Aerospace Science and Technology 16 (2012) 47–55

Contents lists available at ScienceDirect

Aerospace Science and Technology

www.elsevier.com/locate/aescte

A new vibrational genetic algorithm enhanced with a Voronoi diagram for path planning of autonomous UAV

Y. Volkan Pehlivanoglu *,[1]

Air Force Academy, 34149 Istanbul, Turkey

ARTICLE INFO

Article history:
Received 26 March 2010
Received in revised form 3 February 2011
Accepted 19 February 2011
Available online 3 March 2011

Keywords:
Autonomous UAV
Voronoi diagram
Genetic algorithm
Path planning

ABSTRACT

A new optimization algorithm called multi-frequency vibrational genetic algorithm (mVGA) that can be used to solve the path planning problems of autonomous unmanned aerial vehicles (UAVs) is significantly improved. The algorithm emphasizes a new mutation application strategy and diversity variety such as the global random and the local random diversity. Clustering method and Voronoi diagram concepts are used within the initial population phase of mVGA process. The new algorithm and three additional GAs in the literature are applied to the path planning problem in two different three-dimensional (3D) environments such as sinusoidal and city type terrain models, and their results are compared. For both of the demonstration problems considered, remarkable reductions in the computational times have been accomplished.

1. Introduction

The number of applications that consider the use of autonomous UAVs is increased for civilian or military purposes. There are many activities that must be carried out by a UAV system to enable the execution of the autonomous navigation. These activities are mapping and modeling the environment, path planning, and flight control systems. Path planning is one of the most important problems in the navigation process. Its objective is to find out the optimal flight path starting from the departure location and ending in an arrival location in the proper duration during which UAV is able to accomplish the pre-arranged task and avoid the hostile threats. The selection of an appropriate algorithm in every stage of the path planning process is very important. Optimal path planning heavily relies on time consuming optimization techniques such as numerical computation. It is usually solved offline based on the known information before takeoff.

There are different groups of path planning algorithms in the literature. The first group is called skeleton approach. This approach is also called the roadmap, or highway approach [12]. The second group is cell decompositions. A world space is triangulated into a mesh. An optimized path graph is extracted from this mesh. Dijkstra's algorithm is usually used to search for the shortest path in the path graph. The third group is called the potential field approach, in which a scalar function is constructed from the information of obstacles and contractions. The path is determined by performing the steepest gradient descent on the potential function [4].

In recent years, a path planning problem is introduced to new heuristic or hybrid techniques such as fuzzy logic [9,16], neural network [1], ant colony optimization [7,18], Dijkstra's algorithm and ant system algorithm [25], the artificial potential field approach with simulated annealing [20,24], particle swarm optimization [23], genetic algorithm with simulated annealing [22], and genetic algorithms (GA). Each method has its own strength over others in certain aspects. However, GA has attracted significant attention because of ease of implementation for both continuous and discrete problems, no requirement for the continuity in response functions, efficient use of large numbers of parallel processors, and more robust solution generations to search for global or near global solutions. When we review GA based path planning studies, we see that the initial population generation is commonly based on random number operators. Additionally, in these studies, the classical crossover and mutation operators are mostly chosen to produce offspring individuals. For example, Xiao et al. [26] generated the initial population by using random numbers and used one-point crossover scheme together with a uniform mutation scheme in their evolutionary planner/navigator algorithm. Nikolos et al. [19] preferred to use breeder genetic algorithm including randomly generated initial individuals, one-point and heuristic crossover operators, and uniform and non-uniform mutation operators. Jia and Wagners [13] presented a parallel evolutionary algorithm and chose randomly generated initial population, one-point crossover scheme, and one- or two-point mutation techniques. Zhao and Murthy [27] used a random number operator and information based on problematic knowledge to generate the initial individuals. They also proposed to use a single-

* Tel.: +90 212 663 2490 (4361); fax: +90 212 662 8551.
E-mail address: vpehlivan@hho.edu.tr.
[1] Dr. and Major in Air Force Academy.

1270-9638/$ – see front matter
doi:10.1016/j.ast.2011.02.006

point crossover scheme and one-position mutation operator. Pehlivanoglu et al. [21] presented a new genetic algorithm called vibrational GA. In their study, the initial individuals are produced by using a random number operator, mated with BLX-α crossover operator, and mutated with vibrational mutation operator. Eun and Bang [6] studied the path planning problem of multiple unmanned aerial vehicles. They preferred to generate the initial population based on rough 2D Voronoi diagrams. They also used the uniform mutation operator and cycle crossover technique for the reproduction phase. Fu and Gao [10] offered to use additional genetic operators in addition to classical operators such as one-point crossover and uniform mutation operators. They also used a random number operator to impose the initial population. However, it often takes a long time for planners to escape from local optimal solutions once the evolution converges with a local optimal plan in genetic algorithms. In addition to the premature convergence problem, almost for all reported genetic algorithms, the individuals of the first population are assumed to be generated randomly. This is simple but will lead to large quantities of infeasible paths if used in path planning. Although some of these infeasible paths might become feasible solutions after certain genetic operations, they cause several problems such as more computational time or meaningless work. To overcome these issues, this article emphasizes a new Voronoi supported multi-frequency vibrational genetic algorithm (mVGA). Basic contributions of this study are the improvement of initial population by using fuzzy c-means clustering method and Voronoi diagram in three-dimensional (3D) space, and genetic algorithm enhancement by using multi-frequency vibrational mutation strategy.

To demonstrate, mVGA and three additional GAs are applied and compared in two different test cases: path planning problems in a 3D sinusoidal terrain model and in a 3D city type terrain modeling. Based on the results obtained, it is concluded that present multi-frequency vibrational genetic algorithm is efficient and fast genetic algorithm since it can successfully avoid all local optima within a shorter optimization cycle.

2. Methodology

2.1. Path representation

Path parameterization plays a key role since it defines the design variables. Bezier curves [8] have been widely adopted when computing smooth, dynamically feasible trajectories for UAVs. A smooth path is an important feature for UAV flight dynamics because aerial vehicles cannot fly on line segments like land vehicles. Additionally, smooth path prevents the vehicle from sharp turnings which would be harmful for the vehicle structure. The other advantage of employing Bezier curve is that the path can be represented using a relatively smaller number of parameters than using a complete geometric description of the path. This results in a small computational cost. Bezier curve is defined by

$$
\begin{aligned}
x_1(d) &= \sum_{i=0}^{n} B_i^n d^i (1-d)^{n-i} x_{1,i} \\
x_2(d) &= \sum_{i=0}^{n} B_i^n d^i (1-d)^{n-i} x_{2,i} \\
x_3(d) &= \sum_{i=0}^{n} B_i^n d^i (1-d)^{n-i} x_{3,i} \\
B_i^m &= n!/i!(n-1)! \qquad (1)
\end{aligned}
$$

where d is the step size parameter of the curve whose values vary uniformly between $[0, 1]$, n is the number of control points, $(x_{1,i}, x_{2,i}, x_{3,i})$ are the coordinates of the ith control point which define the path coordinates $(x_1(d), x_2(d), x_3(d))$. Coordinate transformation between aerial vehicle and the terrain model is not taken into consideration because the test cases are implemented in demonstrative terrain models.

