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

Shashi Mittal and Kalyanmoy Deb

Abstract— In this paper, we present 3D offline path planner for Unmanned Aerial Vehicles (UAVs) using Multiobjective Evolutionary Algorithms for finding solutions corresponding to conflicting goals of minimizing length of path and maximizing margin of safety. 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 decision 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 and ii) Path planning for UAV if the vehicle has to necessarily pass through a particular point in the space. The use of a multiobjective evolutionary algorithm 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, thereby making the approach more pragmatic. Although an automated decision-making aid is the next immediate need of research, we defer it for another study.

I. 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 planning such a path, often an optimization problem is constructed either for minimizing the length of path or for maximizing a safety margin from ground obstacles, or others [11]. Other techniques such as a fuzzy-logic strategy [7] or a graph-based strategy [1] are also used. However, the path planning problem is truly a multi-objective optimization problem in which conflicting goals of minimizing the length of path and simultaneously maximizing the margin of safety can be simultaneously important. There exists no past studies in which the path planning problem is considered as a truly multi-objective optimization task.

In this paper, 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 start point and the end point of the flight are known 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 ground 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

S. Mittal is a graduate student at MIT Operations Research Centre, USA (email: shashi.mittal@gmail.com)

K. Deb holds Shri Deva Raj Chair Professor, Department of Mechanical Engineering, Indian Institute of Technology, Kanpur, India (email: deb@iitk.ac.in)

traverse a path with large curvatures. To reduce complexity of computing collision avoidance with the ground obstacle, the UAV is assumed to be a point and the ground obstacle boundary is modified with the size of the UAV. In the present study, the path of the UAV is assumed to be a B-spline curve, which is controlled by manipulating a number of control points. The following two problems are mainly tackled:

1) **Type I: UAV navigation using an offline path planner when no restriction on the path.** 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.

2) **Type II: UAV navigation using an offline path planner when the vehicle has to pass through one or more pre-specified points 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, thereby requiring the UAV to travel through a particular region or point. In this case also, the path planner generates a single B-Spline curve, but enforces that the specified points lies on the generated B-spline curve.

Unlike most path planning strategies, the proposed path planing strategy considers a multiobjective formulation of the path planing task and uses an evolutionary algorithm. The advantage of using such a method is that it generates not just one but many optimal (called as *Pareto-optimal*) solutions (paths), each with a different trade-off among multiple objective functions. This provides the user with more flexibility, since the user now has a knowledge of different possibilities of travel and can choose a particular path of his/her choice which is best suited for the task.

II. PAST STUDIES ON THREE-DIMENSIONAL PATH GENERATION

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], [12], [13]. For an UAV navigation, there exist methods for finding optimal paths using Voronoi diagrams [9], [5]. 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 [11]. This study also uses B-Spline curves for representing paths in 3-D space.

However, all the existing methods for path planning do not take into account the fact that the problem involves a number of conflicting objectives which must be taken into consideration while generating a path. While the time of flight from start to end is one main consideration, distance of the UAV from any ground obstacle (such as a mountain or a structure) or forbidden regions (both in ground or in space) are also important considerations.

In the presence of such multiple conflicting objectives, the existing literature converts the problem into a single objective optimization problem by using one of the two main procedures [10], [3]. In the ϵ-constraint method, only one of the objectives is chosen and rest are used as constraints so that they are restricted to lie within a safe limit. For example, in such a conversion procedure, the time of flight can be minimized by restricting a safe distance (say, d_{crit}) of all intermediate positions of the UAV from corresponding ground obstacle. In the weighted-sum approach, on the other hand, all objectives are aggregated together to form a combined objective. In this approach, a weighted average of time of flight and distance margin of the UAV from ground obstacles can be minimized. It is intuitive that both these methods involve setting of some artificial parameters (such as limiting value d_{crit} used as constraints in the ϵ-constraint approach or relative weights used to combine multiple objectives in the weighted-sum approach). The optimum solution to the resulting single objective optimization problem largely depends on these chosen parameters. More importantly, a setting of these parameters for an important UAV flight task is not often easy, particularly in the absence of any known solution to the problem. We explain some of these matters in the following paragraphs.

There are at least a couple of implementational issues which we discuss first:

1) The conversion of two objectives into one requires that both objectives are of same type (either both are of minimization type or both are of maximization type). In this case, an optimal UAV path generation will require minimization of time of flight and simultaneous maximization of safe distance margin, thereby requiring one of the objectives to be converted to the other type by using the duality principle [2].

