Path Planning of UAVs Under Dynamic Environment based on a Hierarchical Recursive Multiagent Genetic Algorithm

Qiansheng Yang
School of Artificial Intelligence
Xidian University
Xi'an, China
erosyqs@163.com

Jing Liu
School of Artificial Intelligence
Xidian University
Xi'an, China
neouma@163.com

Liqiang Li
School of Artificial Intelligence
Xidian University
Xi'an, China
xidianmrlee@gmail.com

***Abstract*—Path planning is a key technology to realize the automatic navigation of unmanned aerial vehicles (UAV), which has great significance both in theory and practical application. Evolutionary algorithms (EAs) are a type of nature-inspired computational methodologies for addressing complex real-world problems that cannot be solved well by mathematical or traditional modeling. However, the huge calculation of each iteration in the EAs greatly reduces the efficiency of the algorithm. In this paper, to accelerate the search speed of EAs and consider the dynamics of the environment, we propose a hierarchical recursive multi-agent genetic algorithm that can perform path planning in real time, termed as HR-MAGA. We used a strategy of layer-by-layer optimization on 3D maps with different resolution in the optimization process. The local search ability of the algorithm is improved by competition and self-learning process. In addition, the hierarchical recursive optimization process can greatly reduce the amount of computation and effectively deal with the dynamic characteristics of the environment. The experimental results show that HR-MAGA not only has strong global optimization ability, but more importantly, is able to generate collision free paths in real time after considering the physical limitations of the UAV and the dynamics of the environment.**

***Index Terms*—Path Planning of UAVs; Multiagent Genetic Algorithm; Hierarchical Recursive Optimization; Dynamic Environment**

I. Introduction

Unmanned aerial vehicles (UAVs) have demonstrated their irreplaceable role in military and civil applications. The path planning problem is a key element of the UAV autonomous control module [2]. It is a complex optimization problem which allows the UAV to autonomously compute the optimal path from a start point to an end point. The difference with the ordinary commercial route is that the UAV's trajectory must constantly change depending on the particular terrain and conditions prevailing at the time.

In the past research, there are many methods to solve the traditional 2D path planning problem, such as Dijsktra's algorithm [21], A* algorithm [22], D* algorithm [23], and rapidly exploring random tree RRT algorithm [24], [25]. These methods usually model the environment as a network topology connected graph model, and the result of path planning is the shortest path on the connected graph. Compared with 2D path planning, the environment targeted by 3D path planning is more complicated and the traditional 2D environment modeling methods are not applicable. It is very difficult to deal with the dynamic factors in the 3D environment using the traditional 2D path planning methods. In the optimization of 3D path planning, the best path is not always associated with the shortest path simply, more importantly, the factors like the effects of terrain, the requirements of mission and the physical limitations of aircrafts, should be also taken into consideration. A series of methods have been proposed to deal with this complicated optimization problem [2]–[4].

In the past few decades, many new algorithms from various fields have been designed to optimize this problem. Based on the cognitive behavior, Cai *et al.* in [18] proposed an optimization algorithm and designed a tri-level programing (TLP) model to generate a smooth path. Zammit *et al.* in [19] Compared A* with RRT algorithms for the UAV path planning problem. Lin *et al.* in [20] developed and demonstrated a method for UAV collision avoidance based on the closed-loop rapidly-exploring random tree algorithm. As examples of artificial intelligence algorithms, Roberge *et al.* in [6] compared the parallel genetic algorithm (GA) with the particle swarm optimization (PSO) algorithm in the real-time 3D path planning problem. To improve the efficiency of decision-making, Kumar P *et al.* in [7] proposed meta-heuristic optimization schemes for reactive path planning of UAVs while designing a UAV path planning problem using multi-verse optimizer (MVO). Duan *et al.* in [8] developed Hyperid PSO and GA for Multi-UAVs formation reconfiguration. Chen *et al.* in [9] proposed an improved PSO algorithm to optimize the global best solution during the evolutionary process of particles. In this method, the competition strategy was introduced into the standard PSO to improve the convergence speed and the search ability of particles. Phung *et al.* in [10] converted the UAV path planning model into a discrete TSP problem, and the discrete particle swarm optimization (DPSO) algorithm was then proposed to solve the TSP. Cheng *et al.* in [11] also used the GA, but during the evolutionary process, they added the artificial immune system to maintain superior population diversity. Volkan in [12] proposed a vibrational GA to improve the exploration, which can better avoid local minima when searching for the optimal path. Based on a discrete UAV dynamic model, Chen *et al.* in [13] reformed this problem into an optimal control problem. By establishing dynamic and static models, Yin *et al.* in [15] proposed the use of offline and online method to improve the efficiency of UAV path planning in the dynamic urban environment.

In summary, those methods represented by EAs are able to solve the UAV trajectory planning problem quite effectively in the existing researches [6]–[9], [11]–[14]. However, EAs have a high time complexity and lack effective strategies for timely re-planning to cope with the complex and volatile environmental conditions. Besides, using 2D traditional methods need to sample environment as a series of nodes, or a tree, or in other forms [16]–[19], [21]–[25], which is not applicable in the 3D space. Moreover, some of these models consider the static threat only or just give a single path [24], [25]. Therefore, it is hard to improve the efficiency of these algorithms under the complex environmental conditions.