2.2. Terrain modeling

Two different 3D surface representation methods: trigonometric function based terrain modeling and city surface based terrain representation are used to simulate the topology of the artificial terrain. The function of trigonometric based terrain modeling is given by

$$
\begin{aligned}
x_3(x_1, x_2) = {} & \sin(x_2 + b_1) + b_2 \sin(x_1) \\
& + b_3 \cos\bigl(b_4 \sqrt{(x_1)^2 + (x_2)^2}\bigr) + b_5 \cos(x_2) \\
& + b_6 \sin\bigl(b_6 \sqrt{(x_1)^2 + (x_2)^2}\bigr) + b_7 \cos(x_1) \qquad (2)
\end{aligned}
$$

where b_i are real constant numbers.

On the other hand, buildings which are squared prisms with different heights are used to simulate city type environment. It is possible to model the real city or city regions by using a city map and building height info. An example model is generated via three matrices based on the randomly determined building locations and heights. The buildings are defined by the following

$$
\begin{aligned}
& Bu_i^c|^{i=1,2,\ldots,N^B} = (x_{1,i}, x_{2,i}) \\
& [\, x_{1,i} \quad x_{2,i} \,]^T = \begin{bmatrix} rand[x_1^L, x_1^U] \\ rand[x_2^L, x_2^U] \end{bmatrix} \\
& Bu_i^w = rand\bigl[0, H^L\bigr] \qquad (3)
\end{aligned}
$$

where Bu_i^c is the center of the ith building, N^B is the total number of buildings, x^L and x^U are the boundaries for the city plane, Bu_i^w is the width of the ith building, and H^L is the height limitation for the buildings. N^B is taken as 160 in the applications.

Both terrain models are depicted in Fig. 1. Sinusoidal terrain model is a mathematically continuous model. City type terrain model is a discrete and discontinuous model.

2.3. Optimization algorithms

The new multi-frequency vibrational genetic algorithm (mVGA) is an improvement of the vibrational genetic algorithm (VGA), which is described in [11] and applied in [21]. It is an iterative algorithm for which a flow chart is presented in Fig. 2.

To describe the method mathematically, let s be the population size, L be the chromosome (or individual) dimension space, f be the objective function, and $\boldsymbol{C}_i$ be the ith chromosome including genes, $c_{j,i}(t)$, described in tth iteration:

$$
\begin{aligned}
& \boldsymbol{C}_i(t) = \bigl(c_{1,i}(t), c_{2,i}(t), \ldots, c_{L,i}(t)\bigr) \\
& c_{j,i}(t) \in R^3, \ i = 1, 2, \ldots, s \qquad (4)
\end{aligned}
$$

The second step is to evaluate the fitness of the current population via a defined cost function, f. Then, the rank fitness scaling and roulette selection procedure [17] for mating are determined. The elitism concept is applied next to ensure that the best objective function value within the population is not reduced from one generation to the next. The procedure for determining the elite fitness value, f^e, and elite individual, $\boldsymbol{C}^e$, is as follows:

$$
f^e(t) = \arg\min_{\boldsymbol{C}_i} f\bigl(\boldsymbol{C}_i(t)\bigr), \qquad \boldsymbol{C}^e(t) = \boldsymbol{C}_i(t) \qquad (5)
$$

$$
\boldsymbol{C}^e(t) = \begin{cases} \boldsymbol{C}^e(t-1) & \text{if } f^e(t) > f^e(t-1) \\ \boldsymbol{C}^e(t) & \text{if } f^e(t) < f^e(t-1) \end{cases} \qquad (6)
$$

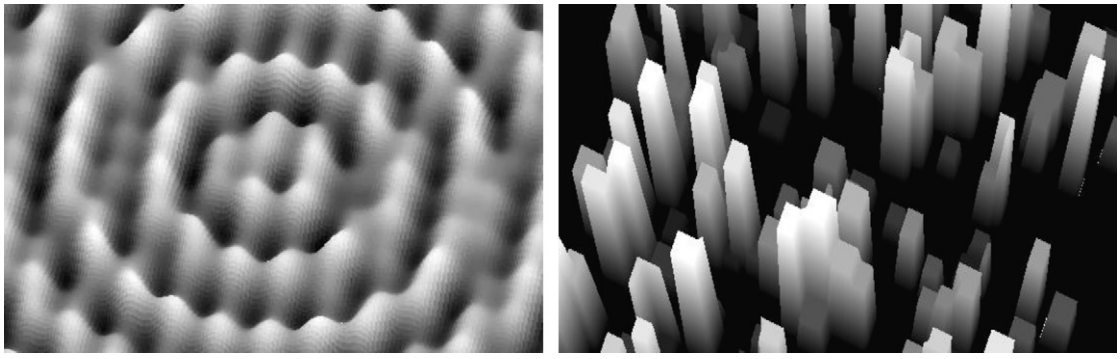

Fig. 1. Sinusoidal and city type terrain models.

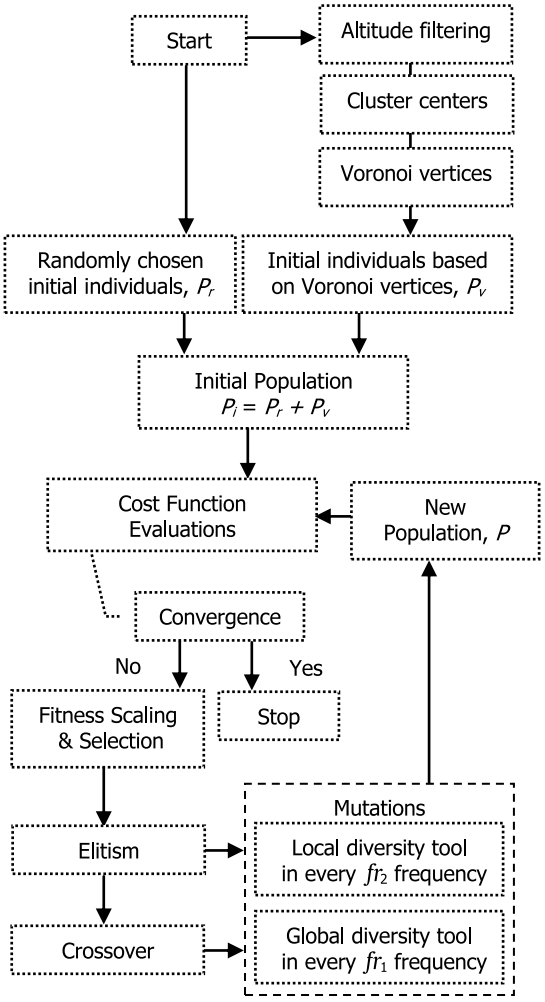

Fig. 2. Flow chart of Voronoi supported mVGA.

