

Evolutionary Algorithm Based 3-D Path Planner for UAV Navigation.

Ioannis K. Nikolos, Nikos C. Tsourveloudis, and Kimon P. Valavanis

Abstract--An off-line path planner for Unmanned Air Vehicles is presented. The planner is based on Evolutionary Algorithms, in order to calculate a curved pathline with desired characteristics in a three-dimensional environment. The pathline is represented using B-Spline curves, with the coordinates of its control points being the genes of the artificial chromosome of the Evolutionary Algorithm. The method was tested in an artificial three-dimensional terrain, for different starting and ending points, providing very smooth pathlines, under difficult constraints. The path planner is under integration with a trajectory tracker, based on fuzzy-logic.

Index Terms--B-Splines, 3-D Path Planning, Evolutionary Algorithms, Navigation, UAV.

I. INTRODUCTION

DURING the last decade, Unmanned Air Vehicles (UAVs) have replaced piloted aircraft in a broad band of missions, showing a high potential for further growing, especially due to the avoidance of human risk in dangerous environments. Typical present and future roles include weather reconnaissance, offshore and border patrol, search and rescue assisting operations in sea and mountains, aerial photographing and mapping, fire detection and coordination of fire fighting, traffic control etc. Advances in telecommunications, control and Artificial Intelligence along with UAV low risk, low cost and long endurance, tailor such missions to the UAV's profile.

Autonomous operation of UAVs requires the development of control systems that operate isolated from human support for extended time periods. The development of such systems has been traditionally focused on ground vehicles. Additionally, the high cost and risk of testing air vehicles necessitates the use of sophisticated analysis and simulation tools.

The desired autonomous operations of UAVs leads to the development of pathline planners, for generating collision free paths in environments with obstacles. Such planners should work either on-board, for local real-time optimization of the flight path, or out-board, for global optimization of the mission, prior to the flight. The optimization of the flight path is, in general, a constrained multi-objective optimization

problem. Additionally, a trajectory tracker is needed, in order to schedule the movement of the UAV along the optimized planned path.

In this work, an off-line UAV trajectory planner, based on an Evolutionary Algorithm, is described. Traditionally, evolutionary algorithms have been used for the solution of the path-finding problem in ground based or sea surface navigation [1]. Evolutionary algorithms have been used for the solution of the path-finding problem in a 3-D environment for underwater vehicles, using line segments for the representation of the pathline [2]. B-Spline curves have been used for the pathline representation in a 2-D environment, using simulated annealing for the optimization of the line [3]. In the current work the path-finding problem is three-dimensional, with the pathline being a continuous 3-D B-Spline curve, while the solid boundaries are 3-D surfaces.

The pathline of a flying vessel is actually a curve with curvature continuity and, therefore, cannot be modeled using straight-line segments, which is the usual practice for ground robots. The B-Spline curves, used here for path representation, have the advantage of being described using a small amount of data (actually the coordinates of their control points), although they may produce very complicated curves. The coordinates of their control points form the chromosome of each individual in the Evolutionary Algorithm.

II. THE EVOLUTIONARY ALGORITHM

A. Basic Features

Evolutionary Algorithms (EA) simulate the process of genetic evolution found in nature to perform artificial evolution, in order to provide optimized solutions in various problems. Since their development they have been successfully applied to numerous different and complex search space problems.

In nature, the external characteristics (phenotype) of each organism are encoded in their genes (genotype). Individuals with external characteristics better fitted to the environment have better chances to survive and increase their numbers over a number of generations, while the less fit individuals tend to eliminate. In this way *natural selection* ensures that genes from a better-fitted individual have better chances to survive and consequently their population increases.

The natural selection process is simulated in EA, using a number (population) of individuals (solutions to the problem) to evolve through certain procedures. Each individual is represented through a string of numbers (bit strings, integers

This work was partially supported by the Hellenic General Secretary of Research and Technology.

