J Intell Robot Syst (2014) 73:737–762
DOI 10.1007/s10846-013-9895-6

3D Path Planning for Multiple UAVs for Maximum Information Collection

Halit Ergezer · Kemal Leblebicioğlu

Received: 24 June 2013 / Accepted: 11 September 2013 / Published online: 4 October 2013

Abstract This paper addresses the problem of path planning for multiple UAVs. The paths are planned to maximize collected amount of information from Desired Regions (DR) while avoiding Forbidden Regions (FR) violation and reaching the destination. The approach extends prior study for multiple UAVs by considering 3D environment constraints. The path planning problem is studied as an optimization problem. The problem has been solved by a Genetic Algorithm (GA) with the proposal of novel evolutionary operators. The initial populations have been generated from a seed-path for each UAV. The seed-paths have been obtained both by utilizing the Pattern Search method and solving the multiple-Traveling Salesman Problem (mTSP). Utilizing the mTSP solves both the visiting sequences of DRs and the assignment problem of "*which DR should be visited by which UAV*". It should be emphasized that all of the paths in population in any generation of the GA have been constructed using the dynamical mathematical model of an UAV equipped with the autopilot and guidance algorithms. Simulations are realized in the MATLAB/Simulink environment. The path planning algorithm has been tested with different scenarios, and the results are presented in Section 6. Although there are previous studies in this field, this paper focuses on maximizing the collected information instead of minimizing the total mission time. Even though, a direct comparison of our results with those in the literature is not possible, it has been observed that the proposed methodology generates satisfactory and intuitively expected solutions.

Keywords Path planning · Unmanned aerial vehicles · Evolutionary computation · Optimization · Maximum information collection · Multiple-Travelling Salesman Problem (mTSP)

H. Ergezer (✉)
MiKES Microwave Inc., Ankara, Turkey
e-mail: halit.ergezer@mikes.com.tr

K. Leblebicioğlu
Electrical and Electronics Engineering Department,
Middle East Technical University, Ankara, Turkey

1 Introduction

Autonomous systems in robotics can be described as the automation of mechanical systems that have sensing, actuation, and computation capabilities. One of the fundamental needs in robotics is to have algorithms that convert high-level specifications of tasks into low-level descriptions of how to move. The terms motion planning and path planning are often used for these kinds of problems [17]. Path planning is the problem of designing the path a vehicle is supposed to follow in such a way that a certain objective is maximized and a goal is reached.

In our case, the main objective is "*maximizing the collected information about desired regions*". In [7], the algorithm is proposed for single UAV case with level-flight assumption. The first part of the algorithm for determining the visiting sequence of DRs has been modified to find the visiting sequence of DRs for each UAV. The essence of the idea is solving fixed-destination multiple-Traveling Salesman Problem (mTSP). mTSP is a generalization of the well-known traveling salesman problem (TSP) [4], where more than one salesman is allowed to be used in the solution [1]. By finding the visiting sequence of DRs for each UAV, the problem is transformed into multiple single-UAV-path-planning problems. The mTSP adaptation also solves the assignment problem of "*which DR should be visited by which UAV*".

Many methods exist for solving the basic trajectory planning problem [16]. Most of them use a basic kinematic model of an UAV [21, 29], and [14]. Although it is possible to use de-coupled equations of motion in path planning problems, we prefer using fully coupled equations of motion [29] to be more realistic and to obtain more accurate simulations. To the best our knowledge, none of the existing methods consider the topic of trajectory planning in its full generality. For instance, some methods require the workspace to be two-dimensional. Despite many external differences, the methods are based on a few different general approaches: the roadmap methodology [13], cell decomposition [2, 24–26], the potential field [15], sampling-based [18] and evolutionary methods [5, 19, 22, 29]. In [14]—one of the studies closer to ours—the process of information collection is expressed by the Signal-to-Noise-Ratio (SNR) value of a sensor, which is assumed to be on the UAV. They attempt to minimize the total mission time instead of maximizing the collected information. In contrast, in many real-time applications, the mission duration is given, and the main objective is to maximize collected information in this fixed time. In this study, we have concentrated on information maximization given the mission time.

In [10] and [11], Shannon information is used to search for a single stationary target. Their approach is based on minimizing the entropy of the target distribution at each time step. Pitre et al. [23] defines new objective function that utilizes Fisher information due to its flexibility of handling multiple targets case. But it is computationally intensive. It is possible to extend the list of possible studies, but it is necessary to emphasize that there is no benchmark problem to compare path planning algorithms in general. It is in fact difficult to define a benchmark problem because path planning is done with respect to many different criteria, the topology of the scenarios varies, and the computation times differ significantly.

The Genetic Algorithm (GA) is a search heuristic that mimics the process of natural evolution. Genetic Algorithms belong to the larger class of evolutionary algorithms. Evolutionary algorithms, which imitate natural selection and survival of the fittest, are efficient and effective ways to solve the optimization problem associated with path planning in general. The major advantage of evolutionary algorithms is that there is no need to compute the gradient of the cost or the constraint functions. These algorithms have already been used to solve different UAV path-planning problems, including optimizing the paths of UAVs flying over a given terrain [5, 9, 20–22] and searching for optimal UAV paths in military missions [29]. All of these studies formulated the problem as finding the trajectory that minimizes and fulfils a set of optimization indices and constraints. Besada-Portas et al. [5], which is one of the most recent studies among those, studies a path planning algorithm for a multi-UAV problem that can run both online [27] and offline. Similar to our study, forbidden regions are defined, though, in our case, the Regions of Interest (ROI) are more complex. The discussion of the complexity level of the ROI is given in [7]. In [5] and [29], multiple objectives are considered at the same time, as in our case. However, we are attempting to maximize the information gathered from the desired regions. In doing so, instead of using a simple kinematical model, we use a full dynamical model. Studies based on kinematical models require extra processes such as curve fitting and guidance algorithms. It has been observed in our study that the path associated with the kinematical model is different than that of the maximum information collection path. The main novelties of this is study are listed below;

- Our algorithm consists of three main steps. In the first step, the problem is reduced to the multiple *single-UAV-path-planning problems* by solving the assignment problem of "*which DR should be visited by which UAV*". In this step, the DR visiting sequence for each UAV is also determined. *PatternSearch* algorithm is utilized to find the distances between the centers of DRs. The simplified form of the problem is modelled as an mTSP (see Section 5.3.2). By solving mTSP, both problem mentioned above are solved.
- In the second step, instead of using randomly generated population for path search as in previous studies [5, 22, 28, 29], and [19], seed paths are formed, for each UAV, to satisfy the physical constraints of the problem as much as possible (see Sections 5.1 and 5.4). It provides a good starting point for path search.
- Two new mutation operators (different from those in [7]) are defined and implemented: *Ascend-to-EScape (ATES), and Change ALTitude (CALT)*. These operators also mimic the thinking process of a human path planner as the operators defined in [7].
- The main difficulty of the problem is due to the dynamic constraints of the UAVs. Otherwise, the first step (*PatternSearch* and mTSP step) of the algorithm might have been sufficient to solve the problem. Even though using a full dynamic model introduces much more complex constraints to the problem, it makes the simulations more realistic, and it guarantees that the generated path does not violate dynamic limitations of UAV. In addition, the outputs of the controllers are saturated to handle the physical constraints of UAV actuators.

Our study contributes to the UAV path planning literature considerably for the reasons below:

- Because an mTSP solver used for the simplified form of the actual problem, it is almost a guarantee that our path planning algorithm produces a global or a nearly global optimum solution.
- The topology of the scenarios in our simulation studies have been chosen intentionally to be very complicated. The degree of success of our algorithm with respect to the existing algorithms was considered qualitatively.
- Our algorithm is *intelligent* not only because very special evolutionary operators are used but also because the weights in the construction of the cost function change according to what has been obtained at a given generation (see Section 3).

2 Problem Description

Path planning can be defined as finding a route to visit a given set of points or areas under some constraints. In our case, the critical constraints are "Desired Region(s)" (DR) and "Forbidden Region(s)" (FR), as shown in Fig. 1. There is a camera at the bottom of each UAV to capture images from regions of interest. The regions in red are FR, and the blue regions are DR. Our goal is to design an algorithm that finds an optimal route from a given starting point to a given final point within a certain mission time for each UAV. The mission time may be different for each UAV. The assumptions used to define the problem are as follows:

- There is a camera at the bottom of each UAV to capture images from the region of interest.
- There is no overlap between the regions.
- The tilting angle of the camera is neglected during turning.
- The starting and final positions are neither in DR nor in FR.
- Masking effect of the terrain has been neglected.

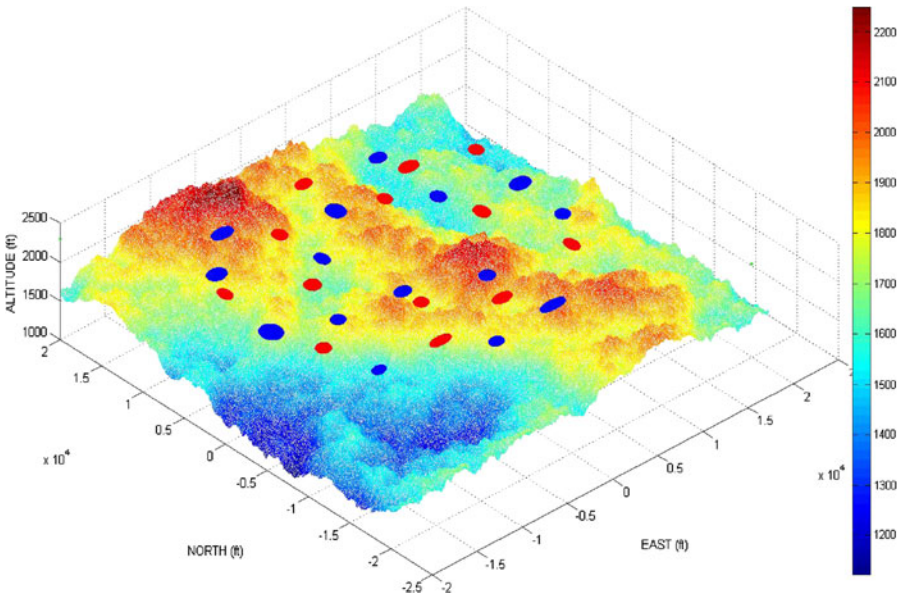


Fig. 1 Region of interest, desired regions (*blue*), forbidden regions (*red*)

- Each DR, should be visited by one UAV is enough for information collection. For a DR, visited by more than one UAV is preferable, and the total contribution to the collected amount of information is taken into account.
- Each UAV has limited energy, therefore the maximum number of DR assigned to each UAV should be stinted. This assumption raises the lower limit of the number of visited DRs. Also, the path length for each should have an upper limit.
- To use the advantages of multi-UAV, another limitation is adopted to the problem. For an UAV, to minimize the control effort, the heading angle change should be minimized. This limitation also provides less time spent in the region outside the DRs and FRs.

The construction of the objective function is described briefly in the following section.

3 Objective Function Construction

There is a camera at the bottom of the UAV to capture images from the region of interest. The camera is fixed to the UAV. The dimensions of the region of interest are 20,000 (ft) $\times$ 20,000 (ft). The region of interest has been divided into 20 ft by 20 ft cells to perform the calculations.

3.1 Definition of Collected Information and FR Violation Penalty

The captured images from desired regions are evaluated as the collected information with special regards to the resolution in a methodology similar to what has been discussed in [8].

How these captured images are evaluated as information will be described briefly. First, the floor area of a square based pyramid, of which the camera is the vertex, is calculated.

There are three different areas in Fig. 2, which we call different resolution cells associated with three different view angles (10°, 20°, and 30°),

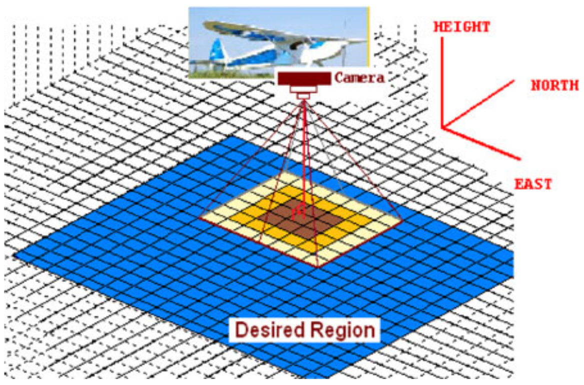


Fig. 2 Information collection, resolution cells (the inner 3-by-3 area is the highest resolution cell)

where the view angle of a square based pyramid is the maximum angle between two generatrix lines.