In [26], Zhong *et al.* proposed an effective evolutionary algorithm, namely Multi-Agent Genetic Algorithm (MAGA), which has been proved to be an excellent algorithm for solving complex optimization problems and has been widely applied to various fields [27]–[29]. In this article, the method that MAGA is directly applied to the path planning problem is called P-MAGA. To deal with the uncertainties timely in the environment and improve the efficiency of the algorithm, we use a hierarchical recursive optimization strategy, which improve convergence speed by reducing the number of path points in each optimization. Moreover, uncertainties in the environment can be effectively addressed in the deep recursive optimization process. Based on this, we propose an optimization algorithm termed as HR-MAGA. In HR-MAGA, during the process of evolution, agents can sense the environment and interact with its neighbors and reduce its own loss by utilizing the corresponding operators, which instantly find a good solution. In the hierarchical recursive process, the algorithm is able to optimize the local path to get a more refined path. By comparing P-MAGA and HR-MAGA, the experiment verifies the effectiveness of the hierarchical recursive strategy, which can be also effectively applied with other EAs. In summary, HR-MAGA has higher efficiency compared with P-MAGA and GA [6] , and can be effectively applied in a complex and volatile environment.

The rest of this paper is organized as follows. In Section II, we briefly introduce the model of UAVs path planning. In Section III, the details of the proposed HR-MAGA are introduced. The experimental results and discussions are presented in Section IV. Finally, conclusions and future work are given in Section V.

II. Path Planning Model

A. Environmental model

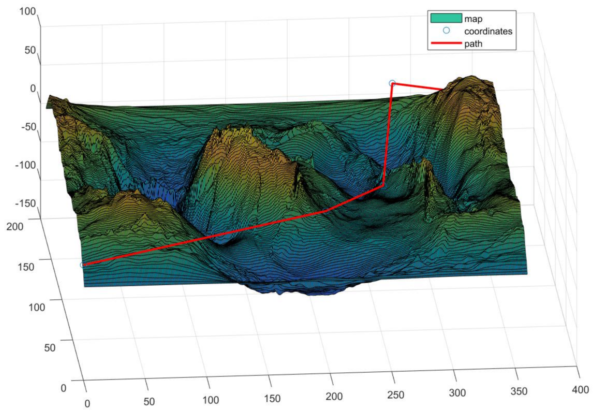
The first step of three-dimensional path planning is to discretize the world space into a representation which is meaningful to the path planning algorithm. With the development of simulation technology, there are many approaches to build a three dimensional environment in this area [5], [18], [30]. In this paper, we approximate the terrain using a 2D matrix, where each element of the matrix represents the elevation of terrain. Then, we can add c cylindrical hazardous areas to the environment, where coordinates (x_i, y_i) and radii d_i can be saved to a matrix as shown in Eq. (1).

$$
sampling\_map = \begin{bmatrix} -10 & -10 & -10 & -10 & -10 & -11 & -11 & -10 & -9 \\ 32 & 43 & 36 & -9 & -36 & -40 & -16 & 12 & 24 \\ 44 & 65 & 46 & -3 & 7 & 3 & 10 & 26 & 21 \\ 42 & 32 & 8 & -1 & 56 & 25 & 10 & 29 & 12 \\ 31 & -13 & -57 & 57 & 84 & 30 & 8 & 30 & 9 \\ 42 & 9 & -78 & 74 & 55 & 30 & -20 & 18 & 4 \\ 56 & 28 & -3 & 29 & 22 & 10 & -22 & -15 & 40 \\ 65 & 30 & -24 & 18 & 27 & 28 & 3 & -21 & 67 \\ 69 & 29 & 8 & 12 & 23 & 19 & 4 & 5 & 66 \end{bmatrix}.
$$

(a)

$$
agent = \begin{bmatrix} 0 & 0 & 0 \\ 220 & 40 & 30 \\ 250 & 80 & 45 \\ 280 & 170 & 40 \\ 360 & 180 & 15 \end{bmatrix}
$$

(b)



(c)

Fig. 1: Path planning example: (a) 2D matrix representation of the map after 40 times sampling. (b) agent matrix. (c) agent representation on the map.

$$
danger\_areas = \begin{bmatrix} x_1 & y_1 & d_1 \\ \vdots & \ddots & \vdots \\ x_c & y_c & d_c \end{bmatrix} \tag{1}
$$

To represent m uncertainty regions, we use a rectangle starting at (xr_i, yr_i) with width w_i and height h_i, and save them as a matrix shown in Eq.(2).

$$
uncertain\_areas = \begin{bmatrix} xr_1 & yr_1 & w_1 & h_1 \\ xr_2 & yr_2 & w_2 & h_2 \\ \vdots & \vdots & \vdots & \vdots \\ xr_m & yr_m & w_m & h_m \end{bmatrix} \tag{2}
$$

In the planning space, we discretize the trajectories into a series of waypoints connecting by straight lines. Then, the path planning task is transformed into function optimization problems by optimizing the coordinate point sequence. When optimizing this problem, each route represents an agent whose encoding is the coordinates of the waypoint as shown in Eq. (3).

$$agent = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix} \tag{3}$$

The 2D matrix in Fig. 1(a) shows the terrain map after sampling 40 times, which is available from geoshow [1]. Each element in the matrix represents the terrain elevation at that location. Fig. 1(c) shows a simple example of planning a route on a map, and the corresponding agent matrix is shown in Fig. 1(b). As shown in Fig. 1, the first line element of agent is [0,0,0], indicating that the starting point of the route is set as the origin. The coordinates of the next point [220,40,30] represent the point to be passed at the next moment, and the coordinates at the end point are [360,180,15].