The authors are with the Dept. of Production Engineering and Management, Technical University of Crete, 73100 Chania, Greece.
(e-mail: jnikolo@central.ntua.gr, (nikost, kimonv)@dpem.tuc.gr).

or floating point numbers), in a similar way with chromosomes in nature. The quality of each individual is represented by a function (the fitness), tailored to the problem to be optimized.

The EA starts by generating in a random way the initial population of chromosomes, with their genes taking values inside the desired constrained space. After the evaluation of the fitness function for each individual, certain operators are applied to the population, simulating the according natural processes. These operators include various forms of selection, recombination and mutation, which are used in order to provide the chromosomes of the next generation. The process of evaluation and creation of a new generation is successively repeated, providing individuals with high values of fitness function.

Four main categories of Evolutionary Algorithms exist: Genetic Algorithms (GAs), Evolution Strategies (ES), Breeder Genetic Algorithms (BGAs) and Evolutionary Programming (EP). The Evolutionary Algorithm discussed in this work is a modified Breeder Genetic Algorithm, incorporating some characteristics of the classic GAs.

The Breeder Genetic Algorithms (BGAs) use floating point representation of variables and both recombination and mutation operators. The truncation model is used as the selection scheme, with the best T% of P initial individuals to give origin to the individuals of the next generation, with equal probability.

B. The Representation of the Individual

The representation of each individual can be realized using artificial chromosomes with either binary or floating point coding. In the present work floating point coding is used, as it allows for better physical representation of the path line control points and easier control of the space constraints.

The path line is represented using a B-Spline curve, with the Cartesian coordinates of its control points being the genes of the artificial chromosome. The starting and ending points of the path are fixed, while the internal control points are free to change in a constrained space. The Cartesian coordinates of the non-fixed control points form the genes of each individual.

C. The Initial Population

The initial population is created randomly in the constrained space of each gene. The lower and higher constraints of each gene can be chosen in a way that specific undesirable solutions may be avoided, such as pathlines with a higher than the desired altitude. Although the shortening of the search space reduces the computation time, it can also lead in local optima. The algorithm may be initialized using also the last population of a previous run.

D. The Selection Scheme

The selection scheme is a combination of the truncation model of Breeder Genetic Algorithms and the roulette procedure of the traditional Genetic Algorithms. Starting with the truncation model, only T% elements showing the best

fitness are chosen in order to give origin to the individuals of the next generation. The parameter T is the threshold of the procedure. Once chosen these individuals are used for the generation of a new population through the roulette wheel selection [1]. The selection probability of each of the T% remaining individuals is directly proportional to its relative fitness, with reference to the average fitness of the T% selected individuals. The fitter the individual, the more chances has for being chosen. Using this hybrid scheme, only the individuals with the best fitness are allowed to pass to the next generation.

The adopted hybrid selection scheme provides high flexibility to the evolutionary algorithm. When threshold takes values close to 100% the scheme actually serves as a classic roulette scheme. For values of T close to 10% the scheme serves close to a classic truncation model. In the current work values of T between 40% and 70% were used. Lower values of T tend to trap the procedure in local optima.

An elitist model assures that the best individual of each generation always survives the selection procedure and reproduces its structure in the next generation.

E. The Recombination and Mutation Operators

In order to provide fine local tuning, non-uniform mutation and heuristic crossover are used, along with the classic mutation and crossover schemes [1]. The first operator chooses randomly, with a predefined probability, the gene of a chromosome to be mutated. Contrary to the uniform mutation, the search space for the new gene is not fixed, but it shrinks close to the previous value of the corresponding gene as the algorithm converges. The search is uniform initially, but very local at later stages.

The second operator generates a single offspring x_3 from two parents x_1 and x_2 . If x_2 is not worse than x_1 , then x_3 is given as

$$x_3 = r(x_2 - x_1) + x_2 \quad (1)$$

where r is a random number between 0 and 1. In this way a direction for search is adopted, providing fine local tuning and search in the most promising direction.