When the UAV flies over a DR, the intersected area between the base of the pyramid and the DR is evaluated as information. The information from the three different resolution cells is different.

$$r_1 = \cos(10°) * altitude \tag{1}$$

$$r_2 = \cos(20°) * altitude \tag{2}$$

$$r_3 = \cos(30°) * altitude \tag{3}$$

$$CI = \frac{b\left(\alpha_1 {}^*4r_1^2 + \alpha_2 {}^*4\left(r_2 - r_1\right)^2 + \alpha_3 {}^*4\left(r_3 - r_2\right)^2\right)}{\mathrm{h_{UAV}} - \mathrm{h_{terrain}}} \tag{4}$$

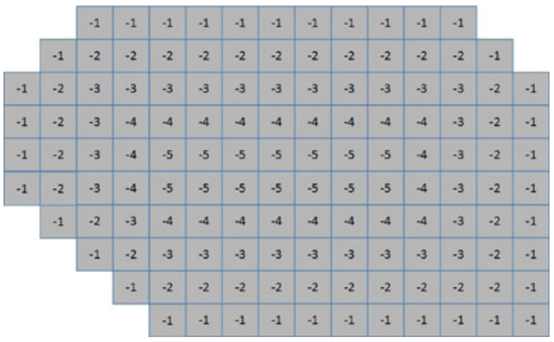
$$\alpha_1 \geq \alpha_2 \geq \alpha_3 \tag{5}$$

$$b = \begin{cases} 1 & if\, UAV\, is\, flying\, over\, DR \\ 0 & otherwise \end{cases} \tag{6}$$

where r$_1$, r$_2$, and r$_3$ are half of the base edge of the pyramid corresponding to viewing angles of 10°, 20°, and 30°, respectively. CI represents *Collected Information*, and b is the binary value that represents whether the UAV is in a DR or not. When the UAV descends the area of the base of the pyramid will be smaller, but the resolution will be increased.

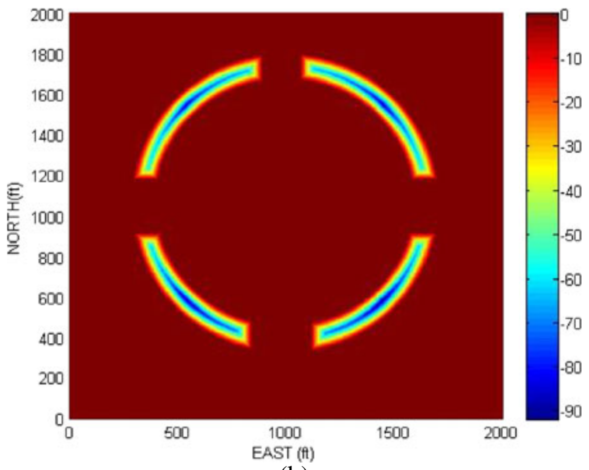
If the UAV flies over the same location of a DR more than once, the collected information from this region will be evaluated by comparing the resolution of the previously captured images. If the UAV captures higher resolution images than the previously captured images, then the former one will be ignored and latter will contribute to the objective function. Figure 2 illustrates the resolution of captured images.

The UAV should not fly over FRs. Entering an FR has a penalty. The penalty value increases from the borders of FRs towards their center by considering the 4-connectivity relation between pixels (see Fig. 3a), and also increases while the difference between the altitude of UAV and the height of the terrain decreases. In Fig. 3b, penalty function for scenario is given. In Fig. 3a, calculation of penalty values is illustrated.



		-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1		
	-1	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-1	
-1	-2	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-3	-2	-1
-1	-2	-3	-4	-4	-4	-4	-4	-4	-4	-4	-4	-3	-2	-1
-1	-2	-3	-4	-5	-5	-5	-5	-5	-5	-5	-4	-3	-2	-1
-1	-2	-3	-4	-5	-5	-5	-5	-5	-5	-5	-4	-3	-2	-1
	-1	-2	-3	-4	-4	-4	-4	-4	-4	-4	-4	-3	-2	-1
		-1	-2	-3	-3	-3	-3	-3	-3	-3	-3	-3	-2	-1
			-1	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-1
				-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1

(a)



(b)

Fig. 3 Penalty function of FR violation for scenario 3

3.2 Attractive and Repellent Forces

While the UAV flies over the region outside of DRs and FRs there should be some contribution or penalty to the objective function to direct its flight to maximize the objective function (i.e., maximize the collected information). Otherwise, the objective function value remains constant when the heading angles of the UAV are changed. To handle this problem, two kinds of forces have been suggested: attractive forces and repellent forces. Attractive forces pull the UAV to the DR, and repellent forces push the UAV away from the FR. These forces have been calculated as proportional to the areas of the regions and inversely proportional to the distance to the center of the regions [7].

During our preliminary studies, it was observed that, once the UAV entered a DR, it remained inside or might enter repeatedly into this DR instead of flying to other DRs and the final point. To eliminate this unacceptable situation, the attractive force of this particular DR is used now as a repellent force. However, this remedy was not a complete solution for the case where the UAV enters and exits a particular DR close to a corner.

In this case, it may still be preferable to re-enter the same DR when a very small area of the DR is captured. Instead of using attractive forces as repellent forces after leaving the region, attractive forces are re-calculated by updating the area that has been used in the calculation of these forces. The area from which the UAVs have collected information is extracted from the entire area of this DR. Therefore, when the collected information from one DR increases, the attractive force of this DR decreases.

3.3 The Final Point Constraint

The UAV must be at a given coordinate at the end of the simulation. The final point requirement is incorporated into the problem as follows. Let ($east_f$, $north_f$) be the final position of the UAV at the end of a simulation, and let ($east_{final}$, $north_{final}$) be the desired final point.

The distance between these two points is calculated and inserted into the objective function as a penalty. Initially, the weight of this distance penalty is taken as the mean value of the areas of the DRs.

3.4 Calculation of Dissipated Energy

The dissipated energy has been calculated for each UAV. At each time step of the simulation, the energy of the UAV, the sum of its potential and kinetic energy, is calculated. If the energy difference between successive time periods is positive, this energy is supplied by the UAV's motor, the energy difference, taking into account a certain coefficient of efficiency, be considered as the energy consumed.

Then, the problem can be formulated as;

$$\text{“max} \quad \sum_{j}^{\#UAV} \{w_1^* CI_j\} \quad -$$

$$w_2{}^* FRPenalty_j(\chi_j, V_j, h_j) - w_3{}^* Distance\_To\_FinalPosition_j - w_4{}^* Dissipated\_Energy_j$$

$$\begin{array}{ll} (\chi_j : heading\ angles\ of\ jth\ UAV) & j = 1 \ldots \#UAV \\ (V_j : velocity\ values\ of\ jth\ UAV) & j = 1 \ldots \#UAV \\ (h_j : altitude\ values\ of\ jth\ UAV) & j = 1 \ldots \#UAV \end{array}$$

$$Subject\ to:$$

$$\begin{array}{cl} 0 \le \chi_j \le 2\pi & j = 1 \ldots \#UAV \\ 30\,ft/s \le V_j \le 90\,ft/s & j = 1 \ldots \#UAV \\ \min\_altitude \le h_j \le \max\_altitude & j = 1 \ldots \#UAV \\ Given\ (east_{start}, north_{start}, altitude_{start}) & \\ Given\ (east_{final}, north_{final}, altitude_{final}) & \\ Fly\ inside\ the\ region\ of\ interest. & \\ Simulation\ time\ is\ constant. & \\ The\ dynamics\ of\ UAVs.\text{”} & \end{array}$$

where χ_i are the heading angles of the UAV to the North and w_1, w_2, w_3, and w_4 are the problem weights. The weight w_4 is increased from its nominal value if the best path at the end of a certain number of generations requires too much energy dissipation. The weight w_3 is increased from its nominal value if the best path at the end of a certain number of generations is not sufficiently close to the final point. The weight w_2 is increased from its nominal value if the best path at the end of a certain number of generations still passes through an FR. The weight w_1 is increased from its nominal value if the best path at the end of a certain number of generations does not enter all of the DRs.

4 Autopilot Design

The twelve equations of motion [12], which are non-linear, fully coupled ordinary differential equations, are used to completely and accurately model the true motion of an aircraft, which moves with six degrees of freedom along three axes. The motion caused by gravity, propulsion, and aerodynamic forces contributes to the forces and moments that act upon the body. These ordinary differential equations were constructed in [12] by using four major assumptions. First, the aircraft is rigid. Although aircrafts are truly elastic in nature, modeling the flexibility of the UAV will not contribute significantly to our research. Second, the earth is an inertial reference frame. Third, the aircraft mass properties are constant throughout the simulation. Finally, the aircraft has a plane of symmetry. The first and third assumptions allow for the treatment of the aircraft as a point mass.

A system of twelve state variables, expressed in terms of stability, or flight path components are obtained at the end. As explained in [12], the flight path components are defined by an inertial system. With these equations of motion, the UAV response to any command input is accurately modeled.

Before developing the path planning algorithms, we must design the autopilot for each UAV. The autopilot is required to have the abilities of altitude controller, speed controller, and turn controllers.

Our UAVs have four control inputs: the thruster, elevator, aileron, and rudder. The controllers are designed using the approximating linear model, since dealing with non-linear models is too complicated. Design of the autopilot for UAV is started with the linearization of the model around trim points. For different velocity values of between 30–90 ft/s trim points are determined. The linear approximations of the UAV model around these trim points are constructed. The controllers are designed using linear models and are combined using “Gain Scheduling” to construct autopilot. The autopilot has been designed using the “Linear Quadratic Regulator (LQR)” [6] method.

The first controller is the altitude-hold controller. First, only the elevator control input is used. Then, the altitude-hold controller is designed using only the thrust control input. The coupling between the control responses to the control input is optimized.

5 Path Planning of the UAV Using Evolutionary Computations

5.1 The Discretization of the Mission Time for Updating Heading Angles

To simulate the flight path of the UAV, the mission time is discretized. The total mission time $[t_0, t_f]$ has been subdivided into $N > 0$ subintervals $[t_0; t_1]; [t_1; t_2]; \ldots; [t_{N-1}; t_f]$ of an equal duration of 1 second. That is, the heading angle of the UAV is assumed to be changing in 1 second time intervals. In each subinterval, the control inputs are assumed to be constant. Thus, the UAV is assumed to fly with the same heading angle in each subinterval. During our studies we try different time intervals. It is very obvious that using the shorter time intervals gives flexibility to design a path. But, our studies have shown that in time intervals shorter than 1 s UAV has not the ability to perform the command.

5.2 The Discretization of the Mission Time for Velocity and Altitude Commands

The total mission time $[t_0, t_f]$ has been subdivided into M $(0<M<N)$ subintervals $[t_0; t_1]; [t_1; t_2]; \ldots; [t_{M-1}; t_f]$ of an equal duration of 5 s. That is, the velocity and altitude of the UAV are assumed to be changing in 5 s time intervals. In each subinterval, the control inputs are assumed to be constant. The rapid changes in velocity and altitude cause phugoid mode. The phugoid has a nearly constant angle of attack but varying pitch, caused by a repeated exchange of airspeed and altitude. We notice also the time intervals shorter than 5 s do not contribute to the creation of better paths.

5.3 Determining the Visiting Sequence of DRs

5.3.1 Calculation of Distance Matrix

In this step, the distances between the centers of DRs have been calculated. This calculation does not find the Euclidean distance between any two points directly. FRs have also been taken into account [7]. The algorithm works on 2D projection of the ROI and controls whether the sub-path intersects any FR. It is possible to run the algorithm on a 3D terrain and to take the possibility of flying over FRs into account; but it would not change the order of DRs in the following step of the algorithm.

If the path intersects any FR, then the algorithm will try to find a point to connect the centers of DRs with two line segments. The distance

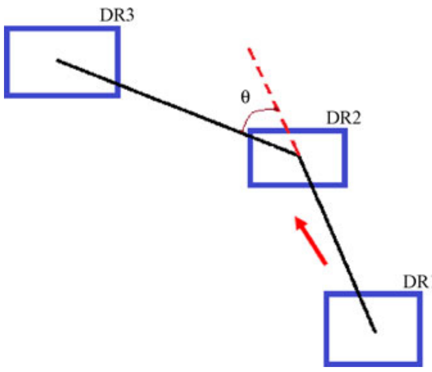


Fig. 4 The illustration of angle between lines connecting the centers of DRs

between these DRs is the sum of the lengths of the line segments. The *PatternSearch* algorithm [2] has been used to find the end points of the line segments [7]. The maximum number of line segments connecting any two DRs is (#FR + 1).