B. Loss function

It is known that searching for the best path is often associated with searching for the shortest path. In the situation of UAV path planning, the optimal path is more complex and includes many different characteristics, like the complexity of environment and the physical limitations of aircrafts. To take into account these desired characteristics, we design a comprehensive loss function consisting of multiple optimization goals for path planning to minimize. A path that meets various conditions has lower loss at a high probability. Our loss function is defined as follows:

$$\begin{aligned} &F_{loss}(agent, map, danger\_areas) = w_1 \times L_{path} + \\ &w_2 \times L_{altitude} + w_3 \times L_{danger} \end{aligned} \tag{4}$$

where F_{loss} consists of three parts, namely the path's length loss L_{path}, the loss of average altitude of path $L_{attitude}$, and L_{danger} penalizing the path going through danger areas. w_1, w_2, and w_3 denote the weight on each part.

Considering the maneuver performance of UAVs and making the path as smooth as possible, we design a vector Θ to evaluate the corner of each path. Eq.(5) is designed to calculate the track's angle θ_k.

$$\theta_k = \frac{\lambda_{k+1}\lambda_k}{|\lambda_{k+1}||\lambda_k|} \tag{5}$$

where λ_k means $(x_{k+1} - x_k, y_{k+1} - y_k, z_{k+1} - z_k), k = 1, 2, ..., n - 1$. Then, the optimization model of the path planning problem can be defined as follows:

$$\begin{cases} \min \quad F_{loss}(agent, map, danger\_areas) \\ s.t. \quad \theta_k > 0 \quad k = 1, 2, \cdots, n-1 \end{cases} \tag{6}$$

The length of the planning path is a key factor for most path planning missions. Obviously, the shorter path can save more fuel and more time. In our cost function, the term associated with the length of a path is defined as Eq. (7):

$$L_{path} = \sum_{k=1}^{n-1} ||(x_{k+1}, y_{k+1}, z_{k+1}) - (x_k, y_k, z_k)||_2 \tag{7}$$

where (x_k, y_k, z_k) means the k-th waypoint of the $agent$.

The altitude of the path is also an important characteristic. On one hand, the flight height is limited by UAV's physical constraints. On the other hand, the low altitude flight can also reduce the probability of being captured by enemies, and this means that the safety of tracks is increased. The term associated with the altitude of path is defined as Eq. (8):

$$L_{altitude}(agent, map) = \sum_{k=1}^{n-1} mean(map \cap agent_k) \tag{8}$$

where $mean$ represents the averaging function and $map \cap agent_k$ represents the altitude of the position which is the k-th path of the agent projected on the map. $L_{altitude}$ depends mainly on the altitude of the area where subsections of the trajectory go through.

In the real life, there are many no-fly zones or dangerous areas. Therefore, the term associated with the violation of the danger areas is defined as Eq. (9):

$$L_{danger} = \sum_{i=1}^{c}\sum_{k=1}^{n-1} length(danger\_areas_i \cap agent_k) \tag{9}$$

where c is the total number of danger areas, $danger\_areas_i \cap agent_k$ represents the intersection between the i-th $danger\_area$ and the $agent_k$ on the map , and $length$ is a function calculating the length of the intersection. The result of L_{danger} is the total length of subsections of trajectory which goes through danger areas.

III. HR-MAGA

In this section, we describe HR-MAGA for optimizing the UAV path planning problem in details.

A. Definition of agent-lattice

In HR-MAGA, an agent is defined in Eq. (3). All agents live in a lattice-like environment L called agent-lattice with the size of $L_{size} \times L_{size}$, where L_{size} is an integer, and the agent has the ability to perceive and respond to the environment it lives [23]. Each agent is fixed on a lattice-point, and can only exchange information with its neighbors. We define the agent located at (i, j) as $L_{ij}, i, j = 1, 2, ..., L_{size}$, then the set of neighbors of L_{ij} is denoted as $Neighbor_{ij}$,

$$Neighbor_{i,j} = \{L_{i',j}, L_{i,j'}, L_{i'',j}, L_{i,j''}\} \tag{10}$$

where $i' = \begin{cases} i-1 & i \neq 1 \\ L_{size} & i = 1 \end{cases}$, $j' = \begin{cases} j-1 & j \neq 1 \\ L_{size} & j = 1 \end{cases}$, $i'' = \begin{cases} i+1 & i \neq L_{size} \\ 1 & i = L_{size} \end{cases}$, $j'' = \begin{cases} j+1 & j \neq L_{size} \\ 1 & j = L_{size} \end{cases}$.

Fig. 2 shows the model of agent-lattice, where each circle represents an agent. The data in a circle denotes the agent's location, and the agent can communicate with each other when they are connected directly.

In traditional EAs, the individuals used to generate offspring are selected from the population based on their fitness. However, this kind of behavior based on fitness for global

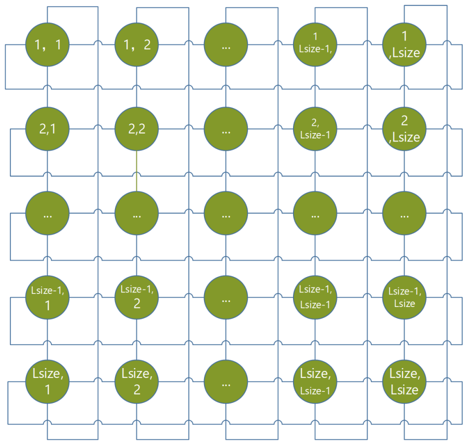


Fig. 2: Model of the agent-lattice.

