Chapter 10: Evolutionary Algorithm Based Path Planning for Multiple UAV Cooperation[1]

This Chapter describes an evolutionary based off-line / on-line path planner for cooperating Unmanned Aerial Vehicles (UAVs). It considers the environment characteristics, the flight envelope and mission constraints of cooperating UAVs. The scenario under consideration assumes that several UAVs are launched from the same or different but known initial locations. Then, the main goal is to produce 3-D trajectories that ensure a collision free operation with respect to mission constraints. The path planner produces curved routes that are represented by 3-D B-Spline curves. Two types of path planner are discussed: i) the off-line planner that generates collision free paths in environments with known characteristics and flight restrictions; ii) the on-line planner, based on the off-line one, that generates collision free paths in unknown static environments by using acquired information from the UAV on-board sensors. This information is exchanged between cooperating UAVs in order to maximize knowledge of the environment. For each UAV, the on-line planner generates rapidly a near optimum path that guides the vehicle safely to an intermediate position within the already scanned territory, taking into account mission and cooperation objectives and constraints. The process is repeated for each UAV until the final position is reached by one of them. Then, each remaining UAV uses the acquired information about the environment in order to compute a curved path that connects its current position to the final one. Both off-line and on-line path planning problems are formulated as optimization problems, with a differential evolution algorithm serving as the optimizer.

10.1 Introduction

Path planning refers to the generation of a space path between an initial location and the desired destination, with an optimal or near-optimal per-

[1] Written by I. K. Nikolos, N. C. Tsourveloudis, K. P. Valavanis

Kimon P. Valavanis (ed.), Advances in Unmanned Aerial Vehicles, 309–340.

formance under specific constraints [1]. A path planning algorithm may produce different candidate plans, which should be compared and evaluated based on specific criteria. Such criteria are generally related to *feasibility* and *optimality* of the path generation. The first criterion relates to derivation of a plan that moves safely a UAV (an object) to its final state, without taking into account the quality of the produced plan. The second criterion refers to derivation of optimal, yet feasible, paths, with optimality defined according to the problem under consideration [2]. However, searching for optimal paths is not a trivial task; in most cases this requires excessive computational time; in some cases even computation of just one feasible path is a rather involved task. Therefore, the search focuses mostly on suboptimal or just feasible solutions.

Compared to the path-planning problem in other application areas, path planning for UAVs requires special characteristics that need be considered [3] [4] [5]: i) feasibility, which refers to limitations from using UAVs, such as limited endurance and range, minimum turning angle, minimum and maximum speed, etc; ii) stealth, in order to minimize the probability of detection by hostile sensors; iii) acceptable performance related to assigned mission, which imposes special requirements, including maximum climbing / diving angle, minimum and/or maximum flying altitude, etc; iv) real-time implementation, which asks for computationally efficient algorithms, and, v) cooperation between UAVs in order to maximize the possibility of mission accomplishment.

Cooperation between UAVs has recently gained increased interest as systems of multiple vehicles engaged in cooperative behavior show specific benefits compared to a single vehicle [6]. Such systems of cooperating UAVs may be used for fire fighting applications, military missions, search and rescue scenarios or exploration of unknown environments (space-oriented applications).

In order to establish a robust framework for efficient and reliable cooperation of multiple UAVs, several issues must be addressed: i) the task assignment problem that relates to the number of UAVs required to perform a mission, with predefined order, on a number of targets; ii) the path planning problem to provide feasible, flyable and near optimal trajectories that connect starting to destination points; iii) the exchange of information between cooperating UAVs, and data fusion expected to enhance team effectiveness; iv) cooperative sensing of the targets defined as how to 'use' UAV sensors in terms of their locations to achieve optimal estimation of the state of each target [7], and, v) cooperative sensing of the environment for better situation awareness.

10.1.1 Related Work

UAV path planning algorithms were initially derived for solving the single vehicle case. However, the continuously increasing interest for cooperating UAVs has resulted in development of algorithms that take into account special characteristics of multi-UAV missions.

Path planning problems are computationally demanding multi-objective multi-constraint optimization problems [8]. Problem complexity increases when multiple UAVs are used. Several approaches have been reported for coordinated route planning, such as Voronoi diagrams [9], mixed integer linear programming [10] [11], and dynamic programming [12] formulations.

In [9] a group of UAVs is required to transition through a number of known target locations, with a number of threats present in the region of interest. Some threats were known a priori, while some others 'popped up' or became known only when a UAV flew near them. The motion planning problem was decomposed into a waypoint path planner and a dynamic trajectory generator. The path planning problem was solved via a Voronoi diagram and Epstein's *k*-best paths algorithm. The trajectory generator problem was solved via a real-time nonlinear filter that explicitly accounted for the dynamic constraints of the vehicle and modified the initial path.

In [13], the motion planning problem for a limited resource of Mobile Sensor Agents (MSAs) was investigated in an environment with a number of targets larger than the available MSAs. The problem, being a combination of problems of sensor resource management and robot motion planning, was formulated as an optimization one with the objective to minimize the average time duration between two consecutive observations of each target.

Computational intelligence methods, such as Neural Networks [14], Fuzzy Logic [15] and Evolutionary Algorithms (EAs) [5] [16] have been successfully used to derive trajectories for guiding mobile robots in known, unknown or partially known environments.

Besides their computational cost, EAs are considered as a viable candidate to solve path planning problems effectively, because of their high robustness compared to other existing directed search methods, their ease of implementation in problems with a relatively high number of constraints, and their high adaptability to the special characteristics of the problem under consideration [16].

EAs have been successfully used in the past for the solution of the path finding problem in ground based or sea surface navigation [17]. A common practice was to model the path using straight line segments, connecting successive waypoints. In the case of partially known or dynamic envi-

ronments a feasible and safe trajectory was planned off-line by the EA and the algorithm was used on-line whenever unexpected obstacles were sensed [18] [19]. EAs have been also used for solving the path finding problem in a 3-D environment for underwater vehicles, assuming that the path is a sequence of cells in a 3-D grid [20] [21].

In [5] a route planner for UAVs was proposed based on evolutionary computation. The generated routes enabled the vehicles to arrive at their destination simultaneously by taking into account the exposure of UAVs to potential threats. The flight route consisted of straight-line segments, connecting the waypoints from the starting to the goal points. A real coded chromosome representation with certain design variables and state variable were used. These variables provided information on the feasibility of the corresponding way point and the route segment composed of these points. The cost function penalized the route length, high altitude flights or routes that come dangerously close to known ground threats. The imposed constraints on route segments were relevant to minimum route leg length, maximum route distance, minimum flying height, maximum turning angle, maximum climbing/diving angle, simultaneous arrival at target location and no collision between vehicles.

In [22] a multi-task assignment problem for cooperating UAVs was formulated as a combinatorial optimization problem; a Genetic Algorithm was utilized for assigning the multiple agents to perform different tasks on multiple targets. Integer encoding was used for the chromosomes, which were composed of two rows; the first row presented the assignment of a vehicle to perform a task on the target appearing on the second row. The proposed algorithm solved the problem efficiently, while taking into account requirements, such as task precedence and coordination, timing constraints, and flyable trajectories.

In [16] an EA based framework was utilized to design an off-line / on-line path planner for UAVs autonomous navigation. The path planner calculated a curved path line, represented using B-Spline curves in a 3-D terrain environment; the coordinates of B-Spline control points served as design variables. The off-line planner produced a single B-Spline curve that connected the starting and target points with a predefined initial direction. The on-line planner gradually produced a smooth 3-D trajectory aiming at reaching a predetermined target in an unknown environment; the produced trajectory consisted of smaller B-Spline curves smoothly connected with each other. For both off-line and on-line planners, the problem is formulated as an optimization one; each objective function is formed as the weighted sum of different terms, which take into account the various objectives and constraints of the corresponding problem. Constraints are formulated using penalty functions.

10.1.2 Problem Definition and Proposed Solution

The main scenario considered in this Chapter is the following: A number of UAVs are launched from the same or different but known initial locations with predefined initial directions. The main objective is to derive 3-D trajectories, represented by 3-D B-Spline curves, which connect the initial locations with a single destination location and ensure a collision free operation with respect to mission constraints. The UAVs are assumed to be equipped with a set of on-board sensors, including radar, GPS (Global Positioning System) / DGPS (Differential GPS), INS (Inertial Navigation System) and gyroscopes, through which they sense their surroundings and position. Each vehicle is assumed to be a point and its actual size is taken into account by equivalent obstacle – ground growing.

Two types of path planner are discussed: The off-line planner that generates collision free paths in environments with known characteristics and flight restrictions. The on-line planner, being an extension of the off-line one based on already reported research in [16]. Knowledge of the environment is gradually acquired through the on-board sensors that scan the area within a certain range from each UAV. This information is exchanged between cooperating UAVs in order to maximize sensor effectiveness. The on-line planner rapidly generates a near optimum path for each UAV that will guide the vehicle safely to an intermediate position within the known territory, taking into account the mission and cooperation objectives and constraints. The process is repeated until the corresponding final position is reached by an UAV. Then, each one of the remaining UAVs uses acquired information about the environment and the off-line planner output to compute a path that connects its current position to the final destination. Both path planning problems are formulated as optimization (minimization) problems, where specially constructed functions take into account mission and cooperation objectives and constraints, with a Differential Evolution algorithm to serve as the optimizer.