5.3.2 Visiting Sequence Determination and DR Assignment by Solving mTSP

Even with evolutionary algorithms, it is very important to initialize the optimization from a "good" initial solution. We have produced the initial population of our GA algorithm using the mTSP. The **DISTANCE** matrix which is found in the previous step, is used to represent the distances between the cities in the mTSP problem. The definition of the mTSP as an optimization problem is given below.

$$PathLength_j = 0.5^* \sum_{k}^{\#DR} \sum_{i}^{\#DR} D_{ki}{}^* X_{kj}{}^* X_{ij}$$

$$j = 1..\#UAV \tag{7}$$

$$\min_X \left\{ \sum_{j}^{\#UAV} PathLength_j + \alpha.\theta_j \right\}$$

Subject to:

$$miDRs_j \le \sum_{i}^{\#DR} X_{ij} \le maDRs_j \qquad j = 1\ldots\#UAV$$

$$\sum_{j}^{\#UAV} X_{ij} = 1 \qquad i = 1\ldots\#DR$$

$$PathLength_j \le mPL_{UAV_j} \qquad j = 1\ldots\#UAV$$

$$X_{ij} \in \{0, 1\}$$

where X_{ij} represents whether UAV_j visits DR_i it is 1 when UAV_j visits the UAV_i, and 0 otherwise. miDRs and maDRs represent the lower and upper bound of the visited number of DRs by each UAV, respectively. mPL_{UAVj} represents the upper bound of the path length for the UAV_j. θ_i is the angle between two lines connecting to DR centers, see Fig. 4. The parameter α is an input parameter that is used to adjust the degree of importance of sharp turns. If sharp turns are not desired then the value of α should be entered high. Depending on the problem *PathLength* can take a wide range of values. Therefore the value of α is entered as a percentage to the *PathLength*. The binary variables X_{ij} represent whether UAV_j visits DR_i.

The mTSP is solved using a Genetic Algorithm. The chromosome structure is shown in Fig. 5. Genetic operators used in this step are explained in the following section.

5.3.3 Chromosome Structure for Solving mTSP:

Each desired region has a unique label starting from 2. The chromosome is encoded using these labels. The label "1" represents the starting point. In the figure below the sample chromosome used in GA is presented. The chromosome encodes the number of UAVs (salesmen) implicitly. The

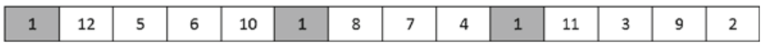


Fig. 5 Chromosome structure for solving mTSP with GA

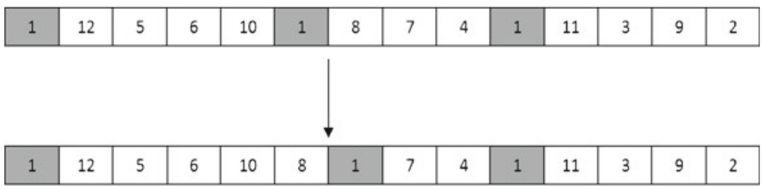


Fig. 6 Changing the assigned number of DR

number of "1"s in a chromosome is the number of UAVs in scenario. Each UAV is assigned to DRs starting by the label "1" to the next "1", except for the last UAV. The last UAV is assigned to DRs from right-most "1" to end of the chromosome. In Fig. 6, there are 3 UAVs and the number of DRs is 11. The DR-x is labeled as (x + 1). The UAV-1 is assigned to DRs of labels (12, 5, 6, and 10), UAV-2 is assigned to DRs of labels (8, 7, and 4), and UAV-3 is assigned to DRs of labels (11, 3, 9, and 2).

5.3.4 GA Operators for Solving mTSP: Changing the Assigned Number of DRs

This operator changes the DR assignments between UAVs, as shown in Fig. 6. In this figure, the input chromosome (upper one) of the operator DR-7 (of label 8) is assigned to the UAV-2, in the output chromosome (lower one) the DR-7 is assigned to the UAV-1. Using this operator provides correct assignment of DRs to UAVs.

The Fig. 7 illustrates the action of this operator. The output chromosome seems to be not good enough, but it is better than the input chromosome. This will be seen more clearly after the application of other operators.

5.3.5 GA Operators for Solving mTSP: Changing the Sequence of DRs for each UAV

This operator swaps the DRs for each UAV at the same time. The visiting sequence of the DRs is changed at the output chromosome. In Fig. 8, the input chromosome (upper one) represents the visiting sequence of DRs for UAV-1 which is {12, 5, 6, and 10}, then in the output chromosome DR of label 5 and DR of label 6 are swapped.

This operator is used in TSP also. This is the parallel version of it, and provides solution rapidly with slight changes. In Fig. 9. the illustration of the operator is given.

5.3.6 GA Operators for Solving mTSP: Swap DRs

This operator realizes similar action as previous one, but the difference is that the swapped DRs are assigned to different UAVs (Figs. 10 and 11).

5.3.7 GA Operators for Solving mTSP: TSP for Sub-paths

This operator actually tries to solve the problem of TSP for each UAV. This operator, using crossover

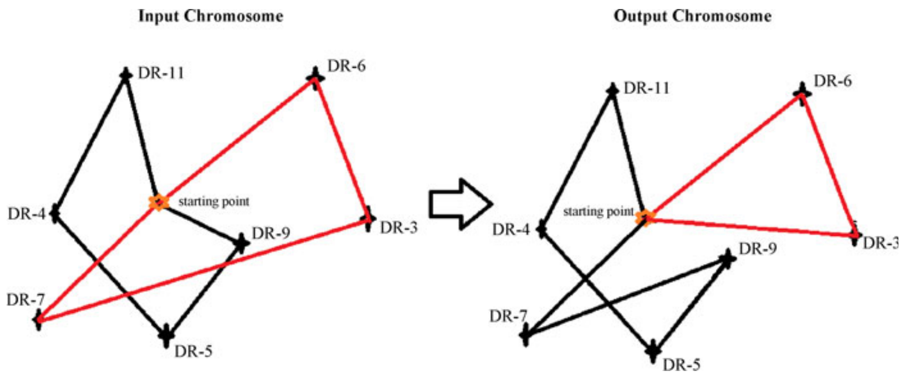


Fig. 7 Illustration of the operator "*changing the assigned number of DR*"

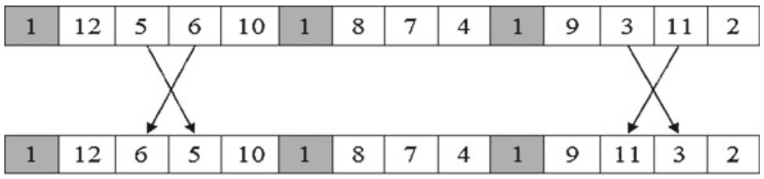


Fig. 8 Changing the sequence of DR for each UAV

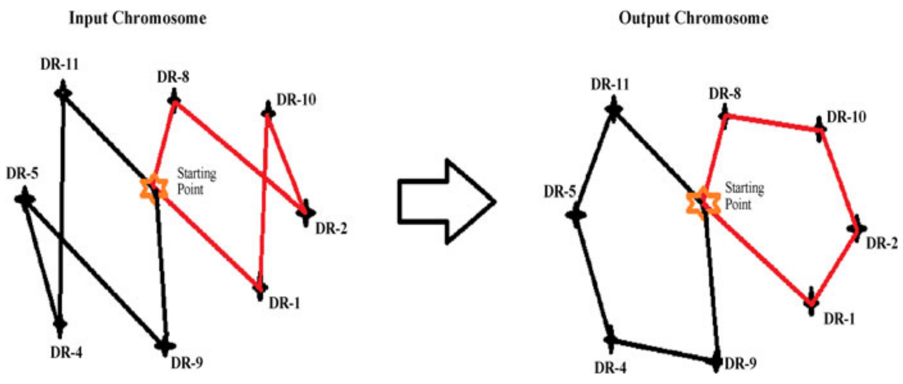


Fig. 9 Illustration of the operator "*changing the sequence of DR for each UAV*"

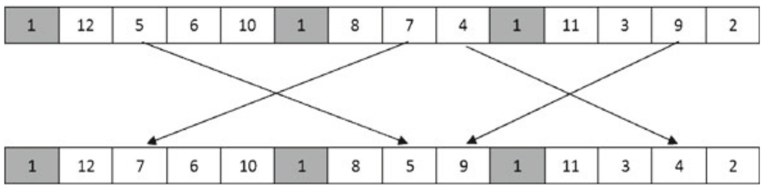


Fig. 10 Swap DRs

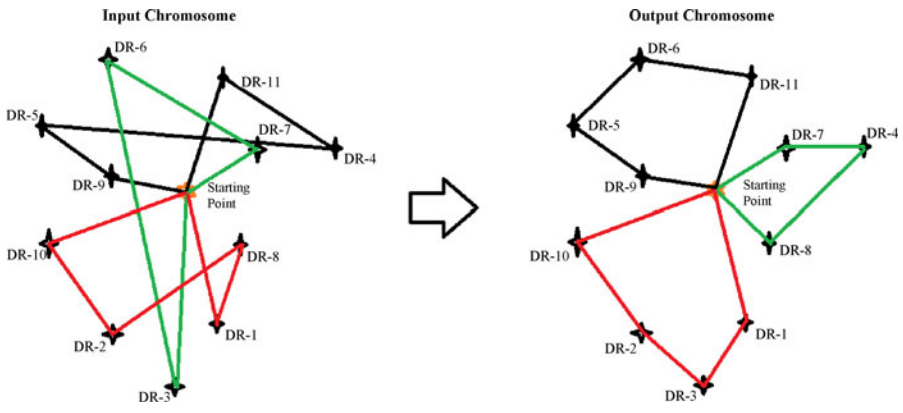


Fig. 11 Illustration of the operator "*Swap DRs*"

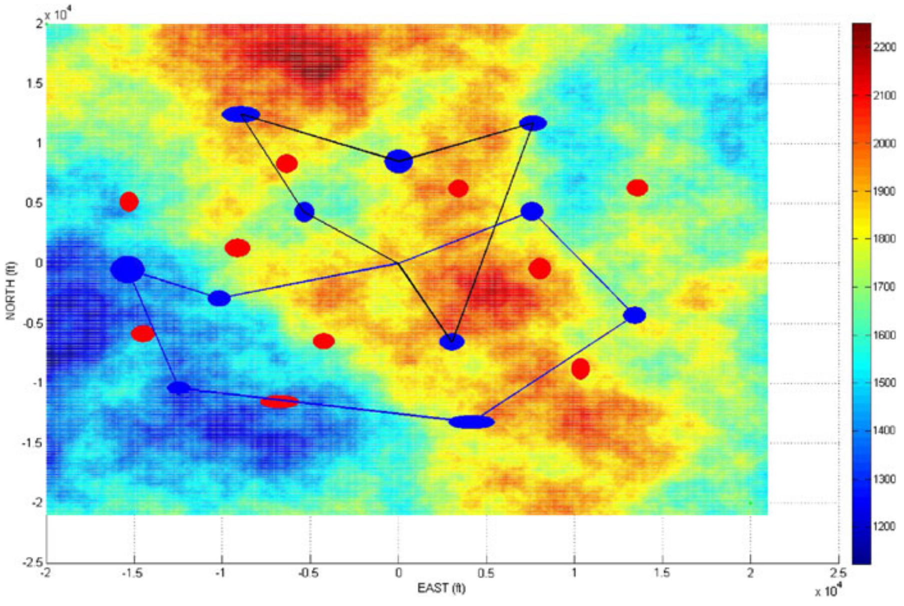


Fig. 12 2D view of the visiting sequence of DRs for 2 UAVs

and mutation operators, optimizes the sequence of DRs allocated for a UAV. Here, the use of crossover operator requires some attention. In TSP, each city must be visited once. In an uncontrolled application of the classical crossover operator it is possible that invalid chromosomes will be generated. For this reason, the operator should be applied in a controlled manner and invalid chromosomes must be disposed of.

