Three-Dimensional Offline Path Planning for UAVs Using Multiobjective Evolutionary Algorithms

Shashi Mittal and Kalyanmoy Deb
Kanpur Genetic Algorithms Laboratory (KanGAL)
Indian Institute of Technology Kanpur
Kanpur, PIN 208016, India
{mshashi,deb}@iitk.ac.in
http://www.iitk.ac.in/kangal

KanGAL Report Number 2004008

Abstract

In this paper, we present 3D offline path planner for Unmanned Aerial Vehicles (UAVs) using multiobjective evolutionary algorithms. In particular we have chosen the commonly used NSGA-II algorithm for this purpose. The algorithm generates a curved path which is represented using B-Spline curves. The control points of the B-Spline curve are the variables in the genetic algorithm. In particular, we solve two problems, assuming the normal flight envelope restriction : i) path planning for UAV when no other constraint is assumed to be present ii) path planning for UAV if the vehicle has to necessarily pass through a particular point in the space. The use of multiobjective evolutionary algorithms helps in generating a number of feasible paths with different trade-offs between the objective functions. The availability of a number of trade-off solutions allows the user to choose a path according to his/her needs easily.

Keywords: 3-D path planning, B-Spline curves, Multiobjective Optimization, UAV.

1 Introduction

The main motivation behind this work is to develop an offline path planner for UAVs. Such a path planner can be used for navigation of UAVs over rough terrain, where a traversal otherwise can be difficult and prone to accidents.

In this work it is assumed that the terrain topography is known beforehand. The details of the terrain can be acquired using satellite data or from surveillance data. The starting point and the end point of the flight are given and it is required to find a feasible path connecting these two points. The path should be feasible in the sense that it should be free from obstacles and its curvature at any point along the path should not exceed beyond a limit. This restriction is necessary as it is not always possible for an aerial vehicle to traverse a path with large curvatures. The UAV is assumed to be a point. In this work the following two problems are tackled.

Case I: UAV navigation using an offline path planner when no other constraints are assumed. In this case it is assumed that the path of the UAV can take any shape and can pass through any point in the space. The path so obtained is a single B-Spline curve between starting and end points.

Case II: UAV navigation using an offline path planner when the vehicle has to pass a particular point in the space. Such a situation can arise, for example, when the UAV is required for surveillance in a particular part of the terrain, or it has to drop a bomb at a particular point. In this case also, the path planner generates a single B-Spline curve, which passes through this particular point.

The path planing strategy uses a multiobjective evolutionary algorithm here. The advantage of using such a method is that it generates not just one but many optimal (called as *Pareto-optimal*) solutions, each with a different trade-off among the objective functions. It should be mentioned that the objective functions (which will be described later) of the path planning problems are conflicting to each other, hence the use of such a strategy is desirable. After briefly describing the specific multiobjective optimization algorithm used in this study, we present simulation results on a number of case studies. The advantages of posing the problem as a multiobjective optimization will be evident from the simulation results.

2 Related work

A number of methods for generating paths in known 2-D or 3-D environment already exist. Neural networks had been extensively used for solving such problems. In recent years, path planning for robots in a 2-D environment using evolutionary algorithms are also studied in great detail [8]. For an UAV navigation, there exist methods for finding optimal paths using Voronnoi diagrams [9, 10].

A more recent work in UAV path planning uses a class of single objective genetic algorithms (called *breeder genetic algorithms*) for offline as well as online path planning [1]. It uses B-Spline curves for representing paths in 3-D space. Our work uses several of the models employed in this work. In particular, the representation of the terrain and the B-Spline curves representation for UAV paths is the same as in [1].

3 Representation of terrain and UAV path

The representation of the terrain is the same as in [1]. For UAV path, as it has been mentioned earlier, B-Spline curves are used. The details of both these are given below.

3.1 Representation of terrain

The terrain is represented as a meshed 3-D surface, produced using a mathematical function of the form

$$z(x,y) = \sin(y+a) + b.\sin(x) + c.\cos(d.\sqrt{y^2+x^2}) + e.\sin(e.\sqrt{y^2+x^2}) + f.\cos(y) \quad (1)$$

where a, b, c, d, e, f are constants experimentally defined in order to produce a surface simulating a rough terrain with valleys and mountains. A sample terrain generated using this function is shown in Figure 1.

3.2 Representation of UAV path using B-Spline curves

B-Spline curves are used to represent the UAV path. More details on B-Spline curves can be found in [7]. B-Spline curves are well suited for this problem because of the following reasons:

1. B-Spline curves can be defined using only a few points (called *control points*). This makes their encoding in genetic algorithm easier.
2. Very complicated path, if needed, can easily be produced by using a few control points.
3. B-Spline curves guarantee differentiability at least up to the first order. Thus, the curves so obtained are quite smooth, without any kinks or breaks.

