median value on any of the remaining instances. In terms of statistical tests, MA-FD shows significant performance improvement over GA, SPSO, QPIO, KNDE, MA1, and MA2 on all the 15 instances, over MA3 on 14 instances (except instance 4), over MA4 on 13 instances (except instances 6 and 10), and over MA5 and MA6 on 14 instances (except instance 6). Such significant performance advantages demonstrate that different LS procedures have different contributions in different stages of problem-solving and, therefore, the adaptive selection among them can effectively improve the search ability of the algorithms.

Among the five adaptive memetic algorithms, the overall performance of MA-S is relatively poor due to its simple meme selection strategy. The performances of the remaining four algorithms vary from instance to instance. On instances 4–18, MA-S obtains the best median values on 2 instances, MA-BS obtains the best median values on 3 instances, MA-B obtains the best median values on 9 instances and best mean values on 5 instances, MA-L obtains the best median values on 8 instances and best mean values on 5 instances, and our MA-FD obtains the best median values on 11 instances and best mean values on 6 instances. In terms of statistical tests, MA-FD shows significant performance improvement over MA-S on 5 instances, and there is no significant difference between the two algorithms on the remaining 10 instances; MA-FD shows significant performance improvement over MA-BS on 7 instances, and there is no significant difference on the remaining 8 instances; MA-FD shows significant performance improvement over MA-B on instance 5, and there is no significant difference on the remaining 14 instances; there is no significant difference between MA-L and MA-FD on all 15 instances. In summary, the experimental results demonstrate that the proposed MA-FD exhibits the best performance among all comparative algorithms on the test instances.

It should also be noted that, the B&B algorithm obtains the exact optimal solutions on instances 1–10, 13, and 14, but fails to stop within 12 h on the other 6 instances due to the combinatorial explosion of the solution spaces. That is why on the large instances 11, 12, and 15–18, the benchmark solutions are even much worse than the solutions of the heuristic algorithms obtained within 10 min.

6 Conclusions

In this paper, we have presented a UAV search planning problem where the search area has been divided into a set of subareas with different prior target existence probabilities and, for each subarea, the UAV can search at different altitudes with different precisions. The problem was to determine the search sequence of the subareas and the search mode on each subarea, such that the total weighted probability of finding the target was maximized. For this problem, we have proposed an adaptive memetic algorithm that combines a GA with six local search procedures and dynamically determines which procedure to apply based on its suitability in a previous period of problem-solving. Computational experiments showed that the proposed method outperforms a number of state-of-the-art works on a wide set of test instances.

Our current problem formulation has assumed that we have complete knowledge about the probability distribution of target existence. In some cases, some information may be uncertain or unavailable. Therefore, our future work will incorporate incomplete and uncertain knowledge of the prior probabilities into search planning, and extend our method to effectively handle such incompleteness and uncertainty using fuzzy inference and machine learning (Li et al., 2021).

Contributors

Libin HONG and Yujun ZHENG designed the research. Yue WANG processed the data. Yue WANG and Yichen DU developed the algorithm. Yichen DU drafted the paper. Xin CHEN and Yujun ZHENG revised and finalized the paper.

Compliance with ethics guidelines

Libin HONG, Yue WANG, Yichen DU, Xin CHEN, and Yujun ZHENG declare that they have no conflict of interest.

References

Al-Helal H, Sprinkle J, 2010. UAV search: maximizing target acquisition. Proc 17th IEEE Int Conf and Workshops on Engineering of Computer Based Systems, p.9-18. https://doi.org/10.1109/ECBS.2010.9

Altshuler Y, Yanovsky V, Wagner IA, et al., 2008. Efficient cooperative search of smart targets using UAV swarms. *Robotica*, 26(4):551-557. https://doi.org/10.1017/S0263574708004141

Bertuccelli LF, How JP, 2006. UAV search for dynamic targets with uncertain motion models. Proc 45th IEEE

Conf on Decision and Control, p.5941-5946. https://doi.org/10.1109/CDC.2006.377010

Bourgault F, Furukawa T, Durrant-Whyte HF, 2006. Optimal search for a lost target in a Bayesian world. In: Yuta S, Asama H, Prassler E, et al. (Eds.), Field and Service Robotics: Recent Advances in Research and Applications. Springer, Berlin, Heidelberg, p.209-222. https://doi.org/10.1007/10991459_21