Since the number of cities is comparably small, the time required to find the shortest path is short. We try to find the shortest path to avoid navigating too much in areas outside of DRs. When such a sequence is not known, at the points where the

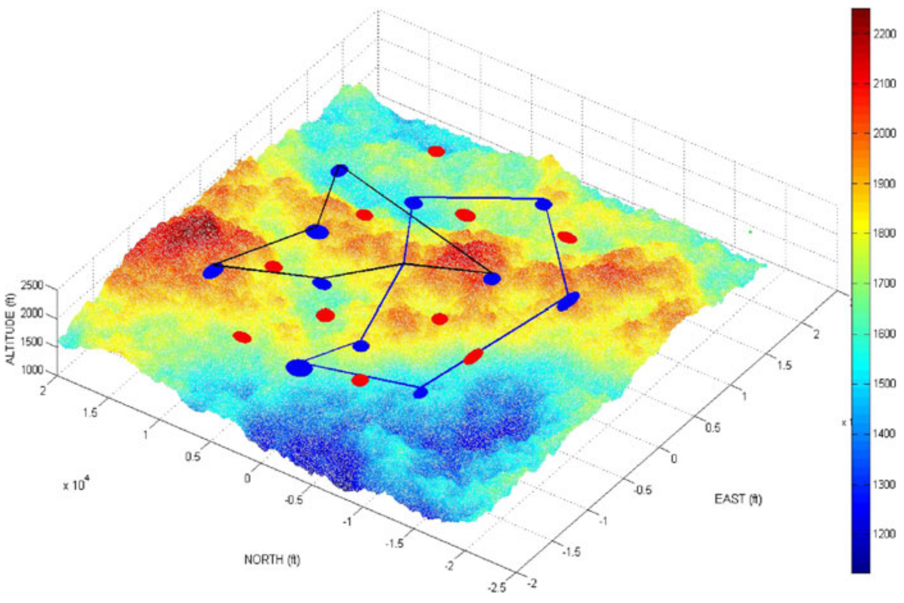


Fig. 13 3D view of the visiting sequence of DRs for 2 UAVs

attractive force is the same for different DRs, the UAV is unable to decide which direction to turn, and the UAV may move along any curve that has an equal effect. The resultant sequence of visit for each UAV has been shown in Figs. 12 and 13.

5.4 Finding SEED_PATH

The population generation for GA is accomplished by changing the randomly selected heading angles of one path. This path is called the seed-path. To find the seed-path, a GA has been used based on the output of the mTSP algorithm. The chromosome construction and operators used in this step are different than those in Section 5.5.

5.4.1 Population Generation

In this section, the population generation procedure will be explained briefly. First of all, each chromosome of the population has three main parts: the first part is encoded for heading angles, the second part is encoded for the velocity, and the third part is encoded for the altitude of UAV. Each main part also has three parts.

For the heading part of the chromosome; the first part designates how many steps (i.e., seconds, time step = 1 s) that the UAV flies with the heading angle values in the second part. The second part contains heading angles to be used for periods of time in the first part. Finally, the third part is the time values at which the UAV starts to change its heading angles towards the next destination (DR or the final point). The structure of a chromosome is shown in Fig. 14. According to the example in the figure, the UAV flies with the heading angle of pi/2 for $120 * (1 - 0.15) = 102$ s and starts to turn from pi/2 (East) to pi/4 (North-East) at the 103rd second.

Each of the straight paths connecting to the centers of DRs is divided into four equally spaced sub-paths between two consecutive points (points: The starting position, center of DR, and the final

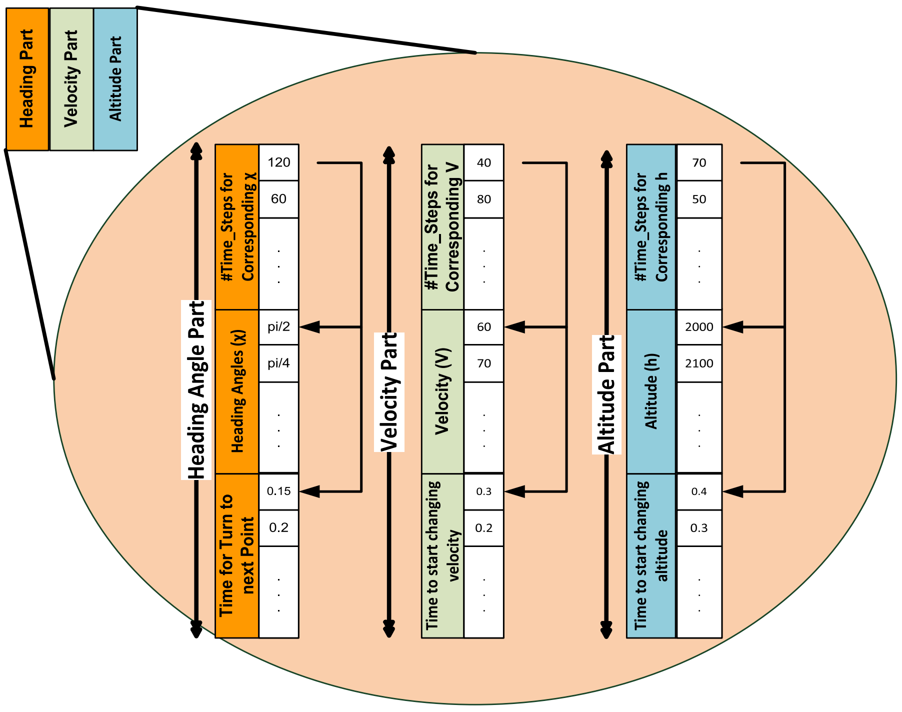

Heading Part
Velocity Part
Altitude Part
Heading Angle Part
#Time_Steps for Corresponding χ
120
60
Heading Angles (χ)
pi/2
pi/4
Time for Turn to next Point
0.15
0.2
Velocity Part
#Time_Steps for Corresponding V
40
80
Velocity (V)
60
70
Time to start changing velocity
0.3
0.2
Altitude Part
#Time_Steps for Corresponding h
70
50
Altitude (h)
2000
2100
Time to start changing altitude
0.4
0.3

Fig. 14 Chromosome structure for the *SEED_PATH* finding step

position). Then, the total length of the tour is calculated.

The altitude and velocity values of each chromosome is obtained by the same procedure and then filtered to guarantee the velocity and altitude of the UAV do not change in 5 s time intervals.

5.4.2 Genetic Operators: Crossover

New chromosomes are obtained from two randomly selected chromosomes using the crossover operation. Two new offspring paths are obtained by crossing over the parts of parents, as shown in Fig. 15. The same operation is valid for subparts also. For a chromosome, the probability of being a parent of a crossover operation is proportional to its evaluation function value. A higher evaluation function value means a higher probability of selection.

5.4.3 Genetic Operators: Mutation

The mutation rate is fixed to 5 %. For the first main block (i.e., heading angle block), randomly selected heading angles (the second part of the main block) are changed by adding random numbers between pi/5 and pi/10 rad. The first part of the main block is changed by an amount that is selected randomly between the minimum and maximum of this part in each chromosome. Then, the normalization procedure is executed to keep the total number of steps between each point constant. The last part of the main block is changed by a number that is selected randomly between 0 and 0.1.

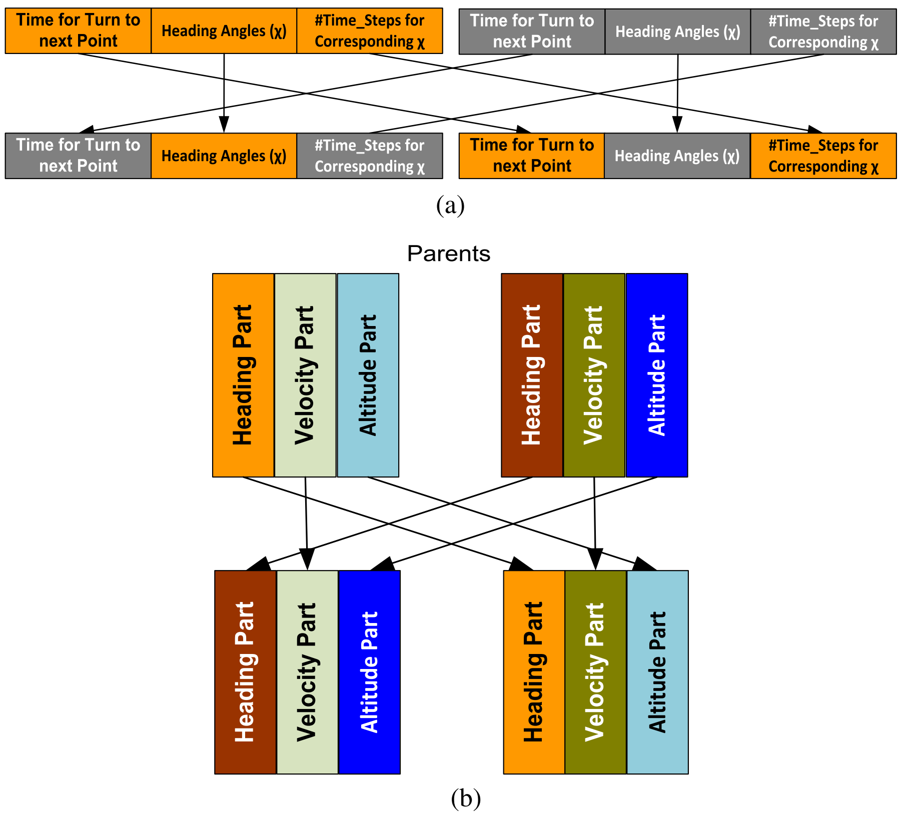


Fig. 15 Crossover operators (**a**) for sub-parts, (**b**) for main blocks

For the velocity part of the chromosome, randomly selected velocity values (the second part of the main block) are changed by adding random numbers between −20 ft/s and 20 ft/s. For the altitude part of the chromosome, randomly selected altitude values are changed by adding random numbers between −200 ft and 200 ft. Other parts of the second and third main blocks are changed in the same way as the first main block.

5.5 Final Path Finding with the Proposed GA Operators

This step is the third and the final step of our path planning algorithm. At this step, the path of the UAV is planned using the proposed operators in addition to the classical GA operators. The proposed operators are described briefly and by presenting illustrations. The chromosomal structure used in this step is different than that of used in the previous step. The chromosome is composed of only the heading angles of the UAV.

5.5.1 Population Generation

The population in this step is created by changing the randomly selected position of SEED_PATH, which is calculated in the previous step.

5.5.2 Proposed GA Operators: Push-From-Forbidden-Region (PFFR)

The purpose of these operators is to modify the undesirable parts of a path intersecting any FR [7]. Depending on how much the UAV enters a particular FR, more than one PFFR operator is recommended. This process changes the heading angles, which are the parameters of the optimization algorithm. One sample is shown in Fig. 16. The operators in this sample work as follows: First, any chromosome (path) that enters any FR is detected. Second, the entrance point of that chromosome to the FR is detected. Third, the coordinates ($\text{east}_{\text{beforeFR}}$, $\text{north}_{\text{beforeFR}}$) and heading

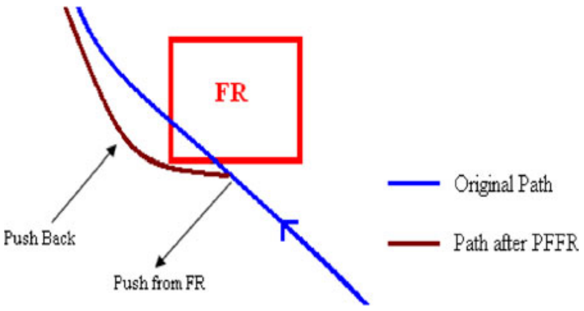


Fig. 16 Push-From-Forbidden-Region (PFFR)

angle χ_{beforeFR} b steps before that particular FR is entered (to remove the UAV from the FR) are calculated. The value of b is calculated ($1 \le b \le 8$) by using the length of the diagonal of the FR and the length of the path inside the FR. The angle between the NORTH axis and the line segment passing through these coordinates and the center of the FR ($\text{east}_{\text{center\_of\_FR}}$, $\text{north}_{\text{center\_of\_FR}}$) is calculated as

$$\chi_d = \arctan\left(\frac{east_{\text{center\_of\_FR}} - east_{\text{beforeFR}}}{north_{\text{center\_of\_FR}} - north_{\text{beforeFR}}}\right) \tag{8}$$

The heading angle of the UAV before entering the FR is χ_{beforeFR}. The UAV is prevented from entering the FR by changing the angle χ_{d} to $\chi_{\text{d}} + \Theta$ (where $\pi/2 \le \Theta \le \pi$). However, there is another step to be performed: due to this special "mutation", unintended changes in the remaining path should be corrected. This part of the path should be as close to the path before the mutation operation as possible. We find the point that the UAV exits from the FR and the heading angle at the position of {*floor* (b/2)} steps before it is changed by subtracting $\chi_{\text{d}} + \Theta$. Thus, the UAV becomes closer to the rest of the path before the mutation operation, as illustrated in Fig. 16. As we have stated before, the strength of the pushing force depends on the length of path intersecting with the FR. In addition, it depends on Θ as well. Another parameter of the operator is b. This parameter determines how many points before the entry point the UAV should be pushed from the FR. It increases if the UAV becomes closer to the center of the FR because, when it flies near the center, the length of the path intersecting the FR increases, and the UAV is forced to make a turn with a greater radius.