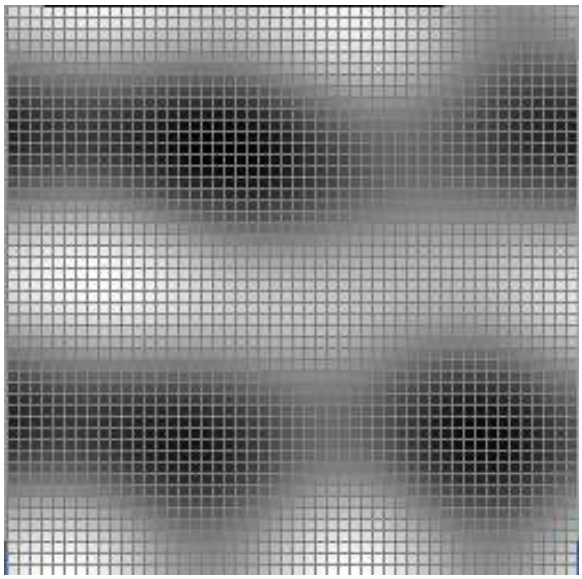

Figure 1: A sample terrain generated using eq. 1. Note that the lighter regions represent greater heights (i.e. hills), where as the darker regions represent valleys. The same shading convention has been adopted for the B-Spline curves in subsequent figures.

4. Changing the position of one of the control points affects the shape of the curve in the vicinity of that control point only.

Besides, B-Spline curves also provide us an easy model for representing curves to solve the Case 2. The curve representation in this problem is tackled as follows: Suppose that UAV has to pass through the point P, and the control points of the B-Spline curve are $P_0, P_1, \ldots P_n$. Choose k such that $0 < k < n$. Fix $P_k = P$. If $P_{k-1}P_k = P_kP_{k+1}$ and P_{k-1}, P_k, P_{k+1} are collinear then the B-Spline curve is guaranteed to pass through the point P. Using such a model for the curve, Case 2 can be easily tackled, to force the path to traverse through one or more points.

3.3 Genetic encoding of B-Spline Curves

The coordinates of the free-to-move control points of the B-Spline curve (i.e. the control points excluding the starting and the end points) are the genes of the chromosomes used in the genetic algorithm. Each control point in 3-D space is represented using three coordinates, one each for the (X, Y, Z) axis. Thus, if there are n free-to-move control points, then each chromosomes will have $3n$ genes. Each coordinate is represented using a floating point representation.

3.3.1 Crossover operator

Simulated Binary crossover(SBX) [2] has been used in the genetic algorithm. This operator is ideal for genes encoded using floating point representation.

3.3.2 Mutation operator

For mutation, a non-uniform mutation operator is used. This operator is described in detail in [2]. In this operator, the probability of creating a solution closer to the parent is more than the probability of creating away from it, and also the perturbation caused by the mutation operator decreases with increase in generation. The mathematical formulation of the operator is

$$y = x + \tau(x_u - x_i)(1 - r^{(1-t/t_{max})^b}), \qquad (2)$$

where y is the offspring sample, x is the parent sample, r is a random number in the range $[0, 1]$, τ takes a Boolean value $(-1, 1)$, each with a probability of 0.5. The parameter t_{max} is the maximum number of allowed generations, while b is a user defined parameter, generally set to 1.0. x_u and x_i are the upper and the lower limits, respectively for gene variable i.

4 Objective functions and constraints

The optimization problem to be solved minimizes two functions subject to three constraints. These are mentioned in detail below.

4.1 Objective functions

We consider two objective functions which are conflicting in nature.

4.1.1 Length of the curve

The first objective function is the length of the overall path (curve), which should be as small as possible. The length of the curve length is calculated by summing over the distances between the successive discrete points of the curve. This objective function is called as f_1.

4.1.2 Risk factor

It is designed to provide flight paths with a safety distance from the solid boundaries. This objective function is the same as in [1]. A mathematical formulation of the objective function is

$$f_2 = \sum_{i=1}^{nline} \sum_{j=1}^{nground} 1/(r_{i,j}/r_{safe})^2, \qquad (3)$$

where *nline* is the number of discrete curve points, *nground* is the number of discrete mesh points of the terrain boundary, $r_{i,j}$ is the distance between the i-th curve points and j-th node point on the terrain boundary, and r_{safe} is the minimum safety distance the UAV should have above the terrain. This function is henceforth referred to as the risk factor of the curve. It can be observed that smaller the value of this risk factor, larger is the safety.

Thus, we minimize both objective functions for achieving a short-length path having a smaller risk factor.

4.2 Constraints

The constraints of the problem are enumerated below:

1. The first constraint is that the path should be plausible to realize, that is, the vehicle should not collide with the terrain boundary in the course of its flight.

2. The angle between the two successive discrete segments of the curve should not be less than a certain cut-off angle. This constraint is imposed to provide the path with a prescribed minimum curvature radius.

3. The maximum height of any point in the curve should not exceed a specified upper limit.

The last constraint is imposed for several reasons. It is generally taxing for a UAV to traverse at high altitudes, because thrust at higher altitude is low. Moving at a low height also avoids the UAV getting detected by the radars, and is also helpful in carrying out surveillance tasks in a better way.

5 Algorithms for Path Planning

The path planner employs the following three-step hybrid algorithm for finding optimal paths:

Step 1: First, an evolutionary multiobjective optimizer is used to obtain a set of Pareto-optimal curves.

Step 2: Next, about 8 to 10 well-dispersed solutions from the Pareto-optimal solutions are picked by using a K-mean clustering algorithm, so that picked solutions provide a good trade-off between the objectives.

Step 3: Finally, a local search is performed on these solutions to obtain a set of solutions close to the true Pareto-optimal solutions.

The non-dominated solutions so obtained after the local search are finally displayed using a graphics display system. The algorithms are described in detail below.