Burcin Ozsoydan F, Sağir M, 2021. Iterated greedy algorithms enhanced by hyper-heuristic based learning for hybrid flexible flowshop scheduling problem with sequence dependent setup times: a case study at a manufacturing plant. *Comput Oper Res*, 125:105044. https://doi.org/10.1016/j.cor.2020.105044

Cabassi F, Locatelli M, 2016. Computational investigation of simple memetic approaches for continuous global optimization. *Comput Oper Res*, 72:50-70. https://doi.org/10.1016/j.cor.2016.01.015

Chak CK, Feng G, 1995. Accelerated genetic algorithms: combined with local search techniques for fast and accurate global search. Proc IEEE Int Conf on Evolutionary Computation, p.378-383. https://doi.org/10.1109/ICEC.1995.489177

Chandler P, Rasmussen S, Pachter M, 2000. UAV cooperative path planning. Proc AIAA Guidance, Navigation, and Control Conf and Exhibit, p.1-11. https://doi.org/10.2514/6.2000-4370

Dong ZN, Chen ZJ, Zhou R, et al., 2011. A hybrid approach of virtual force and A* search algorithm for UAV path re-planning. Proc 6th IEEE Conf on Industrial Electronics and Applications, p.1140-1145. https://doi.org/10.1109/ICIEA.2011.5975758

Du YC, Zhang MX, Ling HF, et al., 2019. Evolutionary planning of multi-UAV search for missing tourists. *IEEE Access*, 7:73480-73492. https://doi.org/10.1109/ACCESS.2019.2920623

Duan HB, Li P, 2014. Bio-inspired Computation in Unmanned Aerial Vehicles. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-41196-0

Eiben AE, Smit SK, 2011. Parameter tuning for configuring and analyzing evolutionary algorithms. *Swarm Evol Comput*, 1(1):19-31. https://doi.org/10.1016/j.swevo.2011.02.001

Goodrich MA, Morse BS, Gerhardt D, et al., 2008. Supporting wilderness search and rescue using a camera-equipped mini UAV. *J Field Robot*, 25(1-2):89-110. https://doi.org/10.1002/rob.20226

Hansen SR, McLain TW, Goodrich MA, 2007. Probabilistic searching using a small unmanned aerial vehicle. Proc AIAA Infotech@Aerospace Conf and Exhibit, p.1-16. https://doi.org/10.2514/6.2007-2740

Hu CH, Xia Y, Zhang JG, 2019. Adaptive operator quantum-behaved pigeon-inspired optimization algorithm with application to UAV path planning. *Algorithms*, 12(1):3. https://doi.org/10.3390/a12010003

Hutter F, Hoos HH, Leyton-Brown K, et al., 2009. ParamILS: an automatic algorithm configuration framework. *J Artif Intell Res*, 36:267-306. https://doi.org/10.1613/jair.2861

Jin Y, Liao Y, Minai AA, et al., 2006. Balancing search and target response in cooperative unmanned aerial vehicle (UAV) teams. *IEEE Trans Syst Man Cybern Part B Cybern*, 36(3):571-587. https://doi.org/10.1109/TSMCB.2005.861881

Kamrani F, Ayani R, 2009. UAV Path Planning in Search Operations. INTECH Open Access Publisher, Rijeka, Croatia.

Krasnogor N, Smith J, 2005. A tutorial for competent memetic algorithms: model, taxonomy, and design issues. *IEEE Trans Evol Comput*, 9(5):474-488. https://doi.org/10.1109/TEVC.2005.850260

Lai XJ, Hao JK, 2016. A tabu search based memetic algorithm for the max-mean dispersion problem. *Comput Oper Res*, 72:118-127. https://doi.org/10.1016/j.cor.2016.02.016

Li W, Yang BW, Song GH, et al., 2021. Dynamic value iteration networks for the planning of rapidly changing UAV swarms. *Front Inform Technol Electron Eng*, 22(5):687-696. https://doi.org/10.1631/FITEE.1900712

Lin J, 2019. Backtracking search based hyper-heuristic for the flexible job-shop scheduling problem with fuzzy processing time. *Eng Appl Artif Intell*, 77:186-196. https://doi.org/10.1016/j.engappai.2018.10.008