5.5.3 Proposed GA Operators: Pull-to-Desired-Region (PTDR)

This operator is proposed to shove the UAV towards the DR in cases when it flies close to a DR without entering it [7]. For each DR, the PTDR operator searches for paths that do not enter the DR. Then, for each path, the closest point of the path to the center of the DR is found. Among these points, the closest to the center of the DR is selected for mutation. For this individual and mutation point, the heading angles are changed to pull the UAV to the center of the DR, as illustrated in Fig. 17.

Let the minimum distance between the center of a DR and the mutation point on the path be $\text{dist}_{\text{center\_of\_DR}}$. The number of heading angles to be changed is calculated as

$$NoHeadingAngles = floor\left(dist_{\text{center\_of\_DR}} / Velocity_{\text{UAV}}\right) \tag{9}$$

where $\text{Velocity}_{\text{UAV}}$ is the constant velocity of the UAV. Afterward, the starting index for the angle

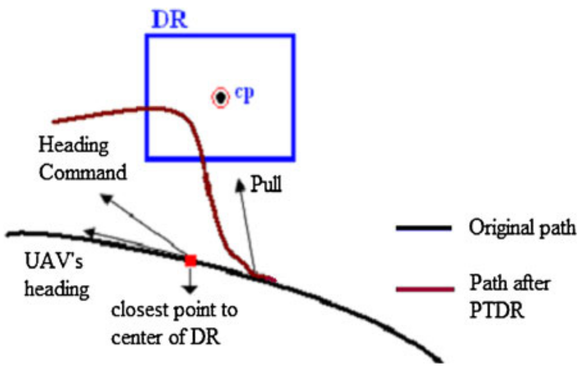


Fig. 17 Pull-To-Desired-Region (PTDR)

mutation should be calculated. This point is calculated using the point closest to the center of the DR, as shown in Fig. 17, and the heading command at this point (χ_{command}) (Eq. 10). The critical idea here is the use of the heading command at this coordinate instead of the heading angle of the UAV. It is necessary to use this heading command because of the dynamical constraints of the UAV. In the case where a simple kinematic model is used, it will not be an issue because the heading command and the heading responses of the model UAV are similar. However, in real life applications, they are not similar, especially for rapidly changing heading commands. Let the angle between the North axis and the line passing through the closest point and center of the DR be χ_c.

$$floor\left(\left(r_{\min} * \left(abs\left(\chi_c - \chi_{\text{command}}\right)\right) + dist_{\text{center\_of\_DR}}\right)\right) / Velocity_{\text{UAV}}) \tag{10}$$

where $r_{\min}$ is the minimum turning radius of our UAV.

5.5.4 Proposed GA Operators: Pull-to-Desired-Region2 (PTDR2)

This operator is defined to pull the UAV to the next DR as soon as possible when it exits another DR [7]. It finds the point that the UAV leaves a DR and determines the angle at this point. Then, it changes this heading angle and the consecutive K ($1 \leq K \leq 4$) heading angles to pull the UAV to the next DR, as illustrated in Fig. 18.

If the next DR is the first DR to be visited, then, starting from the first heading angles, K heading angles are mutated. K is calculated using the distance between the starting point and the center of the next-DR, and the velocity of the UAV in a similar calculation given in Eq. 4.

5.5.5 Proposed GA Operators: Ascend-to-EScape (ATES)

This operator is defined to increase altitude of the UAV to avoid hitting terrain. There is a critical

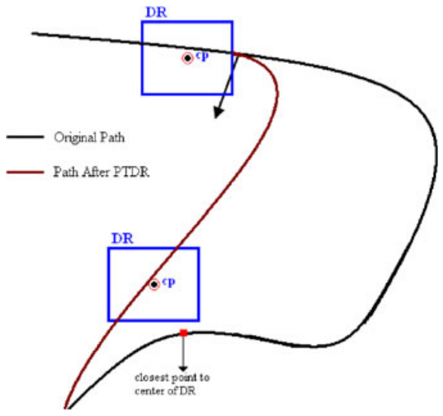


Fig. 18 Pull-To-Desired-Region (2)

altitude limit for the UAV, if the altitude of the UAV falls below this value, ATES operator decides on ascending by controlling the heights on the route. Since the ascending decision increases energy dissipation, there is trade-off between critical altitude level and energy. Our algorithm adjusts the weight of energy dissipation and weight of the critical altitude penalty according to what has been obtained. If ascending decision is not feasible due to the energy dissipation then operator decides on heading around to escape possible obstacles.

5.5.6 Proposed GA Operators: Change ALTitude (CALT)

This operator is defined to change the altitude of the UAV to ensure the optimal flight level. The operator is utilized with the other operators to adjust the altitude. For example if the UAV flying over FR and it is not possible to avoid FR or flying over it increases the collected amount of information then operator will carry out UAV to ascend. When the operator is called with the PTDR then it decides on ascending or descending, firstly. Since descending increases the resolution of the images taken, and ascending provides the image of the wider area. The energy dissipation is also taken into account when deciding on changing altitude. The operator is utilized with the PTFP operator to achieve final altitude constraint also. It starts to change the altitude of the UAV A steps before the destination point. The parameter A is an integer which is selected randomly between 1 and 9.

Path Planning Algorithm:

```
1:  Call Compute_Forces
2:  Call Compute_DISTANCE_Matrix
3:  Call mTSP
4:  Call Seed_Path_Finder
5:  Init Population
6:  Set Best_Objective_Value to minus Infiniti.
7:  Set SameResult to 0
8:  Read SameResult_Limit
9:  Set POP_SIZE to 50
10: Set Operation_List as {Proposed_Operators,
Crossover, Mutation}
11: Repeat
12:   For each path in the population
13:     Call Simulink_Model
14:     Compute Objective_Values
15:     Determine whether it enters to FR or not.
16:   End For
17:   Sort Objective_Values and keep best three for next
18:   If Objective_Values (1) greater than
      Best_Objective_Value Then
19:     Set Best_Objective_Value to Objective_Values (1)
20:     Set SameResult to 0
21:   Else
22:     Increment SameResult
23:   End If
24:   Repeat
25:     Select a path from present population randomly.
26:     Select Operation from Operation_List Randomly
27:     Case Selected operation is "Proposed_Operators"
28:     For three times apply proposed operators
29:       If it flies over to any FR Then
30:         Call PFFR and CALT
31:       End If
32:         If the altitude is at critical level Then
33:           Call ATES
34:       End If
35:       If the distance to next DR>thrDR Then
36:           Call SRDR
37:       End If
38:       Call PTDR and CALT
39:       If it enters to the DR Then
40:         Call RIDR
41:       End If
42:       Call PTFP and CALT
43:       End For
44:       Case Selected Operation is Crossover
45:         Select another path from present population
            randomly to generate new chromosomes.
46:         Call Crossover_Routine
47:       Case Selected Operation is Mutation
48:         Call Classical_Mutation_Routine
49:   Until new generation is created.
50: Until SameResult equals to SameResult Limit
```

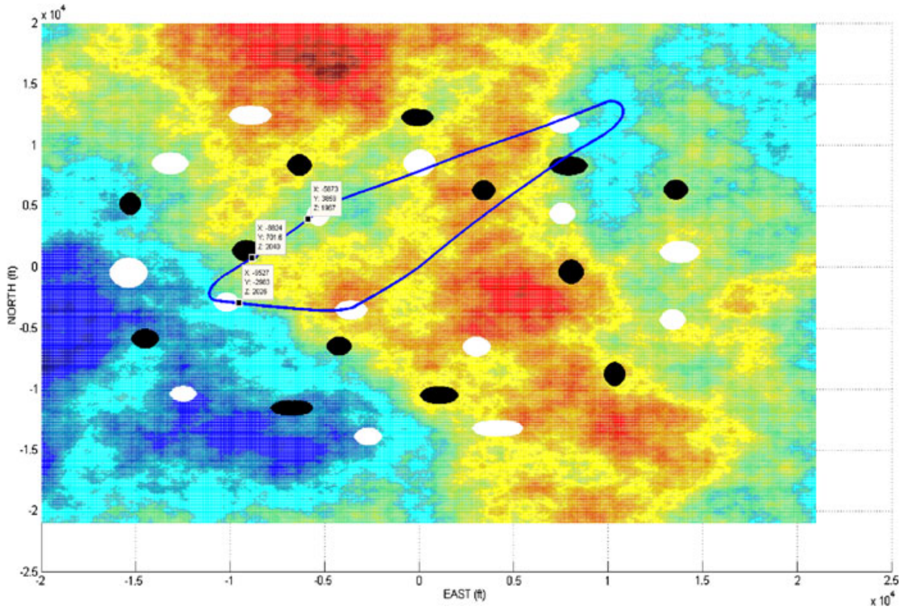


Fig. 19 2D view of the path of UAV1 in scenario1

5.5.7 Description of the Algorithm

The path planning algorithm and autopilot are implemented in MATLAB/Simulink. The controller and UAV dynamic model are constructed in Simulink and heading angles, speed and altitude inputs are given determined using the algorithm implemented in MATLAB. The pseudo code of the main algorithm is given below. As shown in this code, the proposed operators are applied to the path *at most* three times successively. The reason for this repetitive loop is that the proposed operators may cause unintentional effects. For example, the path that does not enter any FR may enter to one of them after applying PTDR even if there is a correction part in the operator.

The specific iteration number throughout the experiments has been decided by trial and error,

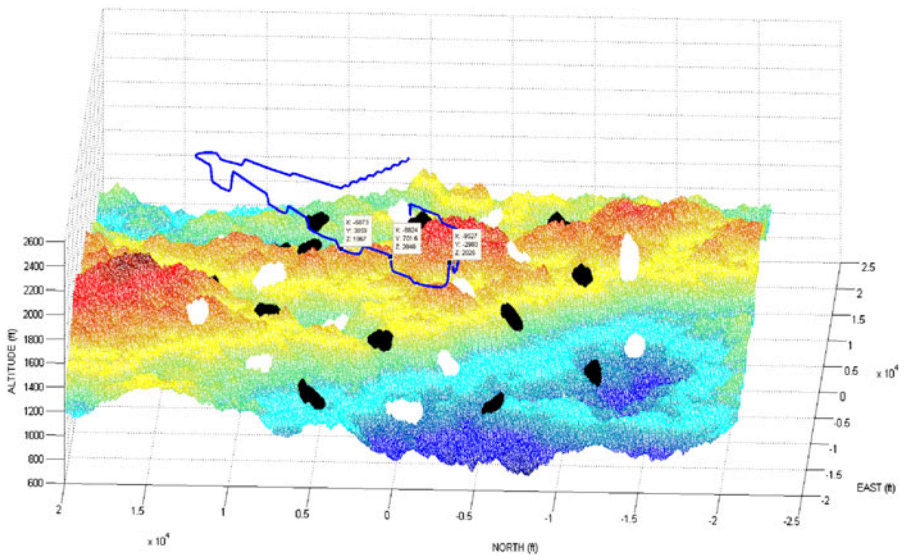


Fig. 20 3D view of the path of UAV1 in scenario1

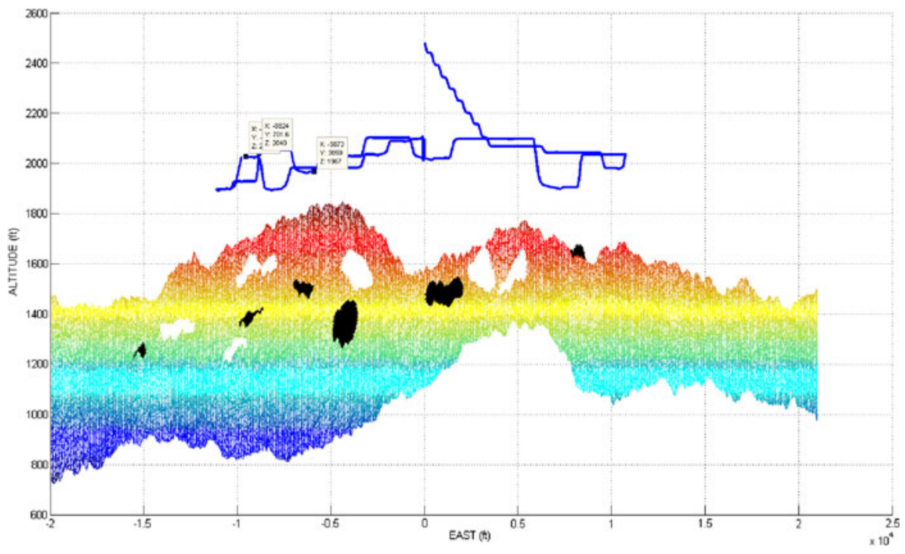


Fig. 21 The East-Altitude view of the path of UAV1 in scenario1

e.g., [3]. These successive applications of operators should be limited; otherwise, there will be an infinite loop for unfeasible paths. Our experiments show that three successive iterations are sufficient to make remarkable changes in the path. The operator is utilized with the other operators to adjust the altitude, so in lines of 30, 38, and 42 call statement has two function references.