The crossover technique denoted by BLX-α and described in [5] with $\alpha = 0.5$ is applied for the individuals.

2.3.1. Mutation applications

The present mVGA mutation strategy is applied right after this crossover phase. At this step, there are two tools. As the first tool, the goal of the first mutation application is to provide a global random diversity in the population. For this reason, all the genes in all the chromosomes are mutated as follows:

$$c_{j,i}(t) = \begin{cases} c_{j,i}(t)[1 + w_1\beta_1(1-h)]_{j=1,2,\ldots,L}^{i=1,2,\ldots,s} \\ \quad \text{if } t = gfr_1,\ g = 1,2,\ldots \\ c_{j,i}(t) \quad \text{if } t \neq gfr_1,\ g = 1,2,\ldots \end{cases} \tag{7}$$

where fr_1 is the application frequency, β_1 is a user-defined amplitude parameter, h is a random real number between 0 and 1, and w_1 is a user-defined scale factor. Implementing the mutation starts from the first gene position of the first chromosome, and continues throughout the genes at the same positions in the other chromosomes. As a second tool, the goal of the second mutation application is to provide a local diversity in the neighborhood of the elite individual. In the applications, modified elite individuals, $\boldsymbol{c}^{m.e.}$, are generated as given below:

$$c_{j,i}^{m.e.}(t) = \begin{cases} c_{j,i}^{e}(t)[1 + w_2\beta_2(1-h)]_{j=1,2,\ldots,L}^{i=1,2,\ldots,s} \\ \quad \text{if } t = gfr_2,\ g = 1,2,\ldots \\ \emptyset \quad \text{if } t \neq gfr_2,\ g = 1,2,\ldots \end{cases} \tag{8}$$

where fr_2 is the application frequency, β_2 is the amplitude, w_2 is a user-defined scale factor. The modified elite individual is randomly located within the population as follows and applied I times:

$$\big(\boldsymbol{C}_k(t)\big)_j = \boldsymbol{C}^{m.e.}(t)_{j=1,2,\ldots,I}^{k=rand[1,s]} \tag{9}$$

The frequencies fr_1, fr_2, and I are user-defined constants and their typical values are 5, 3, and 2, respectively. We can figure the population in terms of t generations. Assume that fr_1 is equal to 5, fr_2 is equal to 3. For the first six, the generation includes the following populations:

t	1	2	3
P	$P_{In}(\boldsymbol{C}_i)\|^{i=1,2,\ldots,s}$	$P_{Cr}(\boldsymbol{C}_i)\|^{i=1,2,\ldots,s}$	$P_{Cr}(\boldsymbol{C}_i)\|^{i=1,2,\ldots,s-I}$ $P_{ML}(\boldsymbol{C}_i)\|^{i=1,2,\ldots,I}$
t	4	5	6
P	$P_{Cr}(\boldsymbol{C}_i)\|^{i=1,2,\ldots,s}$	$P_{MG}(\boldsymbol{C}_i)\|^{i=1,2,\ldots,s}$	$P_{Cr}(\boldsymbol{C}_i)\|^{i=1,2,\ldots,s-I}$ $P_{ML}(\boldsymbol{C}_i)\|^{i=1,2,\ldots,I}$

where P_{In} is the initial population, P_{Cr} is the population generated by crossover operations, P_{ML} is the population locally generated by the second mutation operator, P_{MG} is the population globally generated by the first mutation operator. At first, the population is composed of randomly generated initial individuals. In the second generation, the population only includes the individuals generated by crossover operations. The third generation corresponds to the second frequency; $t \leftarrow gfr_2$, $g = 1$. In the third generation, the population includes $(s - I)$ individuals generated by crossover operations and I individuals generated by the second vibrational mutation operator. We need to point out that the vibrational mutation operations are applied to individuals right after the crossover

Table 1
RGA and vibrational genetic algorithm features.

Mutation	T	Crossover	s
Uniform: R_m, 0.05 Vibrational: fr, 10; w, 1; β, 5	300	BLX-α	10
Fitness scaling	Elite	Selection	Run
Rank	1	Roulette	20

operations. Mutation operations are active on the individuals generated by crossover operations. Therefore, new elite based individuals replace randomly selected individuals generated by crossover operations. In the fourth generation, the population only includes the individuals generated by crossover operations. The fifth generation corresponds to the first frequency; $t \leftarrow gfr_1$, $g = 1$. In the fifth generation, the population only includes the individuals generated by the first vibrational mutation operator. We recall that these mutated individuals are the individuals generated by crossover operations in the fifth generation. The sixth generation corresponds to the second frequency; $t \leftarrow gfr_2$, $g = 2$. In the sixth generation, the population includes $(s - I)$ individuals generated by crossover operations and I individuals generated by the second vibrational mutation operator. After the mutation operations, a new population is evaluated via the cost function. The algorithm repeats all of the above steps as necessary until the convergence criteria are satisfied.

Apart from mVGA, two regular genetic algorithms (RGAs) are used in the test cases. In these algorithms, the same algorithm settings are used except for mutation operators. Gaussian and uniform mutation operators [17] are preferred to use in each one. In Gaussian mutation operator applications, the important control parameters are ξ which is a user-defined scale factor and γ which is another user-defined shrink factor. In uniform mutation applications, randomly selected I_M genes are mutated depending on the user-defined mutation ratio, R_m.

2.3.2. Effects of vibrational mutations

To see the effect of vibrational mutations on individuals, a simple test case is constructed. One of the benchmark test functions, Rastrigin test function, is selected as a test case and selected mutation operators including uniform and the first vibrational mutation operators are compared in terms of generation and individual positions. Rastrigin test function is a multi-modal function, and it is described as the following:

$$10L + \sum_{i=1}^{L} x_i^2 - 10\cos(2\pi x_i) \tag{10}$$

The search range and optimal values x^*, $f(x^*)$ are given in the following:

search range	x^*	$f(x^*)$
$[-5.12, 5.12]^L$	$[0, 0, \ldots, 0]$	0

The problem dimension is equal to 3 for visual observation of the individuals' trajectories. The features of RGA and vibrational GA are given in Table 1. The individuals' positions in 4D domain can be seen in Fig. 3.