The rest of the Chapter is organized as follows: Section 10. 2 discusses B-Spline and differential EA fundamentals; the solid terrain formulation, used for experimental simulations, is also presented. The off-line path planner for a single UAV is briefly discussed in Section 10.3. Section 10.4 deals with the concept of cooperating UAV on-line path planning using differential evolution. The problem formulation is described, including assumptions, objectives, constraints, cost function definition and path modeling. Simulation results are presented in Section 10.5, followed by discussion and conclusions in Section 10.6.

10.2 B-Spline and Differential Evolution Fundamentals

10.2.1 Path Modeling Using B-Spline Curves

Straight line segments that connect a number of waypoints have been used in the past to model UAV paths in 2-D or 3-D space [23] [5]. However, these simplified paths cannot be used for accurate simulation of a UAV flight, unless a large number of waypoints are adopted. Furthermore, if an optimization procedure is used, the number of design variables explodes, especially if cooperating flying vehicles are considered. As a result, computation time becomes impractical for real world applications.

In [9], paths from the initial vehicle location to the target location are derived from a graph search of a Voronoi diagram that is constructed from the known threat locations. The resulting paths consist of line segments, which are subsequently smoothed around each way point, in order to provide feasible trajectories within the dynamic constraints of the vehicle.

In [24] car formulation has been proposed as an alternative approach to modeling UAV dynamics. Each UAV is assumed to fly with constant altitude, constant flight speed and to have continuous time kinematics [25]. This approach seems inefficient to model real world scenarios, such as those including 3-D terrain avoidance and following of stealthy routes; however, it seems to be sufficient for task assignment purposes with cooperating UAVs flying at safe altitudes [13] [22] [25].

B-Spline curves have been used for trajectory representation in 2-D [26] or 3-D environments [16] [27]. They fit in an optimization procedure as they need a few variables, the coordinates of their control points, to define complicated curved paths. Each control point has a local effect on the curve's shape and small perturbations in its position produce changes in the curve only in the neighborhood of the repositioned control point if the degree of the curve remains small.

B-Spline curves are parametric curves; their construction is based on blending functions [28] [29]. Their parametric construction produces non-monotonic curves, like the trajectories of moving objects.

If the number of control points of the corresponding curve is $n+1$, with corresponding coordinates $(x_0, y_0, z_0),\ldots, (x_n, y_n, z_n)$, the coordinates of the B-Spline curve may be written as:

$$x(u) = \sum_{i=0}^{n} x_i \cdot N_{i,p}(u), \tag{10.1}$$

$$y(u) = \sum_{i=0}^{n} y_i \cdot N_{i,p}(u), \tag{10.2}$$

$$z(u) = \sum_{i=0}^{n} z_i \cdot N_{i,p}(u), \tag{10.3}$$

where u is the free parameter of the curve, $N_{i,p}(u)$ are the blending functions of the curve and p is its degree, which is associated with the curve's smoothness, $p+1$ being its order. Higher values of p correspond to smoother curves.

The blending functions are defined recursively in terms of a *knot* vector $U=\{u_0,\ldots,u_m\}$, which is a non-decreasing sequence of real numbers, with the most common form being the *uniform non-periodic* one, defined as:

$$u_i = \begin{cases} 0 & if \quad i < p+1 \\ i-p & if \quad p+1 \le i \le n \\ n-p+1 & if \quad n < i. \end{cases} \tag{10.4}$$

The blending functions $N_{i,p}$ are computed using the knot values defined above, as:

$$N_{i,0}(u) = \begin{cases} 1 & if \ u_i \le u < u_{i+1} \\ 0 & otherwise, \end{cases} \tag{10.5}$$

$$N_{i,p}(u) = \frac{u-u_i}{u_{i+p}-u_i} N_{i,p-1}(u) + \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}} N_{i+1,p-1}(u). \tag{10.6}$$

If the denominator of either of the fractions is zero, that fraction is defined to have zero value. Parameter u varies between 0 and $(n-p+1)$ with a constant step, providing the discrete points of the B-Spline curve. The sum of the values of the blending functions for any value of u is always 1.

The use of B-Spline curves for the determination of a flight path provides the advantage of describing complicated non-monotonic 3-D curves with controlled smoothness with a small number of design parameters, i.e. the coordinates of the control points. Another valuable characteristic of the adopted B-Spline curves is that the curve is tangential to the control polygon at the starting and ending points. This characteristic may be used in order to define the starting or ending direction of the curve, by inserting an extra fixed point after the starting one, or before the ending control point. Figure 10.1 shows a quadratic B-Spline curve (p=2) in 2-D with its control points and the corresponding control polygon.

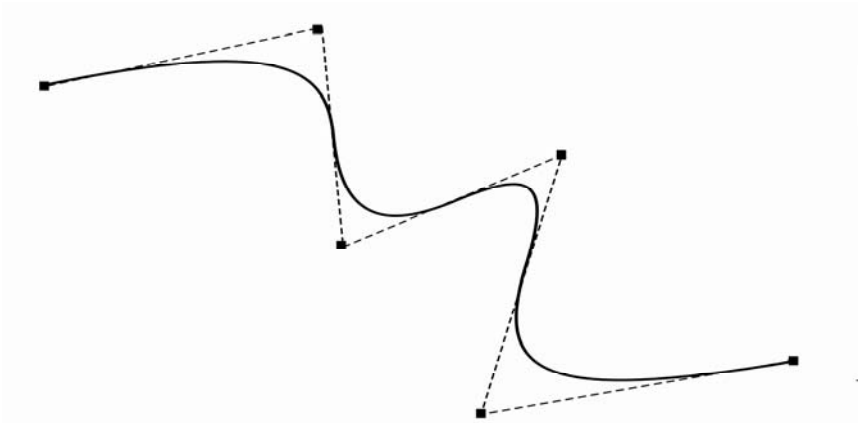

Fig. 10.1. Quadratic 2-D B-Spline curve, produced using a uniform non-periodic knot vector, and its control polygon.

10.2.2 The Solid Boundary Representation

The terrain is represented by a meshed 3-D surface using mathematical functions of the form:

$$z(x,y) = \sin(y+a) + b\cdot\sin(x) + c\cdot\cos\left(d\cdot\sqrt{x^2+y^2}\right) + \tag{10.7}$$
$$e\cdot\cos(y) + f\cdot\sin\left(f\cdot\sqrt{x^2+y^2}\right) + g\cdot\cos(y),$$

where a, b, c, d, e, f, g are constants experimentally defined, in order to produce a surface with mountains and valleys, as shown in Figure 10.2.

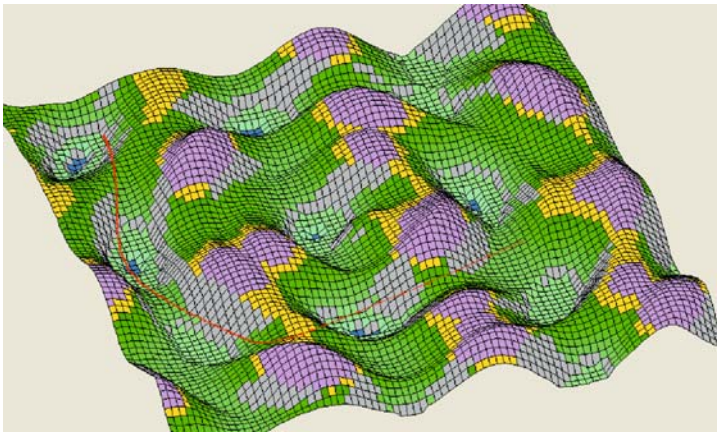

Fig. 10.2. A typical terrain used for simulation studies.

A graphical interface has been developed for visualization of the terrain surface along with the path lines [16]. The corresponding interface deals with different terrains produced either artificially or based on real geographical data, providing an easy verification of the feasibility and the quality of each solution. The path planning algorithm considers the boundary surface as a group of quadratic mesh nodes with known coordinates.

10.2.3 Differential Evolution Algorithm

Differential Evolution (DE) [30] [31] is used as an optimization tool. A DE algorithm, being a recent development in the field of optimization algorithms, represents a type of evolutionary strategy, especially formed so that it can effectively deal with continuous optimization problems. The classic DE algorithm evolves a fixed size population, which is randomly initialized. After initializing the population, an iterative process is started and at each generation G, a new population is produced until a stopping condition is satisfied. At each generation, each element of the population can be replaced with a new generated one. The new element is a linear combination between a randomly selected element and the difference between two other randomly selected elements. In detail, given a cost function f :

$$f(X): R^n \to R \text{ , } n=n_{param} \tag{10.8}$$

the optimization target is to minimize the value of this cost function by optimizing the values of its parameters (design variables):

$$X = \left(x_1, x_2, \ldots, x_{n_{param}}\right),\ x_j \in R \tag{10.9}$$

where X denotes the vector composed of n_{param} cost function parameters (design variables). These parameters take values between specific upper and lower bounds:

$$x_j^{(L)} \le x_j \le x_j^{(U)},\ \ j = 1, \ldots, n_{param} \tag{10.10}$$

The DE algorithm implements real encoding for the values of the objective function design variables. In order to obtain a starting point for the DE algorithm, initialization of the population takes place. Initialization ($G = 1$) is established by randomly assigning values to the parameters j of each i member of the population within the given boundaries, as follows:

$$x_{i,j}^{(1)} = r \cdot \left(x_j^{(U)} - x_j^{(L)}\right) + x_j^{(L)},\ \ i = 1, \ldots, n_{pop},\ \ j = 1, \ldots, n_{param} \tag{10.11}$$

r is a uniformly distributed random value within the range [0, 1]. The DE algorithm mutation operator is based on a triplet of randomly selected different individuals. For each i member of the population a new parameter

vector $V_i^{(G)}$ is generated by adding the weighted difference vector between the two members of the triplet to the third one, the donor:

$$\begin{aligned} V_i^{(G)} &= X_{r3_i}^{(G)} + F \cdot \left(X_{r1_i}^{(G)} - X_{r2_i}^{(G)} \right) \\ V_i^{(G)} &= \left(v_{i,1}^{(G)}, v_{i,2}^{(G)}, \ldots, v_{i,n_{param}}^{(G)} \right) \end{aligned} \tag{10.12}$$

