Risk-Based Path Planning Optimization Methods for Unmanned Aerial Vehicles Over Inhabited Areas[1]

Eliot Rudnick-Cohen
Department of Mechanical Engineering,
University of Maryland,
College Park, MD 20742

Jeffrey W. Herrmann
Department of Mechanical Engineering,
University of Maryland,
College Park, MD 20742

Shapour Azarm
Department of Mechanical Engineering,
University of Maryland,
College Park, MD 20742

Operating unmanned aerial vehicles (UAVs) over inhabited areas requires mitigating the risk to persons on the ground. Because the risk depends upon the flight path, UAV operators need approaches that can find low-risk flight paths between the mission's start and finish points. Because the flight paths with the lowest risk could be excessively long and indirect, UAV operators are concerned about the tradeoff between risk and flight time. This paper presents a risk assessment technique and bi-objective optimization methods to find low-risk and time (flight path) solutions and computational experiments to evaluate the relative performance of the methods (their computation time and solution quality). The methods were a network optimization approach that constructed a graph for the problem and used that to generate initial solutions that were then improved by a local approach and a greedy approach and a fourth method that did not use the network solutions. The approaches that improved the solutions generated by the network optimization step performed better than the optimization approach that did not use the network solutions. [DOI: 10.1115/1.4033235]

1 Introduction

In the U.S., the use of UAVs by government agencies, commercial enterprises, and others requires mitigating the risk to persons on the ground. A UAV operator must demonstrate that the activity poses little risk, that is, the expected number of persons harmed by the activity must be sufficiently small (less than one fatality per 10×10^6 flight hours [1]). The risk depends upon the size and reliability of the UAV, the weather conditions, the number of persons who are on the ground close to the path of the UAV, and other factors.

Because the risk to persons on the ground depends upon the UAV flight path, UAV operators are interested in approaches (techniques) that can find low-risk flight paths between the start and finish points of the activity. Because the flight paths with the lowest risk could be excessively long and indirect, UAV operators are concerned about the tradeoff between risk and flight time. In some cases, risk acceptance criteria may set an upper bound on the risk; in other cases, UAV fuel capacity or other operational issues may set upper bounds on the time. In general, it is important to find the tradeoffs between these two objectives (risk versus time).

A wide variety of methods exist for solving path planning problems for UAVs [2]. Most methods define some form of cost metric to represent the type of risk being minimized and then formulate the problem as a multiobjective optimization problem where the objectives are the risk metric and another metric representing the length of the path (such as distance traversed along the path or time needed to traverse the path). Examples of types of risk considered in such methods include risk posed due to environmental hazards and terrain [3–5], risk posed due to large-scale obstacles such as radar or heavily populated areas [6,7], the risk of a midair collision [8–10], or the risks to persons on the ground [7].

In general, most methods for solving UAV path planning optimization problems utilize either discrete graph-based planning approaches or mathematical optimization techniques that optimize a fixed number of waypoints. A discussion of methods for solving graph-based planning problems with multiple objectives can be found in Ref. [11]. Many mathematical optimization techniques for risk-based planning utilize evolutionary optimization algorithms [12,13].

The risk posed by a UAV to people on the ground can be described in terms of the expected number of fatalities associated with a given flight, which can be determined by identifying the possible crash locations and multiplying the probability of a UAV crash by the number of people present in the potential crash location [1]. Typically, this is quantified as a two-dimensional probability distribution representing the likelihood of crashing at a certain distance away from the point of the failure. For example, Pikaar et al. [14] used data about historical crashes at airports to generate a crash location distribution for the specific scenarios of takeoff and landing. For the more general case of a UAV in flight, Wu and Clothier used worst case assumptions to bound the potential crash area [15], which can be used as a distribution with the assumption of a uniform distribution in those bounds. Ford and McEntee [16] generated a bivariate crash location distribution using simple assumptions about the flight dynamics of an unpowered UAV. Lum et al. [17] determined a nonuniform distribution of potential crash locations for a particular UAV by performing Monte Carlo simulations of that UAV failing and crashing to the ground.

This paper presents a risk-based optimization approach for exploring the tradeoffs between the risk to persons on the ground and flight time and describes the results of a computational study that evaluated the performance of these optimization algorithms for some specific instances. The approach is a novel combination of multiple elements: (1) a flight dynamics model that predicts the crash location for a UAV that loses power at a given altitude and velocity, (2) a Monte Carlo simulation to generate a probability distribution of crash locations, (3) a risk assessment method that incorporates the crash location distribution (not the worst case) and the population density near the flight path (based on census

[1]An earlier version of this paper appeared in the Proceedings of the ASME 2015 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE 2015) in Boston, MA.

Contributed by the Design Engineering Division of ASME for publication in the Journal of Computing and Information Science in Engineering. Manuscript received June 3, 2015; final manuscript received March 22, 2016; published online April 27, 2016. Editor: Bahram Ravani.

data), (4) an efficient algorithm for finding a flight path that minimizes both time and risk, (5) two different solution improvement techniques, (6) a bi-objective framework for generating a set of nondominated solutions, and (7) a set quality metric for evaluating and comparing sets of bi-objective solutions.