5.1 NSGA-II Algorithm

NSGA-II is a fast and elitist multiobjective evolutionary algorithm that has been found to be highly suitable for solving multiobjective optimization problems [3]. It not only finds solutions close to the true Pareto-optimal frontier, but also provides a good spread of solutions.

Like in a genetic algorithm (GA), NSGA-II starts with a population of N_{pop} random solutions. In the N-th iteration of NSGA-II, the offspring population Q_N is first created by using the parent population P_N and the usual genetic operators – reproduction, recombination, and mutation [4]. Thereafter, both populations are combined together to form R_N of size $2N_{\text{pop}}$. Then, a non-dominated sorting procedure [2] is applied to classify the entire population R_N into a number of hierarchical non-dominated fronts. Figure 2 shows a schematic of one iteration of NSGA-II. Fronts are marked as 1,2,...in the decreasing level of non-domination of solutions. A property of such fronts is that a solution lying on a better non-dominated front (say $\mathcal{F}$) does not get dominated by any solution of a front worse than $\mathcal{F}$. For M objectives, there exists an algorithm requiring $O(N \log^{M-1} N)$ computations to make such a non-dominated sorting of the entire population of size N [5].

Once the non-dominated sorting of the set R_N is over, the new population is filled with solutions of different non-dominated fronts, one at a time. The filling starts with the best non-dominated front (marked as front 1) and continues with solutions of the second non-dominated front, followed by the third non-dominated front, and so on. Since the overall population size of R_N is $2N_{\text{pop}}$, not all fronts may be accommodated in N slots available in the new population. This scenario is shown in Figure 2. All fronts which could not be accommodated (such as, fronts 4, 5 and 6 in the figure) are simply deleted. When the last allowed front (front 3 in the figure)is being considered, there may exist more solutions in the front than the remaining slots in the new population. Instead of arbitrarily discarding some members from the last front, the solutions which will make the *diversity* of the selected solutions the largest are chosen. In this step, a *crowded-sorting* of the solutions of front i (the last front which could not be accommodated fully) is performed by using a *crowding distance metric* and the adequate number of solutions are picked from the top of the list. The crowding distance of a solution in a non-dominated front is a measure of crowding by other members of the front. In the NSGA-II implementation, a metric totalling the objective-wise distances between neighboring solutions is used. For details, readers are encouraged to refer to the original NSGA-II study [3]. The reproduction operator uses a *constrained tournament* selection method in which two solutions are compared and the winner is declared based on the following three conditions:

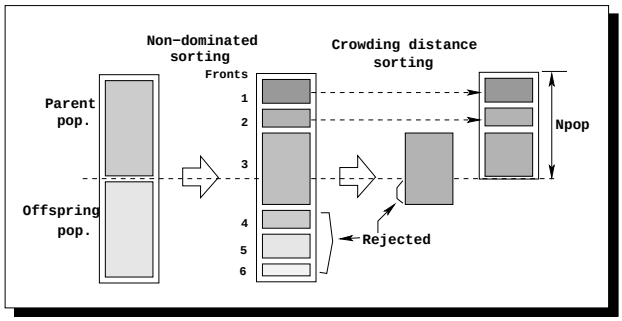

Figure 2: Schematic of the NSGA-II procedure.

1. When both solutions are feasible, the one residing on a better non-dominated front, or the one having a larger crowding distance value wins.

2. When both solutions are infeasible, the one having a smaller overall constraint violation wins.

3. When one solution is feasible and other is infeasible, the feasible solution wins.

The computational complexity of one iteration of NSGA-II is dictated by the non-dominated sorting complexity mentioned above.

NSGA-II algorithm, with the constrained tournament selection, non-dominated sorting of population based on constraint violations and objective function values and crowding in objective value space [2], is employed in the first stage to get a set of non-dominated solutions with a good spread in the objective values.

5.2 Clustering

After NSGA-II run, clustering of the solutions is done to pick up 8 to 10 solutions from the non-dominated front. More details about clustering can be found in [6]. It is worthwhile mentioning here that the extreme solutions (i.e. the individual champions) are always retained in the final set of solutions.

5.3 Local Search

After clustering, a local search is done from each of the clustered solutions. The algorithm is described below:

Let S = current solution under investigation.
Calculate the weights for the two objective functions using the following equation

$$w_j = \frac{(f_j^{max} - f_j(S))/(f_j^{max} - f_j^{min})}{\sum_{k=1}^{M} (f_k^{max} - f_k(S))/(f_k^{max} - f_k^{min})}$$

(For our problem there are two objectives, hence $M = 2$)
Let $COUNT = 0$
while ($COUNT < N$)

M = *mutate*(S)
if$(w_1.f_1(M) + w_2.f_2(M) < w_1.f_1(S) + w_2.f_2(S))$
if(M *has no constraint violation*)
Let $S = M$
endif
end if
$COUNT = COUNT + 1$
end while

For solving the first problem, a value of a maximum limit on evaluations of $N = 10,000$ was used. The local search in most of the cases gives a better non-dominated front after the NSGA-II search and clustering.

6 Simulation results

The above algorithm is run on a number of problems having terrains of varying shapes. The algorithm is successful in finding feasible optimal paths in almost all cases. Here, the results of the simulation runs on a few of the problems are presented.