6 Results

The Region of Interest has been generated by using the diamond-square algorithm defined in [11]. It is also known as the random midpoint displacement fractal, the cloud fractal or the plasma fractal, because of the plasma effect produced when applied. The algorithm starts with a 2D grid

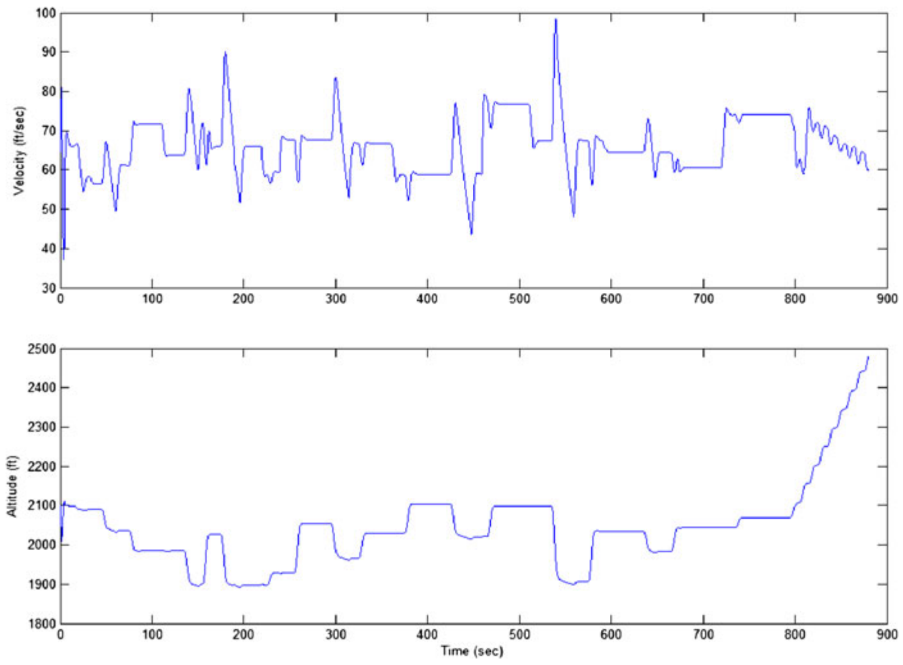


Fig. 22 The velocity and the altitude of UAV1 in scenario1

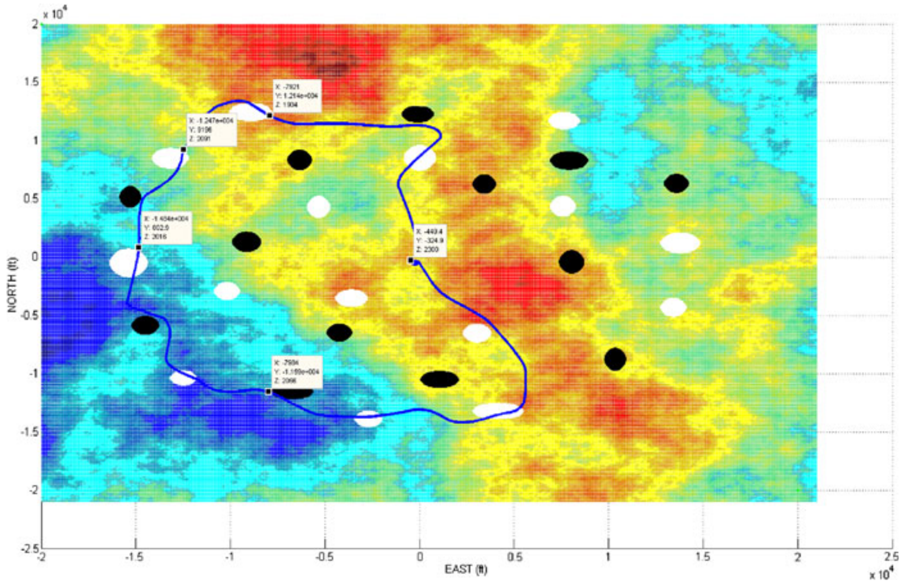


Fig. 23 2D view of the path of UAV2 in scenario1

then randomly generates terrain height from four seed values arranged in a grid of points so that the entire plane is covered in squares.

We have tested our algorithm on three different scenarios. In the first scenario there are three UAVs, 14 DRs, and 13 FRs. The minimum number of the DRs for each UAV is selected as 3. In the Figs. 19, 20 and 21 the resultant path for UAV1 has been presented from different views.

In Fig. 19, the 2D projection of the path has been shown. In this figure there are three datatip points. Both these marked points and the noticeable parts of the path (such as U turns) help observing the effects of the proposed operators. For example, the UAV flies over the right-most FR after making U turn close to the North-East of the ROI. But, we can easily observe from Fig. 20, the UAV climbs over this FR to decrease violation penalty. The starting and finish points are (East, North, Altitude) = (0, 0, 2000) ft, and (East, North, Altitude) = (0, 0, 2500) ft, respectively.

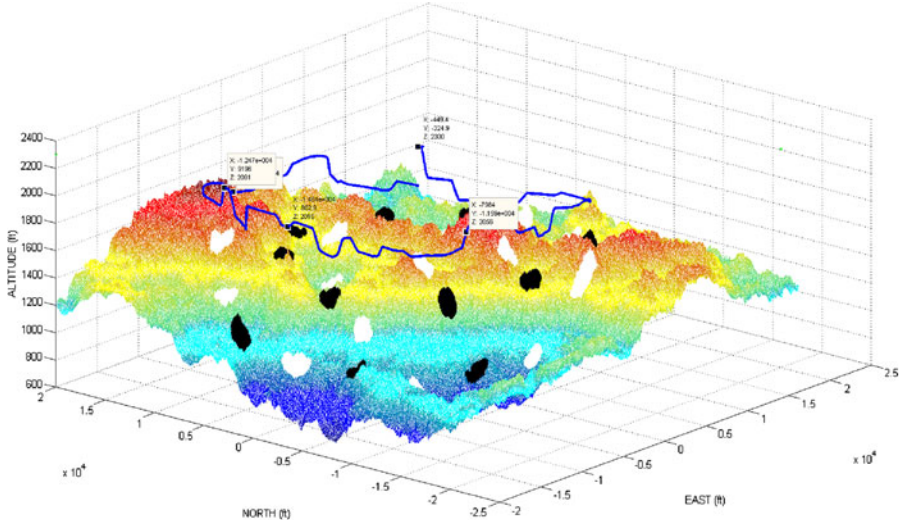


Fig. 24 3D view of the path of UAV2 in scenario1

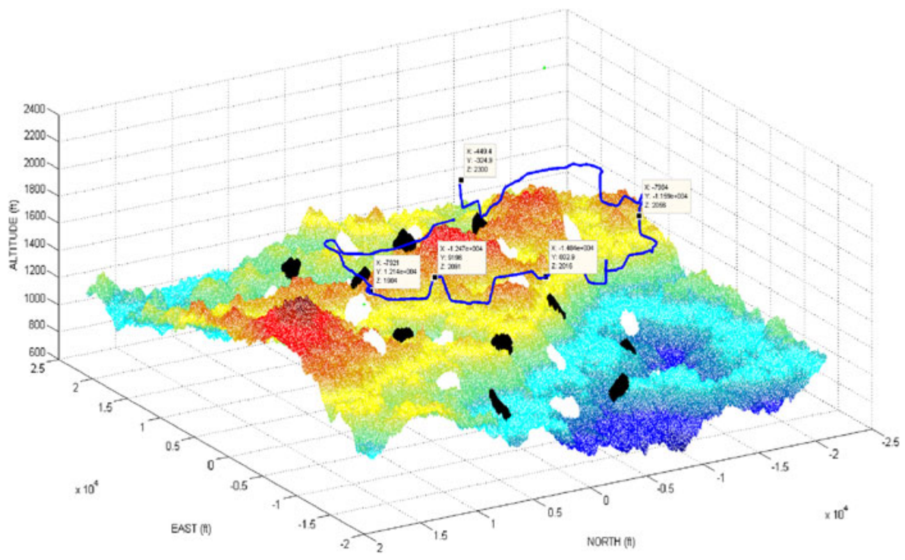


Fig. 25 3D view of the path of UAV2 in scenario1

The Fig. 22 presents the altitude and velocity of UAV1 in scenario 1.

In the Figs. 23, 24 and 25 the resultant path for UAV2 has been presented from different views. In Fig. 23. the 2D projection of the path has been shown. In this figure there are three datatip points. Both five marked points and the noticeable parts of the path helps observing the effects of the proposed operators. The starting and finish points are (East, North, Altitude) = (0, 0, 2000) ft, and (East, North, Altitude) = (0, 0, 2300) ft, respectively.

In the Figs. 26 and 27 the resultant paths for all UAVs have been presented in 2D and 3D, respectively. The path of each UAV has been indicated in the same figures.

In the second scenario there are two UAVs, 11 DRs, and 10 FRs. The minimum number of