In these plots, the x axis of an individual is ridden on the generations (t) with a certain range, δx. Totally 20 runs including 60,000 individual positions are plotted in each figure. In the upper part of the figure, we can see some drawbacks of the regular genetic process. It is clear that there are multiple local minimums and the algorithm prematurely converges to these minimums. Activated mutations based on uniform mutation operator cause random global perturbations within the population and it seems that they don't provide a beneficial diversity for the population. The other point is the relatively homogeneous distribution of the mutated individuals. On the other hand, the dispersions of the individuals of vibrational genetic process at each mutation period are clearly observed like fishbone during the generations. The dispersions become smaller while the iterations go on. The mutated individuals are heterogeneously distributed. There are also a few local minimum accumulations observed until a certain generation. However, these local convergences disappear after 130th generation. Almost all runs are converged to a global minimum with certain accuracies.

2.4. Initial population enhancement

The aim of the optimization process is to optimize design variables which represent the optimal path. The time of optimization process and also the result of it heavily depend on the initial points. Improper initial population may cause a time consuming process and premature convergence. Therefore, proper and robust initial population is very important for the optimization. Initial population is described as follows

$$P = P_r + P_v \tag{11}$$

where P_r is the random based population, and P_v is the remaining population. Random based population includes the individuals generated by using random number operator. The design variables are randomly generated within the terrain model bounds and altitude bound of UAV. However, the first and the last control point coordinates are fixed to the start point and the end point, respectively. The remaining population includes the individuals generated by using Voronoi diagram. Voronoi diagrams are used to find threat avoiding paths. However, Voronoi diagrams are efficient in two-dimensional environments; therefore, they don't care about the altitude restrictions. Additionally, these diagrams include straight lines and UAV cannot fly on these straight lines due to flight kinematics. To solve 2D plane problem, Krozel and Andrisani [15] studied to polygonize the counter data of mountains. Vertices from mountain polygon obstacles are used as Delaunay point locations to model the obstacles. To solve line segments problems, Jung and Tsiotras [14] used B-spline path templates. In their study, in conjunction with the high-level path planner, the proposed algorithm finds the corresponding local path segments and stitches them together, while preserving the smoothness of the composite curve. As a high-level path planner, they used a cell decomposition algorithm. Differently from these studies, we propose a simpler method. Voronoi diagram needs Delaunay points. To get these points, we can use a simple approach. Assume that we have a finite set of data which is described below

$$S(x_i), \quad x_i \in R^3, \quad i = 1, 2, \ldots, N^S \tag{12}$$

where N^S is the total number of surface points. $S(x_i)$ is depicted in Fig. 4(a). This data set can be partitioned by using A^L, the altitude limitation of UAV. Simply,

$$D(x_j), \quad x_j \in R^3, \quad i = 1, 2, \ldots, N^D$$
$$\text{if } x_{i,3} > A^L|^{i=1,2,\ldots,N^S}, \quad x_j = x_i \tag{13}$$

where N^D is the total number of dangerous points. $D(x_j)$ is depicted in Fig. 4(b). This new data set, $D(x_j)$ can be clustered using fuzzy c-means (FCM) clustering method [3]. FCM is formulated as follows:

$$\min_{\substack{x_k \in D(x) \\ x_k \in R^2 \\ u_k \in [0,1]}} J_{FCM} = \sum_{k=1}^{N^D} \sum_{i=1}^{c} (u_{ik})^m (\boldsymbol{x}_k - \boldsymbol{\vartheta}_i)^2 \tag{14}$$

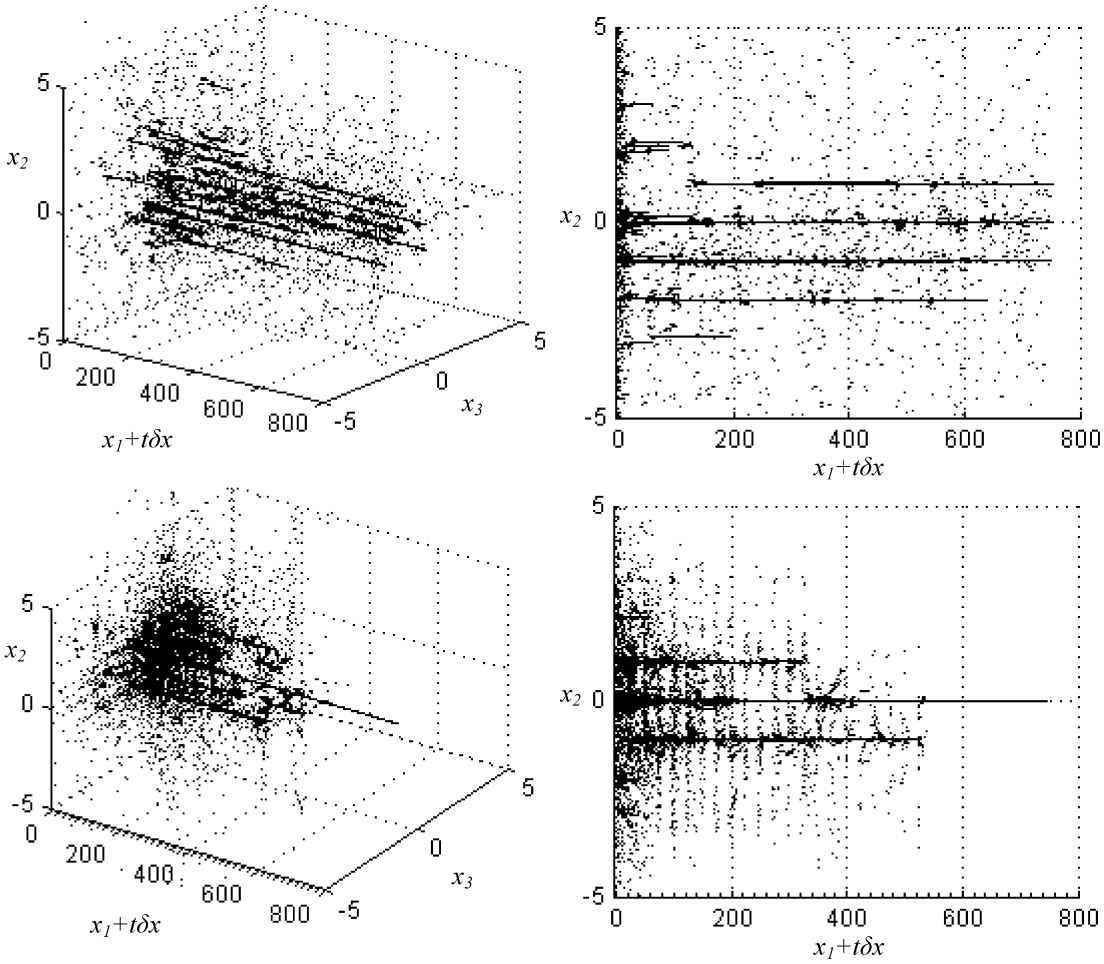

Fig. 3. Individuals' positions of regular and vibrational genetic algorithm processes.