6.1 Results for the problem in which no other constraint is assumed

The following parameters are used for solving the Case 1 :

- Population size = 100
- Maximum number of generations = 150
- Crossover probability = 0.8
- Mutation probability is fixed as the reciprocal of the number of variables used in the chromosome.
- Degree of B-Spline curve = 5
- Minimum allowed angle between two successive discrete segments of the B-Spline curve = 150 °.
- r_{safe} (in eq. 3) = 1.0

Results for two sample problems are shown here. For both the problems a terrain with a 50×50 mesh is used.

6.1.1 Problem 1

In the first problem, the UAV is required to traverse over a hill which is surrounded by valleys on two sides. For this problem seven control points (two fixed and five free-to-move control points) are used. Step 1 of the proposed hybrid algorithm has found 30 different trade-off solutions, as shown with circles in Figure **??**. Step 2 chooses a few well-dispersed solutions from this set using a K-mean clustering method. Thereafter, in Step 3, the local search method improves each of these solutions and finds eight non-dominated solutions which are marked using triangles in Figure **??**. The final eight

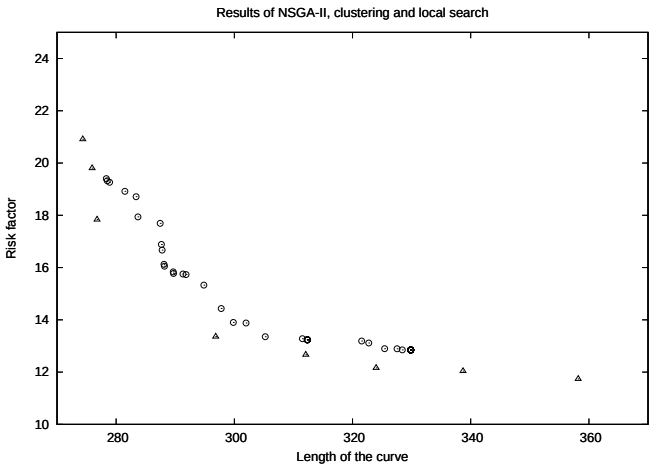

Figure 3: The plot of the non-dominated front obtained using NSGA-II (shown using circles) and after clustering and local search (shown using triangles) for Problem 1.

solutions are better converged and better distributed than the solutions obtained at the end of Step 1. Variations of 33% and 75% in length of curve and risk factor, respectively, to the individual minimum solutions are observed among the obtained Pareto-optimal solutions. A local search coupled with an evolutionary algorithm is a better approach than using evolutionary or a local search method alone.

The path planner generated eight non-dominated solutions of which three are further investigated. The path with the smallest length is shown in Figure 4. As expected, the path is nearly a straight line (causing smallest distance), and passes over the hill to reach the destination point. A more safe path is shown in Figure 5. In this figure the path avoids the peak of the hill and tends to go along the valleys. In the final figure shown in Figure 6 which has the minimum risk factor, the path completely avoids the hill and goes almost exclusively through the valley. The trend in the change of the shape of the curve clearly demonstrates that the algorithm is successful in obtaining paths with different trade-offs in the objective function values. The availability of such trade-off solutions provides different possibilities of achieving the task and helps a user to compare them and choose one for a particular application.

6.1.2 Problem 2

For the second problem, a wider terrain with a number of hills is chosen. In this case nine control points (two fixed and seven free-to-move control points) are chosen. A low upper limit for the curve points is used in this problem. Figure **??** shows the NSGA-II solutions and the final modified solutions after the local search. Once again, the local search is able to find a much wider and better-converged set of solutions.

As expected, the shortest path generated, as shown

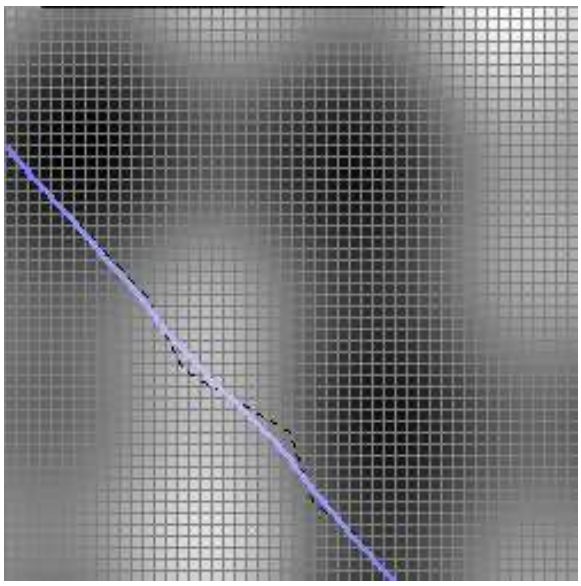

Figure 4: Path of minimum length for Problem 1. However this path is quite risky as it goes near the peak of the hill.