$X_{r3_i}^{(G)}$ is called the 'donor', G is the current generation, and:

$$\begin{aligned} & i = 1, \ldots, n_{pop}, \; j = 1, \ldots, n_{param} \\ & r1_i \in \left[1, \ldots, n_{pop} \right], \; r2_i \in \left[1, \ldots, n_{pop} \right], r3_i \in \left[1, \ldots, n_{pop} \right] \\ & r1_i \neq r2_i \neq r3_i \neq i \\ & F \in \left[0, 1+ \right], \; r \in \left[0, 1 \right] \end{aligned} \tag{10.13}$$

In this way a perturbed individual is generated. The perturbed individual $V_i^{(G)}$ and the initial population member $X_i^{(G)}$ are then subjected to a cross-over operation that generates the final candidate solution $U_i^{(G+1)}$:

$$\begin{aligned} u_{i,j}^{(G+1)} &= \begin{cases} v_{i,j}^{(G)} & if \; \left(r \leq C_r \vee j = k \right) \forall \; j = 1, \ldots, n_{param} \\ x_{i,j}^{(G)} & otherwise \end{cases} \\ C_r &\in \left[0, 1 \right] \end{aligned} \tag{10.14}$$

where k is a random integer within [1, n_{param}], chosen once for all members of the population. The random number r is seeded for every gene of each chromosome. F and C_r are DE algorithm control parameters, which remain constant during the search process and affect the convergence behaviour and robustness of the algorithm. Their values also depend on the objective function, the characteristics of the problem and the population size.

The population for the next generation ($G+1$) is selected between the current population and the final candidates. If each candidate vector is better fitted than the corresponding current one, the new vector replaces the vector with which it was compared. The DE selection scheme is described as follows (for a minimization problem):

$$X_i^{(G+1)} = \begin{cases} U_i^{(G+1)} & if \; f\left(U_i^{(G+1)} \right) \leq f\left(X_i^{(G)} \right) \\ X_i^{(G)} & otherwise \end{cases} \tag{10.15}$$

A new scheme [32] to determine the donor for the mutation operation has been adopted to accelerate the convergence rate. In this scheme, the donor is randomly selected (with uniform distribution) from the region within the 'hyper-triangle', formed by the three members of the triplet:

$$donor_i^{(G)} = \sum_{k=1}^{3}\left(\lambda_k \Big/ \sum_{m=1}^{3}\lambda_m\right) X_{rk_i}^{(G)}, \quad \lambda_m = rand[0,1] \tag{10.16}$$

where $rand[0,1]$ denotes a uniformly distributed value within the range [0, 1]. With this scheme the donor comprises the local information of all members of the triplet, providing a better starting point for the mutation operation that result in a better distribution of the trial vectors.

The random number generation (with uniform probability) is based on the algorithm presented in [33]. For each different operation inside the DE algorithm that requires a random number generation, a different sequence of random numbers is produced by using a different initial seed for each operation and a separate storage of the corresponding produced seeds.

The off-line path planner is presented next.

10.3 Off-line Path Planner

The off-line path planner is presented to introduce the concept of UAV path planning using EAs. The off-line planner generates collision free paths in environments with known characteristics and flight restrictions. The derived path line for each UAV is a single continuous 3-D B-Spline curve, while the solid boundaries are interpreted as 3-D surfaces. The starting and ending control points of each B-Spline curve are fixed. A third point close to the starting one is also fixed, determining the initial flight direction of the corresponding UAV; this control point is placed in a pre-specified distance from the starting control point. Between the fixed control points, additional free-to-move control points determine the shape of the curve. For each path, the number of the free-to-move control points is user-defined.

10.3.1 Path Modeling Using B-Spline Curves

Each path is constructed using a 3-D B-Spline curve; each B-Spline control point is defined by its three Cartesian coordinates $x_{k,j}$, $y_{k,j}$, $z_{k,j}$, $k=0,\ldots,n$, $j=1,\ldots,N$; N is the number of UAVs, while $n+1$ is the number of control points in each B-Spline curve, the same for all curves. The first ($k=0$) and last ($k=n$) control points of the control polygon are the initial and target points of the j^{th} UAV, which are predefined by the user. The second ($k=1$) control point is positioned in a pre-specified distance from the initial control point, in a given altitude, and in a given direction, in order to define the initial direction of the corresponding path.

The control polygon of each B-Spline curve is defined by successive straight line segments as shown in Figure 10.3. For each segment, its length $seg\_length_{k,j}$ and its direction $seg\_angle_{k,j}$ are used as design variables ($k=2,...,n-1$, $j=1,...,N$). Design variables $seg\_angle_{k,j}$ are defined as the difference in degrees between the direction of the current segment and the previous one. For the first segment ($k=1$) of each control polygon $seg\_angle_{1,j}$ is measured with respect to the x-axis. Additionally, the control points' altitudes $z_{k,j}$ are used as design variables, except for the three predefined fixed points ($k=0$, $k=1$, and $k=n$). In the first segment ($k=1$), $seg\_length_{1,j}$, and $seg\_angle_{1,j}$ are pre-specified in order to define the initial direction of the path, and they are not included in the design variables of the optimization procedure. The lower and upper boundaries of each independent design variable are predefined by the user.

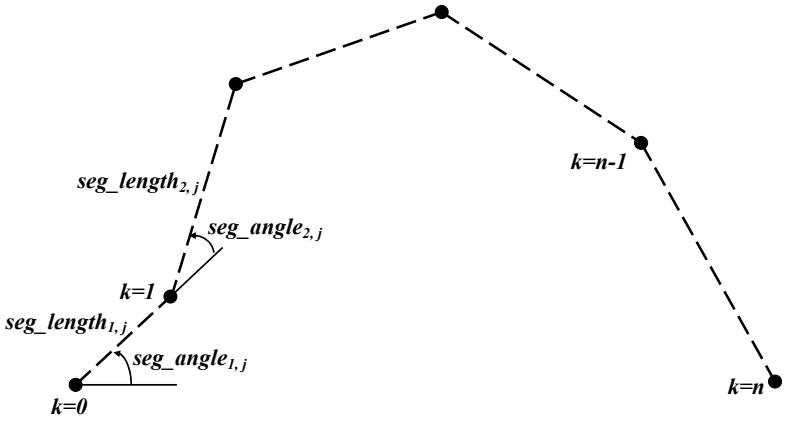

Fig. 10.3. B-Spline control polygon in the horizontal plane.

The coordinates of each B-Spline control point $x_{k,j}$ and $y_{k,j}$ can be easily calculated by using $seg\_length_{k,j}$ and $seg\_angle_{k,j}$ along with the coordinates of the previous control point $x_{k-1,j}$ and $y_{k-1,j}$. The use of $seg\_length_{k,j}$ and $seg\_angle_{k,j}$ as design variables instead of $x_{k,j}$ and $y_{k,j}$ was adopted for three reasons: i) abrupt turns of each flight path can be easily avoided by explicitly imposing short lower and upper bounds for the $seg\_angle_{k,j}$ design variables; ii) by using the proposed design variables, a better convergence rate is achieved compared to the case using the B-Spline control points' coordinates as design variables, a consequence of shortening the search space following the proposed formulation; iii) by using $seg\_length_{k,j}$ as design variables, an easier determination of the upper bound for each curve's length is achieved, along with a smoother variation of the lengths of each curve's segments.

10.3.2 Cost Function Formulation for a Single UAV

For the case of a single UAV the optimization problem to be solved minimizes a set of five terms based on set objectives and constraints associated with the feasibility of the curve, its length and a safety distance from the ground. The cost function to be minimized is defined as:

$$f = \sum_{i=1}^{5} w_i f_i \tag{10.17}$$

Term f_1 penalizes the non-feasible curves that pass through the solid boundary. In order to compute this term, discrete points along each curve are computed, using B-Spline equations (10.1) to (10.6) and a pre-specified step for B-Spline parameter u. The value of f_1 is proportional to the number of discrete curve points located inside the solid boundary; consequently, non-feasible curves with fewer points inside the solid boundary show better cost function than curves with more points inside the solid boundary.

Term f_2 is the length of the curve (non-dimensional with the distance between the starting and destination points) and it is used to provide shorter paths.

Term f_3 is designed to provide flight paths with a safety distance from solid boundaries. For each discrete point i ($i=1,\ldots,n_{line}$, where n_{line} is the number of discrete curve points) of the B-Spline curve its distance from the ground is calculated (recall that the ground is described by a mesh of n_{ground} discrete points). Then the minimum distance of the curve and the ground d_{min} is computed. Term f_3 is defined as:

$$f_3 = \left(d_{safe} / d_{\min}\right)^2 \tag{10.18}$$

while d_{safe} is a safety distance from the solid boundary.