III. USING THE EVOLUTIONARY ALGORITHM FOR PATH PLANNING

A. The Solid Boundary Representation

The solid terrain under the flying vessel is represented by a meshed three-dimensional surface. For simplicity reasons this surface is produced using mathematical functions of the form

$$z(x, y) = \sin(y + a) + b \times \sin(x) + c \times \cos(d \times (y^2 + x^2)^{0.5}) + e \times \cos(y) + f \times \sin(f \times (y^2 + x^2)^{0.5}) + g \times \cos(y) \quad (2)$$

where a, b, c, d, e, f, g are proper constants. The produced surface simulates a terrain with mountains and valleys, as can be seen in Fig. 7.

A graphical environment was developed for the visualization of the terrain surface, along with the pathline curve [5]. The corresponding environment can deal with different terrains, produced either artificially or based on real geographical data. Horizontal sections of the surface in different heights can be plotted, visualizing the solid boundaries in the UAVs flight height, as presented in Fig. 4.

B. The B-Spline Modeling of the Pathline

The pathline of a flying object cannot be represented by straight-line segments, as it is usually the case for mobile robots, sea and undersea vessels. In the present work B-Splines are adopted in order to define the desired path of the UAV, providing continuity at least of the second derivative of the curve [4], [5]. B-Spline curves are well fitted to the evolutionary procedure, as they need a few variables (coordinates of the control points) in order to define complicated curved paths. Additionally, each control point has a very local effect on the curve's shape, and small perturbations in its position produce changes in the curve only in the neighborhood of the changing control point.

B-Spline curves are parametric curves, with their construction based on blending functions [4]. Their parametric construction provides the ability to produce non-monotonic curves. If the number of control points of the corresponding curve is $n+1$, with coordinates $(x_0, y_0, z_0), \dots, (x_n, y_n, z_n)$, the coordinates of the B-Spline can be written as:

$$X(t) = \sum_{i=0}^n x_i * B_{i,K}(t) \quad (3)$$

$$Y(t) = \sum_{i=0}^n y_i * B_{i,K}(t) \quad (4)$$

$$Z(t) = \sum_{i=0}^n z_i * B_{i,K}(t) \quad (5)$$

where $B_{i,K}(t)$ the blending functions of the curve and K the degree of the curve, which is associated with curve's smoothness. Higher values of K correspond to smoother curves, as it is demonstrated in figures 1 and 2. Parameter t varies between 0 and $n-K+2$ with a constant step, providing the discrete points of the B-Spline curve. The coordinates (x_i, y_i, z_i) of the control points form the artificial chromosome of the evolutionary algorithm. The sum of the values of the blending functions for any value of t is always 1.

The blending functions are defined recursively in terms of a set of knot values, with the most common form being the uniform non-periodic one, defined as:

$$\begin{aligned} \text{Knot}(i) &= 0 & \text{if } & i < K \\ \text{Knot}(i) &= i - K + 1 & \text{if } & K \leq i \leq n \\ \text{Knot}(i) &= n - K + 2 & \text{if } & n < i \end{aligned} \quad (6)$$

The blending functions $B_{i,K}$ are defined recursively, using the knot values given by eq.(6):

$$B_{i,K}(t) = \begin{cases} 1 & \text{if } \text{Knot}(i) \leq t < \text{Knot}(i+1) \\ 1 & \text{if } \text{Knot}(i) \leq t \leq \text{Knot}(i+1) \text{ and } t = n - K + 2 \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

$$B_{i,K}(t) = \frac{(t - \text{Knot}(i)) \times B_{i,K-1}(t)}{\text{Knot}(i+K) - \text{Knot}(i)} + \frac{(\text{Knot}(i+K) - t) \times B_{i+1,K-1}(t)}{\text{Knot}(i+K) - \text{Knot}(i+1)} \quad (8)$$

If the denominator of either of the fractions is zero, that fraction is defined to have zero value.