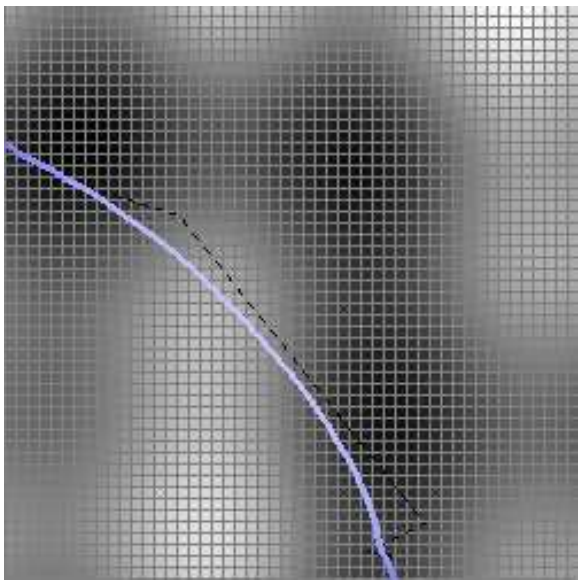

Figure 5: A safer path for Problem 1. The path tends to avoid the hill and goes more through the valley.

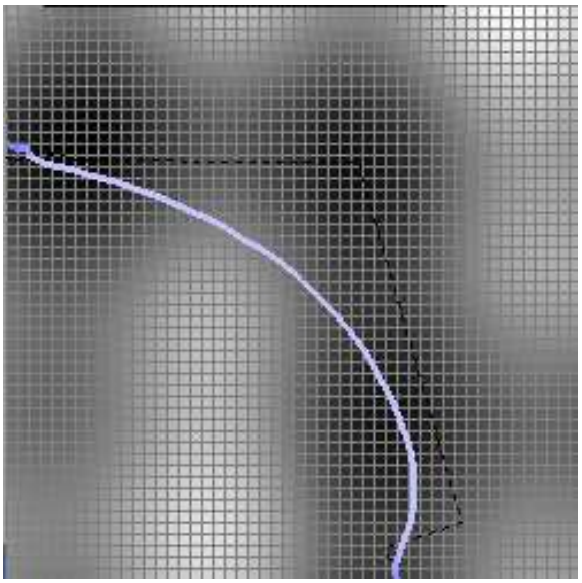

Figure 6: Safest path for Problem 1. The path completely avoids the hill and goes almost exclusively through the valley.

in Figure 8, is nearly a straight line. But this curve has a high risk factor, as it passes very close to the terrain boundary. The path shown in Figure 9 is a safer path (though a longer one) as it goes more through the valleys. Finally the path shown in Figure 10 is the longest one, though it is the safest as it is at a safe distance from the terrain boundaries, as visible from the figure.

6.2 Results for the case in which the UAV has to pass through particular point

The parameters chosen for this problem are the same as in the previous case. Results for two sample problems are presented here. The point through which the UAV has to pass is shown with a black dot in the figures.

6.2.1 Problem 3

Nine control points are used for defining the B-Spline curves. Out of these nine control points, two were the starting and the end point, one was the point through which the UAV has to compulsorily pass, and three other control points are there in the two segments of the B-Spline curve. Figure **??** shows 10 clustered trade-off solutions obtained using NSGA-II. The trade-off between the two objectives is clear from this figure. In this problem, a local search from these clustered solutions did not produce any better solutions. Fixing a point along the path imposes a strict constraint on the shape of the path. As a result, a simple mutation-based local search method is not able to find any better solution to this problem.

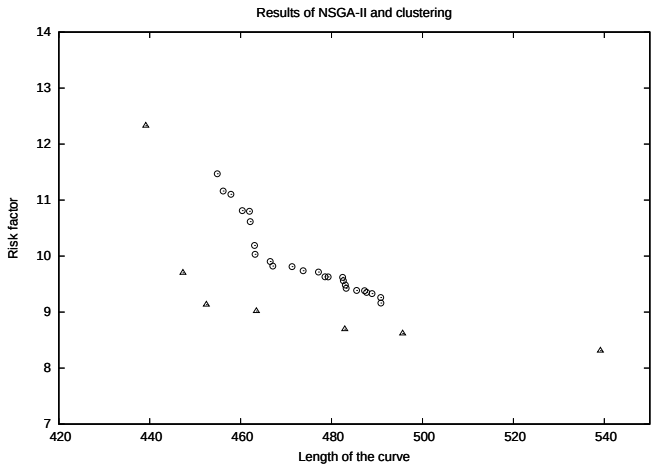

Figure 7: The plot of the non-dominated front obtained using NSGA-II (shown using circles) and after clustering and local search (shown using triangles) for Problem 2.

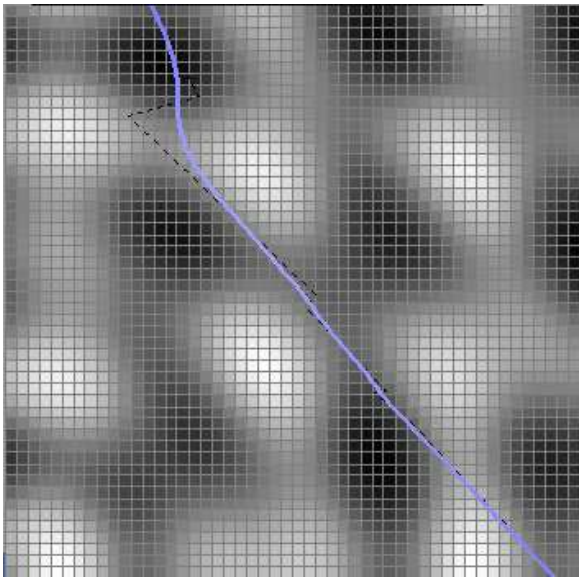

Figure 8: Path of minimum length for Problem 2. This is a risky path as it goes quite close to the terrain boundaries.