The rest of the paper is organized as follows: Section 2 formulates the problem, and Sec. 3 describes the solution approaches. Section 4 presents the design of the experiments that were conducted, and Sec. 5 discusses the results. Section 6 presents the summary and conclusions.

2 Problem Definition

Given a start point A, a finish point B, a planned altitude, and the UAV velocity, the objective is to find the UAV's flight plan from A to B to minimize risk and time. In theory, the flight plan can be any continuous path from A to B. However, here, it is treated as a piecewise linear path passing through n waypoints (x_i, y_i). The first waypoint is the start point $(x_0, y_0) = (x^S, y^S)$, and the last waypoint is the end point $(x_{n+1}, y_{n+1}) = (x^F, y^F)$.

In theory, there are no constraints on the locations of the waypoints. In practice, of course, flight plans must avoid different types of restricted airspace, which are ignored in this study (but these could easily be added as constraints if needed). For computational purposes, locations of the waypoints are restricted to remain within upper and lower bounds on the x- and y-coordinates, in order to place a limit on the size of the region being considered.

The total time of a flight path is the sum of the time for each leg. In this study, the time $t(i, i+1)$ equals the distance from (x_i, y_i) to (x_{i+1}, y_{i+1}) divided by the vehicle's airspeed V.

In this study, the risk measure is the expected number of deaths. The total risk for a flight plan equals the sum of the risk for each leg. The risk measure depends upon the population density at the potential crash locations, which are determined by the flight path. This study did not consider the influence of shelter.

3 Optimization Approaches

The risk-based path planning optimization problem has two stages: (stage 1) estimate the probability distribution of the crash location based on planned altitude and velocity of the UAV and (stage 2) determine the flight paths that minimize time and risk.

To obtain a crash location distribution, a Monte Carlo simulation of a UAV crashing is used to generate sample crash locations by randomly perturbing the initial conditions of the UAV randomly about a fixed initial state. A list of the state variables used in the model, the baseline case, and the distributions of the random perturbations can be found in Table 1. To model a UAV crashing, an unpowered UAV with freely moving control surfaces is simulated using nonlinear ordinary differential equations [18,19] and solved numerically using MATLAB's ODE45 solver [20]. Figure 1 shows a typical trajectory. The UAV's crash location was the point at which the UAV's height (z) becomes zero. The simulated UAV's aerodynamic coefficients and physical properties are based on those of a Cessna 182 aircraft [21]. The crash location distribution is discretized and normalized to generate a two-dimensional discrete probability distribution that specifies, for each discrete point in an m-by-m grid, the probability that the UAV will land at that spot. Figure 2 shows a heat map of this distribution. The probabilities of landing in the central cells are much greater than those of other cells, but the small cell size (relative to the lengths of the edges and the size of the census tracts) makes the distribution adequate.

To compute the risk for a single leg of the flight plan, the risk was sampled at the midpoints of N intervals along the leg. Next, the points in the crash location probability distribution are

Table 1 Initial conditions for Monte Carlo simulations

Velocity (m/s)	Mean	Deviation
$\dot{x}$	50	50
$\dot{y}$	0	10
$\dot{z}$	0	10
Position (m)		
x	0	0
y	0	0
z	1524	0
Orientation, Euler angles (deg)		
Φ	0	11.25
Θ	0	11.25
Ψ	0	11.25
Angular velocity (deg/s)		
P	0	11.25
Q	0	11.25
R	0	11.25
Control surface deflection (deg)		
Elevator deflection (δ_E)	0	11.25
Rudder deflection (δ_R)	0	11.25
Aileron deflection (δ_A)	0	11.25
Control surface deflection rates (deg/s)		
Elevator deflection rate ($\dot{\delta}_E$)	0	0
Rudder deflection rate ($\dot{\delta}_R$)	0	0
Aileron deflection rate ($\dot{\delta}_A$)	0	0

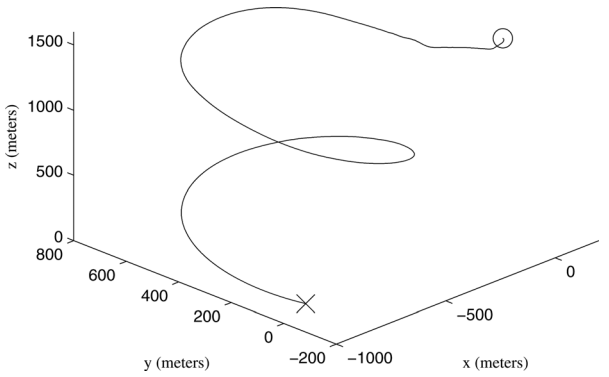

Fig. 1 Example UAV crash trajectory. The circle denotes the start point and the "×" denotes the final crash location.