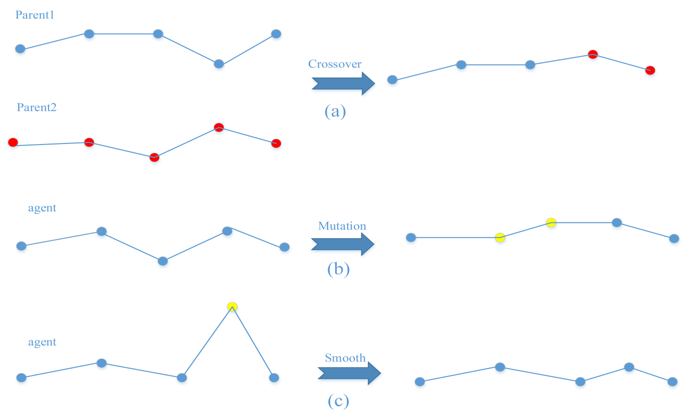


Fig. 3: Genetic behaviors used in HR-MAGA: (a) single-point crossover, (b)mutation, (c) smooth.

selection and global comparison does not exist in the real world. In fact, the process of real natural selection to generate children always happens in a local environment, and each individual can only interact with those around it [26]. For the purpose of achieving a better simulation of this natural law, the agent need to compete with its neighbors so that they can gain more resources. Based on this, a series of behaviors are designed for agents to realize their purposes. At the agent level, the competitive behavior and the self-learning behavior are designed as individual behaviors, which are directly related to the fitness of agents. During the execution of individual behavior, the algorithm further operates on the encoding of the agent. Therefore, the crossover behavior, the mutation behavior and the smooth behavior are designed as genetic behaviors.

B. Competitive and self-learning behavior

Let L_{ij} be an agent in L. If L_{ij} satisfies Eq. (11), it is a winner; otherwise, it is a loser.

$$F_{loss}(L_{i,j}) < F_{loss}(Min_{i,j}) \tag{11}$$

where Min_{ij} is the neighbor of L_{ij} with the lowest loss. If L_{ij} is a winner, it can live in the lattice. If L_{ij} is a loser, it will be occupied by a new agent called New_{ij}. This new agent is generated using the genetic behavior based on parent1 L_{ij} and parent2 Min_{ij} , and the details are shown in Fig. 3.

First, the new agent is created from the selected parents using the single point crossover as shown in Fig. 3(a). In the process of crossover, the algorithm needs to randomly select the intersection and connects Parent1's coordinates before the intersection with Parent2's coordinates after this position. Then, the child agent is subject to the mutation as shown in Fig. 3(b). The algorithm randomly selects a pair of genes and replaces it with the adjacent coordinate point during the mutation process. Finally, in the smooth behavior, the algorithm calculates the various corners of the new path according to Eq. (5). The process of smoothing the route by changing the coordinates of points whose θ_k less than 0 is shown in Fig. 3(c).

Each agent has local information of this problem. To obtain a better solution, the agent changes independently for lower loss. In this paper, all agents live in the lattice. In the process of competitive behavior, the position with the smallest loss value is marked. Those agents whose location are marked in competitive behavior can execute self-learning behavior with a probability of 0.5. In this process, eight agents are generated separately by mutating from each selected agent. These nine agents make up a new local lattice, labeled as L_{new}. Then, the competitive behavior, the crossover behavior, the mutation behavior and the smooth behavior are implemented on L_{new}. In addition, elite retention strategy is used in this process. At last, the initial agent is replaced by the best performing agent in L_{new}.

C. Summary of P-MAGA

Hence, based on the above description, the detail process of P-MAGA is summarized in Algorithm 1, where $trajectories$ represent 3 agents with the smallest loss value among L, and $U(0, 1)$ returns a uniformly distributed random real number ranged from 0 to 1. In P-MAGA, first, the algorithm performs the competitive behavior and marks each winning agent. For a failed agent, it forms a pair of parental agent with the winning agent, and the parent agents generate superior offspring agent through crossover behavior, mutation behavior and smoothing behavior. Then, on those agents winning in the competitive behavior, the self-learning behavior is conducted. In the self-learning behavior, agents mutate to produce multiple variants and form a new lattice. In the new lattice, agents perform crossover, mutation and smoothing to complete the self-learning process. The new lattice outputs the optimal agent to replace the agent in the current position after several iterations.

D. Hierarchical recursive strategy

During the process of evolution, there is no way to optimize the uncertain factors of the future. When using EAs to optimize this problem, the resulting trajectory is approximate due to the influence of agent size. To reduce the complexity of the

Algorithm 1 P-MAGA

```
Input: 3D environment, hazardous areas and uncertainties: $maps$;
    start point: $St$; end point: $Ed$
Parameters: Lattice size: $L_{size} \times L_{size}$; maximum number of
    iterations: $G$; number of waypoints optimized for each recursion:
    $num$; Crossover rate: $P_c$; Mutation rate: $P_m$.
Output: $trajectories$.
 1: $L^1 \leftarrow$ Initial-population($maps$, $St$, $Ed$, $G$, $L_{size} \times L_{size}$ )
 2: $Fitness(L^1, L_{size})$ according to Eq. (4)
 3: for $k = 1$ to $G$ do
 4:     for $L_{ij}^k$ in $L^k, i, j = 1, 2, ..., L_{size}$ do
 5:         $Competition(L_{ij}^k)$
 6:         if $L_{ij}^k$ loses in $Competition(L_{ij}^k)$ then
 7:             $L_{ij}^k \leftarrow Evolutionary\text{-}operate(L_{ij}^k, P_c, P_m)$
 8:             $L_{ij}^k \leftarrow Smooth(L_{ij}^k)$
 9:         else
10:             if $U(0,1) < 0.5$ then
11:                 $L_{ij}^k \leftarrow Self\text{-}learning(L_{ij}^k, P_c, P_m)$
12:             end if
13:         end if
14:     end for
15:     $Fitness(L^k, L_{size})$ according to Eq. (4)
16: end for
17: three agents with the minimum loss value in $L^G$ are recorded as
    $trajectories(1:3, 1:num)$
18: return $trajectories(1:3, 1:num)$
```