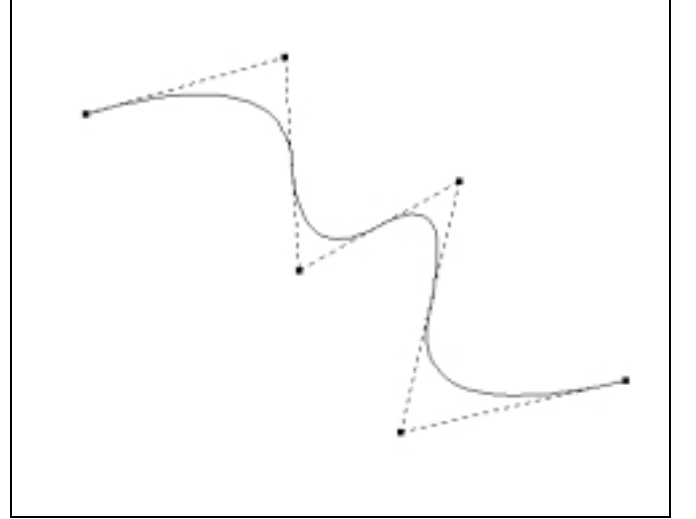


Fig. 1. A 2-D cubic B-Spline curve (K=3), with its control points and the control polygon.

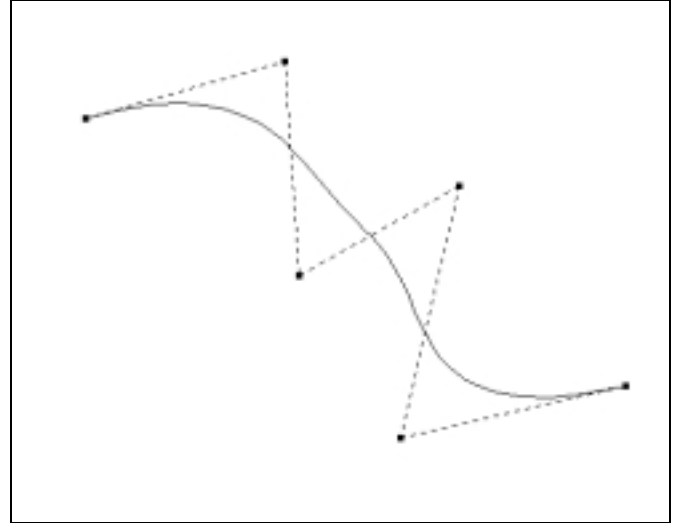


Fig. 2. The corresponding B-Spline curve with the same control points as in Fig. 1 for K=5.

A valuable characteristic of the adopted B-Spline curves is that the curve is tangential to the control polygon at the starting and ending points (see Fig. 1 and Fig. 2 for the polygon formed by the control points). This characteristic can be used in order to define the starting direction of the curve, by inserting an extra fixed point after the starting one. These two points can define the direction of the curve at the corresponding region. This is essential for the path planning of flying vessels, as their flight angles are continuously defined. Consequently the direction of the designed pathline in the starting position must coincide with the current

direction of flight in this position, in order to ensure curvature continuity of the whole pathline.

The B-Spline curve is discretized, using a constant step dt equal to 0.01, and it is used in this form for the calculation of its fitness.

C. The Fitness Function

The fitness function of the evolutionary algorithm is formed in order to evaluate four different aspects of the selected curve:

1. The feasibility of the curve with respect to collision free path generation.
2. The minimization of the total length of the path.
3. The curve's minimum distance from solid boundaries.
4. The minimum curvature of the B-Spline curve.

The problem to be solved is a minimization one, while the algorithm has been constructed for maximization problems. For this reason the fitness function is the inverse of the weighted sum of four different terms:

$$f = \frac{1}{\alpha_1 f_1 + \alpha_2 f_2 + \alpha_3 f_3 + \alpha_4 f_4} \quad (9)$$

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

The term f_1 in (9), penalizes the non-feasible curves that pass through the solid boundary. The penalty value is proportional to the number of discretized curve points (not control points) located inside the solid boundary. In this way non-feasible curves with fewer points inside the solid boundary show better fitness than the curves with more points inside the solid boundary. Additionally, the fitter of the non-feasible curves may survive the selection procedure and produce acceptable offsprings through the heuristic crossover operation.