Term f_4 is designed to provide B-Spline curves with control points inside the pre-specified space. If a control point results with an x or y coordinate outside the pre-specified limits, a penalty is added to term f_4 which is proportional to violating the following constraints:

$$\begin{aligned}
&if\ x_{k,j} > x_{\max} \Rightarrow f_4 = f_4 + C_1 * \left|x_{k,j} - x_{\max}\right| \\
&if\ y_{k,j} > y_{\max} \Rightarrow f_4 = f_4 + C_1 * \left|y_{k,j} - y_{\max}\right| \\
&if\ x_{k,j} < x_{\min} \Rightarrow f_4 = f_4 + C_1 * \left|x_{k,j} - x_{\min}\right| \\
&if\ y_{k,j} < y_{\min} \Rightarrow f_4 = f_4 + C_1 * \left|y_{k,j} - y_{\min}\right| \\
&\forall k,\ k = 0,\ldots,n,\ \forall j,\ j = 1,\ldots,N
\end{aligned} \tag{10.19}$$

C_1 is a constant, and x_{min}, x_{max}, y_{min}, y_{max} define the borders of the working space. An additional penalty is added to f_4 in case its value is greater than

zero, in order to ensure that curves inside the pre-specified space have a smaller cost function than those having control points outside of it. This can be formally written as:

$$if \;\; f_4 > 0 \Rightarrow f_4 = f_4 + C_2 \tag{10.20}$$

where C_2 is a constant.

Term f_5 is defined to provide path lines within the known terrain. This characteristic is particularly useful when the off-line path planner is used together with the on-line one, as it will be explained later. Each control point of the B-Spline curve is checked for whether it is placed over a known territory. The ground is modeled as a mesh of discrete points and the algorithm computes the mesh shell (on the *x*-*y* plane) that includes each B-Spline control point. If the corresponding mesh shell is characterized as unknown then a constant penalty is added to f_5. A mesh shell is characterized as unknown if all its 4 nodes are unknown (have not been scanned by a sensor).

Weights w_i are experimentally determined, using as criterion the almost uniform effect of the last four terms in the objective function. Term $w_1 f_1$ has a dominant role in (10.17) providing feasible curves in few generations, since path feasibility is the main concern. The minimization of (10.17), through the DE procedure, results in a set of B-Spline control points, which actually represent the desired path.

Initially, the starting and ending path-line points are determined, along with the direction of flight. The limits of the physical space, where the vehicle is allowed to fly (upper and lower limits of their Cartesian coordinates), are also determined, along with the ground surface. The determined initial flight direction is used to compute the third fixed point close to the starting one.

The DE randomly produces a number of chromosomes to form the initial population. Each chromosome contains the *z* coordinates of the free-to-move B-Spline control points ($k=2,\ldots, n-1$), along with the corresponding seg_length_k and seg_angle_k design variables ($k=2,\ldots, n-1$). For each chromosome the Cartesian coordinates of all B-Spline control points are then computed. Using (10.1) to (10.6), with a constant step of parameter *u*, a B-Spline curve is calculated for each chromosome of the population in the form of a sequence of discrete points. Subsequently, each B-Spline is evaluated using the aforementioned cost function *f*. The population of candidate solutions evolves during the generations; at the last generation the population member with the smallest value of the cost function is the solution to the problem and corresponds to the path line with the best characteristics according to the aforementioned criteria.

The simulation runs have been designed in order to search for path lines between 'mountains'. For this reason, an upper ceiling for flight height has been enforced by providing an upper limit to the z coordinates of the B-Spline control points.

10.4 Cooperating UAVs On-line Path Planning

This Section describes the development and implementation of an on-line path planner for cooperating UAV navigation and collision avoidance in completely unknown static environments. The problem formulation is described, including assumptions, objectives, constraints, cost function definition and path modeling.

Given N UAVs launched from the same or different known initial locations, the objective is to derive N 3-D trajectories, aiming at reaching a predetermined target location while ensuring collision avoidance with the environmental obstacles. Additionally, produced flight paths should satisfy specific route constraints. Each vehicle is assumed to be a point, while its actual size is taken into account by equivalent obstacle – ground growing.

The general problem constraint is collision avoidance between UAVs and the ground. The route constraints are based on:

- Predefined initial and target coordinates for all UAVs;
- Predefined initial directions for all UAVs;
- Predefined minimum and maximum limits of allowed-to-fly space, expressed in terms of minimum and maximum allowed Cartesian coordinates for all path points.

The first two route constraints are explicitly taken into account by the optimization algorithm. The third route constraint is implicitly handled by the algorithm through the definition of the cost function. The cooperation objective is that all UAVs should reach the same target point.

The on-line planner is based on the ideas developed in [16] for a single UAV. It uses acquired information from all UAV on-board sensors (that scan the area within a certain range from the corresponding UAV). The on-line planner rapidly generates a near optimum path, modeled as a 3-D B-Spline curve that will guide each vehicle safely to an intermediate position within the already scanned area. The information about the already scanned area by each UAV is passed to the other cooperating UAVs in order to maximize environment knowledge. The process is repeated until the final position is reached by a single UAV. Then the other UAVs turn to the off-line mode and a single B-Spline path for each UAV is computed to guide it from its current position, through the already scanned territory to

the common final destination. As a result, each path line from the corresponding starting point to the final goal is a smooth, continuous 3-D line that consists of successive B-Spline curves smoothly connected to each other.

10.4.1 Path Modeling

As the terrain is completely unknown and radars (or equivalent sensors) gradually scan the area, it is impossible to generate feasible paths that connect each starting point with the target point. Instead, at certain moments, each sensor scans a region around the corresponding moving UAV and this region is added to the already scanned regions by all cooperating UAVs. For the UAV under consideration a path line is generated that connects a temporary starting point with a temporary ending point. Each temporary ending point is also the next curve's starting point for the corresponding vehicle. Therefore, what is finally generated is a group of smooth curve segments connected to each other, eventually connecting the starting point to the final destination for each UAV. This procedure is represented in Figures 10.4 to 10.6 for a single UAV.

In the on-line problem only four control points define each B-Spline curve, the first two of which are fixed and determine the direction of the current UAV path. The remaining two control points are allowed to take any position within the scanned by the radars known space, taking into consideration given constraints.

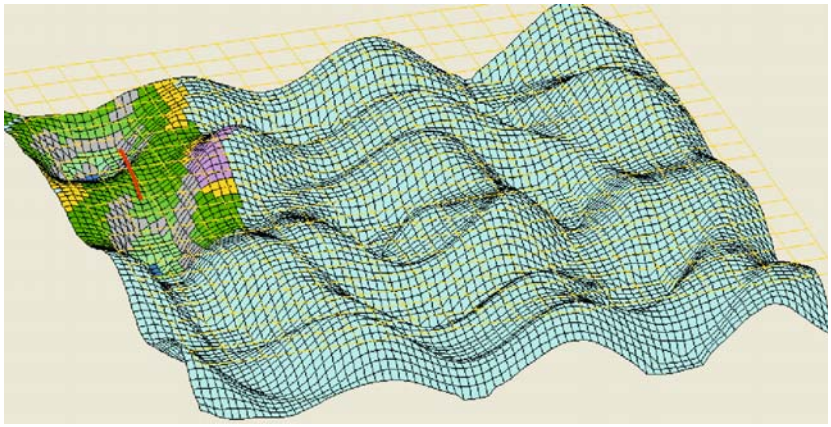

Fig. 10.4. Scanned area (in color), single UAV, movement along first segment.

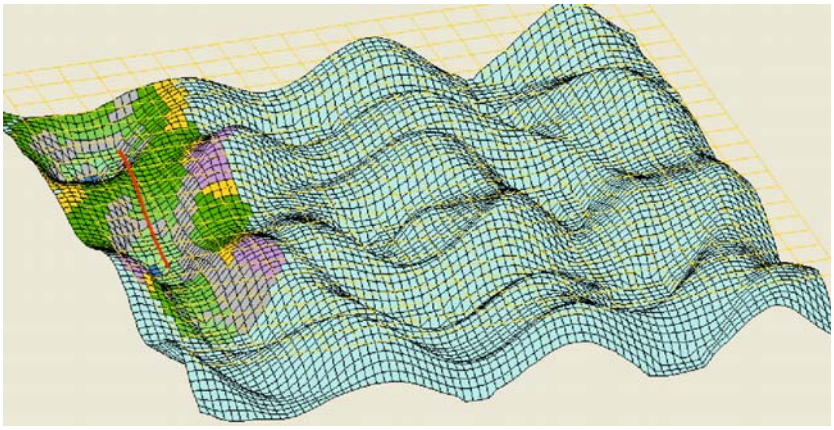

Fig. 10. 5. Scanned area (in color), single UAV, movement along second segment.

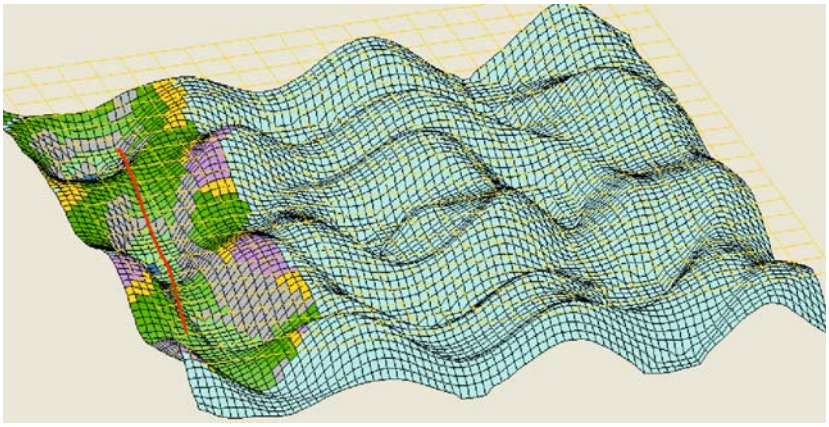

Fig. 10.6. Scanned area (in color), single UAV, movement along third segment.