search space, we use a recursive approach. When the algorithm recurs to a deeper level, it calls a finer map to optimize the sub-paths of the previous hierarchical result in turn. If there is an uncertain area on the planned point of the upper layer, the point can be reinitialized as the coordinate closest to the point and outside the uncertain area in the deeper planning. In the process of optimization, the algorithm first plans on a local high-precision map, and then returns the partial results to the upper layer for integration. The great benefit of using hierarchical recursive optimization is that the length of the agent for each level of optimized sub-path length can be set to a small value. Therefore, this strategy can fundamentally solve the dimensionality disaster caused by the increase of optimization parameters in EAs. As shown in Fig. 4, we adopt the hierarchical recursive strategy to optimize the segments obtained from the previous layer optimization. Fig. 4(a) shows the optimization process of several points on a low-precision map using HR-MAGA, using straight lines to form the result of path planning. Fig. 4(b) shows the algorithm's more refined recursive optimization process for the results of the previous layer. The process of integrating the optimization results of sub-paths is shown in Fig. 4(c). This recursive strategy creates new flight routes by adding new flight points to the sub-flight segments and makes the algorithm more adaptable in complex environments.

E. Summary of HR-MAGA

Algorithm 2 summarizes the details of HR-MAGA. The algorithm first plans on a low-precision map and then gradually recursively optimizes the path. To deal with uncertain factors, the algorithm can update the planned path on a higher-

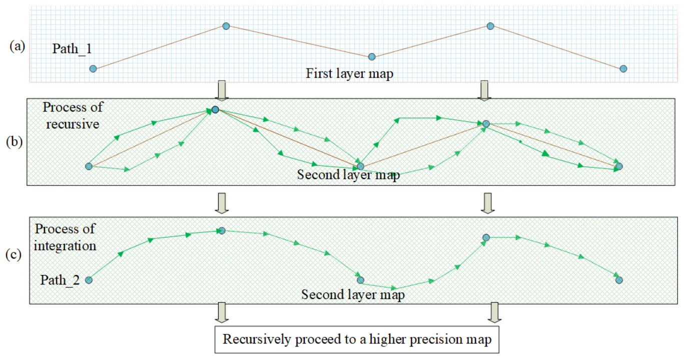


Fig. 4: Process of hierarchical recursion: (a) result of path planning on the first layer. (b) example of recursion. (c) the process of integrating paths and selecting the optimal path.

resolution map through a layered recursive strategy. To ensure the global optimality of the path, the algorithm integrates the multiple optimized path segments. Fig. 4(c) shows the integration process. The algorithm first connects all the sub-paths, and then calculates the loss values of all the global paths obtained by the permutation and combination. The final output is the $newpath$ with three smallest loss value agent.

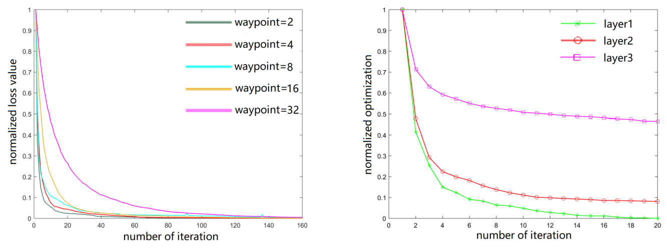
Algorithm 2 HR-MAGA

```
Input: 3D environment, hazardous areas and uncertainties:$maps$; start point:
    $St$; end point :$Ed$; value of recursive depth :$v$
Parameters: number of waypoints optimized for each recursion: $num$;
Output: 3 agents with the smallest loss value in current layer: $newpath$.
 1: $v = v - 1$
 2: $trajectories(1:3, 1:num) \leftarrow P\text{-}MAGA\ (maps, St, Ed)$
 3: if $v == 0$ then
 4:     return $trajectories(1:3, 1:num)$
 5: end if
 6: update $maps$ to improve resolution
 7: for $i = 1$ to 3 do
 8:     for $j = 1$ to $num$ do
 9:         obtain the coordinates of the starting point $St_{ij}$ and the end-
            ing point $Ed_{ij}$ of the $j$-th segment of the $i$-th agent path in
            $trajectories$
10:         if $St_{ij}$ or $Ed_{ij}$ are in the uncertain area of new $maps$ then
11:             reinitialize $St_{ij}$ or $Ed_{ij}$ as the coordinate closest to the original
                point and outside the uncertain area on $maps$
12:         end if
13:         use $HR\text{-}MAGA(maps, St_{ij}, Ed_{ij}, v)$ to recursively optimize
            the $j$-th segment of the $i$-th agent to get 3 paths with the smallest
            loss value $path(j, 1:3)$
14:     end for
15:     $newpath(i) \leftarrow Integration(path(1:num, 1:3))$
16: end for
17: return $newpath(1:3)$
```

IV. Experimental Result

In the experiment, we model a series of matrix from the digital elevation maps which are available from the Geo Base [31]. In the simulation process, the algorithm can construct virtual maps of different precision layers by sampling digital elevation maps at different frequencies. And then, we factitiously add the dangerous areas to the map, and the hazardous areas are marked with yellow. This section first shows the results of the path planning of HR-MAGA at various levels in the recursive process, showing the outstanding global planning ability of the algorithm. We analyze the relationship between the size of the agent and the convergence speed of the algorithm, and illustrate the necessity of using hierarchical recursion. Then, we compare the experimental results and prove the efficient search ability of HR-MAGA.