The term f_2 in (9), is the total length of the curve and is used in order to provide shorter paths.

The term f_3 in (9), is designed to provide flight paths with a safety distance from solid boundaries, given as:

$$f_3 = \sum_{i=1}^{nline} \sum_{j=1}^{nground} \frac{1}{(r_{i,j} / r_{safe})^2} \quad (10)$$

where $nline$ is the number of discrete curve points, $nground$ is the number of discrete mesh points of the solid boundary, $r_{i,j}$ is their distance and r_{safe} is the minimum safety distance from the solid boundary.

The term f_4 of (9) is designed in order to provide curves with a prescribed minimum curvature angle. This characteristic is essential for a flying vessel, as the minimum radius of curvature is determined by its flight envelope. The angle determined by two successive discrete segments of the curve is calculated. In case that the calculated angle is less than a prescribed one (175°), a penalty is added to the fourth term of the fitness function.

The weights α_i are experimentally determined, using as criterion the almost uniform effect of the last three terms in

the fitness value. The term $\alpha_1 f_1$ has a dominant role in (9) because provides feasible curves in few generations, since path feasibility is the main concern.

IV. EXPERIMENTAL RESULTS

The pathline planner was tested in various experiments, using the simulation environment produced by (2). The settings of the evolutionary algorithm are as follows: population size = 100, threshold = 0.5, heuristic crossover probability = 0.75, classic crossover probability = 0.15, mutation probability = 0.05, non-uniform mutation probability = 0.15. The algorithm was defined to terminate after 50 generations, although feasible solutions can be reached in less than 20 iterations.

The experiments were designed in order to search for pathlines between the "mountains". For this reason, an upper limit for flight height was enforced.

The degree K of the B-Spline curve was set equal to 5, providing adequate smoothness to the calculated curve. The free-to-move control points were taking values between 4 and 6, resulting in a total number of B-Spline control points equal to 7-9 (along with the fixed starting and target points, and the fixed second point, used for the determination of the initial direction). The four free-to-move control points correspond to $3 \times 4 = 12$ genes of each chromosome and the six control points to 18 genes, a relatively low number of genes, for this kind of applications. Higher number of control points resulted in higher computation time and slower convergence rate, without any significant profit, concerning the fitness of the curve.

In all test cases the terrain was produced by the same function, with prefixed starting curve directions and various starting and ending positions. Four free-to-move control points were used for test cases 1, 2, 3, 4, 5, while six control points were used for the last test case. In all cases considered, an upper ceiling was set for the z-coordinate of the control points. This ceiling is represented in the graphical environment by the horizontal section of the terrain. The test cases 5 and 6, shown in Fig. 8 and Fig. 9 respectively, were designed with the ceiling set to a low altitude, increasing the path planning difficulty. The starting position in all test cases is marked with a circle. The minimum distance from the mountain-like boundaries was set equal to 1/30 of the x-dimension of the terrain, in all the cases, except for case 4, which is presented in Fig. 6 and Fig. 7. In test case 4, the minimum distance was set equal to the 1/15 of the terrain's x-dimension. As it is demonstrated in Fig. 6, a higher distance from the solid boundaries was achieved, compared to test case 3 (Fig. 5). For the test case 6, shown in Fig. 9, a wider terrain was used.

Relatively high values of mutation probabilities were adopted, in order to ensure the ability of the algorithm to overcome local optima. As it was observed, initial feasible solutions, provided by the evolutionary algorithm, were progressively replaced by fitter ones, with a completely different structure. Figures 6 and 7 demonstrate the above observation.

Figures 3 to 9 demonstrate the ability of the proposed

method to provide collision-free, smooth pathlines, with a desired initial direction, even for complicated environments with very narrow passages in both horizontal and vertical directions.