2) Since a weight-vector (for example, a weight-vector $(0.75, 0.25)$ signifies the minimization of time of flight is *three* times more important than the maximization of safe distance margin) is to be used for the conversion procedure, the objectives must also be normalized so that they are of similar magnitudes before a weighted average is computed.

Similar issues exist with other approaches such as the ϵ-constraint approach in terms of fixing limiting values for constraints.

Besides the above implementational issues, there is a deeper problem which we discuss next. It is important to mention that the fixing of the weight-vector (or ϵ-vector) above must be done before an optimization algorithm is applied and before any optimal solution is obtained. Thus, an user has to entirely guess or rely on his past experience of assigning a relative weight (or ϵ) vector to combine important objectives. This is certainly a difficult task. After obtaining the optimal solution corresponding to the chosen weight vector (say $(0.75, 0.25)$), the user will always wonder what other optimal path would have been generated if (s)he had specified a slightly different weight vector, such as $(0.70, 0.30)$, for example! Moreover, a responsible and reliable user would also like to know how the path would change if an entirely different weight vector (such as $(0.25, 0.75)$) is chosen, instead. The corresponding optimal solutions will provide valuable information of solving the task. In the presence of such paths, each trading off the objectives in a different manner, the user will be in a better position to choose a particular path.

In this paper, we suggest a NSGA-II based procedure for finding multiple such paths in a single simulation. Next, we describe the representation scheme for specifying the terrain and a UAV path in our optimization algorithm.

III. REPRESENTATION OF TERRAIN AND UAV PATH

Our work uses several of the models employed in [11]. In particular, the representation of the terrain and the B-Spline curves representation for UAV paths is the same as in [11]. The representation of the terrain is the same as in [11]. For UAV path, as it has been mentioned earlier, B-Spline curves are used. The details of both these are given below.

A. 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. Such terrains have also been previously used by other researchers [11].

Note that in the figures, the light areas represent greater height and dark areas represent lower height.

B. 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 [6]. 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 be easily 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 smooth, without any kinks or breaks.

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

The start and the end points of the path are two of the control points of the B-Spline Curve, and rest of the control points (also called *free-to-move control points*) are generated by the path planning algorithm. Using the control points, the curve is generated as a sequence of discrete points, so that the length of the curve and other objective functions can be calculated.

B-Spline curves also provide us an easy model for representing curves to solve Type II tasks, in which the path has to pass through a pre-specified point in the space. 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 $0 < k < n$ and $P_k = P$. Then, if the distance between P_{k-1} and P_k is identical to that between P_k and P_{k+1}, or $d(P_{k-1}, P_k) = d(P_k, P_{k+1})$ and the control points P_{k-1}, P_k, P_{k+1} are collinear, the B-Spline curve is guaranteed to pass through the point P. Using such a model for the curve, Type II problems can be easily tackled, to force the path to traverse through one or more points in the 3-D space. We discuss this aspect in Section V-A.

IV. OBJECTIVE FUNCTIONS AND CONSTRAINTS

The optimization problem to be solved is the minimization of two functions subject to three constraints. In this section, we analyze in detail its components.

A. Objective functions

We consider two objective functions which are conflicting with each other.

1) Length of the curve: The first objective function , f_1, 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.

2) Risk factor: The goal of constructing the shortest path clashes with the desire of actually reaching the intended destination. The closer the UAV flies to the terrain, the higher is the risk of crashing. We therefore introduce a second objective function, f_2, to capture this desired property of the solution. It has the following form

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

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.
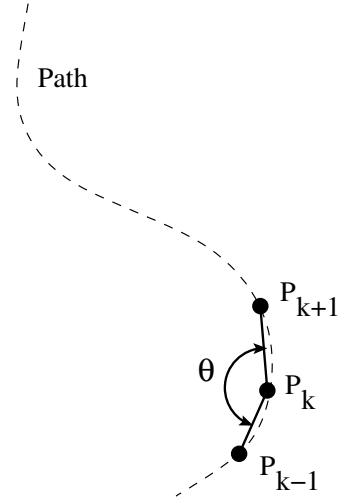

Fig. 1. Schematic representation of the curvature angle used in constraint 2.