Lin L, Goodrich MA, 2009. UAV intelligent path planning for wilderness search and rescue. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.709-714. https://doi.org/10.1109/IROS.2009.5354455

López-Ibáñez M, Dubois-Lacoste J, Pérez Cáceres L, et al., 2016. The irace package: iterated racing for automatic algorithm configuration. *Oper Res Persp*, 3:43-58. https://doi.org/10.1016/j.orp.2016.09.002

López-Ortiz A, Maftuleac D, 2015. Optimal strategies for search and rescue operations with robot swarms. https://arxiv.org/abs/1410.1077v1

Moscato P, Cotta C, 2003. A gentle introduction to memetic algorithms. In: Glover F, Kochenberger GA (Eds.), Handbook of Metaheuristics. Springer, Boston, USA, p.105-144. https://doi.org/10.1007/0-306-48056-5_5

Murphy RR, Tadokoro S, Nardi D, et al., 2008. Search and rescue robotics. In: Siciliano B, Khatib O (Eds.), Springer Handbook of Robotics. Springer, Berlin, Heidelberg, p.1151-1173. https://doi.org/10.1007/978-3-540-30301-5_51

Nikolos IK, Zografos ES, Brintaki AN, 2007. UAV path planning using evolutionary algorithms. In: Chahl JS, Jain LC, Mizutani A, et al. (Eds.), Innovations in Intelligent Machines 1. Springer, Berlin, Heidelberg, p.77-111. https://doi.org/10.1007/978-3-540-72696-8_4

Ong YS, Lim MH, Zhu N, et al., 2006. Classification of adaptive memetic algorithms: a comparative study. *IEEE Trans Syst Man Cybern Part B Cybern*, 36(1):141-152. https://doi.org/10.1109/TSMCB.2005.856143

Özcan E, Drake JH, Altıntaş C, et al., 2016. A self-adaptive Multimeme Memetic Algorithm co-evolving utility scores to control genetic operators and their parameter settings. *Appl Soft Comput*, 49:81-93. https://doi.org/10.1016/j.asoc.2016.07.032

Phung MD, Ha QP, 2021. Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization. *Appl Soft Comput*, 107:107376. https://doi.org/10.1016/j.asoc.2021.107376

Qi YT, Hou ZT, Li H, et al., 2015. A decomposition based memetic algorithm for multi-objective vehicle routing problem with time windows. *Comput Oper Res*, 62:61-77. https://doi.org/10.1016/j.cor.2015.04.009

Qin AK, Huang VL, Suganthan PN, 2009. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Trans Evol Comput*, 13(2):398-417. https://doi.org/10.1109/TEVC.2008.927706

Ragi S, Chong EKP, 2013. UAV path planning in a dynamic environment via partially observable Markov decision process. *IEEE Trans Aerosp Electron Syst*, 49(4):2397-2412. https://doi.org/10.1109/TAES.2013.6621824

Ruan WY, Duan HB, 2020. Multi-UAV obstacle avoidance control via multi-objective social learning pigeon-inspired optimization. *Front Inform Technol Electron Eng*, 21(5):740-748. https://doi.org/10.1631/FITEE.2000066

Ryan A, Hedrick JK, 2005. A mode-switching path planner for UAV-assisted search and rescue. Proc 44th IEEE Conf on Decision and Control, p.1471-1476. https://doi.org/10.1109/CDC.2005.1582366

Sheng WG, Chen SY, Sheng MM, et al., 2016. Adaptive multi-subpopulation competition and multiniche crowding-based memetic algorithm for automatic data clustering. *IEEE Trans Evol Comput*, 20(6):838-858. https://doi.org/10.1109/TEVC.2016.2524555

Syswerda G, 1991. Schedule optimization using genetic algorithms. In: Davis LD (Ed.), Handbook of Genetic Algorithms. van Nostrand Reinhold, New York, USA.