V. DISCUSSION AND FUTURE WORK

The trajectory of an UAV cannot be, adequately, represented using line segments. Additionally, a pathline formed with line segments, cannot be followed by a flying vessel, without giving rise to control and stability problems. The proposed method uses a parametric curve to produce a continuous pathline, which is described by a small set of parameters – the coordinates of its control points. The construction of the B-Splines based on control points, proved suitable for coupling with an evolutionary algorithm. The direction of the curve can easily be prescribed at its starting position, by inserting a second fixed point. The direction described by these initial points is the initial direction of the curve and must coincide with the current flight direction.

The proposed method is capable for providing feasible (collision-free) solutions after a small number of generations (less than 20), rendering it suitable for almost real-time calculations. The ability of the curve, to follow the direction of flight at its starting position, can be incorporated within a real time path planning generalization, which is under development. The latter is particularly useful in applications where the global terrain geography is unknown and a local knowledge is gained through onboard sensors.

The presented off-line path planner is going to be integrated with a trajectory tracker under development, in order to schedule the movement of the UAV along the optimized planned path and to deal with unexpected situations (such as flying obstacles).

VI. ACKNOWLEDGMENT

This work was partially funded by the Hellenic General Secretary of Research and Technology, under a PENED grant.

VII. REFERENCES

- [1] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer Publ., Third Edition, 1999.
- [2] K. Sugihara and J. Smith, "A Genetic Algorithm for 3-D Path Planning of a Mobile Robot", Tech. Rep. No. 96-09-01, Software Engineering Research Laboratory, Department of Information and Computer Sciences, University of Hawaii at Manoa, September 1996.
- [3] H. Martinez-Alfaro and S. Gomez-Garcia, "Mobile robot path planning and tracking using simulated annealing and fuzzy logic control", *Expert Systems with Applications*, vol. 15, pp. 421-429, 1988.
- [4] G. Farin, *Curves and Surfaces for Computer Aided Geometric Design, A Practical Guide*, Academic Press, 1988.
- [5] R. Stephens, *Visual Basic Graphics Programming*, Wiley Publ., Second Edition, 1999.

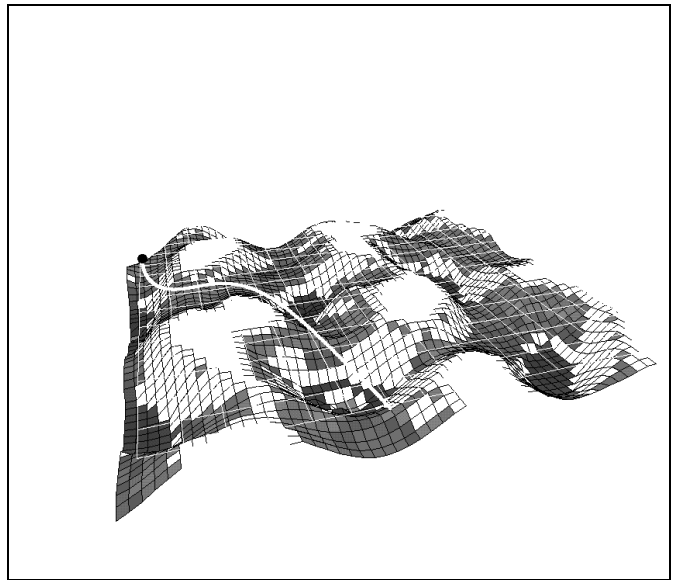


Fig. 3. The first test case: An upper ceiling was imposed for the control points of B-Spline curve, represented by the horizontal section of the terrain. The starting position is marked with a circle.