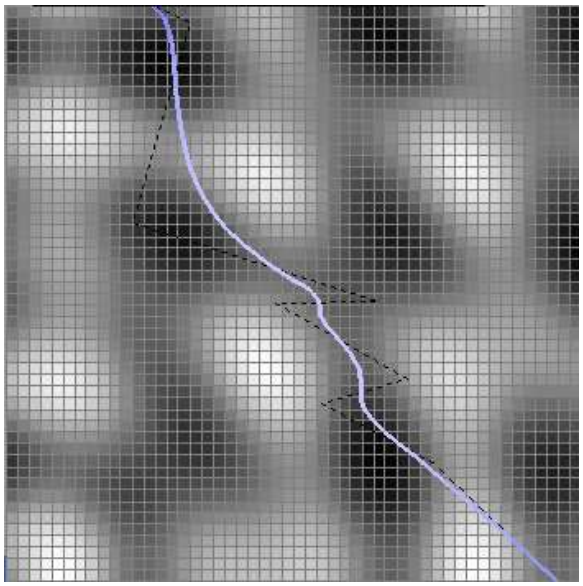

Figure 9: A safer path for Problem 2. A relatively safer path as it avoids the hills.

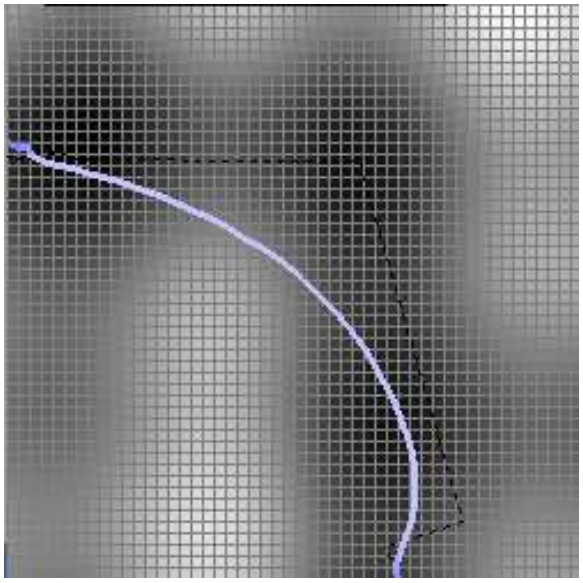

Figure 10: The safest path for Problem 2. The path totally avoids the hills and goes through the valleys.

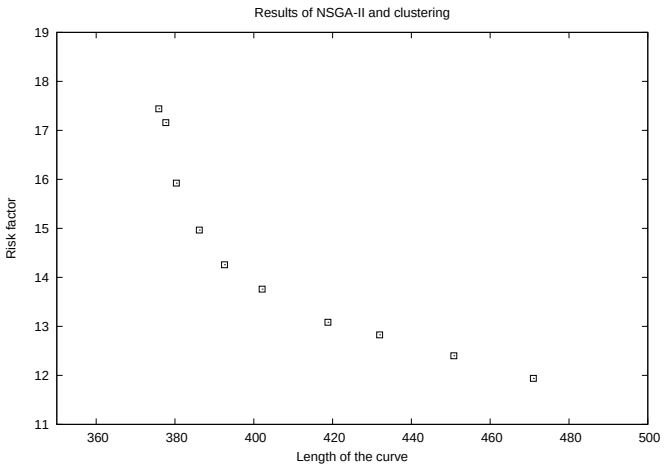

Figure 11: The plot of the non-dominated front obtained after NSGA-II and clustering for Problem 3.

The shortest path is shown in Figure 12. In this path the two segments (from the starting point to the fixed point, and from the fixed point to the destination point) are almost straight lines, as expected. However, the paths in Figure 13 and Figure 14 tend to go round the hills in the terrain and are therefore safer paths as compared to the previous short path.

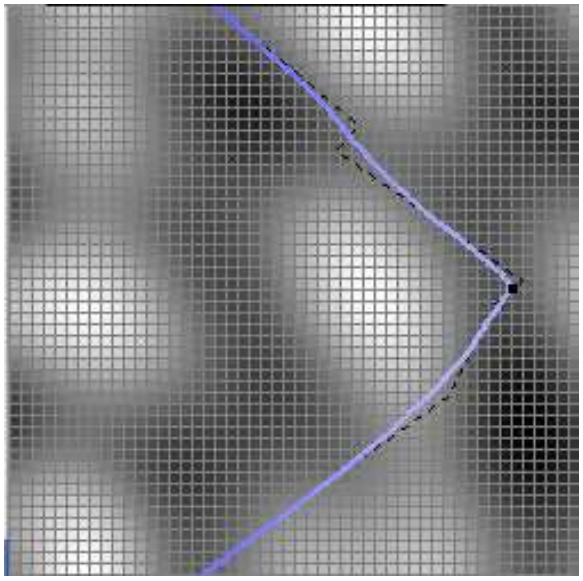

Figure 12: Path of minimum length for Problem 3. The point through which the UAV has to compulsorily pass is shown with a black dot.