subject to

$$\sum_{i=1}^{N^D} u_{ik} = 1$$

u and $\boldsymbol{\vartheta}$ can be computed as:

$$u_{ik} = 1 \Big/ \sum_{j=1}^{c} \left(\frac{d_{ik}}{d_{jk}} \right)^{\frac{2}{m-1}}, \quad d_{ik} = (\boldsymbol{x}_k - \boldsymbol{\vartheta}_i)^2 \tag{15}$$

$$\boldsymbol{\vartheta}_i = \sum_{k=1}^{N^D} (u_{ik})^m \boldsymbol{x}_k \Big/ \sum_{k=1}^{N^D} (u_{ik})^m \tag{16}$$

where i is the cluster, c is the number of clusters, $\boldsymbol{\vartheta}_i$ is a cluster center vector of the cluster i, m is a fuzzification number which is equal to 2, u_{ij} is the membership value of $\boldsymbol{x}_k$ in the ith cluster, and d_{ik} is the Euclidean distance between $\boldsymbol{x}_k$ and $\boldsymbol{\vartheta}_i$. c is a user-defined parameter and predicted before the FCM process. Example cluster centers are depicted in Fig. 4(c). At the end of the clustering process new data set, $C(\boldsymbol{\vartheta})$ is constructed as the following

$$C_i(\boldsymbol{\vartheta}), \quad \boldsymbol{\vartheta} \in R^2 \Big|^{i=1,2,\ldots,c} \tag{17}$$

$C(\boldsymbol{\vartheta})$ is used as Delaunay points to construct Voronoi diagram. We denote by C a set of $n \geqslant 3$ point sites $p, q, \ldots$ in the plane based on A^L. For points p and r let

$$e(p,r) = \sqrt{\left(\vartheta_1^p - \vartheta_1^r\right)^2 + \left(\vartheta_2^p - \vartheta_2^r\right)^2} \tag{18}$$

denote their Euclidean distance. By $|pq|$ we denote the line segment from p to q. For p, q let

$$B(p,q) = \left\{ r \mid e(p,r) = e(q,r) \right\} \tag{19}$$

be the bisector of p and q. $B(p,q)$ is the perpendicular line through the center of the line segment $|pq|$. It separates the halfplane

$$E(p,q) = \left\{ r \mid e(p,r) < e(q,r) \right\} \tag{20}$$

containing p from the halfplane $E(q,p)$ containing q. We call

$$VR(p,F) = \bigcap_{q \in C,\, q \neq p} Ep,q \tag{21}$$

the Voronoi region of p with respect to F. Finally, the Voronoi diagram of F is defined by

$$V(F) = \bigcup_{p,q \in F,\, q \neq p} \overline{VR(p,F)} \cap \overline{VR(q,F)} \tag{22}$$

By definition, each Voronoi region $VR(p,S)$ is the intersection of $n-1$ open halfplanes containing the site p. The common boundary of two Voronoi regions belongs to $V(F)$ is called a Voronoi edge. The endpoints of Voronoi edges are called Voronoi vertices [2].

A Voronoi diagram is constructed from the known cluster center locations. However, there are some defective cluster centers due to the prediction of the number of the cluster centers. These centers cause unsafe Voronoi edges. But this is just the beginning of the optimization process and genetic process eliminates these

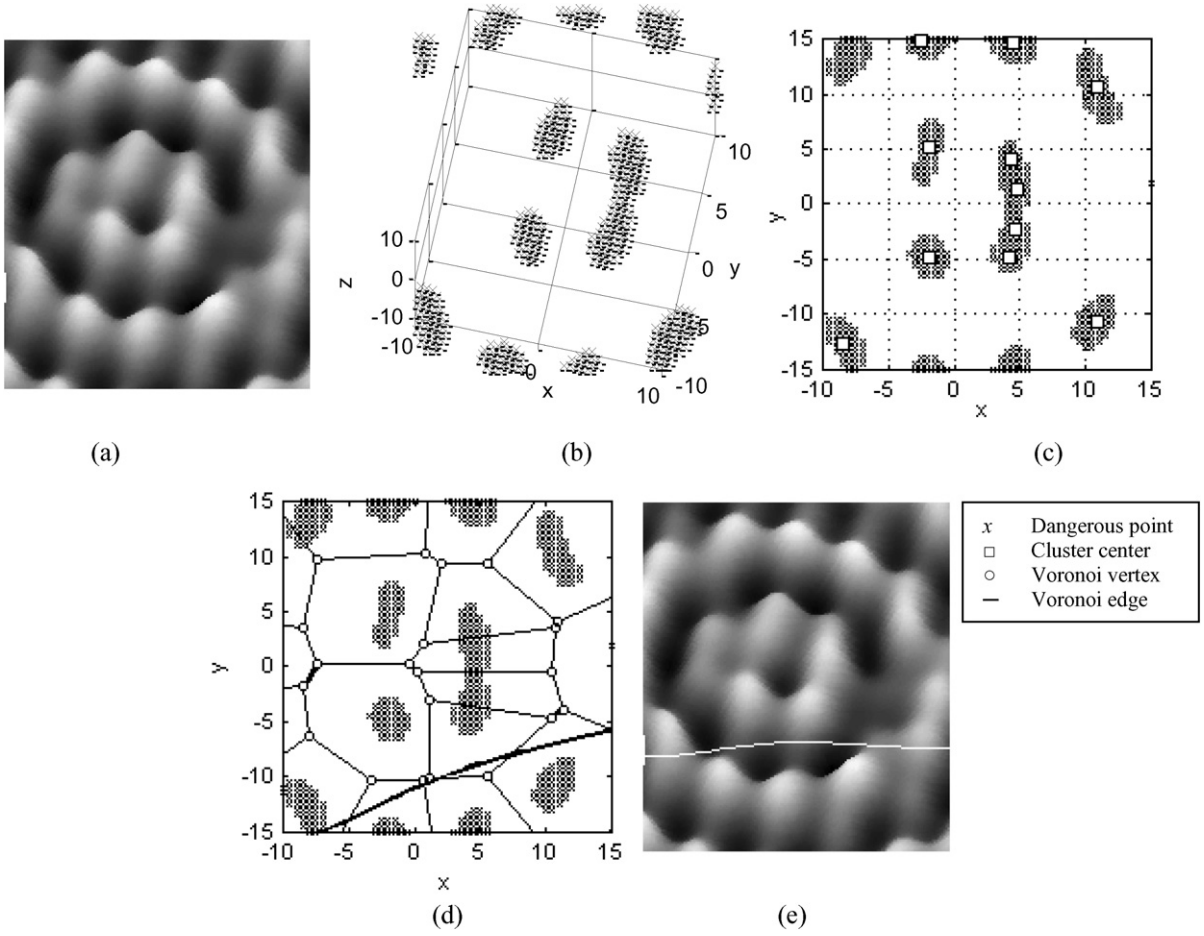

Fig. 4. (a) A sample surface data set, $S(x)$; (b) The data set, $D(x)$, for dangerous points; (c) Cluster centers for $D(x)$; (d) An example Voronoi diagram based on $C(\vartheta)$; (e) An example candidate path (white colored line) on terrain model.

deficiencies because of penalties related to constraints during the optimization. An example of constructed Voronoi diagram is depicted in Fig. 4(d). After constructing a Voronoi diagram, the start location and the end location need to be connected to the Voronoi vertices. We simply connect the start location and the end location to the nearest several nodes of the Voronoi diagram. Similarly, proper Voronoi vertices between start and end points of the path are selected and these vertices are used as control points of the Bezier curves. An example implementation is depicted in Fig. 4(e).