Tisdale J, Kim Z, Hedrick JK, 2009. Autonomous UAV path planning and estimation. *IEEE Robot Autom Mag*, 16(2):35-42. https://doi.org/10.1109/MRA.2009.932529

Tomlin JA, 1971. Technical note—an improved branch-and-bound method for integer programming. *Oper Res*, 19(4):1070-1075. https://doi.org/10.1287/opre.19.4.1070

Turky A, Sabar NR, Dunstall S, et al., 2020. Hyper-heuristic local search for combinatorial optimisation problems. *Knowl-Based Syst*, 205:106264. https://doi.org/10.1016/j.knosys.2020.106264

van Willigen WH, Schut MC, Eiben AE, et al., 2011. Online adaptation of path formation in UAV search-and-identify missions. Proc Int Conf on Adaptive and Natural Computing Algorithms, p.186-195. https://doi.org/10.1007/978-3-642-20267-4_20

Volgenant T, Jonker R, 1982. A branch and bound algorithm for the symmetric traveling salesman problem based on the 1-tree relaxation. *Eur J Oper Res*, 9(1):83-89. https://doi.org/10.1016/0377-2217(82)90015-7

Waharte S, Trigoni N, 2010. Supporting search and rescue operations with UAVs. Proc Int Conf on Emerging Security Technologies, p.142-147. https://doi.org/10.1109/EST.2010.31

Waharte S, Symington A, Trigoni N, 2010. Probabilistic search with agile UAVs. Proc IEEE Int Conf on Robotics and Automation, p.2840-2845. https://doi.org/10.1109/ROBOT.2010.5509962

Wang XP, Tang LX, 2017. A machine-learning based memetic algorithm for the multi-objective permutation flowshop scheduling problem. *Comput Oper Res*, 79:60-77. https://doi.org/10.1016/j.cor.2016.10.003

Wang Y, Zhang MX, Zheng YJ, 2017. A hyper-heuristic method for UAV search planning. Proc Int Conf on Swarm Intelligence, p.454-464 https://doi.org/10.1007/978-3-319-61833-3_48

Yu XB, Li CL, Yen GG, 2021. A knee-guided differential evolution algorithm for unmanned aerial vehicle path planning in disaster management. *Appl Soft Comput*, 98:106857. https://doi.org/10.1016/j.asoc.2020.106857

Zhang B, Duan HB, 2014. Predator-prey pigeon-inspired optimization for UAV three-dimensional path planning. Proc Int Conf on Swarm Intelligence, p.96-105. https://doi.org/10.1007/978-3-319-11897-0_12

Zhang H, Xin B, Dou LH, et al., 2020. A review of cooperative path planning of an unmanned aerial vehicle group. *Front Inform Technol Electron Eng*, 21(12):1671-1694. https://doi.org/10.1631/FITEE.2000228

Zhang YZ, Mei Y, Tang K, et al., 2017. Memetic algorithm with route decomposing for periodic capacitated arc routing problem. *Appl Soft Comput*, 52:1130-1142. https://doi.org/10.1016/j.asoc.2016.09.017

Zheng YJ, 2015. Water wave optimization: a new nature-inspired metaheuristic. *Comput Oper Res*, 55:1-11. https://doi.org/10.1016/j.cor.2014.10.008

Zheng YJ, Ling HF, Xue JY, 2014. Ecogeography-based optimization: enhancing biogeography-based optimization with ecogeographic barriers and differentiations. *Comput Oper Res*, 50:115-127. https://doi.org/10.1016/j.cor.2014.04.013

Zheng YJ, Zhang MX, Ling HF, et al., 2015. Emergency railway transportation planning using a hyper-heuristic approach. *IEEE Trans Intell Transp Syst*, 16(1):321-329. https://doi.org/10.1109/TITS.2014.2331239

Zheng YJ, Du YC, Ling HF, et al., 2020. Evolutionary collaborative human-UAV search for escaped criminals. *IEEE Trans Evol Comput*, 24(2):217-231. https://doi.org/10.1109/TEVC.2019.2925175

Zheng YJ, Du YC, Su ZL, et al., 2021. Evolutionary human-UAV cooperation for transmission network restoration. *IEEE Trans Ind Inform*, 17(3):1648-1657. https://doi.org/10.1109/TII.2020.3003903

Zhou R, Feng Y, Di B, et al., 2020. Multi-UAV cooperative target tracking with bounded noise for connectivity preservation. *Front Inform Technol Electron Eng*, 21(10):1494-1503. https://doi.org/10.1631/FITEE.1900617