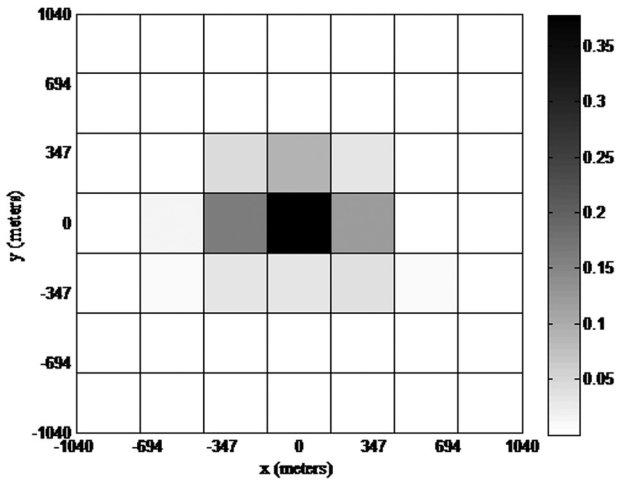

Fig. 2 Discretized heat map of crash density distribution. The scale corresponds to the probability that the vehicle will land in that cell.

rotated by the bearing along the leg for which the risk is being evaluated.

A "cloud" of $(m+N-1)\times m$ points is created as follows:

Step 1: For $a=1,\ldots,m$, the following is carried out:

For $b=1,\ldots,N, \tilde{x}_{ab}=x_b+\Delta x_{a1}, \tilde{y}_{ab}=y_b+\Delta y_{a1}$.

For $b=N+1,\ldots,N+m-1, \tilde{x}_{ab}=x_N+\Delta x_{a,b-N+1}, \tilde{y}_{ab}=y_N+\Delta y_{a,b-N+1}$.

Step 2: If $m\le N$, then the probabilities for each point can be determined as follows:

For $b=1,\ldots,m-1, \tilde{p}_{ab}=\frac{1}{N}\sum_{k=1}^{b} p_{ak}$.

For $b=m,\ldots,N, \tilde{p}_{ab}=\frac{1}{N}\sum_{k=1}^{m} p_{ak}$.

For $b=N+1,\ldots,N+m-1, \tilde{p}_{ab}=\frac{1}{N}\sum_{k=b-N+1}^{m} p_{ak}$.

Step 3: If $m>N$, then the probabilities for each point can be determined as follows:

For $b=1,\ldots,N-1, \tilde{p}_{ab}=\frac{1}{N}\sum_{k=1}^{b} p_{ak}$.

For $b=N,\ldots,m, \tilde{p}_{ab}=\frac{1}{N}\sum_{k=b-N+1}^{b} p_{ak}$.

For $b=m+1,\ldots,N+m-1, \tilde{p}_{ab}=\frac{1}{N}\sum_{k=b-N+1}^{m} p_{ak}$.

Step 4: Loop over the census tracts. For each census tract k, determine which points in the cloud are in that tract's polygon Γ_k and, for $(\tilde{x}_{ab},\tilde{y}_{ab})\in\Gamma_k$, set $\tilde{d}_{ab}=D_k$. Calculate the likelihood of crashing into census tract k

$$\Pi_k=\sum_{(\tilde{x}_{ab},\tilde{y}_{ab})\in\Gamma_k}\tilde{p}_{ab} \tag{1}$$

Step 5: Determine the expected population density along this leg

$$\bar{D}=\sum_{a=1}^{m}\sum_{b=1}^{N+m-1}\tilde{p}_{ab}\tilde{d}_{ab}=\sum_{k}\Pi_k D_k \tag{2}$$

The risk of flying from (x_i,y_i) to (x_{i+1},y_{i+1}) can thus be determined as shown in the following equation:

$$r(i,i+1)=t(i,i+1)\left(\frac{K_1}{100,000}\right)K_2\bar{D}(i,i+1) \tag{3}$$

The approaches used for stage 2 generated a set of flight paths by solving a set of path-planning problems. The overall objective function ("cost") was the weighted sum of the scaled risk and time objectives, as detailed in Eq. (4). This requires a time-weighting constant w_t and a risk-weighting constant w_r. These are non-negative and satisfy $w_t+w_r=1$

$$f(X)=w_t\sum_{i=0}^{n}\frac{t(i,i+1)}{\bar{t}}+w_r\sum_{i=0}^{n}\frac{r(i,i+1)}{\bar{r}} \tag{4}$$

By varying the weights w_t and w_r and minimizing the value of Eq. (4), it is possible to generate a set of different flight paths with the optimization approaches, which include network-based approaches and a non-network approach that used only continuous variable optimization methods.

3.1 Network Optimization Approach. The network optimization step creates a network with a grid of nodes and the start and finish points, evaluates the time and risk of every edge in the graph, and then finds the minimum-cost path from the start to the finish point. The network consists of a uniformly spaced grid of nodes with horizontal spacing $\Delta_x=(x^U-x^L)/(n_x-1)$ and vertical spacing $\Delta_y=(y^U-y^L)/(n_y-1)$ and the points (x^S,y^S) and (x^F,y^F). Nodes outside the census tracts of states being considered in the optimization are deleted. This type of network is chosen for its simplicity, which makes it easy to create.