A. Parameter analysis



(a) (b)

Fig. 5: Analysis of HR-MAGA parameters: (a) The relationship between the number of waypoint and iterations of algorithm convergence. (b) Evaluation of global path optimization by each layer

To better evaluate the relationship between the length of the agent and the convergence of HR-MAGA, we normalize the average iteration loss value of the 40 experimental results according to Eq. (12)

$$loss_i^* = \frac{mean\_loss_i - \min(mean\_loss)}{\max(mean\_loss) - \min(mean\_loss)} \tag{12}$$

where $mean\_loss_i$ represents the mean value of the i-th iteration loss function obtained by performing 40 repeated experiments.

According to Eq. (13), we evaluate the optimization of the path by HR-MAGA at each layer.

$$loss\_layer_i^* = 1 + \frac{loss\_layer_i - \max(loss\_layer)}{\max(loss\_layer1) - \min(loss\_layer1)} \tag{13}$$

where $loss\_layer1$ represents the loss value in the initial hierarchical iterative optimization process, and $loss\_layer$ represents the loss value in the deeper optimization process. $loss\_layer_i$ means the smallest loss value in the population at the i-th iteration.

Fig. 5(a) shows the relationship between the number of points optimized at each layer and the iterative evolution of HR-MAGA. When the number of optimized waypoint exceeds 8, the convergence speed of the algorithm is greatly affected. In EAs, the calculation scale of each iteration is usually large. Therefore, reducing the number of waypoint optimized each time through hierarchical recursive methods can greatly improve the real-time efficiency of HR-MAGA. According to Eq. (13), Fig. 5(b) analyzes the respective optimization quantities at three levels. The results show that as the recursion level deepens, the part of global optimization is gradually decreases. And compared with the second layer, the normalization optimization of the third layer decreases sharply.

TABLE I: Parameter setting in the experiment

Parameter	Definition	Values
G	Number of generations	200
Pop	Population size	256
L_{size}	Size of the Lattice-like environment L	7×7
P_c	Crossover rate	0.5
P_m	Mutation rate	0.3
v	value of recursive depth	3
num	number of waypoints optimized for each recursion	3
$length$	Individual encoding length	27
$weight$	$w_1 : w_2 : w_3$	1:10:15

B. Parameter setting

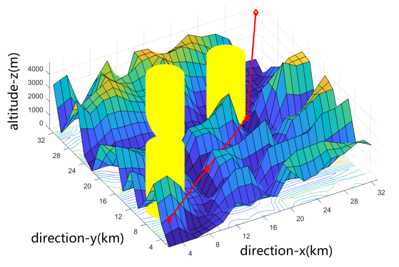
Our experiments are conducted in Matlab programing environment, and a series of related parameters used in our implementation are listed in Table I. To increase the convergence speed, the number of optimized waypoint per level num is set to 3 refer to the results in Fig. 5(a), and the recursive hierarchical $layer$ in HR-MAGA is set to 3. Therefore, the agent coding length is equal to 27, which is calculated by the $layer$-th power of num. In the comparative methods, the length of the individual is also set to 27. Setting the same optimization point in the path guarantees the consistency of the calculation scale required in each encoding and decoding process. Referring to MAGA [26], the parameter L_{size} is set to 7, and to ensure the optimal performance of GA [6], the parameter Pop is set to 256, which is consistent with the original GA instead of 49.

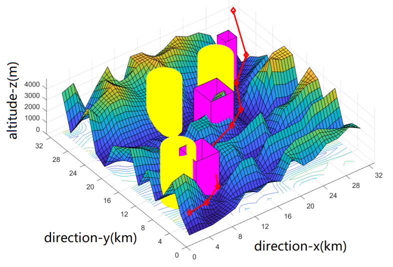
C. Analysis of HR-MAGA algorithm

Fig. 6 shows the experimental optimization results of HR-MAGA at different layers. The yellow part of the figure represents the dangerous area, and the purple rectangular area represents the deep uncertainty. The experimental results show that the planning path of HR-MAGA in the hierarchical planning process becomes more and more refined. At the initial optimization level, the path optimization results are more focused on the global optimality. Deeper optimization is more focused on the refinement of the path. In dealing with the uncertainty factor, HR-MAGA can effectively process the burst situation in real time by correcting the local path, as shown in Fig. 6(b)(c).

D. Comparison against other methods

To evaluate the performance of HR-MAGA, we conduct experiments on four different 3D map environments. The comparison of GA, P-MAGA and HR-MAGA is shown in Fig. 7. In different map environments, all methods show good results. These approaches have strong similarity, as we find the results of the two methods have partial overlap, which is largely due to the fact that the two methods are ideologically consistent. It is hard to distinguish the advantages and disadvantages of the


(a)


(b)

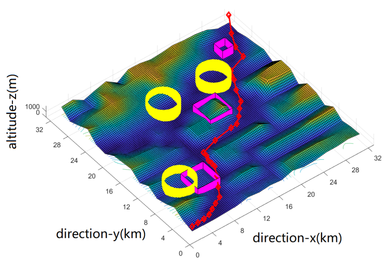
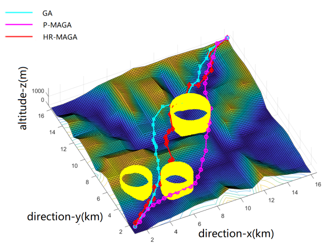
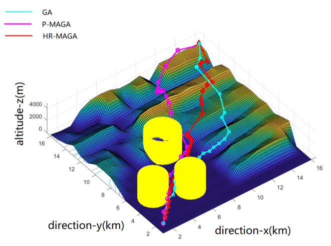
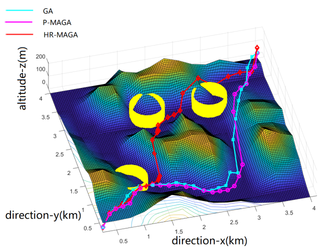

(c)

Fig. 6: Examples of optimization results of HR-MAGA at various layers: (a) Optimized path in the first layer. (b) Optimized path in the second layer. (c) Optimized path in the third layer.

two algorithms directly from the optimized path on the map. So we analyze the convergence curves of various methods through 40 independent experiments. The experimental results are normalized according to Eq. (12).