When the next path segment is generated, only the first control point of the B-Spline curve is known; it is the last control point of the previous B-Spline segment. The second control point is not random, since it is used to guarantee at least first derivative continuity of the two connected curves at their common point. Hence, the second control point of the next curve lies on the line defined by the last two control points of the previous curve as shown in Figure 10.7. It is also desirable that the second control point is near the first one, so that the UAV may easily avoid any obstacle suddenly sensed in front of it. This may happen because the radar scans the environment not continuously, but at intervals. The design variables that define

each B-Spline segment are the same as in the off-line case, i.e., $seg\_length_{k,j}$, $seg\_angle_{k,j}$, and $z_{k,j}$ ($k=2, 3$, and $j=1,...,N$).

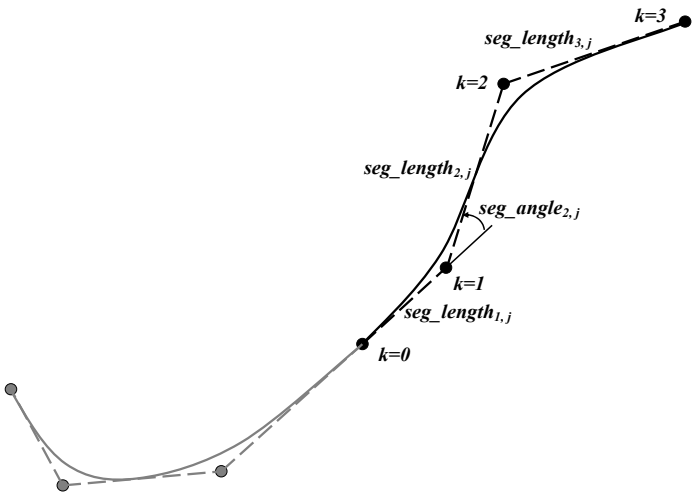

Fig. 10.7. Schematic representation of formation of a complete path by successive B-Spline segments projected on the horizontal plane.

The path-planning algorithm considers the scanned surface as a group of quadratic mesh nodes. All ground nodes are initially assumed to be unknown.

An algorithm is used to distinguish between nodes visible by a radar and nodes that are not visible by it. A node is not visible by a radar if it is not within the range of the radar, or even if it is within its range it is hidden by a ground section that lies between the radar and the UAV. The corresponding algorithm, simulates the radar and checks whether the ground nodes within the radar range are 'visible' or not and consequently 'known' or not. If a newly scanned node is characterized as 'visible', it is added to the set of scanned ground nodes, which is common for all cooperating UAVs.

Radar information is used to produce the first path line segment for the corresponding UAV. As the vehicle moves along its first segment and until it has traveled about 2/3 of its length, its radar scans the surrounding area, returning a new set of visible nodes, which are subsequently added to the common set of scanned nodes. The on-line planner, then, produces a new segment for each UAV, whose first point is the last point of the previous segment and whose last point lies somewhere in the already scanned area. The on-line process is repeated until the ending point of the current path line segment of one UAV lies close to the final destination. Then the other

UAVs turn into the off-line process, in order to reach the target using B-Spline curves that pass through the scanned terrain.

The position at which the algorithm starts to generate the next path line segment for each UAV (here taken as the 2/3 of the segment length) depends on the radar range, the UAV's velocity and the algorithm computational demands.

10.4.2 Cost Function Formulation

The computation of intermediate path segments for each UAV is formulated as a minimization problem. The cost function to be minimized is the weighted sum of eight different terms:

$$f = \sum_{i=1}^{8} w_i f_i \tag{10.21}$$

where w_i are the weights and f_i are the corresponding terms described below.

Terms f_1, f_2, and f_3 are the same with terms f_1, f_3, and f_4 respectively of the off-line procedure. Term f_1 penalizes the non-feasible curves that pass through the solid boundary. Term f_2 is designed to provide flight paths with a safety distance from solid boundaries. Only already scanned ground points are considered for this calculation. Term f_3 is designed to provide B-Spline curves with control points inside the pre-specified working space.

Term f_4 is designed to provide flight segments with their last control point having a safety distance from solid boundaries. This term was introduced to ensure that the next path segment that is going to be computed will not start very close to a solid boundary (which may lead to infeasible paths or paths with abrupt turns). The minimum distance D_{min} from the ground is calculated for the last control point of the current path segment. Only already scanned ground points are considered for this calculation. Term f_4 is then defined as:

$$f_4 = \left(d_{safe} / D_{\min} \right)^2 \tag{10.22}$$

while d_{safe} is a safety distance from the solid boundary.

The value of term f_5 depends on the potential field strength between the initial point of the UAVs path and the final target [16]. This potential field between the two points is the main driving force for the gradual development of each path line in the on-line procedure. The potential is similar to the one between a source and a sink, defined as:

$$\Phi = \ln \frac{r_2 + c \cdot r_0}{r_1 + c \cdot r_0} \tag{10.23}$$

where r_1 is the distance between the last point of the current curve and the initial point for the corresponding UAV, r_2 is the distance between the last point of the current curve and the final destination, r_0 is the distance between the initial point for the corresponding UAV and the final destination and c is a constant. This potential allows for selecting curved paths that bypass obstacles lying between the starting and ending point of each B-Spline curve [16].

Term f_6 is similar to term f_5 but it corresponds to a potential field between the current starting point (of the corresponding path segment) and the final target.

Term f_7 is designed to prevent UAVs from being trapped in small regions and to force them move towards unexplored areas. It may be possible that some segments of the path lines are concentrated in a small area, away from the final target. In order to help the UAV leave this area, term f_7 repels it from the points of the already computed path lines (of all UAVs). Furthermore, if a UAV is wandering around to find a path that will guide it to its target, the UAV will be forced to move towards areas not visited before by this or other UAVs. This term has the form:

$$f_7 = \frac{1}{N_{po}} \sum_{k=1}^{N_{po}} \frac{1}{r_k} \tag{10.24}$$

where N_{po} is the number of the discrete curve points produced so far by all UAVs and r_k is their distance from the last point of the current curve segment.

Term f_8 represents another potential field, which is developed in a small area around the final target. When the UAV is away from the final target, the term is given a constant value. When the UAV is very close to the target the term's value decreases proportionally to the square of the distance between the last point of the current curve and the target. Thus, when the UAV is near its target, the value of this term is quite small and prevents the UAV from moving away.

Weights w_i in (10.21) are experimentally determined, using as criterion the almost uniform effect of all the terms, except the first one. Term $w_1 f_1$ has a dominant role, in order to provide feasible curve segments in a few generations, since path feasibility is the main concern.

10.5 Simulation Results

The same artificial environment was used for all test cases considered, with different starting and target points. The artificial environment is constructed within a rectangle of 20x20 (non-dimensional lengths). The (non-

dimensional) radar's range for each UAV was set equal to 4. The safety distance from the ground was set to d_{safe}=0.25. The experimentally optimized settings of the DE algorithm during the on-line procedure were as follows: *population size* = 20, F = 0.6, C_r = 0.45, *number of generations* = 70. For the on-line procedure two free-to-move control points were considered, resulting in 6 design variables.

The corresponding settings during the off-line procedure were as follows: *population size* = 30, F = 0.6, C_r = 0.45, *number of generations* = 70. For the off-line procedure eight control points were used to construct each B-Spline curve, including the initial (k=0) and the final one (k=7). These correspond to five free-to-move control points, resulting in 15 design variables. All B-Spline curves have a degree p equal to 3.

All experiments search for path lines between 'mountains'. For this reason, an upper ceiling for flight height has been enforced in the optimization procedure by explicitly providing an upper bound for the z coordinates of all B-Spline control points.

The first test case corresponds to the on-line path planning for a single UAV over an unknown environment. Results of how the path is formed in terms of successive snapshots of path formulation have already been illustrated through Figures 10.4 to 10.6. Figures 10.8 and 10.9 depict the path that finally succeeds in guiding the UAV towards the target location, although the initial flight direction drives the UAV away from the target. In this case, term f_3 is activated when the path line exceeds the borders of the pre-specified workspace as observed in the lower left corner of Figure 10.9, and enforces the path line to return within the limits. Although the complete path is constructed of 15 successive B-Spline curves, the final curve is smooth enough to be followed by a flying vehicle.

Four additional test cases of on-line path planning for a single UAV are shown in Figures 10.10 to 10.13, respectively. For the second case the starting point and the initial direction is the same with the first case. The third and forth test cases have the same initial point close to the center of the terrain. The fifth test case has its starting point near the right lower corner of the terrain. As observed, although the planner some times produces complicated paths, it succeeds in finding the final destination. However, as shown in Figure 10.13, an abrupt turn occurs, which is the result of the effort to avoid exceeding workspace limits.

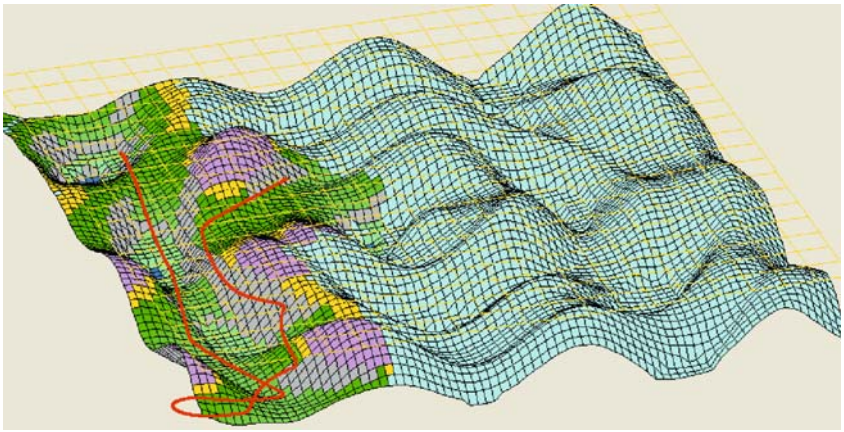

Fig. 10.8. Test case 1, single UAV: On-line path planning with the scanned area shown in color. Path shown in an intermediate position of the flight; consists of 12 B-Spline segments.