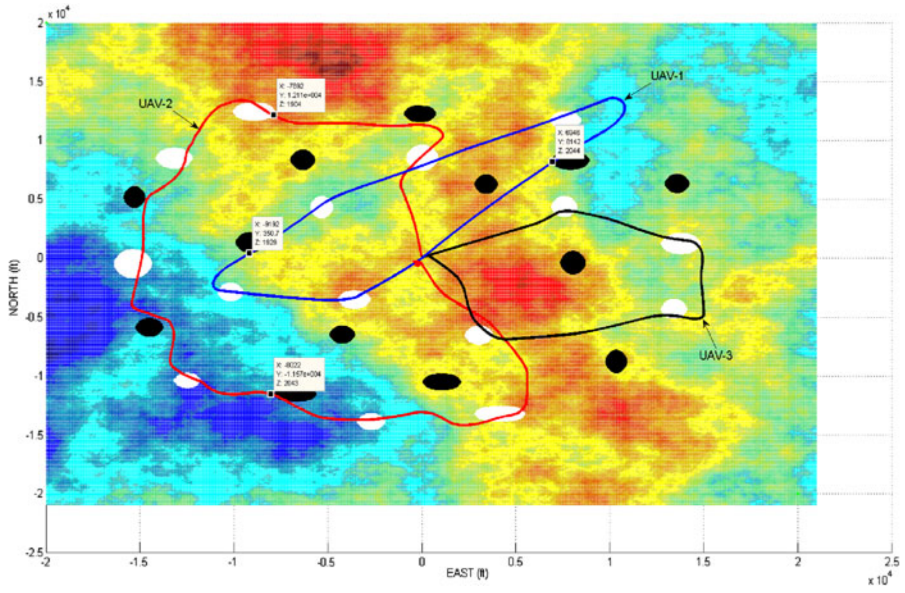


Fig. 26 2D view of the paths of all UAVs in scenario1

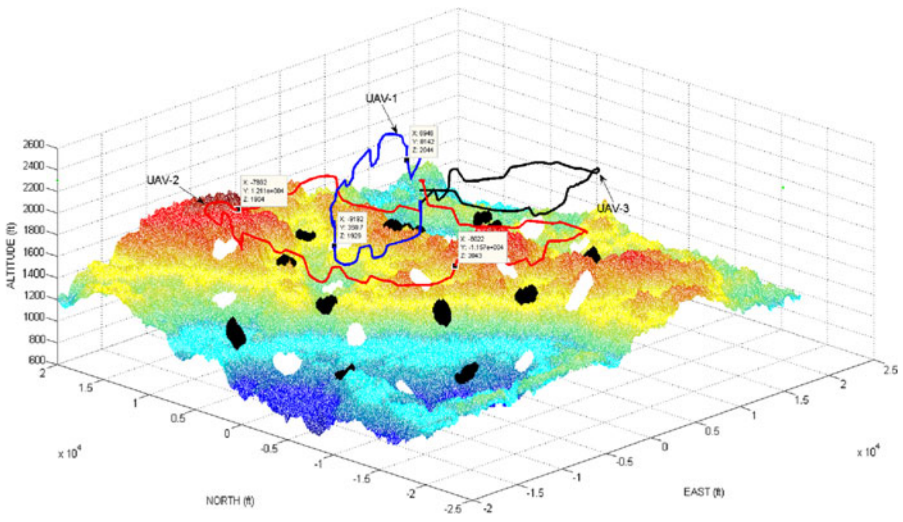


Fig. 27 3D view of the paths of all UAVs in scenario1

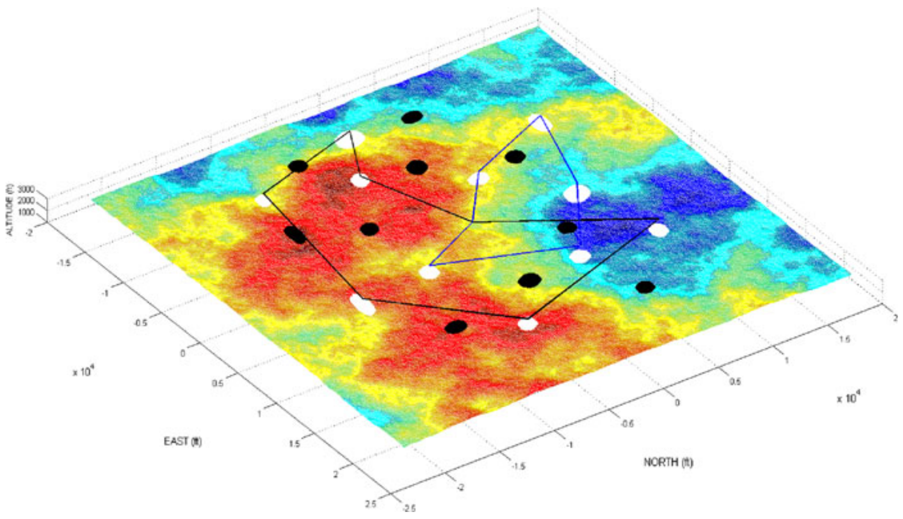


Fig. 28 The result of the mTSP step for scenario2

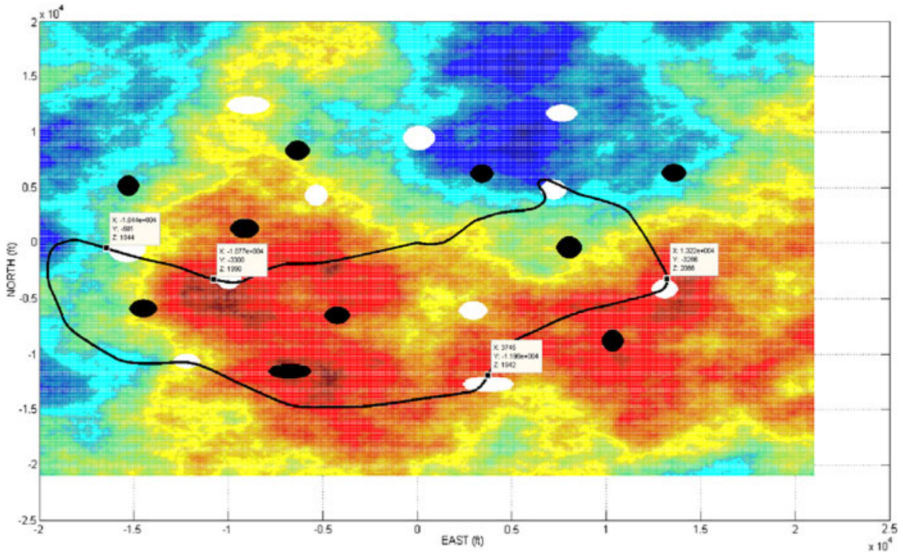


Fig. 29 2D view of the path of UAV1 in scenario2

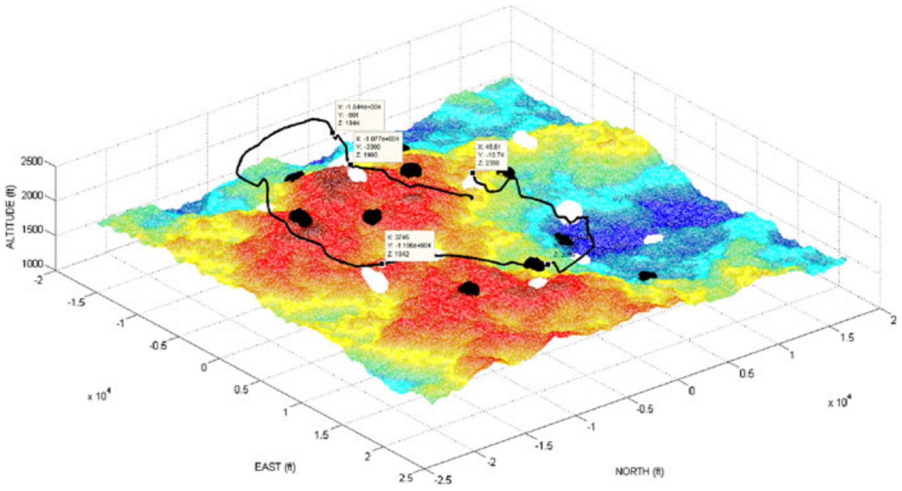


Fig. 30 3D view of the path of UAV1 in scenario2

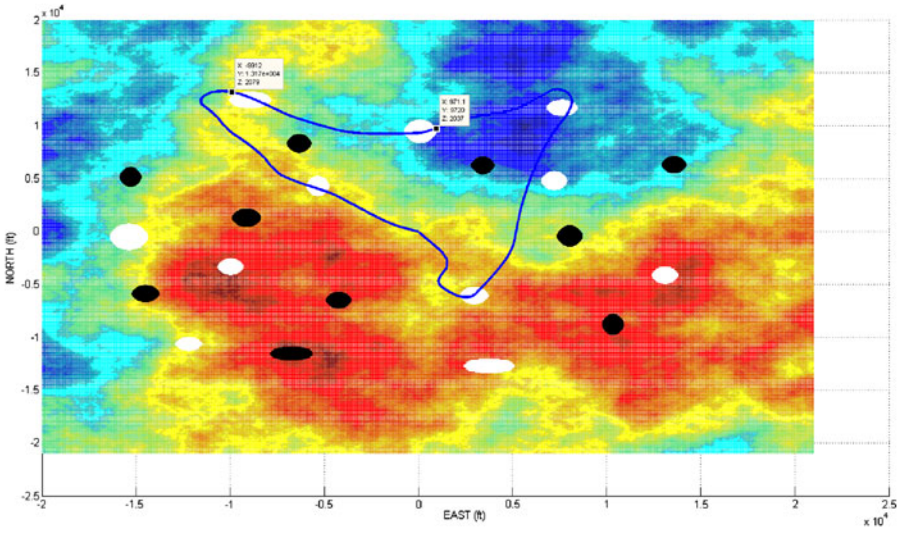


Fig. 31 2D view of the path of UAV2 in scenario2

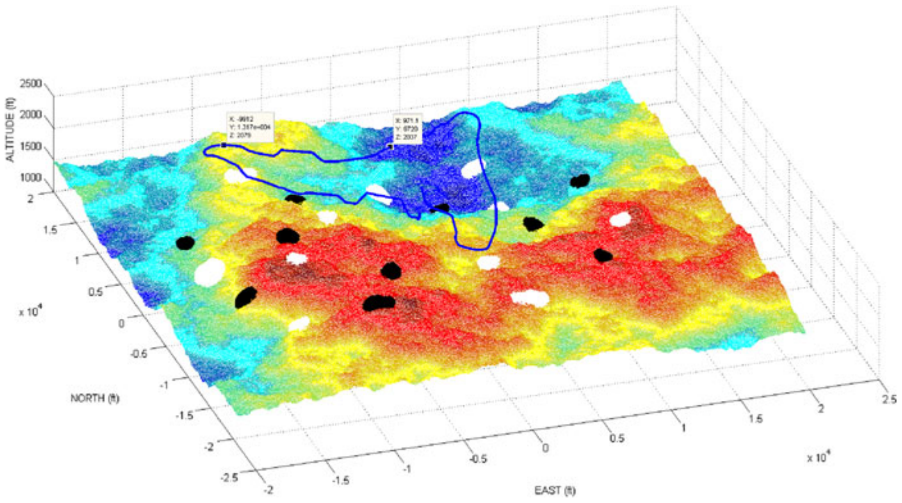


Fig. 32 3D view of the path of UAV2 in scenario2

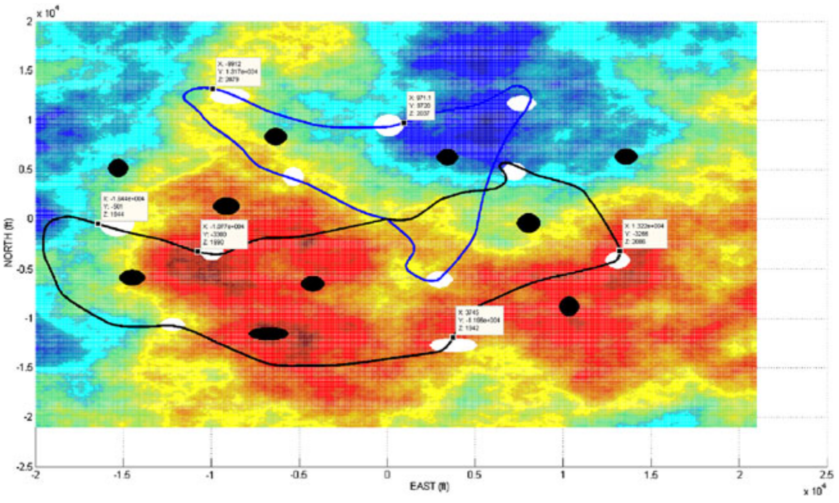


Fig. 33 2D view of the paths of all UAVs in scenario2

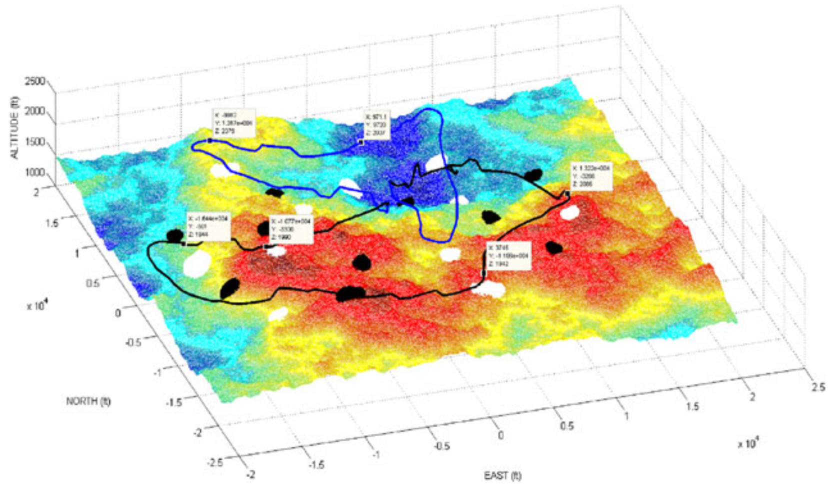


Fig. 34 3D view of the paths of all UAVs in scenario2

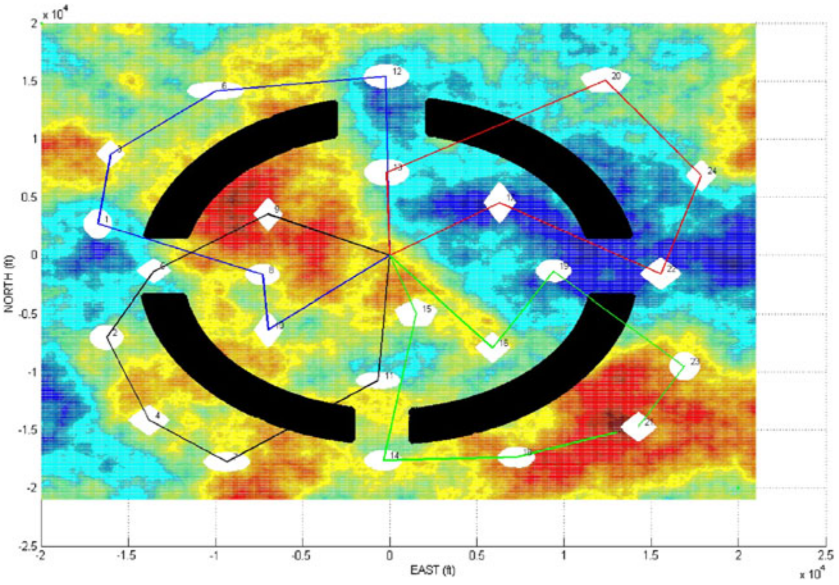


Fig. 35 The result of the mTSP step for scenario3

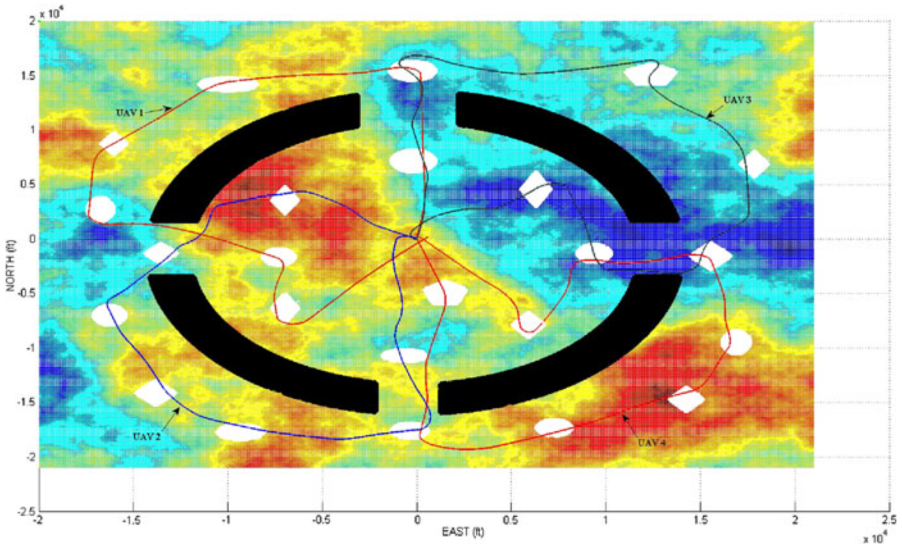


Fig. 36 2D view of the paths of all UAVs in scenario3

the DRs for each UAV is selected as 3. The result of the mTSP step for second scenario has been shown in Fig. 28. In the Figs. 29, 30, 31, 32, 33 and 34 the resultant paths have been presented from different views. The starting and final points are the same for both UAVs. The starting point is (East, North, Altitude) = (0, 0, 2000) ft, and the final point is (East, North, Altitude) = (0, 0, 2400) ft. Both 2D projections of the path and 3D views are presented.

In the third scenario there are four UAVs, 24 DRs, and 4 FRs. The topology of this scenario is the most complex even if the number of FRs is less. The complexity is affected by the number of regions, the area of the regions, and the locations of regions with respect to each other [7]. The minimum number of the DRs for each UAV is selected as 5. The output of the mTSP step is shown in Fig. 35. In the Figs. 36 and 37 the resultant paths have been presented in 2D and 3D views, respectively.

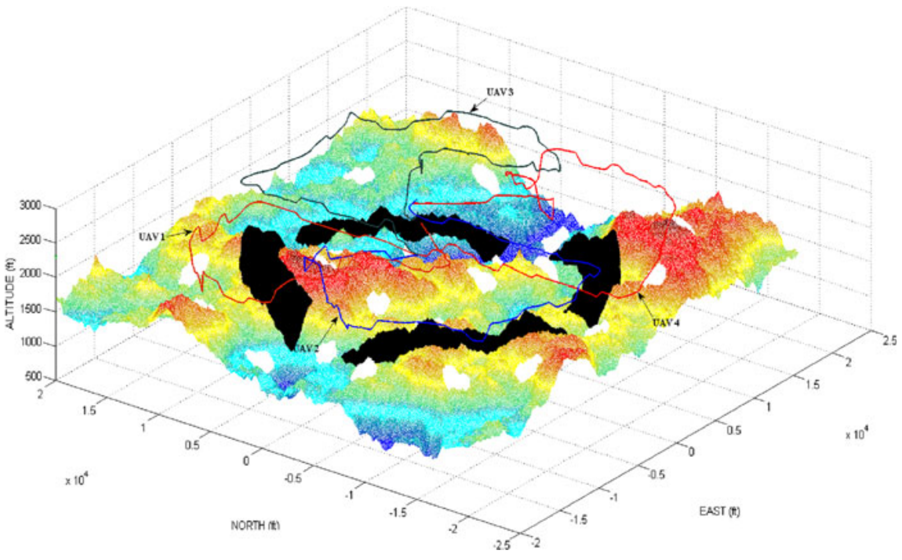


Fig. 37 3D view of the paths of all UAVs in scenario3

7 Conclusions

The paper presents an algorithm for the path planning problem for a multiple Unmanned Air Vehicles (UAVs) in 3D environment. This algorithm is an extension of our previous algorithm [7] with the proposal of additional novel evolutionary operators and the utilization of mTSP. The overall algorithm and the proposed operators are applied to the multi-UAV case. Multiple UAVs may be used when the mission time is not sufficient to visit all DRs. In such a case, more than one UAV should fly to collect information. The mission time of each UAV is different for the fact that each UAV has a limited energy. In this case, the path planning operation starts with finding the **DISTANCE** matrices for each UAV by considering its mission time constraint. The second and the third steps of the algorithm are adopted to use both velocity and the altitude of UAVs to optimize the path for information collection. In the second step, the mTSP adaptation also solves the assignment problem of "which DR should be visited by which UAV". In the third step of the algorithm two novel mutation operators are defined and implemented: Ascend-To-EScape (ATES), Change ALTitude (CALT). As demonstrated by the results, to obtain satisfactory solutions for these types of problems, problem-specific evolutionary operators should be defined. These operators should imitate a human path planner's experience. Furthermore, to increase the chance of the overall algorithm reaching the global optimum solution, initialization using the mTSP and the construction of an intelligently seeded initial population using the SEED-PATH algorithm are among the additional original contributions.

In the future, 3D path planning can also be studied for online path planning to maximize the collected amount of within certain period of time. There may be additional new operators, and constraints. The region of interest may not be known exactly and there may be pop-up DRs or FRs.

Acknowledgement This work is supported by The Scientific and Technological Research Council of Turkey (TUBITAK) under Grant 110E192.

References

1. Alain Fournier, D.F., Carpenter, L.: Computer rendering of stochastic models. Commun. ACM **25**(6), 371–384 (1982)
2. Audet, C., Dennis Jr., J.E.: Analysis of generalized pattern searches. SIAM J. Optim. **13**(3), 889–903 (2003)
3. Barraquand, J., Latombe, J.C.: Robot motion planning: a distributed representation approach. Int. J. Robot. Res. **10**, 628–649 (1991)