B. Constraints

The optimization problem is subject to the following three constraints

1) The vehicle must not collide with the terrain boundary in the course of its flight.
2) The UAV must not perform abrupt changes of direction. We impose this constraint by requiring that the angle between the two successive discrete segments of the curve (Figure 1) should not be less than a certain cut-off angle.
3) The overall 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.

V. PROPOSED PROCEDURE FOR PATH PLANNING

The path planner employs the following three-step hybrid algorithm for finding multiple optimal paths. In the heart of the algorithm is the non-dominated sorting multiobjective optimization algorithm (NSGA-II) [4]. The algorithm is capable of finding a number of optimal paths (depending on the chosen population size used in the simulation), trading off two objectives differently. But for convenience of the users, only about 8-10 well trade-off solutions are selected, and a local search is performed on each of these solutions to improve the optimality property of the obtained paths. The non-dominated solutions so obtained after the local search are finally displayed using a graphics display system.

A. Genetic encoding of B-Spline Curves

The coordinates of the free-to-move control points (that is, the control points excluding the starting and the end points and the pre-specified points in the case of Type II problems) of the B-Spline curve are the variables used in the genetic algorithm. Each control point in 3-D space is represented using three values, one each along the (X, Y, Z) axis. Thus, if there are n free-to-move control points, then a solution will involve $3n$ real-parameter variables, as a three-tuple for each control point in a successive manner, as shown below:

$$(\underbrace{(X_1, Y_1, Z_1)}_{P_1}, \underbrace{(X_2, Y_2, Z_2)}_{P_2}, \ldots, \underbrace{(X_n, Y_n, Z_n)}_{P_n})$$

While computing the entire path from start to end, the start and end points are added in the above list at the beginning and at the end, respectively, and the B-Spline formulation is used to get a mathematical formulation of the entire path for the UAV.

For the Type II problems with one pre-specified point P, the control point ($P_m = (X_m, Y_m, Z_m)$) in the middle of the solution vector ($m = \lceil n \rceil$) is replaced by the pre-specified point P. To ensure the UAV path passes through the point $P_m = P$, we replace $(m + 1)$-th control point as follows:

$$P_{m+1} = 2P_m + P_{m-1}. \tag{3}$$

For more than one pre-specified points, a similar fix-up of the GA solution vector can be accomplished.

B. NSGA-II Operators

NSGA-II procedure is continued till a termination criterion is met. Usually, the NSGA-II procedure is continued for a predefined number of iterations ($T_{\max}$). For recombination, we use the standard SBX operator with $p_c = 0.8$ and a distribution index of $\eta_c = 10$. The following mutation operator is employed:

$$y = x + \eta(x_U - x_L)(1 - r^{(1-t/T_{max})^b}), \tag{4}$$

where y is a member in the offspring solution vector, x is the corresponding member in the parent solution vector, r is a random number in the range $[0, 1]$, η takes a Boolean value ($\{-1, 1\}$), each chosen with a probability of 0.5. The parameter T_{max} is the maximum number of allowed iterations, while b is a user-defined parameter, generally set to 1.0. x_U and x_L are the upper and the lower limits for the variable, respectively.

C. Clustering

If run for an adequate number of iterations, the NSGA-II procedure is likely to find N non-dominated solutions, having trade-off between the objectives. Since usually a population is sized with $N = 100$ to 500 solutions, so many non-dominated solutions are inconvenient for the user of the UAV path generation task to consider, particularly when only one solution has to be chosen from the whole set of N solutions. In this paper, we suggest a clustering technique in which only a handful (say, 8 to 10 solutions) of them

are chosen in a manner so that they are well-dispersed from each other on the objective space. Here, we follow a K-mean clustering algorithm for this purpose.

In the K-mean clustering method applied to N solutions, each solution is assumed to belong to a separate cluster. Thereafter, inter-cluster distances (in the objective space) are computed for each pair of clusters and the two clusters with the smallest distance are merged together to reduce the total count of clusters by one. The procedure is continued till the required number of 8 or 10 clusters are left. Finally, only one representative solution from each cluster is retained and the rest are ignored. For the boundary clusters, a suitable extreme solution is retained and for an intermediate cluster a solution in the middle of the cluster is retained.