Each node in the grid is connected with edges going to the eight nodes neighboring it in the grid. In addition, for the points (x^S,y^S) and (x^F,y^F), edges are added from each point to the four corners of the grid element that contained that point.

Next, the time and risk of each edge (i,j) is determined followed by the calculation of the cost (weighted sum of the time and risk) of an edge:

$$c(G((x_i,y_i),(x_j,y_j)))=w_t t(i,j)/\bar{t}+w_r r(i,j)/\bar{r} \tag{5}$$

The network optimization approach finds the minimum-cost path X^N using the Dijkstra's algorithm [22]. Changing the values of the weights w_t and w_r requires only recalculating the edge costs and optimizing; it is not necessary to build the network and evaluate the time and risk of every edge every time.

3.2 Local Improvement Approach. The local improvement approach uses the output of the network optimization step as its initial solution and then finds a nearby solution by solving a continuous variable optimization problem with Eq. (4) as its objective function and subject to the additional constraints defined by Eq. (6) that keeps each waypoint close to a waypoint of the initial solution. The constraints are determined by the tolerances f_x and f_y

$$\begin{aligned} x_i^N-f_x\Delta_x\le x_i\le x_i^N+f_x\Delta_x\\ y_i^N-f_y\Delta_y\le y_i\le y_i^N+f_y\Delta_y \end{aligned} \tag{6}$$

3.3 Greedy Improvement Approach. The greedy improvement approach also uses the output of the network optimization step as its initial solution and then searches for a nearby solution using a continuous variable optimization method subject to the constraints imposed by Eq. (6). However, the greedy improvement approach solves a sequence of n subproblems, one for each waypoint in turn. Each subproblem has only two variables (the coordinates for one waypoint), which is solved relatively quickly, and this requires evaluating the objective function (Eq. (4)) for only two legs: the ones immediately before and after the waypoint being optimized.

3.4 Non-Network Approach. The non-network approach does not require the network optimization step because it uses a straight-line path between the start and finish points as the initial solution. The number of waypoints is fixed (at 5, 10, 14, or 20), and their coordinates are constrained by the lower and upper bounds (not the nodes of the network). In the initial solution, the waypoints divide the straight-line path into legs with the same distance.

4 Experimental Design

Multiple studies were conducted to compare the performance characteristics of the optimization approaches and understand the tradeoffs between the quality of the solutions that were generated and the computational effort required. Throughout these studies, two different scenarios were considered: a flight traveling from Patuxent River Naval Air Station, St. Mary's County, MD, to Camp David, Thurmont, MD (the "Pax River case"), and a flight traveling from College Park Airport in College Park, MD, to Virginia Tech Executive Airport in Blacksburg, VA (the "College Park case").

A set of solutions were generated by solving the problem with different combinations of weights, with $w_t=0$, 0.1, 0.2, …, 1.0, and $w_r=1-w_t$. For the network optimization step, the dimensions of the grid (the number of points in each direction) were varied between several sizes: 30×12, 40×16, and 50×20. (For example, the 30×12 grid began with 360 nodes arranged in 30 columns and 12 rows.) Solutions for the greedy and local improvement approaches were computed for each grid size and for three different values of the tolerance parameters f_x and f_y: 0.25, 0.5, and 0.75 times the size of each grid element. This gave the College Park case horizontal edge lengths (deg longitude) (Δ_x) of 0.2989, 0.2223, and 0.1769 with vertical edge lengths (deg latitude) (Δ_y) of 0.3707, 0.2875, and 0.2146 for the 30×12, 30×16, and 50×20 grid sizes, respectively. The edge lengths in the Pax River case for the same grid sizes were 0.3092,

0.2299, and 0.1830 for horizontal edges and 0.5414, 0.3970, and 0.3134. The non-network-based approach was used to generate solutions with 5, 10, 14, and 20 waypoints. MATLAB's fmincon [23] function was used to solve the continuous optimization problems in the local improvement, greedy improvement, and non-network-based approaches.

Each approach generated a set of solutions (one for each value of the weights, see Eq. (5)). To quantify and compare the quality of a set of solutions, we developed a closeness metric based on the method detailed in Ref. [24]. To calculate this metric, we scaled the time and risk of every solution generated for that case so that the scaled time and risk ranged from 0 to 1. The metric can be defined as the left-handed Riemann sum of the points comprising a Pareto frontier with two additional points added to the frontier at (max objective 1, min objective 2) and (min objective 1, max objective 2) (where the min and max objective function values are relative to all Pareto frontiers being compared), these two additional points represent the worst case values for any regions not covered by the Pareto frontier being evaluated. Note that if the values of each objective function are scaled onto [0,1] using a min–max scaling, these two added points become (1,0) and (0,1). A lower value for this closeness metric will represent a higher quality solution as the solution set will be closer to the ideal point of (0,0).