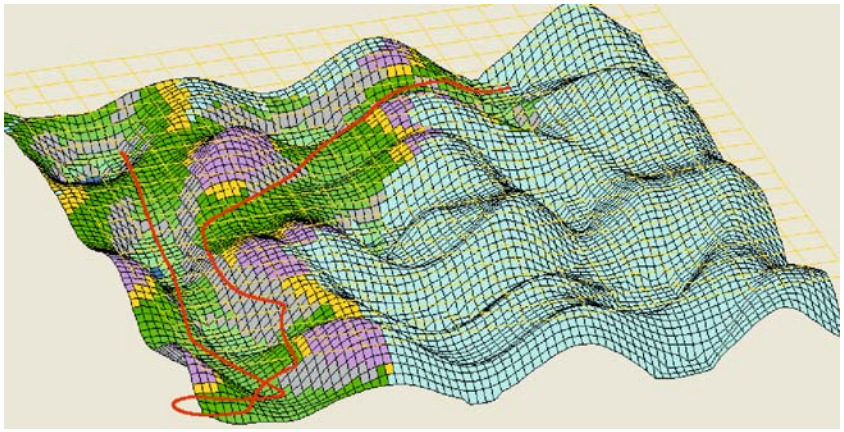

Fig. 10.9. The completed path for test case 1 consisting of 15 B-Spline segments.

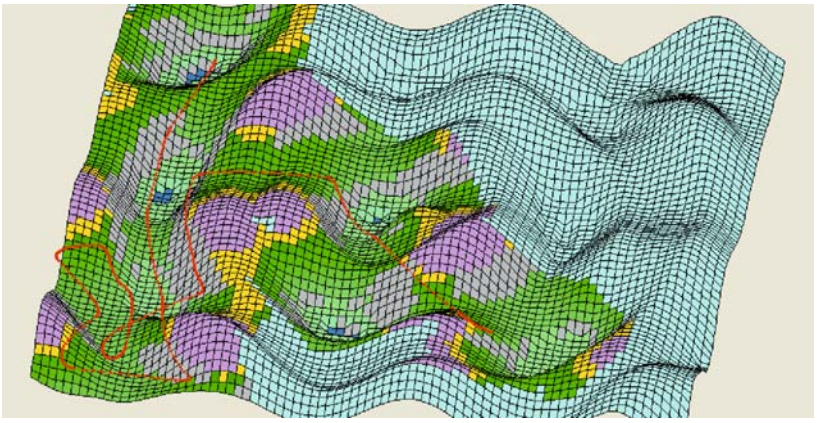

Fig. 10.10. Test case 2, single UAV: On-line path planning; Completed path with the starting point being the same with test case 1, close to the upper left corner.

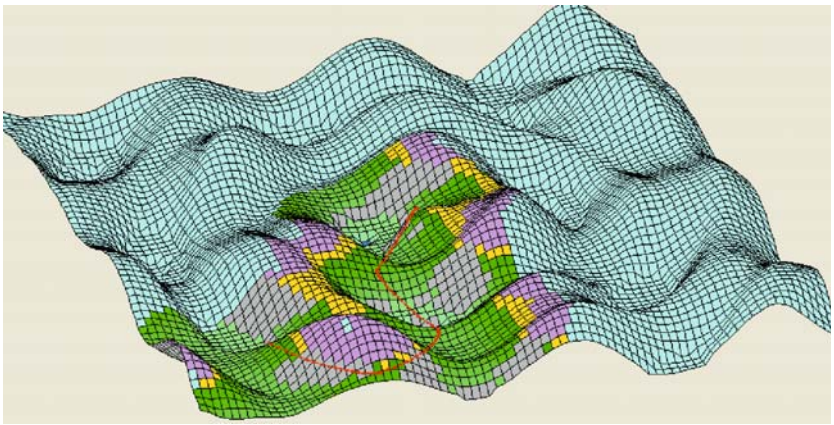

Fig. 10.11. Test case 3, single UAV: On-line path planning; The starting point is close to the center of the terrain.

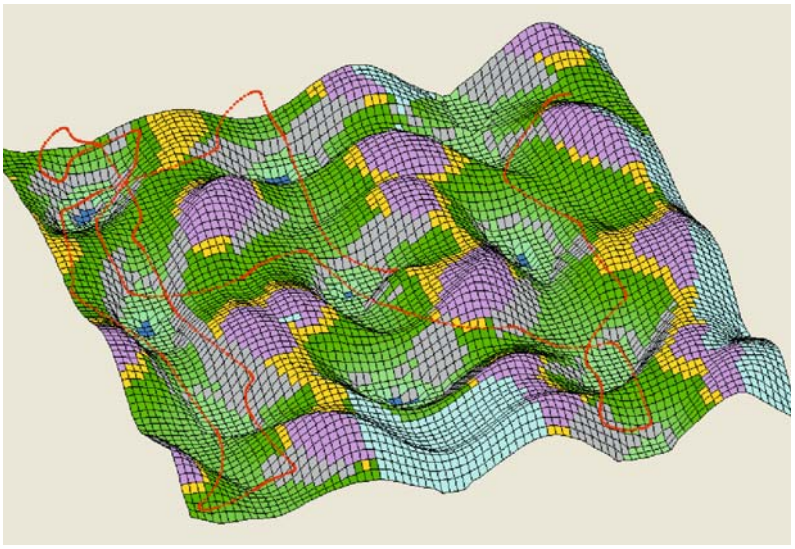

Fig. 10.12. Test case 4, single UAV: On-line path planning; Completed path with the starting point being close to the center of the terrain.

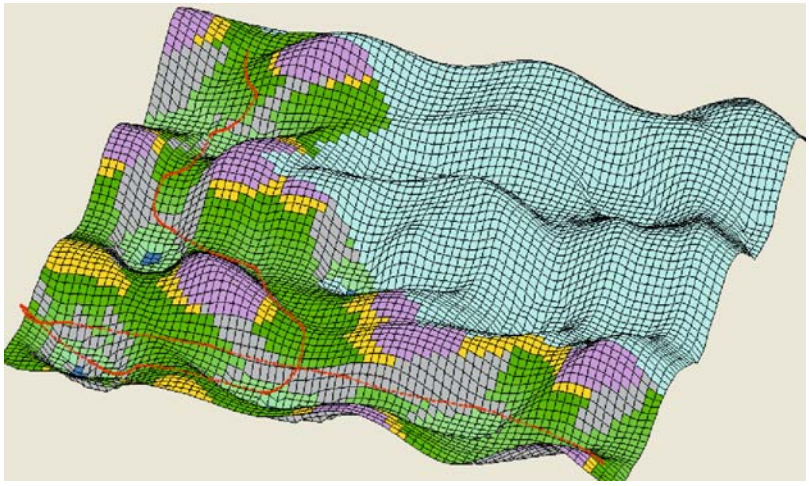

Fig. 10.13. Test case 5, single UAV: On-line path planning with the starting point being close to the right lower corner of the terrain.

Test case 6 corresponds to on-line path planning of 2 UAVs as shown in Figures 10.14 and 10.15. Figure 10.14 shows the path lines when the first UAV (blue line) reaches the target. At that moment the second UAV (red line) turns into the off-line mode, in order to compute a feasible path line

that connects its current position with the target through the already scanned area. The final status is demonstrated in Figure 10.15. The starting point for the first UAV is near the lower left corner of the terrain; the starting point of the second UAV is near the upper left corner.

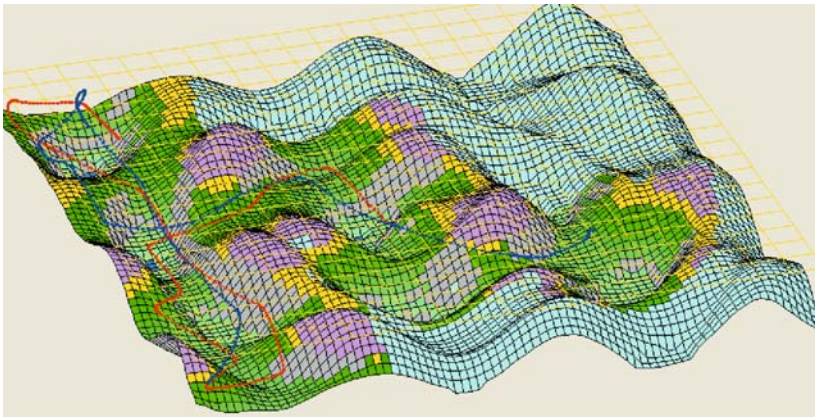

Fig. 10.14. Test case 6, two UAVs. On-line path planning; when the first UAV (blue line) reaches the target the second one turns into the off-line mode.