D. Local Search Procedure

After the NSGA-II procedure finds N non-dominated solutions, the usual K-mean clustering procedure filters out eight to ten solutions. In this procedure, each solution is considered to lie on a separate cluster. Thereafter, a pair of clusters with minimum Euclidean distance are merged together to reduce the number of clusters by one. The same procedure is continued till about 8 to 10 clusters are left. For two clusters having more than one solutions in them, average Euclidean distance between all pairs of solutions between both clusters are used a representative distance between the clusters. Finally, the solution residing near the centroid of all solutions in a cluster is chosen.

After well-spread clusters are found, a local search is applied from each of the chosen clustered solutions for any possibility of further improvement. For each solution, a suitable composite criterion is assigned based on its location on the objective space. The procedure is similar to the weighted-sum approach ($F(\mathbf{x}) = \sum_{j=1}^{2} w_j f_j(\mathbf{x})$), but the weight vector is assigned based on how close the solution (S) is located compared to the optimum of an individual objective [3]:

$$w_j = \frac{(f_j^{max} - f_j(S))/(f_j^{max} - f_j^{min})}{\sum_{k=1}^{2}(f_k^{max} - f_k(S))/(f_k^{max} - f_k^{min})}, \tag{5}$$

where $f_j^{\min}$ and $f_j^{\max}$ are the minimum and maximum function values of the j-th objective, respectively. It is important to note that the above weighted-sum procedure is applicable to convex Pareto-optimal fronts. Fortunately, we observed that minimizations of both length of flight and risk factor in this problem produced a convex trade-off frontier and hence the above procedure is applicable to the problem of this study. For non-convex problems, other suitable composite methodologies, such as *epsilon*-constraint or Tchebyshev metric method [3], [10], can be used, instead.

The local search operation starts with a clustered solution and works by perturbing (or mutating) the solution. If the perturbed solution is better in terms of the assigned composite objective ($F(\mathbf{x})$), the clustered solution is replaced by the perturbed solution. This procedure is continued until no improvement is found in $H = 10,000$ consecutive perturbations. The local search procedure, in general, produces

a better non-dominated front from the front found by the NSGA-II and clustering procedure. It is interesting to note that since the local vicinity of each non-dominated solution is checked by the local search, the resulting solution can be considered to be close to an exact Pareto-optimal solution.

VI. SIMULATION RESULTS

The above hybrid NSGA-II cum local search procedure is applied to 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.

A. Type I: No Intermediate Restriction

The following parameters are used for solving Type I problems:

- Population size, $N = 100$
- Maximum number of iterations, $T_{\max} = 150$
- Degree of B-Spline curve $K = 5$
- Minimum allowed angle between two successive discrete segments of the B-Spline curve = $150°$
- Critical safe distance (in eq. 2), $r_{\text{safe}} = 1.0$

The above parameters were fixed after trial and error, so as to get the optimum performance from the path finding algorithm. The algorithm takes about 3 to 4 minutes to generate the final set of eight to ten different paths when run on a Pentium IV (2.4 GHz) PC. There is no significant variance in the results between different runs of the algorithm starting from different initial populations. This implies that the algorithm presented here is also a robust one.

Results for three sample problems are shown here. For both types of problems, a terrain with a 50×50 mesh is used.

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 finds 30 different trade-off solutions, as shown with shaded circles in Figure 2. The minimum length of curve is 277.54 units and minimum risk factor is 18.53. The trade-off among these solutions is clear from the figure. If any two solutions are compared from this set, one is better in one objective, but is worse in the other objective.

Step 2 chooses a few well-dispersed solutions from this set using a K-mean clustering method mentioned earlier. Ten well-dispersed (marked with boxes) are chosen from the set obtained by NSGA-II. Thereafter, in Step 3, the local search method improves each of these 10 solutions and finds eight non-dominated solutions which are marked using triangles in Figure 2. The figure shows the improvement of each solution by using an arrow. The two extreme solutions are improved to two dominated solutions, thereby leaving eight final trade-off solutions. It is clear from the figure that the final eight solutions (joined with dashed lines) constitute a better non-dominated front than that by NSGA-II alone,