5 Results

The results were generated using a computer equipped with an Intel i5 2400 processor and 4 GB RAM. We used MATLAB's fmincon with its default tolerances and the active set method. When generating the crash distribution, the relevant error tolerances in MATLAB's ODE45 solver were 10^{-3}. For each case, three grids were generated. For each grid, the network optimization and the local and greedy improvement approaches were used, each with three different values for the tolerances (which yielded seven sets of solutions per grid and 21 network-based sets of solutions). The non-network approach was also used with four different values for the number of waypoints, which generated four more sets of solutions. Thus, there were 25 sets of solutions for each case. Figure 3 shows the average computation time required for each approach (the average is taken over the different values for the weights) and the closeness of the sets of solutions that were generated for the College Park case.

The results displayed in Fig. 4 show that different approaches generate very different sets of solutions. For the College Park case, the network optimization approach generated a variety of solutions, including some with moderate values of both time and risk, as shown in Fig. 4. The local improvement and greedy improvement approaches similarly generated a variety of solutions that improved upon those generated by the network approach.

The network optimization approach for the Pax River case generated only two distinct solutions (a nearly straight, minimum-time solution and a wandering minimum-risk solution). As a result, the local improvement and greedy improvement approaches generated sets of solutions that had many solutions near the minimum-time solution and one solution near the minimum-risk solution (as shown in Fig. 4). The non-network approach was unable to find a low-risk solution; it generated solutions near the initial straight-line solution.

The closeness metric shows that the quality of the solutions generated by the local improvement and greedy improvement approaches was superior to the quality of the solutions that the network optimization step generated. This was true for both approaches in the College Park case. The tolerance value did not show any consistent trend in how it affected the closeness of the solutions. As can be seen in Fig. 4, the Pareto frontiers generated by these approaches either dominate or are nondominated by those produced by only using the network approach. The greedy and local approaches both produce superior results to using only the network optimization approach. The non-network approach was unable to construct long, low-risk solutions like those that the network approaches found. The lack of low-risk solutions is due to the non-network approach converging to local optima that are near the initial straight-line solution, which prevents the approach from finding solutions near the better solutions that the network-based approaches find. Several examples of the differences between these two types of solutions can be seen in Fig. 5. The greedy and local approaches appear to be the best of the approaches that were considered in this paper (that is, they produced the best Pareto frontiers of solutions).

As can be seen in Fig. 3, neither the local improvement approach nor the greedy improvement approach was substantially better than the other in terms of solution quality; the computational effort, however, was quite different: the local improvement approach required more effort than the greedy improvement approach (the computational effort for both includes the computational effort for the network optimization step). The computational effort of the non-network approach increased as the number of waypoints increased, which is expected given that an increase in waypoints means that the optimizer has more variables that it needs to manipulate. Additionally, as the grid becomes finer (includes more nodes), the computation time required for the