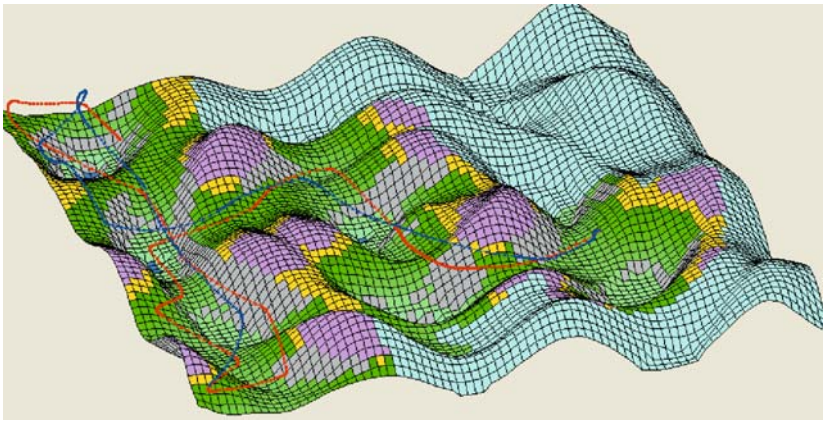

Fig. 10.15. Final status of the path lines for the test case 6.

Test case 7 corresponds to on-line path planning for 3 UAVs as shown in Figures 10.16 and 10.17. Figure 10.16 shows the status of the two path lines when the first UAV (blue line) reaches the target. At that moment the second UAV (red line) and the third one (black line) turn into the off-line mode, in order to compute feasible path lines that connect their positions

with the target. The final status is demonstrated in Figure 10.17. The starting point of the first and second UAV are the same as in test case 6, while the starting point of the third UAV is near the middle of the left side of the terrain.

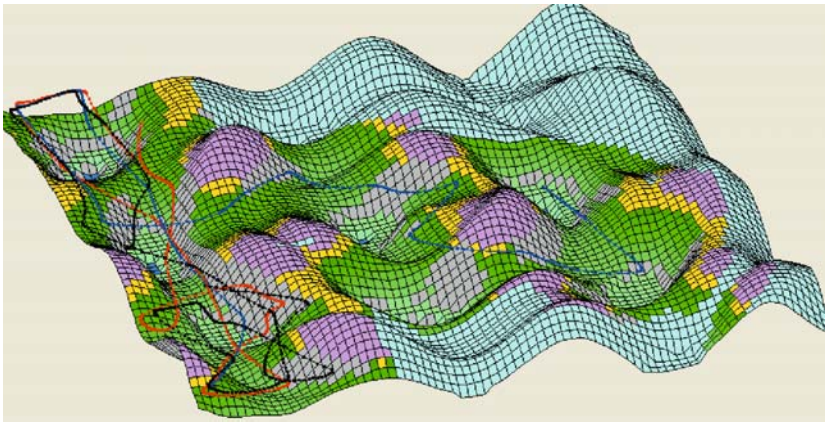

Fig. 10.16. Test case 7, three UAVs. On-line path planning; it shows the paths when the first UAV (blue line) reaches the target.

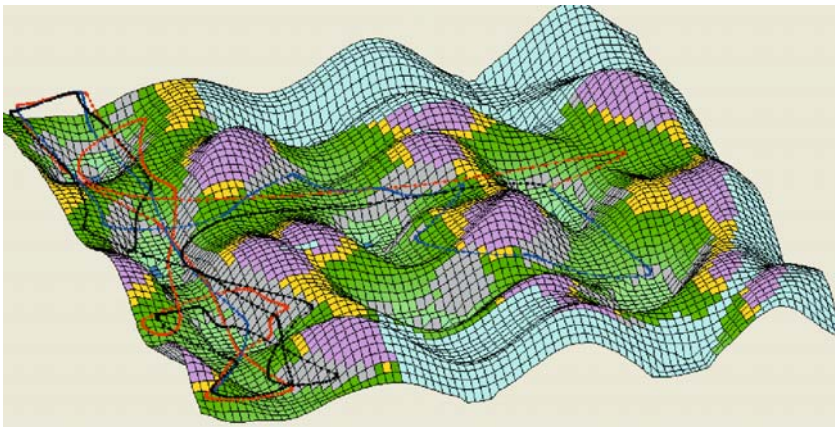

Fig. 10.17. The final status of the path lines of test case 7

Test case 8 refers to another on-line path planning scenario for 3 UAVs as shown in Figures 10.18 and 10.19. Figure 10.18 depicts the status of the two path lines when the first UAV (blue line) reaches the target. The final status is demonstrated in Figure 10.19. As the first UAV (blue line) is

close to the target, it succeeds in reaching it using just one B-Spline segment. The other two UAVs turn into off-line mode to reach the target.

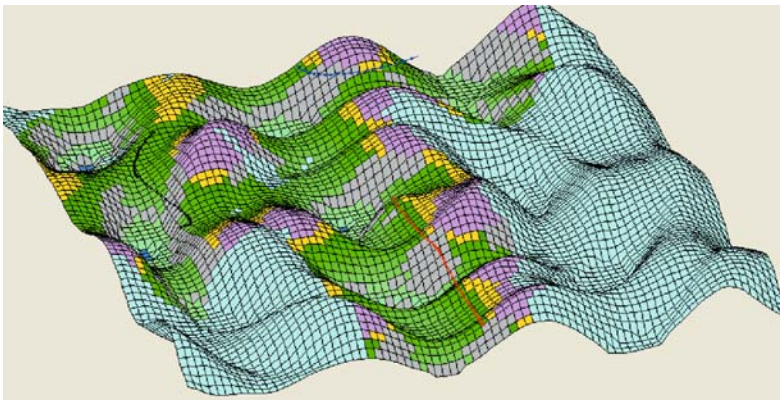

Fig. 10.18. Test case 8, three UAVs. On-line path planning; it shows the path lines when the first UAV (blue line) reaches the target.

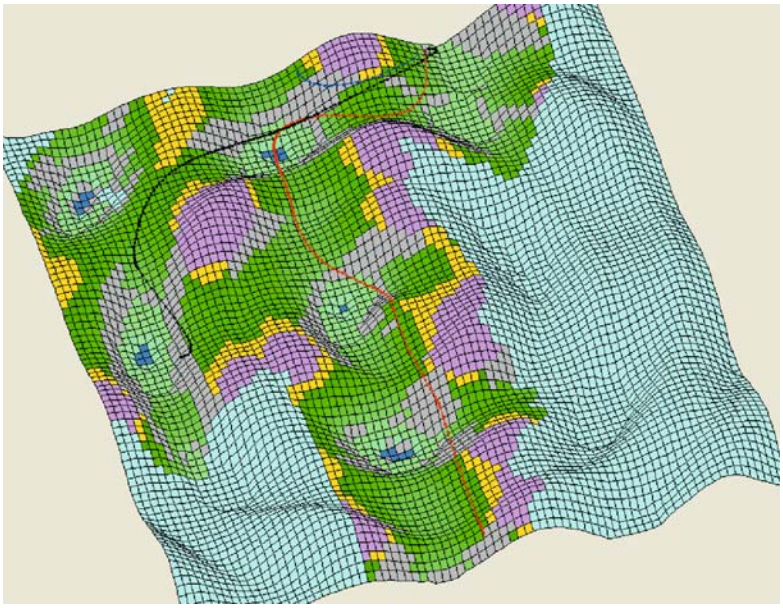

Fig. 10.19. The final status of the path lines of test case 8.

In test case 9 three UAVs are launched from the same point in the center of the working space but with different directions. Figure 10.20 shows the status of the two path lines when the first UAV (blue line) reaches the tar-

get. The final status is demonstrated in Figure 10.21. During the on-line procedure, when the final point of a curve segment is within a small range from the final destination the on-line procedure is terminated; this is the reason for the absence of coincidence between the final points of the first (blue line) and the rest path lines.

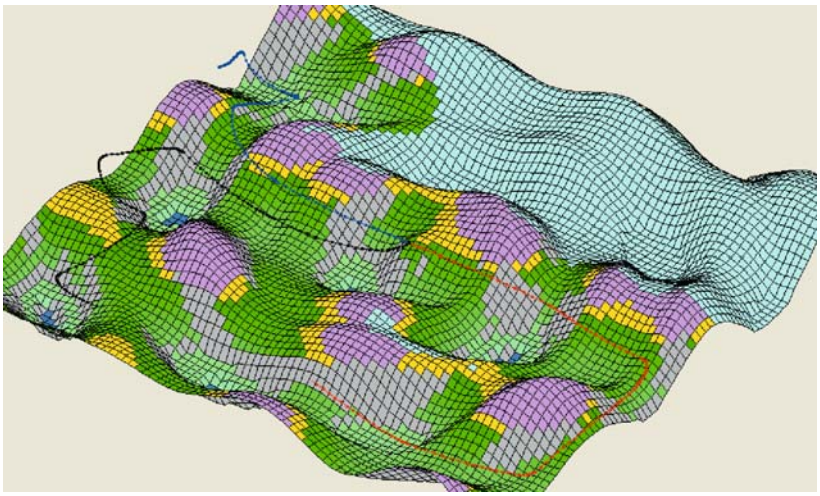

Fig. 10.20. Test case 9, three UAVs. On-line path planning; it shows the status of path lines when the first UAV (blue line) reaches the target.