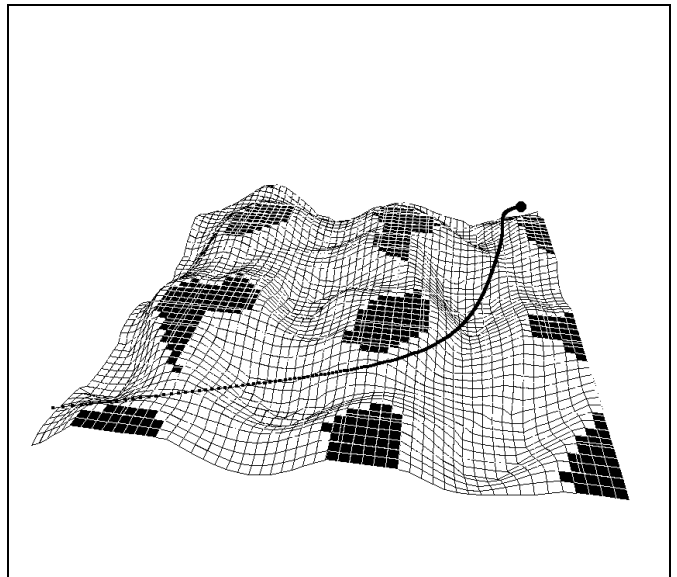


Fig. 4. The second test case: Four free control points with prefixed direction of the curve at the starting position. The minimum acceptable distance from the solid boundary is equal to the one of the first test case.

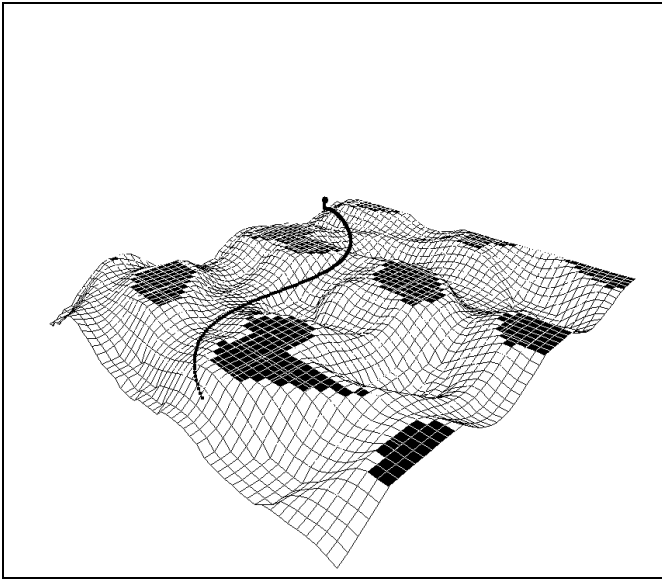


Fig. 5. The third test case.

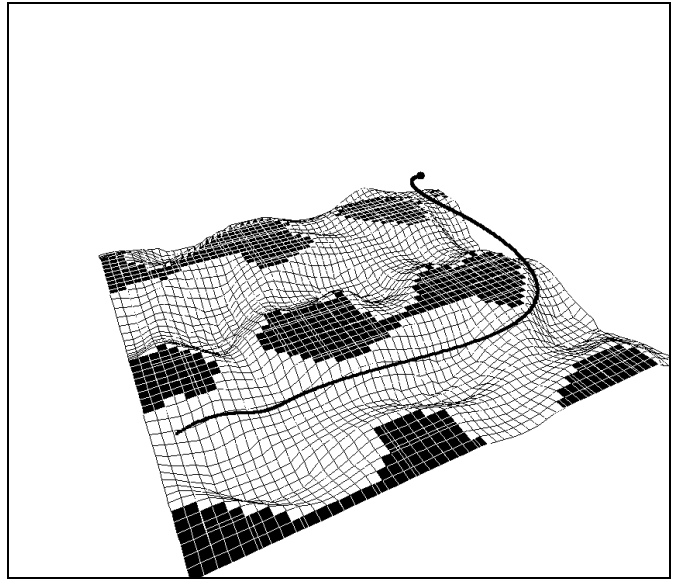


Fig. 8. The fifth test case, with 4 free control points, and a very low upper limit.