4. Bektaş, T.: The multiple traveling salesman problem: an overview of formulations and solution procedures. Omega **34**, 209–219 (2006)
5. Besada-Portas, E., Torre, L., Cruz, J.: Evolutionary trajectory planner for multiple UAVs in realistic scenarios. IEEE Trans. Robot. **26**(4), 619–634 (2010)
6. Burl, J.B.: Linear Optimal Control: H(2) and H (Infinity) Methods. Addison-Wesley Longman Publishing Co., Inc., Boston (1998)
7. Ergezer, H., Leblebicioğlu, K.: Path planning for UAVs for maximum information collection using evolutionary computation. IEEE Trans. Aerosp. Electron. Syst. **49**(1), 502–520 (2013)
8. Göktoğan, A.H., Sukkarieh, S., et al.: Airborne vision sensor detection performance simulation. In: SimTecT'05, Simulation Conference and Exhibition. Sydney, Australia (2005)
9. Hasircioglu, I., Topcuoglu, H.R., Ermis, M.: 3-D path planning for the navigation of unmanned aerial vehicles by using evolutionary algorithms. In: Proc. Genet. Evol. Comput. Conf., pp. 1499–1506 (2008)
10. Hoffmann, G.M., Tomlin, C.J.: Mobile sensor network control using mutual information methods and particle filters. IEEE Trans. Autom. Control **55**(1), 32–47 (2010)
11. Hoffmann, G.M., Waslander, S.L., Tomlin, C.J.: Distributed cooperative search using information-theoretic costs for particle filters, with quadrotor applications. In: Proceedings of the AIAA Guidance, Navigation, and Control Conference, pp. 21–24. Keystone (2006)
12. Honeywell Technology Center. Application of multivariable control theory to aircraft control laws. Final Report: Multivariable Control Design Guidelines, ser. AD-a315 259. Honeywell Incorporated minneapolis mn (1996)
13. Kavraki, L.E., Latombe, P., Overmars, M.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Autom. **12**, 566–580 (1996)
14. Klesh, A.T., Kabamba, P.T., Girard, A.R.: Path planning for cooperative time-optimal information collection. In: 2008 American Control Conference, pp. 1991–1996. Seattle (2008)
15. Koditschek, D.: Exact robot navigation by means of potential functions: some topological considerations. In: IEEE International Conference on Robotics and Automation, pp. 1–6 (1987)
16. Kwag, Y., Kang, J.: Obstacle awareness and collision avoidance radar sensor system for low-altitude flying smart UAV. Digit. Avionics Syst. Conf. **2**, 12.D.2–121-10 (2004)

17. Latombe, J.C.: Robot Motion Planning. Kluwer Academic Press (1990)
18. LaValle, S.M., Kuffner, J.J.: Randomized kinodynamic planning. Proc. IEEE. Int. Conf. Robot. Autom. **1**, 473–479 (1999)
19. Mittal, S., Deb, K.: Three-dimensional offline path planning for UAVs using multiobjective evolutionary algorithms. In: Proc. IEEE Congr. Evol. Comput., vol. 7, pp. 3195–3202 (2007)
20. Nikolos, I.K., Tsourvelouds, N.C.: Path planning for cooperating unmanned vehicles over 3-D terrain. Inf. Control Autom. Robot. **24**, 153–168 (2009)
21. Nikolos, I.K., Valavanis, K.P., Tsourveloudis, N.C., Kostaras, A.N.: Evolutionary algorithm based offline/online path planner for UAV navigation. IEEE Trans. Syst. Man Cybern. B Cybern. **33**(6), 898–912 (2003)
22. Pehlivanoglu, Y.V., Baysal, O., Hacioglu, A.: Vibrational genetic algorithm based path planner for autonomous UAV in spatial data based environments. In: Proc. 3rd Int. Conf. Recent Adv. Space Technol., vol. 7, pp. 573–578 (2007)
23. Pitre, R.R., Li, X.R., Delbalzo, R.: UAV route planning for joint search and track missions- an information-value approach. IEEE Trans. Aerosp. Electron. Syst. **48**(3), 2251–2565 (2012)
24. Schwartz, J.T., Sharir, M.: On the piano mover's problem: I. The case of a two-dimensional rigid polygonal body moving admits polygonal barriers. Commun. Pur. Appl. Math. **36**, 345–398 (1983)
25. Schwartz, J.T., Sharir, M.: On the piano mover's problem: II. General techniques for computing topological properties of real algebraic manifolds. Adv. Appl. Math. **4**, 51–96 (1983). Academic Press
26. Schwartz, J.T., Sharir, M.: On the piano mover's problem: III. Coordinating the motion of several. Independent bodies: the special case of circular bodies. Plan. Geom. Complexity Robot Motion (1987)
27. Szczerba, R.J., Galkowski, P., Glickstein, I., Ternullo, N.: Robust algorithm for real-time route planning. IEEE Trans. Aerosp. Electron. Syst. **36**(3), 869–878 (2000)
28. Zhang, R., Zheng, C., Yan, P.: Route planning for unmanned air vehicles with multiple missions using an evolutionary algorithm. In: Proc. IEEE 3rd Int. Conf. Nat. Comput., pp. 1499–1506 (2007)
29. Zheng, C., Li, L., Xu, F., Sun, F., Ding, M.: Evolutionary route planner for unmanned air vehicles. IEEE Trans. Robot. **21**(4), 609–620 (2005)