6.3 Problem 4

The next problem also uses nine control points for describing the B-Spline curve, however a large number of hills and valleys are introduced to make the problem more difficult. Figure ?? shows nine trade-off solutions. Since a small value of the upper height limit is imposed on the path, the shortest length path does not consist of two straight line segments, but has two segments which are more curved going round the hill. Three paths, illustrating the minimum-length solution, minimum-risk-factor solution and a compromised solution, are shown in Figures 16, 17 and 18, respectively. Although the minimum-length solution seems to have a sharp turn at the specified point, the curvature at this point is well within the specified minimum curvature.

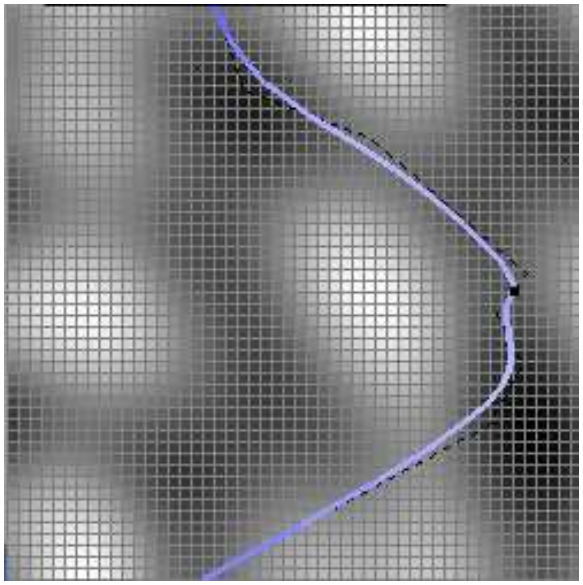

Figure 13: A longer but a relatively safer path for Problem 3.

7 Conclusions

Using multiobjective evolutionary optimization techniques, two problems have been solved here: one in which the UAV can take any path, and the other in which the UAV has to compulsorily pass through a particular point in the three-dimensional space.

The results clearly demonstrate the power of using the hybrid multiobjective evolutionary technique in solving the given problem. This method does not only generate one single optimal solution, but a number of Pareto-optimal solutions are evolved. Different solutions have

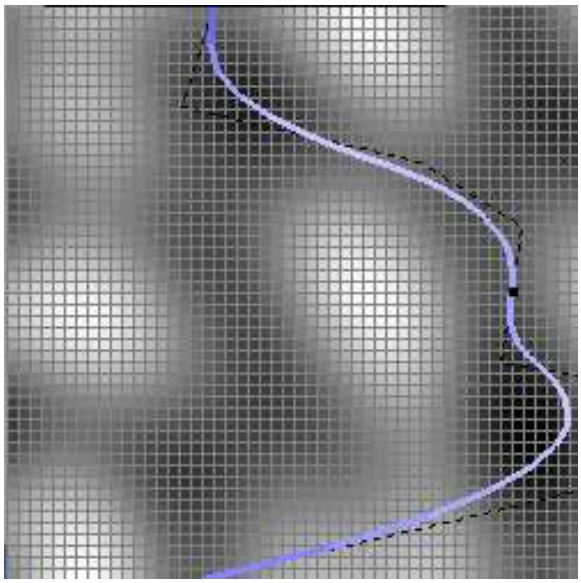

Figure 14: Safest path for Problem 3. Note how this path avoids the hills and goes through the valleys.

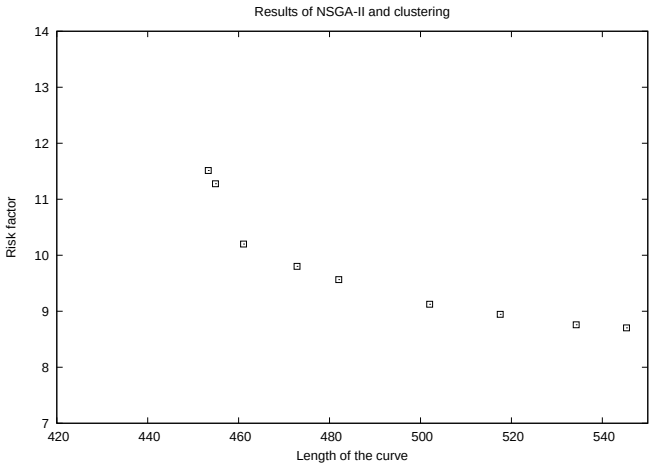

Figure 15: The plot of the non-dominated front obtained after NSGA-II and clustering for Problem 4.

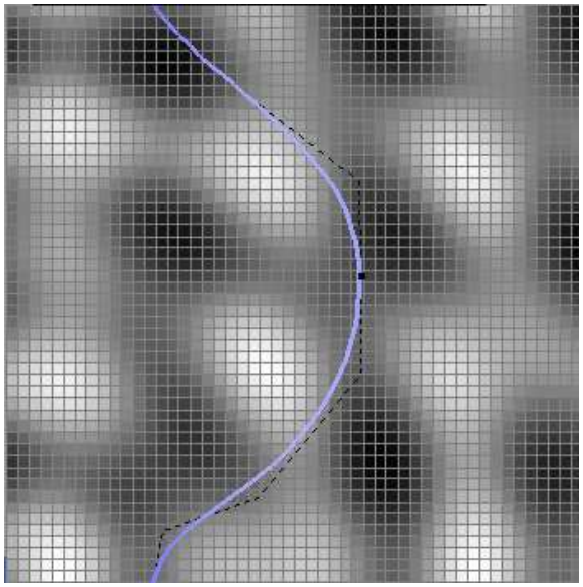

Figure 16: Path of minimum length for Problem 4.