These constructed path solutions are Voronoi based initial individuals. The number of Voronoi vertices on the constructed path determines the number of control points of Bezier curve. The third coordinate $(x_{3,i})$ is determined as A^L.

2.5. Fitness function

The evaluation function of an individual measures the cost of a candidate path. The fitness function is designed to accommodate three different terms: minimize the distance flown, maintain a smooth trajectory preventing sharp turns, and satisfy the clearance providing the safe distance for UAV from terrain. We have proposed a linear combination of these three factors. The general description of the optimization problem is given below

$$\min f = \sum_{i=1}^{3} a_i G_i \tag{23}$$

subject to

$$x_3^{curve} - x_3^{surface} < S_d$$

$$\theta < \theta_S$$

where G_i is the term connected to various concerns, a_i is weighting constant. The first concern is the length of the curve G_1 and is calculated by the given expression

$$G_1 = \sum_{i=1}^{dn-1} \left[(x_{1,i+1} - x_{1,i})^2 + (x_{2,i+1} - x_{2,i})^2 + (x_{3,i+1} - x_{3,i})^2 \right]^{1/2} \tag{24}$$

where dn is the curve discretization number; $x_{1,i}$, $x_{2,i}$, and $x_{3,i}$ are the discrete coordinates of the path curve. The second concern, G_2, is the passing ratio of the curve through the terrain boundary and is calculated by the given expression:

$$G_2 = \sum_{i=1}^{dn} \quad \text{if } x_{3,i}^{curve} - x_3^{surface} < S_d, \quad \text{punishment} + 1 \tag{25}$$

where S_d is a safe distance determined by user, x_3^{curve} is the discrete path curve coordinate, and $x_3^{surface}$ is the terrain model coordinate. This expression penalizes the curve that passes through the solid boundary. So, the penalty is proportional to the number of discretized curve points located under the solid surface. The third concern, G_3, is the curvature angle ratio of the curve which is calculated by the given expression:

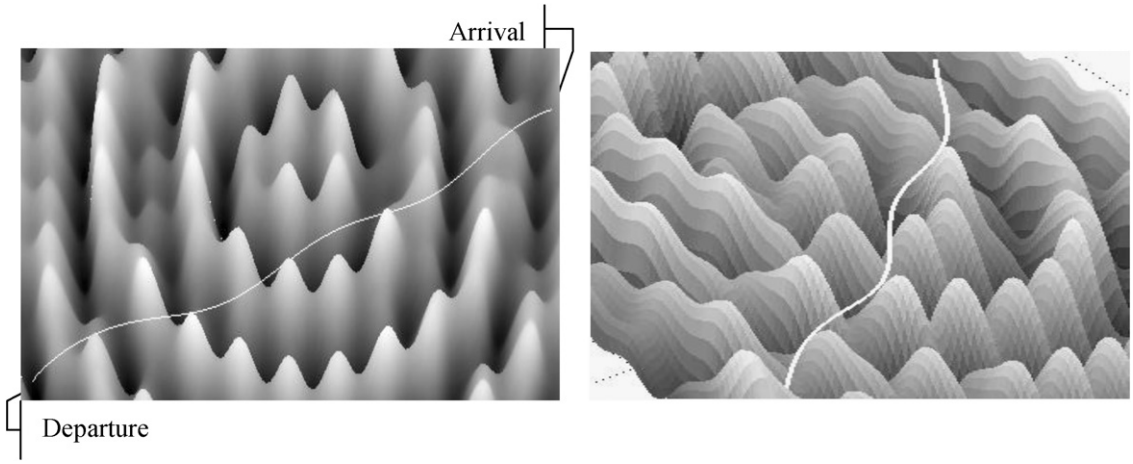

Fig. 5. An example path constructed by mVGA (white color) in different views.

Table 2
Genetic algorithms' features.

Method	Mutation	P_r/P_v	Run
RGA_1	Gaussian: ξ, 0.5; γ, 0.75	10/0	20
RGA_2	Uniform: R_m, 0.05	10/0	
VGA	Vibrational: fr_1, 5; w_1, 1; β_1, 0.5	10/0	
mVGA	Vibrational: fr_1, 5; w_1, 1; β_1, 0.5 fr_2, 2; w_2, 1; β_2, 2.5	10/0	
$mVGA^V$	Vibrational: fr_1, 5; w_1, 1; β_1, 0.5 fr_2, 2; w_2, 1; β_2, 2.5	9/1	

Fitness scaling	Selection	Elite	T	Crossover
Rank	Roulette	1	100	BLX-α; α, 0.5

$$G_3 = \sum_{i=1}^{n-1} \quad \text{if } \theta_{i,i+1} < \theta_S, \quad \text{punishment} + 1 \tag{26}$$

where $\theta_{i,i+1}$ is the angle between the extension of the line segment connecting Bezier control points i and $i+1$, θ_S is the safe turning angle determined by user for the UAV. This concern is designed to prevent the aerial vehicle from exceeding the lateral and vertical acceleration limits, because the flight envelope determines the maximum radius of turns for flying objects. The weight constants, a_i, are determined experimentally. In test cases, a_2 is selected as a high number.

3. Test cases and results

The new improved algorithm is tested and compared in different environments. The first environment is selected as a sinusoidal terrain model. The second environment is selected as a city type surface modeling. For both test cases, the assignment is to fly from a departure point to an arrival point under described conditions.