meaning that the obtained solutions together are better in terms of both objectives. The final set contains solutions with curve-length varying in the range [276, 337] units and risk factor in the range [12.8, 19.6]. A local search coupled with the evolutionary algorithm is found to be a better approach than using an 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 curve-length (solution A) is shown in Figure 3. As expected, the path is nearly a straight line (causing smallest distance between start to end), and passes over the hill to reach the destination point. A more safe path (solution B) is shown in Figure 4. 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 5 which has the minimum risk factor (solution C), 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 capable in obtaining paths with different trade-offs in the objective function values.
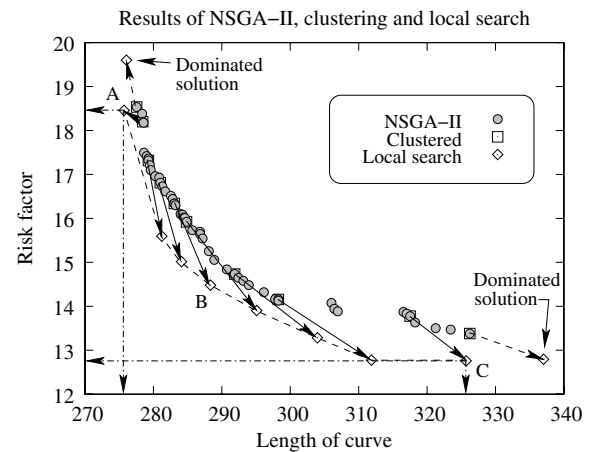

Fig. 2. The plot of the non-dominated front obtained using NSGA-II and after clustering and local search for Problem 1.

Interestingly, these three solutions and five other such trade-off solutions are obtained by a single simulation of an evolutionary multi-objective optimizer (NSGA-II). 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. In these and many other real-world multi-objective optimization problems, it is difficult to find the exact global Pareto-optimal solutions. However, the use of the local search method along with an EMO procedure provides a great deal of confidence about the near-optimality of the obtained solutions.

2) Problem 2: For the second problem, a wider terrain with more than one hill is chosen. In this case, nine control points (two fixed and seven free-to-move control points) are chosen. Figure 6 shows the NSGA-II solutions and the final modified solutions after the local search procedure. Once
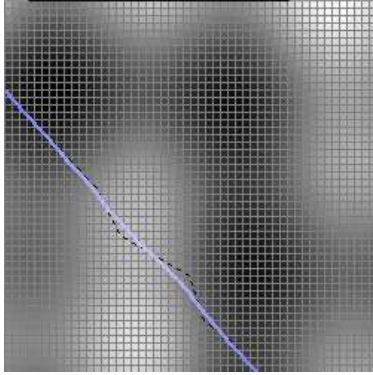
Fig. 3. Path of minimum curve-length (solution A) for Problem 1. However this path is quite risky as it goes near the peak of the hill.
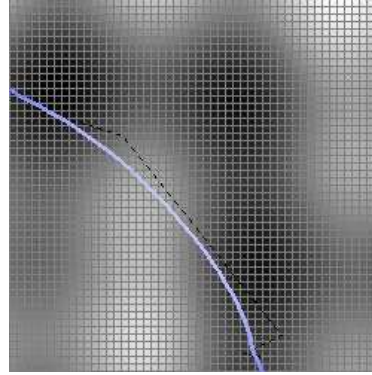

Fig. 4. A safer path (solution B) for Problem 1. The path tends to avoid the hill and goes more through the valley.
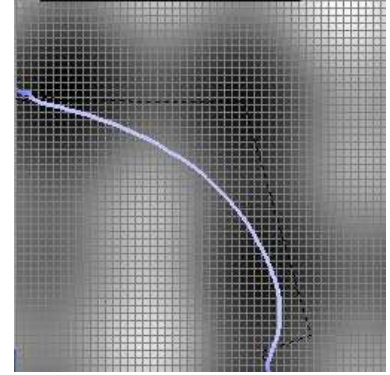

Fig. 5. Safest path (solution C) for Problem 1. The path completely avoids the hill and goes almost exclusively through the valley.
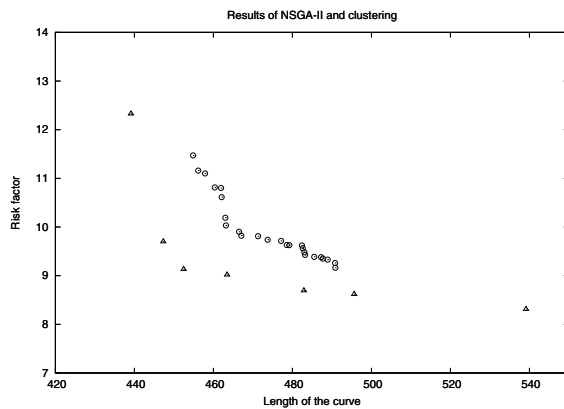

Fig. 6. The plot of the non-dominated front obtained using NSGA-II (shown using circles) and after clustering and local search (shown using triangles) for Problem 2.

again, the local search procedure is able to find a much wider and better-converged set of solutions.