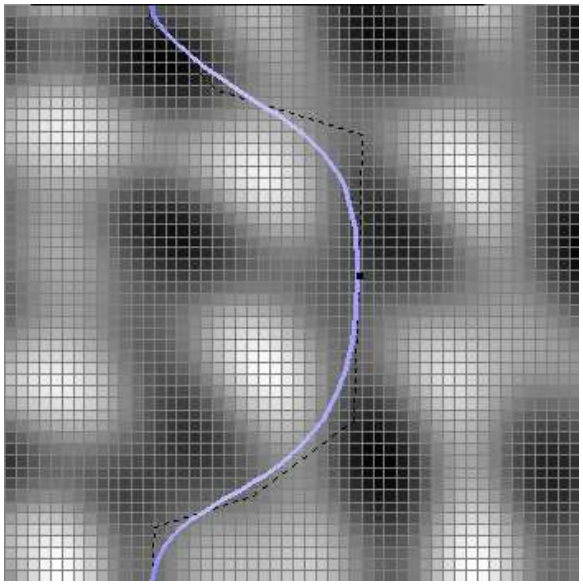

Figure 17: A relatively safer path for Problem 4.

different trade-offs in the objective values. Therefore, a user who has been assigned the task of path planning for the UAV can choose the path best suited to his needs and the capabilities of the UAV. If a minimum flight time is desirable, then (s)he can choose the path with the shortest length. If safety is the main concern, then (s)he can choose the longer path which avoids the terrain boundaries to a large extent. Thus, this procedure provides a lot of flexibility to the user who has to plan a suitable path for the UAV. Also the above results demonstrate that the algorithm can find optimal paths even in complicated terrains.

The problem of finding an optimal path is highly multimodal, that is, there are can be numerous paths with the same objective values. This poses a serious difficulty to the algorithm in finding optimal paths. In general, the paths obtained do not show multimodality, that is, almost all the paths take the same route to reach the destination point. Although the algorithm works pretty well in the sense that it is able to generate trade-off solutions, it fails in finding multimodal solutions. Nevertheless the path finding algorithm is a good one, and generates optimal paths very efficiently.

References

[1] I. K. Nikolos, K. P. Valvanis, N. C. Tsourveloudis and A. N. Kostaras, "Evolutionary Algorithm

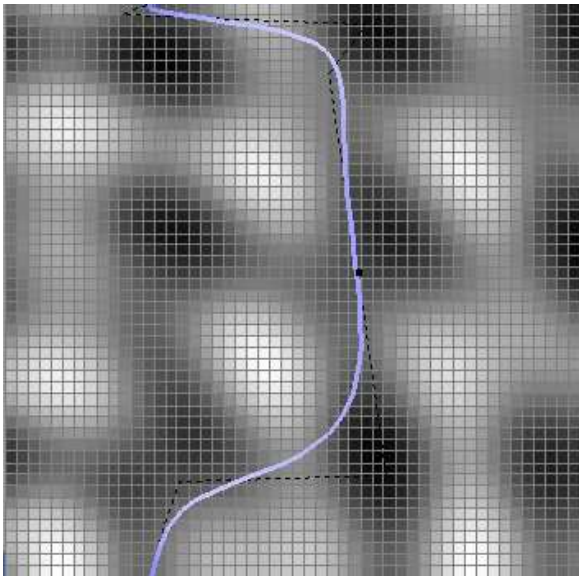

Figure 18: Safest but comparatively longer path for Problem 4.

based Offline/Online Path Planner for UAV Navigation", in *IEEE Transactions on Man, Systems and Cybernetics - Part B: Cybernetics*, 2003, vol. 33, pp. 898-911.

[2] K. Deb, *Multiobjective Optimisation Using Evolutionary Algorithms.* Chichester, U.K.: Wiley, 2001.

[3] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, "A fast and elitist multiobjective algorithm: NSGA-II", in *IEEE transactions on Evolutionary Computation*, 2002, vol. 6, pp. 182-197.

[4] D. E. Goldberg, *G*enetic Algorithms for Search, Optimization, and Machine Learning, Reading, MA: Addison-Wesley, 1989.

[5] M. T. Jensen, "Reducing the run-time complexity of multiobjective EAs", *I*EEE Transactions to Evolutionary Computation, 2003, Vol. 7, No. 5, pp. 503–515.

[6] E. Zitzler, "Evolutionary algorithms for multiobjective optimisation: Methods and applications", Doctoral Dissertation ETH 13398, Swiss Federal Institute of Technology(ETH), Zurich, Switzerland, 1999.

[7] D. Hearn and M. P. Baker, *Computer Graphics.* Pearson Education, 2003, pp. 335-345.

[8] Xiao Jing, Z. Michalewicz, Zhang Lixin and K. Trojanowski, "Adaptive evolutionary planner/navigator for mobile robots", in *IEEE transactions on Evolutionary Computation*, 1997, vol. 1, pp. 18-28.

[9] M. Jun and R. D'Andrea, *Cooperative Control: Models, Applications and Algorithms* Kluwer, Chapter 6.

[10] A. Dogan, "Probabilistic Path Planning For UAVs", in *American Institute of Aeronautics and Astronautics*, 2003.