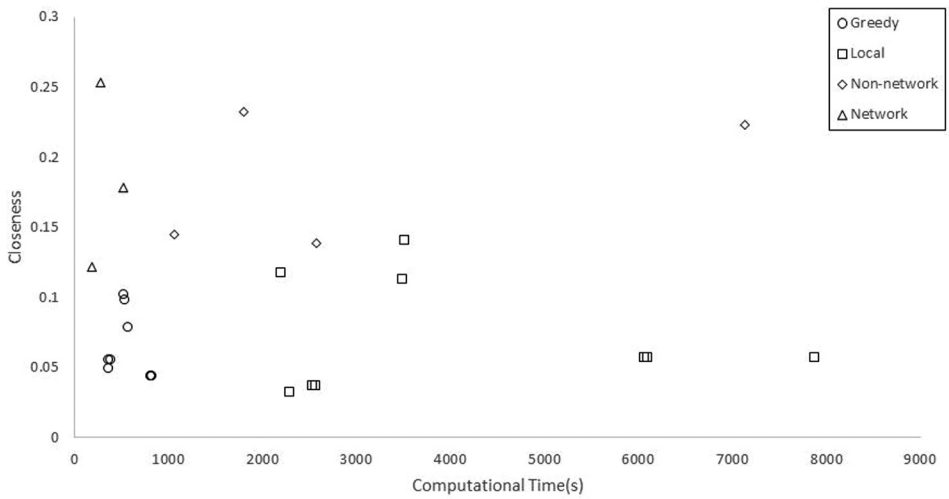

Fig. 3 Closeness against computation time for the College Park case

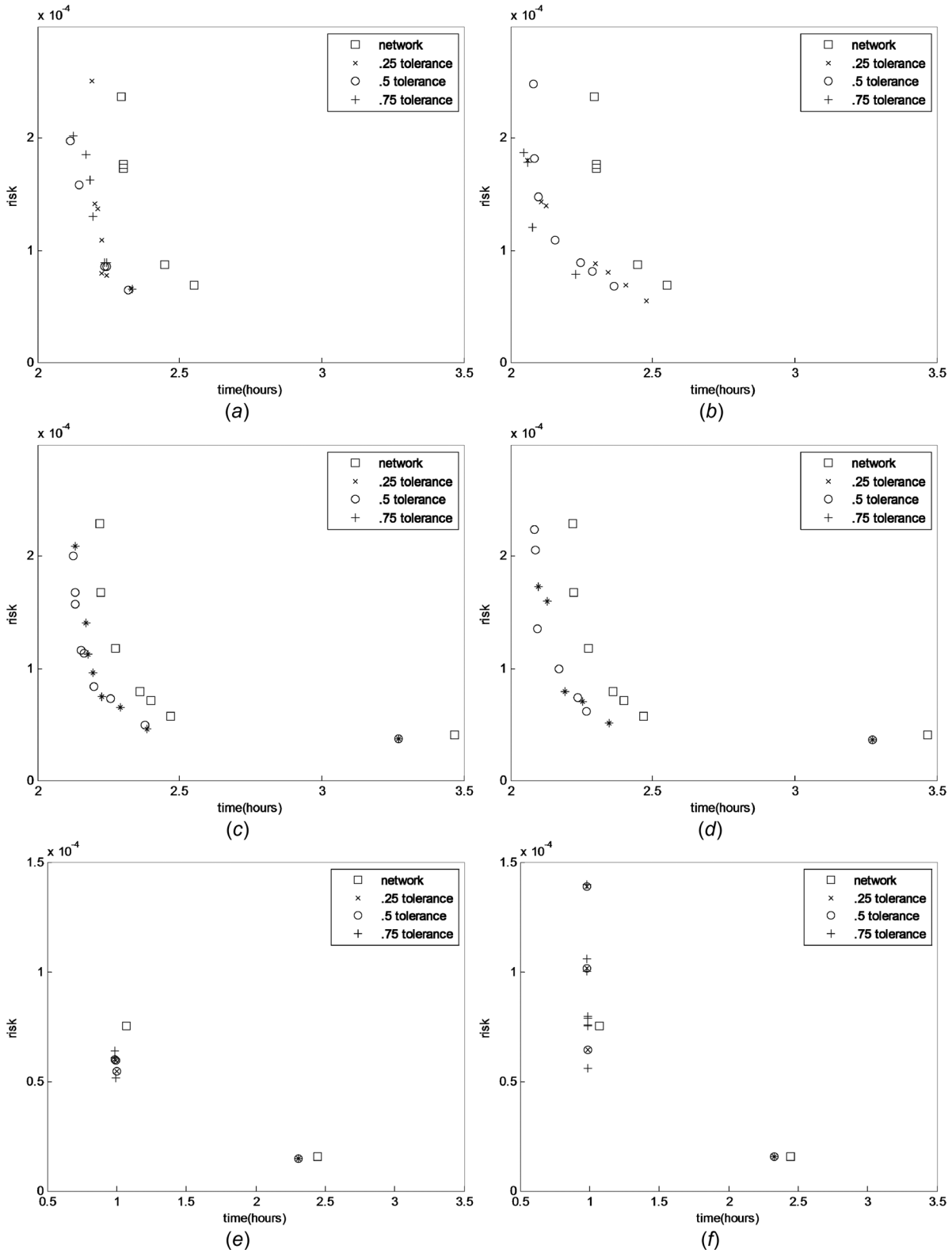

Fig. 4 Selected Pareto frontier results. For the College Park case: (*a*) greedy approach, 30 × 12 GRID; (*b*) local approach, 30 × 12 GRID; (*c*) greedy approach, 40 × 16 GRID; and (*d*) local approach, 40 × 16 GRID. For PAX river case: (*e*) greedy approach, 40 × 16 GRID and (*f*) local approach, 40 × 16 GRID.

greedy improvement approach does not grow at the same rate as the computation time required for the local improvement approach does, which suggests that the difference in the computation time for the two methods would likely increase for larger problems.

The quality of the solutions and the computational effort of the network optimization step varied as the grid size varied, but no trend was evident. In general, the solution quality should improve as the grid resolution becomes finer, but the network optimization step will require more memory to store the larger network.