(a)


(b)


(c)

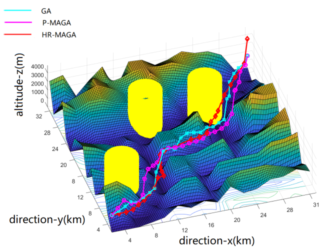
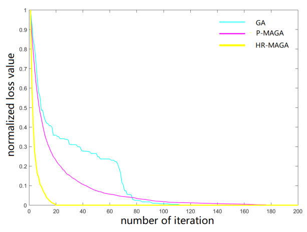

(d)

Fig. 7: 3D visualization of obtained paths by GA, P-MAGA and HR-MAGA: (a) Kinbasket lake, BC, Canada. (b) St-John, NL, Canada. (c) virtual map. (d) Banff, AB, Canada.

In Fig. 8(a), HR-MAGA has an absolute advantage in convergence speed, mainly due to the use of hierarchical recursion to make the number of sub-path points optimized for each time small. Although the use of hierarchical recursion requires optimization of multi-segment sub-paths, the complexity of each sub-path in calculating the loss function value in coding and decoding is proportional to the number of optimized waypoints. Therefore, when the number of waypoints optimized by the last planned path is the same, the optimization of the multi-segment sub-path by HR-MAGA is the same as the total amount of calculation of P-MAGA and GA in each iterative


(a)

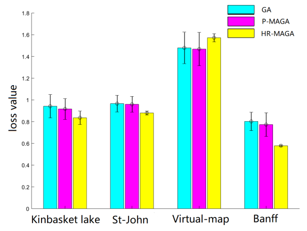

(b)

Fig. 8: Performance comparison of GA, P-MAGA and HR-MAGA: (a) Comparison of convergence curves of GA, P-MAGA and HR-MAGA. (b) Comparison of optimal loss function values of GA, P-MAGA and HR-MAGA.

process. Comparing the convergence curves of red and purple, we can obviously see that using P-MAGA can achieve the convergence result quickly. Of course, the main reason is that we use a self-learning process in P-MAGA, which speeds up the evolution of outstanding individuals. By comparing P-MAGA and HR-MAGA, the experiment illustrates the effectiveness of the hierarchical recursive strategy. Similarly, the hierarchical recursive strategy can be also used for EAs.

In Fig. 8(b), the variance of HR-MAGA is smaller than P-MAGA and GA in 40 independent experiments. This aspect means the stability of path optimization results. In four different 3D maps, HR-MAGA algorithm shows a good performance in three environments. A poor performance appears in one of the environments. The reason is that after the path is recursively segmented, there is some loss in the global optimality of the path.

V. Conclusions

In this paper, we present a path planning method for UAVs in the complex real 3D map which considers both dynamic properties of the environment and the physical limitations of the UAV. First, we design a set of optimization objectives, constraints, and experiments in the 3D dynamic environment. Through the multi-agent competition method and the self-learning strategy, the P-MAGA we use has higher search efficiency than GA. Based on this, the real-time efficiency of the algorithm has been greatly improved by hierarchical recursion strategy. The experimental results show that the optimization at a deeper level makes the path more refined. And compared with P-MAGA and GA, the HR-MAGA has great advantages in search efficiency, and the solution quality is also competitive.

VI. Acknowledgment

This work was supported in part by the Key Project of Science and Technology Innovation 2030 supported by the Ministry of Science and Technology of China under Grant 2018AAA0101302 and in part by the General Program of National Natural Science Foundation of China (NSFC) under Grant 61773300.

References

[1] Geoshow: The Digital Elevation Data example. [Online]. Available: https://ww2.mathworks.cn/help/map/ref/ geoshow.html. 2018 (accessed Jan 2018).

[2] H. Chen, X. M. Wang and Y. Li, "A survey of autonomous control for UAV," In: *Proceedings of the 2009 IEEE International Conference on Artificial Intelligence and Computational Intelligence*, vol. 2, pp. 267-271, 2009.

[3] L. Yang, J. T. Qi and D. Song, "Survey of robot 3D path planning algorithms ," *Journal of Control Science & Engineering*, 2016.

[4] Yijing. Z, Zheng. Z and Yang. L, "Survey on computational-intelligence-based UAV path planning[J]," *Knowledge-Based Systems*, S0950705118302636-, 2018.

[5] H. Shen, J. Chen., H. Li and Z. Zhou, "Research on real-time flight path planning of UAV based on grey prediction ," In:*Proceedings of 2016 IEEE International Symposium on Computational Intelligence and Design (ISCID)*, Vol. 1, pp. 62-67, 2016.

[6] V. Roberge, M. Tarbouchi and G. Labonte, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV Path Planning ," *IEEE Trans. on Industrial Informatics*, vol. 9, no. 1, pp. 132-141, 2013.

[7] Kumar P, Garg S, Singh A, *et al*, "MVO-based two-dimensional path planning scheme for providing quality of service in UAV environment[J]," *IEEE Internet of Things Journal*, 2018.