3.1. Sinusoidal terrain results

The test case is given in Fig. 5. The mission of UAV is to depart from the departure point and arrive at the arrival point under certain conditions. Five genetic algorithms are tested in this case. These algorithms are described in accordance with the mutation type. The features of these algorithms are tabled in Table 2. The first two algorithms are labeled as regular genetic algorithms such as RGA_1 and RGA_2. In these algorithms, classical mutation strategy applications are performed including Gaussian and uniform mutations. VGA is named vibrational genetic algorithm and the first vibrational mutation operator given in Eq. (7) is applied to get global diversity in the population. mVGA is labeled as multi-frequency vibrational genetic algorithm and both vibrational mutation operators given in Eqs. (7) and (8) are applied to provide global and local diversity in the population. Additionally, mVGA is supported by initial population enhancement and put in as the fifth algorithm called $mVGA^V$. The data for fitness function is given in Table 3.

Table 3
Fitness function data.

S_D	θ_S	A^L	a_1	a_2	a_3
0.05	60°	5	5	10	1

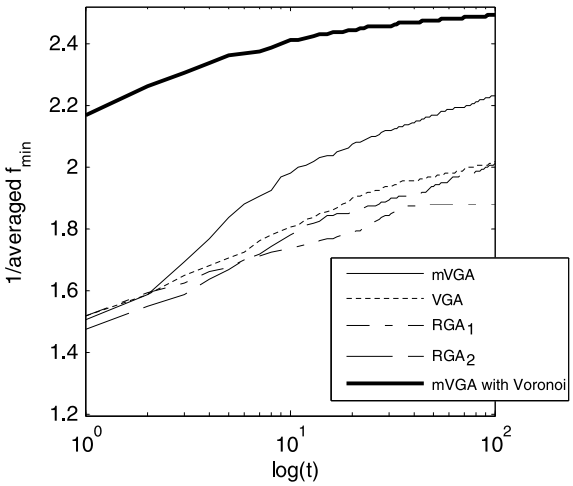

Fig. 6. The fitness values of algorithms versus generations.

All algorithms are run 20 times and the averaged best individual fitness function values versus generations are plotted in Fig. 6. According to this figure, mVGA outperformed VGA and the regular genetic algorithms. VGA reaches the value of 0.002 (1/averaged f_{min}) at 83rd generation, while RGA_2 reaches the same value at 100th generation. Therefore, VGA decreases 17% the required generation number. mVGA looks very efficient in this case. mVGA reaches the value of 0.002 at 12th generation. It decreases by 78% the required generation number.

Fig. 6 also shows that the enhancement of initial population causes a significant improvement in fitness function values. It improves the algorithm almost 100%. mVGA reaches the value of 0.0022 at 100th generation, while $mVGA^V$ starts with the value of 0.0021.

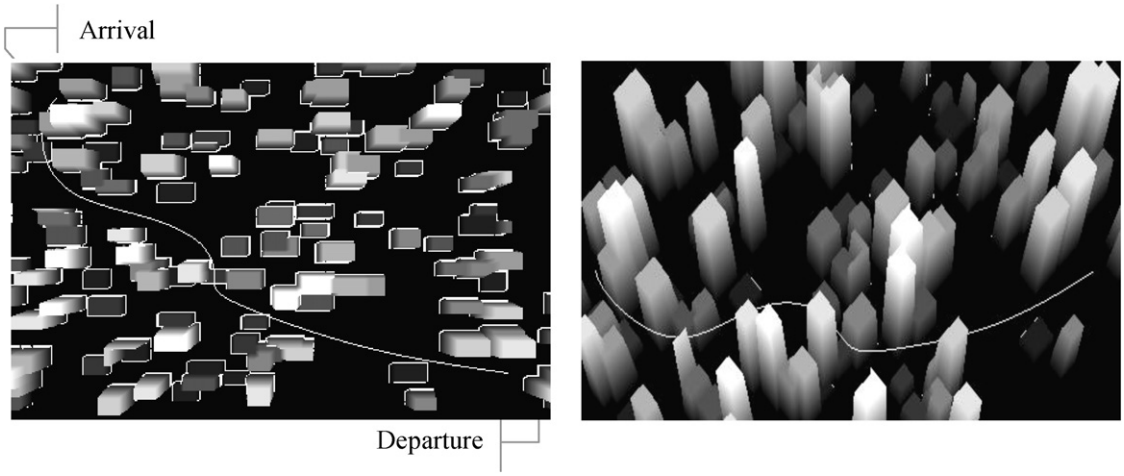

Fig. 7. An example constructed path (white color) with different views by mVGA.

Table 4
Genetic algorithms' features.

Method	Mutation	P_r/P_v	Run
VGA	Vibrational: fr_1, 5; w_1, 1; β_1, 0.5	10/0	20
mVGA	Vibrational: fr_1, 5; w_1, 1; β_1, 0.5 fr_2, 2; w_2, 1; β_2, 2.5	10/0	
mVGA^V	Vibrational: fr_1, 5; w_1, 1; β_1, 0.5 fr_2, 2; w_2, 1; β_2, 2.5	9/1	

Fitness scaling	Selection	Elite	T	Crossover
Rank	Roulette	1	100	BLX-α; α, 0.5

Table 5
Fitness function data.

S_D	θ_T	A^L	a_1	a_2	a_3
0.05	60°	5	5	10	1

3.2. City type terrain results

The test case is given in Fig. 7. Similar to the previous test case, the mission of UAV is to depart from a departure point and arrive at an arrival point above the city under certain conditions. Three genetic algorithms including VGA, mVGA, and mVGA^V are tested in this case. The features of these algorithms are tabled in Table 4. The data for fitness function is given in Table 5.

All algorithms are run 20 times and the averaged best individual fitness function values versus generations are plotted in Fig. 8.

According to this figure, mVGA outperformed VGA. VGA reaches the value of 0.0092 (1/averaged f_{min}) at 100th generation, while mVGA reaches the same value at 50th generation. Therefore, mVGA decreases by 50% the required generation number. The efficiency of mVGA is a bit lower than the previous case (sinusoidal terrain case). This may originate from the terrain data. The sinusoidal terrain model is an exact model and the computation of fitness values is based on continuous functions. However, the city terrain model is a discrete model. Therefore, it is considered to be the reason for the lower efficiency. Fig. 8 also shows that the effect of Voronoi based individually on the process is very significant. mVGA reaches the value of 0.0083 at 100th generation, while mVGA^V reaches the same value at 11th generation. This means an 89% decrease in the required generation number.

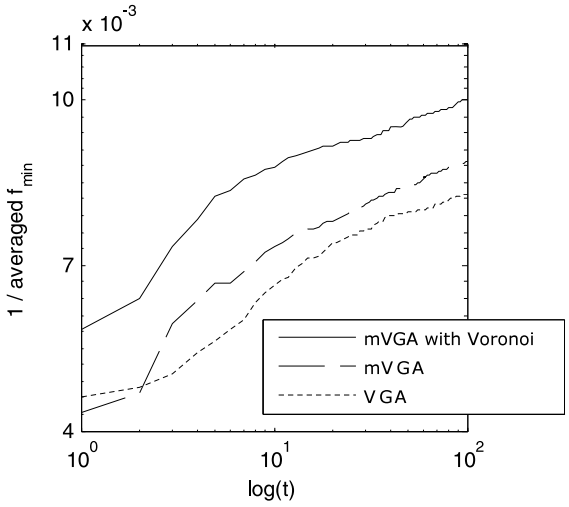

Fig. 8. The fitness values of algorithms versus generations.