As expected, the shortest path generated, as shown in Figure 7, 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 8 is a safer path (though a longer one) as it goes more through the valleys. Finally the path shown in Figure 9 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.

B. Type II: Path Needs to Pass Through a Specified 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.

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 10 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 solution. 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 overall solution to this problem.

The shortest path is shown in Figure 11. 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. Notice how the proposed procedure is able to locate the three consecutive control points ($m - 1$, m, and $m + 1$) very close to each other so that the overall path appears as almost two straight lines. Although this 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. However, the paths in Figure 12 and Figure 13 tend to go round the hills in the terrain and are therefore safer paths as compared to the previous short path. In these solutions, the three control points are places sufficiently away so as to have a smoother transition through the pre-specified point.

VII. CONCLUSIONS

This work has attempted to solve two problems using a hybrid multiobjective evolutionary algorithm and a local search method concerning UAV path planning: (i) the first one finds a suitable path between two points in the space over a rough terrain when no other constraints are specified, and (ii) the second one restricts that the UAV has to necessarily pass through a particular point in the space. The strategy presented here has attempted to overcome the limitations of the existing path planning procedures involving optimizing a single objective. The objective functions (minimum flight distance and risk factor) have been optimized simultaneously by using the NSGA-II algorithm to generate a set of Pareto-optimal paths. Then about eight to ten well-dispersed solu-
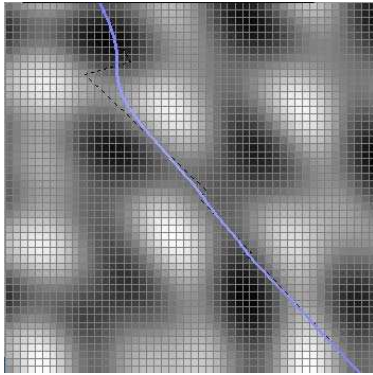
Fig. 7. Path of minimum length for Problem 2. This is a risky path as it goes quite close to the terrain boundaries.
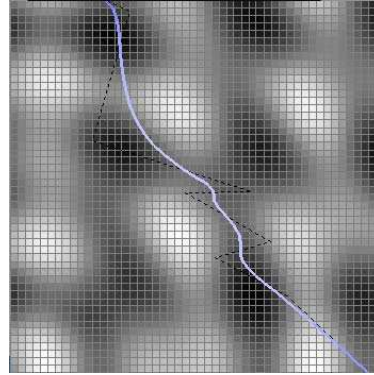

Fig. 8. A safer path for Problem 2. A relatively safer path as it avoids the hills.
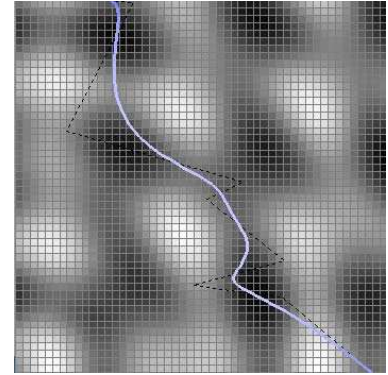

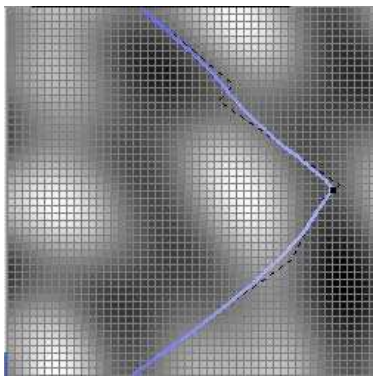Fig. 9. The safest path for Problem 2. The path totally avoids the hills and goes through the valleys.