[8] H. B. Duan, Q. N. Luo, G. Ma and Y. H, "Hybrid particle swarm optimization and genetic algorithm for multi-UAVs formation reconfiguration," *IEEE Computational Intelligence Magazine*, vol. 8, no. 3, pp. 16-27, 2013.

[9] H. Chen and J. Y. Fei, "UAV path planning based on particle swarm optimization with global best path competition," *International Journal of Pattern Recognition and Artificial Intelligence*, 1859008, 2017.

[10] Phung. M. D, Quach. C. H, Dinh. T. H, *et al*, 'Enhanced discrete particle swarm optimization path planning for UAV vision-based surface inspection[J]," *Automation in Construction*, 2017.

[11] Z. Cheng, Y. Sun and Y. Liu, "Path planning based on immune genetic algorithm for UAV," In: *Proceedings of the 2011 IEEE International Conference on Electric Information & Control Engineering*, pp. 590-593, 2011.

[12] P. Y. Volkan, "A new vibrational genetic algorithm enhanced with a voronoi diagram for path planning of autonomous UAV," *Aerospace Science and Technology*, vol. 16, no. 1, pp. 47-55, Jan, 2012.

[13] Y. Chen, G. Luo, Y. Mei,*et al*, "UAV path planning using artificial potential field method updated by optimal control theory[J]," *International Journal of Systems Science*, vol. 47, no. 1: pp. 1407-1420, 2016.

[14] Y, Liu and Y. Zhao, "A virtual-waypoint based artificial potential field method for UAV path planning ," In: *Proceedings of the 2016 IEEE Guidance, Navigation and Control Conference (CGNCC)*, Chinese, pp. 949-953, 2016.

[15] Yin. C, Xiao. Z, Cao. X, *et al*, "Offline and Online Search: UAV multi-objective path planning under dynamic urban environment[J]," *IEEE Internet of Things Journal*, 2017.

[16] Y. Lin and S. Saripalli, "Path planning using 3D dubins curve for unmanned aerial vehicles," In:*Proceedings of the 2014 IEEE International Conference on Unmanned Aircraft Systems*, pp. 296-304, 2014.

[17] Y. B. Chen and J. Q. Yue, "Modified central force optimization (MCFO) algorithm for 3D UAV path planning ," *Neurocomputing*, vol. 171, pp. 878-888, 2016.

[18] Y. Cai, H. Zhao, M. Li,"3D real-time path planning based on cognitive behavior optimization algorithm for UAV with TLP model[J]," *Cluster Computing*, pp.1-10, 2018.

[19] C. Zammit, E. J. Van Kampen,"Comparison between A* and RRT algorithms for UAV Path Planning[C]," In:*Proceedings of the 2018 AIAA Guidance, Navigation, and Control Conference*,1846, 2018.

[20] Lin. Y, Saripalli. S, "Sampling-Based path planning for UAV collision avoidance[J]," *IEEE Transactions on Intelligent Transportation Systems*, 2017.

[21] I. A. Musliman, A. Rahman and V. Coors, "Implementing 3D network analysis in 3D-GIS[J]," *International archives of ISPRS*, part B, vol. 37, 2008.

[22] De Filippis, Luca, Giorgio Guglieri and Fulvia Quagliotti, "Path planning strategies for UAVs in 3D environments ," *Journal of Intelligent & Robotic Systems*,vol. 65, no. 1, pp. 247-264, 2012.

[23] J. Carsten, D. Ferguson and A. Stentz, "3d field d: Improved path planning and re-planning in three dimensions[C] ," In: *Proceedings of the 2006 IEEE International Conference on Intelligent Robots and Systems*, pp. 3381-3386, 2006.

[24] S. Choudhury, S. Scherer and S. Singh, "Rrt∗-ar: sampling based alternate routes planning with applications to autonomous emergency landing of a helicopter," In:*Proceedings of the 2013 IEEE International Conference on Robotics and Automation*, pp. 3947-3952, Karlsruhe, Germany, 2013.

[25] W. G. Aguilar, S. Morales and H. Ruiz. "RRT∗ GL based optimal path planning for real-time navigation of UAVs[C]," In:*Proceedings of the Springer International Work-Conference on Artificial Neural Networks*, 585-595, Cham, 2017.

[26] W. Zhong, J. Liu and L. Jiao,"A multiagent evolutionary algorithm for combinatorial optimization problems ," *IEEE Trans. on Systems*, Man, and Cybern., Part B, vol. 34, no. 2, pp. 1128-1141, 2004.

[27] J. Liu, W. Zhong, and L. Jiao, "A multiagent evolutionary algorithm for combinatorial optimization problems," *IEEE Trans. on Systems*, Man, and Cybernetics, Part B, vol. 40, no. 1, pp. 229-240, 2010.

[28] X. Hao and J. Liu, "A multiagent evolutionary algorithm with direct and indirect combined representation for constraint satisfaction problems," *Soft Computing*, vol. 21, no. 3, pp. 781-793, 2017.

[29] Y. Zhou, J. Liu and Y. Zhang, *et al*, "A multi-objective evolutionary algorithm for multi-period dynamic emergency resource scheduling problems," *Transportation Research Part E: Logistics and Transportation Review*, vol. 99, pp. 77-95, 2017.

[30] K. N. Ioanni, N. B. Athina, "Coordinated UAV path planning using differential evolution ," In:*Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation*, vol. 70, pp. 77-111. Heidelberg, 2005.

[31] GeoBase: Canadian Digital Elevation Data.[Online]. Available: http://www.geobase.ca/geobase/en/index.html. 2018 (accessed Jan 2018).