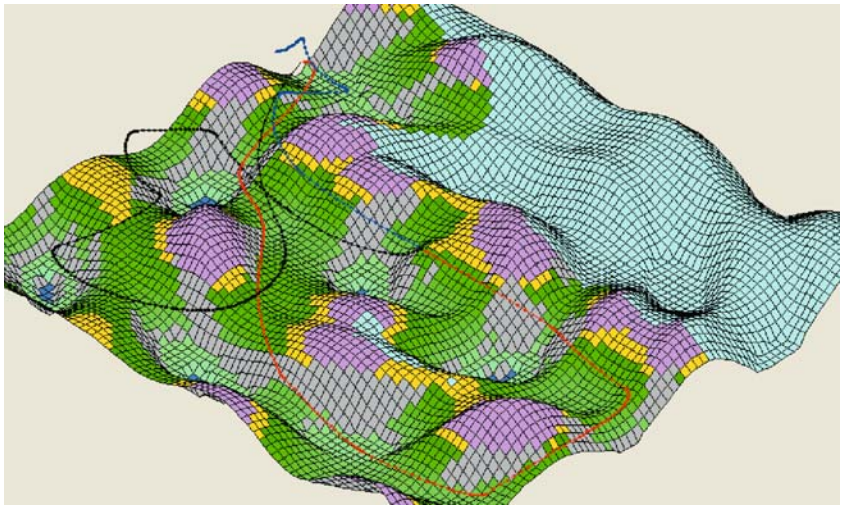

Fig. 10.21. The final status of the path lines of test case 9.

10.6 Conclusions

A path planner has been presented that is suitable for navigating a group of cooperating UAVs avoiding collisions with environment obstacles. The planner is capable of producing smooth path curves in known or unknown static environments.

Two types of path planner were presented. The off-line path planner generates collision free paths in environments with known characteristics and flight restrictions. The on-line planner, which is based on the off-line one, generates collision free paths in unknown environments. The path line is gradually constructed by successive, smoothly connected B-Spline curves, within the gradually scanned environment. The knowledge of the environment is acquired through the UAV on-board sensors that scan the area within a certain range from each UAV. This information is exchanged between the cooperating UAVs; as a result, each UAV utilizes the knowledge of a larger region than the one scanned by its own sensors.

The on-line planner generates for each vehicle a smooth path segment that will guide the vehicle safely to an intermediate position within the known territory. The process is repeated for all UAVs until the corresponding final position is reached by an UAV. Then, the rest vehicles turn into the off-line mode in order to compute path lines consisting of a single B-Spline curve that connect their current positions with the final destination. These path lines are enforced to lie within the already scanned region. Both path planners are based on optimization procedures, and specially constructed functions are used to encounter the mission and cooperation objectives and constraints. A DE algorithm is used as the optimizer for both planners.

The introduced potential fields are the main driving forces for the gradual generation of the path lines in the on-line planner. As demonstrated by the presented test cases, potential fields may be effectively used to generate curves that bypass the solid ground obstacles positioned between the starting and target positions, and when combined with the other terms of the fitness (cost) function, they can produce path lines that escape from concave areas.

The use of a confined workspace for the UAV flight was proven to be another useful characteristic of the on-line procedure. The planner enforces the path lines to be constructed within this confined space; as a result, the search for each path is restricted within a finite area. By utilizing a special term in the cost function, the on-line planner is enforced to explore new areas and prevent each UAV from being trapped in the same region for a long time. As the search space is confined, this term enables the planner to

explore this space, and even if a vehicle is initially driven towards the wrong direction, it eventually returns towards its final destination.

The selection of the B-Spline design variables was also proven very effective in providing smooth curves. For all *seg_angle* variables a range of variation between -90 and +90 degrees was explicitly defined, which resulted in smooth turns for both off-line and on-line planners. If the B-Spline control point coordinates were used as design variables, the avoidance of abrupt turns would be a much more difficult procedure that would require additional special terms in the cost function.

Results are very encouraging and support further research to implement the algorithms to small UAVs in real-time.

References

1. Gilmore J. F., "Autonomous vehicle planning analysis methodology", *Proceedings, Association of Unmanned Vehicles Systems Conference*, 503-509, 1991.
2. LaValle S. M., *Planning Algorithms*, Cambridge University Press, 2006.
3. Bortoff S., "Path planning for UAVs", *Proceedings, American Control Conference*, 364–368, 2000.
4. Szczerba R. J., Galkowski P., Glickstein I. S., and Ternullo N., "Robust Algorithm for Real-time Route Planning", *IEEE Transactions on Aerospace Electronic Systems*, 36, 869–878, 2000.
5. Zheng C., Li L., Xu F., Sun F., Ding M., "Evolutionary Route Planner for Unmanned Air Vehicles", *IEEE Transactions on Robotics*, 21, 609-620, 2005.
6. Uny Cao Y., Fukunaga A. S., Kahng A. B., "Cooperative Mobile Robotics: Antecedents and Directions", *Autonomous Robots*, 4, 7-27, 1997.
7. Schumacher C., "Ground Moving Target Engagement by Cooperative UAVs", *Proceedings, American Control Conference*, Oregon, June 2005.
8. Mettler B., Schouwenaars T., How J., Paunicka J., and Feron E., "Autonomous UAV Guidance Build-up: Flight-test Demonstration and Evaluation Plan", *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2003.
9. Beard R. W., McLain T. W., Goodrich M. A., Anderson E. P., "Coordinated Target Assignment and Intercept for Unmanned Air Vehicles", *IEEE Transactions on Robotics and Automation*, 18, 911-922, 2002.
10. Richards A., Bellingham J., Tillerson M., and How J., "Coordination and Control of UAVs", *Proceedings of the AIAA Guidance, Navigation and Control Conference,* Monterey, CA, 2002.
11. Schouwenaars T., How J., and Feron E., "Decentralized Cooperative Trajectory Planning of Multiple Aircraft with Hard Safety Guarantees", *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004.

12. Flint M., Polycarpou M., and Fernandez-Gaucherand E., "Cooperative Control for Multiple Autonomous UAV's Searching for Targets", *Proceedings, 41st IEEE Conference on Decision and Control*, 2002.
13. Tang Z., and Ozguner U., "Motion Planning for Multi-Target Surveillance with Mobile Sensor Agents", *IEEE Transactions on Robotics*, 21, 898-908, 2005.
14. Gomez Ortega J., and Camacho E. F., "Mobile Robot Navigation in a Partially Structured Static Environment, Using Neural Predictive Control", *Control Engineering Practice*, 4, 1669-1679, 1996.
15. Kwon Y. D., and Lee J. S., "On-line Evolutionary Optimization of Fuzzy Control System Based on Decentralized Population", *Intelligent Automation and Soft Computing*, 6, 135-146, 2000.
16. Nikolos I. K., Valavanis K. P., Tsourveloudis N. C., Kostaras A., "Evolutionary Algorithm Based Offline / Online Path Planner for UAV Navigation", *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, 33, 898-912, 2003.
17. Michalewicz Z., *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, 1999.
18. Smierzchalski R., "Evolutionary Trajectory Planning of Ships in Navigation Traffic Areas", *Journal of Marine Science and Technology*, 4, 1-6, 1999.
19. Smierzchalski R., and Michalewicz Z., "Modeling of Ship Trajectory in Collision Situations by an Evolutionary Algorithm", *IEEE Transactions on Evolutionary Computation*, 4, 227-241, 2000.
20. Sugihara K., and Smith J., "Genetic Algorithms for Adaptive Motion Planning of an Autonomous Mobile Robot", *Proceedings, IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Monterey, California, 138-143, 1997.
21. Sugihara K., and Yuh J., *GA-Based Motion Planning for Underwater Robotic Vehicles*, UUST-10, Durham, NH, 1997.
22. Shima T., Rasmussen S. J., Sparks A. G., "UAV Cooperative Multiple Task Assignments Using Genetic Algorithms", *Proceedings, American Control Conference*, Oregon, June 2005.
23. Moitra A., Mattheyses R. M., Hoebel L. J., Szczerba R. J., Yamrom B., "Multivehicle Reconnaissance Route and Sensor Planning", *IEEE Transactions on Aerospace and Electronic Systems*, 37, 799-812, 2003.
24. Dubins L., "On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Position", *American Journal of Mathematics*, 79, 497–516, 1957.
25. Shima T., Schumacher C., "Assignment of Cooperating UAVs to Simultaneous Tasks Using Genetic Algorithms", *Proceedings, AIAA Guidance, Navigation, and Control Conference and Exhibit*, California, 2005.
26. Martinez-Alfaro H., and Gomez-Garcia S., "Mobile Robot Path Planning and Tracking Using Simulated Annealing and Fuzzy Logic Control", *Expert Systems with Applications*, 15, 421-429, 1988.
27. Nikolos I. K., Tsourveloudis N. C, and Valavanis K. P., "Evolutionary Algorithm Based 3-D Path Planner for UAV Navigation", CD-ROM Proceedings,

9th Mediterranean Conference on Control and Automation, Dubrovnik, Croatia, 2001.
28. Piegl L., Tiller W., *The NURBS Book*, Springer, 1997.
29. Farin G., *Curves and Surfaces for Computer Aided Geometric Design, A Practical Guide*, Academic Press, 1988.
30. Storn R., and Price, K., *DE - A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Space*, ICSI, Technical Report TR-95-012, 1995.
31. Price K. V., Storn R. M., Lampinen J. A., *Differential Evolution, a Practical Approach to Global Optimization*, Springer, 2005.
32. Hui-Yuan F., Lampinen J., Dulikravich G. S., "Improvements to Mutation Donor Formulation of Differential Evolution", *Proceedings of EUROGEN Conference on Evolutionary Methods for Design, Optimization and Control, Applications to Industrial and Societal Problems*, CIMNE, Barcelona, 2003.
33. Marse K. and Roberts S. D., "Implementing a portable FORTRAN Uniform (0, 1) Generator", *Simulation*, 41-135, 1983.