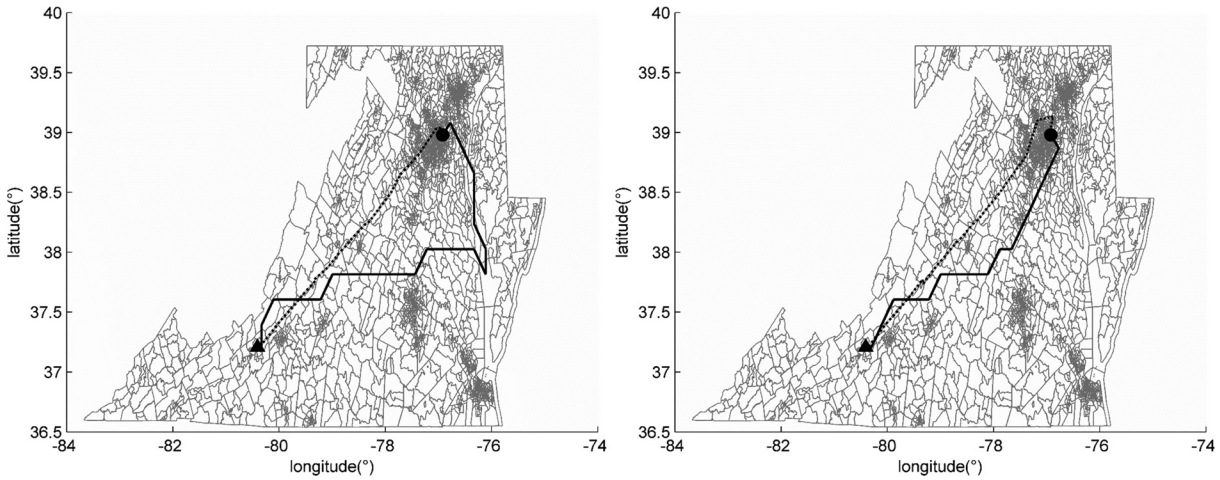

Fig. 5 Examples of the solutions generated by the network approach with the 40 × 16 GRID (solid line) and the non-network approach with 20 waypoints (dashed line) for the College Park case: (*a*) $w_t = 0$, $w_r = 1$ (*b*) $w_t = 0.3$, $w_r = 0.7$

6 Conclusions

This paper presented a bi-objective path planning optimization framework for exploring the tradeoffs between risk and flight time for UAVs. A risk assessment technique and bi-objective optimization methods were developed to find low-risk and time (flight path) solutions. Computational experiments were performed to evaluate the relative performance of the proposed optimization methods. The optimization methods considered were based on a network optimization approach, followed by improvements by a local approach and a greedy approach that used the network optimization results. A fourth approach did not use the network results but locally optimized the coordinates of a fixed number of waypoints.

The results from the computational experiments described the relative performance of the four methods and illustrated the tradeoffs involved. These results indicate that in terms of both computation time and solution quality, the greedy improvement approach produces the best results of the methods considered.

The proposed framework can be extended to incorporate factors such as the shelter provided by buildings that would affect the risk calculations. It can also be extended to incorporate other types of risks (including the risk of midair collisions). Future work will consider testing other approaches for generating the initial solutions for the non-network approach, using approximations to evaluate solutions faster, using higher resolution population data for takeoff and landing patterns, using time-dependent population data (time of day, seasonality, and special events), developing consistent heuristics for risk for use in an A* search, and incorporating shelter data. The problem formulation can be expanded to include selecting the altitude and velocity of each leg (which affects crash location distribution) and avoiding no-fly zones.

Acknowledgment

The authors acknowledge the support of Dr. David Burke and Dr. John Tritschler and the financial support of the Naval Air Warfare Center under cooperative Agreement No. N00421132M006.

Nomenclature

$c(e)$ = cost (weighted sum of the time and risk) of an edge
d = distance between two adjacent points in the discrete probability distribution
D_k = population density of a census tract
$\bar{D}(i, i+1)$ = expected crash location population density along a leg
$f(X)$ = cost objective function
f_x = fraction (tolerance) for the x-coordinates
f_y = fraction (tolerance) for the y-coordinates
$G(n_1, n_2)$ = edge between nodes n_1 and n_2
K_1 = expected number of crashes per 100,000 flight hours
K_2 = expected area in which persons will be killed if the vehicle crashes
n = number of waypoints
N = number of whole intervals in a leg
n_x = number of points in a row in the grid
n_y = number of points in a column in the grid
p_{jk} = probability associated with a point in the bivariate distribution
$r(i, i+1)$ = risk of flying a leg
$\bar{r}$ = normalization constant for risk
$t(i, i+1)$ = time to travel a leg
$\bar{t}$ = normalization constant for time
V = vehicle airspeed
w_r = weight on risk
w_t = weight on time
X = x- and y-coordinates of a list of waypoints
(x_i, y_i) = coordinates of a waypoint
x_L, x_U = lower and upper bounds for waypoint x-coordinates
(x^F, y^F) = finish point of the flight plan
(x^S, y^S) = start point of the flight plan
X^N = list of waypoints in solution obtained from network optimization
y_L, y_U = lower and upper bounds for waypoints y-coordinates
Γ_k = census tract polygon
Δ_x = horizontal distance between adjacent nodes (vertices) in the grid
Δ_y = vertical distance between adjacent nodes (vertices) in the grid
$(\Delta x_{jk}, \Delta y_{jk})$ = rotated coordinates of a point in the bivariate distribution

References

[1] Burke, D., 2010, "System Level Airworthiness Tool: A Comprehensive Approach to Small Unmanned Aircraft System Airworthiness," Ph.D. thesis, North Carolina State University, Raleigh, NC.
[2] Goerzen, C., Kong, Z., and Mettler, B., 2010, "A Survey of Motion Planning Algorithms From the Perspective of Autonomous UAV Guidance," J. Intell. Rob. Syst., **57**(1–4), pp. 65–100.
[3] Mittal, S., and Deb, K., 2007, "Three-Dimensional Offline Path Planning for UAVs Using Multiobjective Evolutionary Algorithms," IEEE Congress on Evolutionary Computation, IEEE, Singapore, pp. 3195–3202.