4. Conclusions

Although genetic algorithms are global search methods, they sometimes suffer from premature convergence. The main reason for premature convergence is the lack of diversity in the population. Instead of using classical mutational applications, this article preferred to use the periodic mutation applications based on vibrational mutation operators, and showed that these applications provide classified but efficient diversity in the population.

The principal role of this multi-frequency approach is to answer the question of which individuals should be mutated and when they should be mutated. The first mutation operator was applied to all genes of the whole population and this application provided global but random diversity in the population. The global diversity afforded a chance for the population to escape from all local optima. The second mutation operator was specifically applied to the genes of an elite individual in the population. This application provided local diversity leading to a fast convergence.

In addition to a new mutation strategy, by using altitude filtration and fuzzy c-means clustering method, the Voronoi diagrams are constructed and some initial individuals are generated based on Voronoi vertices. By doing this, the initial population is significantly improved, and thus the convergence is seriously accelerated.

From the results obtained, it is concluded that a Voronoi supported multi-frequency vibrational genetic algorithm is an efficient and fast algorithm since it avoided all local optima within relatively short optimization cycles.

References

[1] K. Althoefer, Neuro-fuzzy motion planning for robotic manipulators, Ph.D. thesis, King's College, London, 1997.

[2] F. Aurenhammer, R. Klein, Voronoi diagrams, in: J.-R. Sack, J. Urrutia (Eds.), Handbook of Computational Geometry, North-Holland, Amsterdam, Netherlands, 2000, Ch. 5, pp. 201–290.

[3] J.C. Bezdec, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York, 1981.

[4] G. Chen, D. Shen, J. Cruz, C. Kwan, S. Riddle, S. Cox, C. Matthews, A novel cooperative path planning for multiple aerial platforms, AIAA-2005-6948, 2005.

[5] L.J. Eshelman, J.D. Schaffer, Real coded genetic algorithms and interval schemata, in: Foundations of Genetic Algorithms, vol. 2, Morgan Kaufmann Publishers, 1993, pp. 187–202.

[6] Y. Eun, H. Bang, Cooperative task assignment/path planning of multiple unmanned aerial vehicles using genetic algorithms, Journal of Aircraft 46 (1) (2009) 338–343.

[7] X. Fan, X. Luo, S. Yi, S. Yang, H. Zhang, Optimal path planning for mobile robots based on intensified ant colony optimization algorithm, in: Proc. of IEEE Int. Conf. Robotics, Intelligent Systems and Signal Processing, vol. 1, 2003, pp. 131–136.

[8] G. Farin, Curves and Surfaces for Computer Aided Geometric Design; A Practical Guide, Academic Press, Inc., 1993, p. XVI.

[9] T.P. Fries, Fuzzy genetic motion planning under diverse terrain conditions for autonomous robots, in: Proc. Circuits, Signals, and Systems, 2004, pp. 504–508.

[10] X. Fu, X. Gao, Genetic algorithm with adaptive immigrants for dynamic flight path planning, in: IEEE Conf. Intelligent Computing and Intelligent Systems, vol. 1, 2010, pp. 630–634.

[11] A. Hacioglu, I. Ozkol, Vibrational genetic algorithm as a new concept in aerodynamic design, Aircraft Engineering and Aerospace Technology 74 (3) (2002) 228–236.

[12] J.Y. Hwang, J.S. Kim, S.S. Lim, K.H. Park, A fast path planning by path graph optimization, IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans 33 (2003) 121–129.

[13] D. Jia, J. Vagners, Parallel evolutionary algorithms for UAV path planning, in: AIAA 1st Intelligent Systems Technical Conf., Chicago, Illinois, 2004, AIAA 2004-6230.

[14] D. Jung, P. Tsiotras, Online path generation for small unmanned aerial vehicles using B-spline path templates, AIAA 2008-7135, 2008.

[15] J. Krozel, D. Andrisani, Navigation path planning for autonomous aircraft: Voronoi diagram approach, Journal of Guidance: Engineering Notes (Nov.–Dec. 1990) 1152–1154.

[16] Y. Lee, Y. Kim, L.J. Kohou, An intelligent collision avoidance system for autonomous underwater vehicles using fuzzy relational products, Information Sciences – Informatics and Computer Science: An International Journal 158 (1) (2004) 209–232.

[17] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer, Berlin, 1996.

[18] C. Mou, W. Qing-xian, J. Chang-sheng, A modified ant optimization algorithm for path planning of UCAV, Applied Soft Computing 8 (4) (2007) 1712–1718.

[19] I.K. Nikolos, K.P. Valavanis, N.C. Sourveloudis, A.N. Kostaras, Evolutionary algorithm based offline/online path planner for UAV navigation, IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics 33 (6) (2003) 898–912.

[20] M.G. Park, M.C. Lee, Experimental evaluation of robot path planning by artificial potential field approach with simulated annealing, in: Proc. of the 41st SICE Annual Conf., vol. 4, 2006, pp. 2190–2195.

[21] Y.V. Pehlivanoglu, O. Baysal, A. Hacioglu, Path planning for autonomous UAV via VGA, Aircraft Engineering and Aerospace Technology: An International Journal 79 (4) (2007) 352–359.

[22] S. Piao, B. Hong, Path planning of robot using genetic annealing algorithm, in: Proc. of the 4th World Congress on Intelligent Control and Automation, vol. 1, 2002, pp. 493–495.

[23] M. Saska, M. Macas, L. Preucil, L. Lhotska, Robot path planning using particle swarm optimization of Ferguson splines, in: Conf. Emerging Technologies and Factory Automation, IEEE, 2006, pp. 833–839.

[24] F.J. Sharifi, D. Vinke, Integration of the artificial potential field approach with simulated annealing for robot path planning, in: Proc. of IEEE Int. Sym. Intelligent Control, 1993, pp. 536–541.

[25] G. Tan, H. He, A. Sloman, Global Optimal Path Planning for Mobile Robot Based on Improved Dijkstra's Algorithm and Ant System Algorithm, vol. 13, Central South University of Technology, China, 2006, pp. 80–86.

[26] J. Xiao, Z. Michalewicz, L. Zhang, K. Trojanowski, Adaptive evolutionary planner/navigator for mobile robots, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 18–28.

[27] L. Zhao, V.R. Murthy, Optimal flight path planner for an unmanned helicopter by evolutionary algorithms, AIAA 2007-6741, 2007.