Fig. 11. Path of minimum length for Problem 3. The point through which the UAV has to compulsorily pass is shown with a black dot.
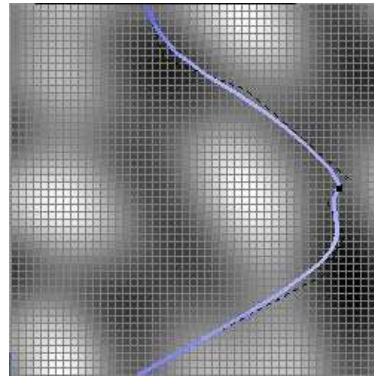

Fig. 12. A longer but a relatively safer path for Problem 3.
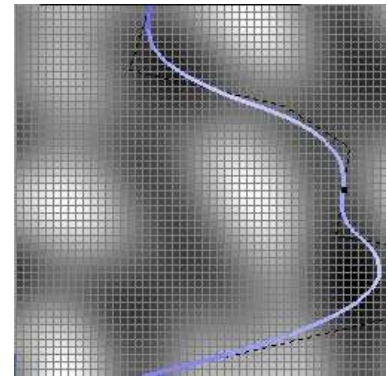
Fig. 13. Safest path for Problem 3. Note how this path avoids the hills and goes through the valleys.

tions have been selected from the NSGA-II solutions and a local optimization has been performed to get solutions as close to the true Pareto-optimal front as possible. Thus, the combined algorithm presents the user with several options for choosing a suitable path for the UAV. If the user is more concerned about the time of flight and confident that the UAV can evade the obstacles, then (s)he can go for the path with the shortest length. If, on the other hand, the user is more concerned about the safety of the vehicle, (s)he can choose the path with a lower risk factor. The multiobjective paradigm of path planning for UAVs is therefore, a more flexible and pragmatic one as compared to the existing methods, and should find more use in path planning technology, as it provides a set of solutions with a decision-making task involved on the part of the user. As an immediate step for further study would be to embed a decision-making tool with the proposed NSGA-II procedure to interactively choose a single Pareto-optimal solution for on-line implementation. This remains as a focus of our future study, which awaits interests from a real application organization, such as military, home-land security, or others.

REFERENCES

[1] W. T. Cerven, F. Bullo, and V. L. Coverstone. Vehicle motion planning with time-varying constraints. *AIAA Journal on Guidance, Control and Dynamics*, 27(3):506–509, 2004.

[2] K. Deb. *Optimization for Engineering Design: Algorithms and Examples*. New Delhi: Prentice-Hall, 1995.

[3] K. Deb. *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley, 2001.

[4] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[5] A. Dogan. Probabilistic path planning for uavs. *American Institute of Aeronautics and Astronautics*, 2003.

[6] D. Hearn and M. P. Baker. *Computer Graphics*. Pearson Education, 2003.

[7] M. Innocenti, L. Polloni, and D. Turra. Guidance of unmanned air vehicles based on fuzzy sets and fixed waypoints. *AIAA Journal on Guidance, Control and Dynamics*, 27(4):715–720, 2004.

[8] Xiao Jing, Z. Michalewicz, Zhang Lixin, and K. Trojanowski. Adaptive evolutionary planner/navigator for mobile robots. *IEEE Transactions on Evolutionary Computation*, 1:18–28, 1997.
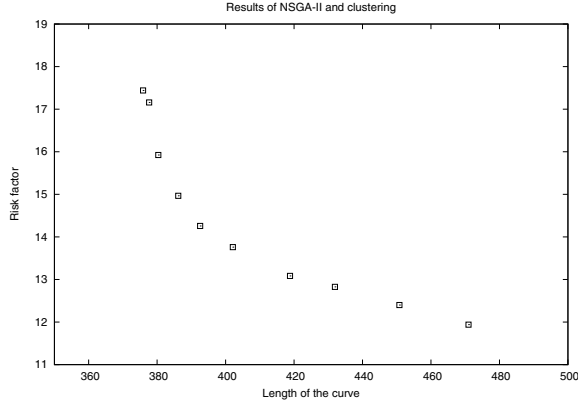
Fig. 10.   The plot of the non-dominated front obtained after NSGA-II and clustering for Problem 3.

[9] M. Jun and R. D'Andrea. *Cooperative Control: Models, Applications and Algorithms*. Boston: Kluwer, 2003.

[10] K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer, Boston, 1999.

[11] I. K. Nikolos, K. P. Valvanis, N. C. Tsourveloudis, and A. N. Kostaras. Evolutionary algorithm based offline/online path planner for uav navigation. *IEEE Transactions on Man, Systems and Cybernetics - Part B: Cybernetics*, 33:898–911, 2003.