[4] Sanders, G., and Ray, T., 2007, "Optimal Offline Path Planning of a Fixed Wing Unmanned Aerial Vehicle (UAV) Using an Evolutionary Algorithm," IEEE Congress on Evolutionary Computation, IEEE, Singapore, pp. 4410–4416.
[5] De Filippis, L., Guglieri, G., and Quagliotti, F., 2011, "A Minimum Risk Approach for Path Planning of UAVs," J. Intell. Rob. Syst., **61**(1–4), pp. 203–219.
[6] Bortoff, S. A., 2000, "Path Planning for UAVs," 2000 American Control Conference, IEEE, Chicago, IL, Vol. 1, pp. 364–368.
[7] Medeiros, F. L. L., and Da Silva, J. D. S., 2011, "Computational Modeling for Automatic Path Planning Based on Evaluations of the Effects of Impacts of UAVs on the Ground," J. Intell. Rob. Syst., **61**(1–4), pp. 181–202.
[8] Weibel, R. E., 2005, "Safety Considerations for Operation of Different Classes of Unmanned Aerial Vehicles in the National Airspace System," M.S. thesis, Massachusetts Institute of Technology, Cambridge, MA.
[9] Lum, C. W., and Waggoner, B., 2011, *A Risk Based Paradigm and Model for Unmanned Aerial Vehicles in the National Airspace*, Infotech@Aerospace, St. Louis, MO.
[10] Cobano, J. A., Conde, R., Alejo, D., and Ollero, A., 2011, "Path Planning Based on Genetic Algorithms and the Monte Carlo Method to Avoid Aerial Vehicle Collisions Under Uncertainties," IEEE International Conference on Robotics and Automation, IEEE, Shanghai, China, pp. 4429–4434.
[11] Reinhardt, L. B., and Pisinger, D., 2011, "Multi-Objective and Multi-Constrained Non-Additive Shortest Path Problems," Comput. Oper. Res., **38**(3), pp. 605–616.
[12] Lamont, G., Slear, J., and Melendez, K., 2007, "UAV Swarm Mission Planning and Routing Using Multi-Objective Evolutionary Algorithms," IEEE Symposium on Computational Intelligence in Multicriteria Decision Making, IEEE, Honolulu, HI, pp. 10–20.
[13] de la Cruz, J. M., Besada-Portas, E., Torre-Cubillo, L., Andres-Toro, B., and Lopez-Orozco, J. A., 2008, "Evolutionary Path Planner for UAVs in Realistic Environments," 10th Annual Conference on Genetic and Evolutionary Computation, ACM, Atlanta, GA, pp. 1477–1484.
[14] Pikaar, A., De Jong, C., and Weijts, J., 2000, *An Enhanced Method for the Calculation of Third Party Risk Around Large Airports: With Application to Schiphol*, Nationaal Lucht-en Ruimtevaartlaboratorium, Amsterdam, The Netherlands.
[15] Wu, P. P., and Clothier, R. A., 2012, "The Development of Ground Impact Models for the Analysis of the Risks Associated With Unmanned Aircraft Operations Over Inhabited Areas," 11th Probabilistic Safety Assessment and Management Conference (PSAM11) and the Annual European Safety and Reliability Conference (ESREL 2012), Helsinki, Finland.
[16] Ford, A., and McEntee, K., 2010, "Assessment of the Risk to Ground Population Due to an Unmanned Aircraft In-Flight Failure," AIAA Paper No. 2010-9056.
[17] Lum, C., Gauksheimy, K., Deseure, C., Vagnersx, J., and McGeer, T., 2011, "Assessing and Estimating Risk of Operating Unmanned Aerial Systems in Populated Areas," AIAA Paper No. 2011-6918.
[18] Stevens, B., 2003, *Aircraft Control and Simulation*, Wiley, Hoboken, NJ, Chap. 3.
[19] Stengel, R., 2004, *Flight Dynamics*, Princeton University Press, Princeton, NJ, Chap. 3.
[20] MATLAB, 2012, "Version R2012b," The MathWorks, Inc., Natick, MA.
[21] Roskam, J., 1995, *Airplane Flight Dynamics and Automatic Flight Controls*, DARcorporation, Lawrence, KS.
[22] Dijkstra, E. W., 1959, "A Note on Two Problems in Connexion With Graphs," Numerische Math., **1**(1), pp. 269–271.
[23] MATLAB, 2012, "Optimization Toolbox User's Guide, Version R2012b," MathWorks, Inc., Natick, MA.
[24] Wu, J., and Azarm, S., 2001, "Metrics for Quality Assessment of a Multiobjective Design Optimization Solution Set," ASME J. Mech. Des., **123**(1), pp. 18–25.