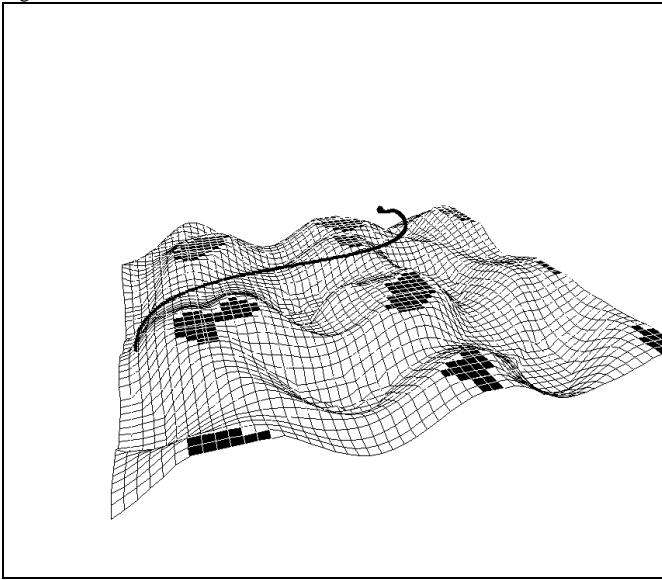


Fig. 6. The fourth test case.

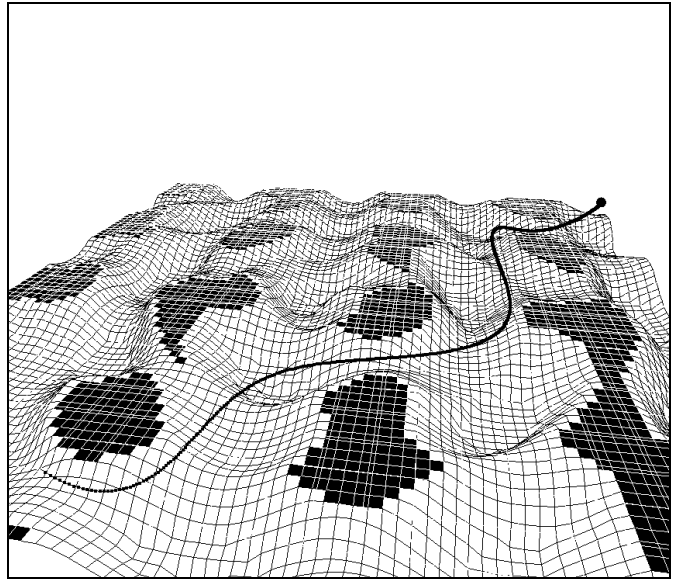


Fig. 9. A wider terrain was used for the sixth test case, along with 6 free control points and very low upper limit.

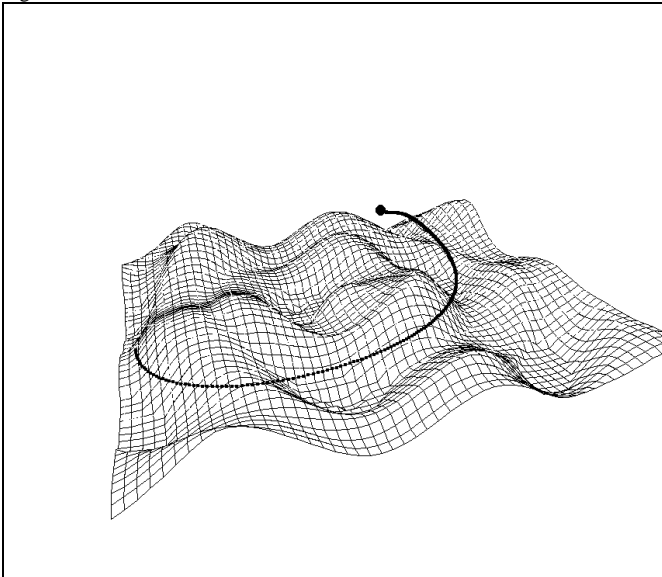


Fig. 7. Early feasible solution of the fourth test case.