[12] D. Pratihar, K. Deb, and A. Ghosh. Fuzzy-genetic algorithms and time-optimal obstacle-free path generation for mobile robots. *Engineering Optimization*, 32:117–142, 1999.

[13] D. K. Pratihar, K. Deb, and A. Ghosh. Optimal path and gait generations simultaneously of a six-legged robot using a ga-fuzzy approach. *Robotics and Autonomous Systems*, 41:1–21, 2002.

APPENDIX

COMPUTING B-SPLINE CURVES FROM CONTROL POINTS

B-Spline curves are parametric curves, and their construction are based on blending functions [6]. Suppose the number of control points of the curve is $(n+1)$, the coordinates being $(x_0, y_0, z_0), \ldots, (x_n, y_n, z_n)$, then the coordinates of the B-Spline curves are given by

$$X(t) = \sum_{i=0}^{n} x_i B_{i,k}(t), \qquad (6)$$

$$Y(t) = \sum_{i=0}^{n} y_i B_{i,k}(t), \qquad (7)$$

$$Z(t) = \sum_{i=0}^{n} z_i B_{i,k}(t), \qquad (8)$$

where $B_{i,K}(t)$ is a blending function of the curve and K is the order of the curve. Higher the order of the curve, smoother the curve. For example, see Figure 14, Figure 15 and Figure 16, shown for different order K of the curve. The parameter t varies between zero and $(n - K + 2)$ with a constant step, providing the discrete points of the B-spline curve.

For our problem, the blending functions have been defined recursively in terms of a set of *Knot* values, which in this case is a *uniform non-periodic* one, defined as:

$$Knot(i) = \begin{cases} 0, & \text{if } i < K \\ i - K + 1, & \text{if } K \le i \le n \\ n - K + 2, & \text{if } n < i \end{cases} \qquad (9)$$
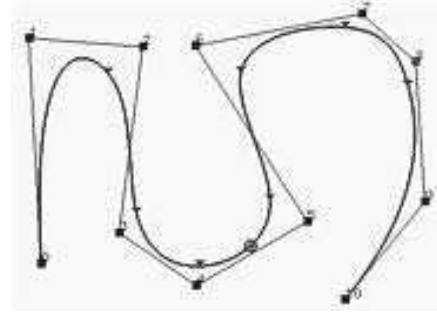

Fig. 14.   B-Spline curves with 11 control points and order 3
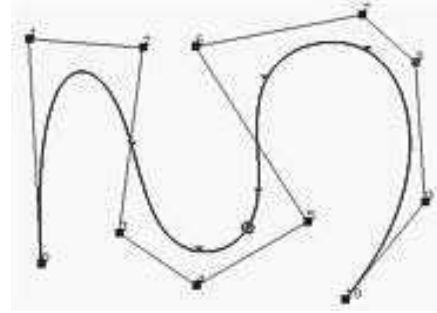

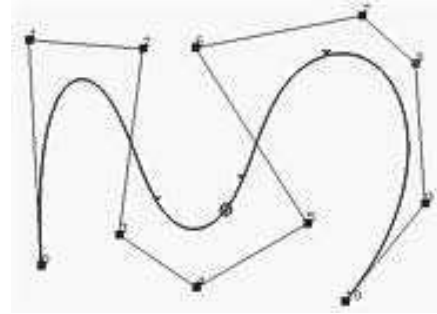Fig. 15.   B-Spline curves with the same set of 11 control points and order 5

Fig. 16.   B-Spline curves with the same set of 11 control points and order 7

The blending function $B_{i,K}$ is defined recursively, using the *Knot* values given above:

$$B_{i,1}(t) = \begin{cases} 1, & \text{if } Knot(i) \le t < Knot(i+1) \\ 1, & \text{if } \begin{cases} Knot(i) \le t \le Knot(i+1) \\ \text{and} \\ t = n - K + 2 \end{cases} \\ 0, & \text{otherwise} \end{cases} \qquad (10)$$

$$B_{i,K}(t) = \frac{(t - Knot(i)) \times B_{i,K-1}(t)}{Knot(i + K - 1) - Knot(i)}$$
$$+ \frac{(Knot(i + K) - t) \times B_{i+1,K-1}(t)}{Knot(i + K) - Knot(i + 1)} \qquad (11)$$

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