A Hybrid Genetic Algorithm for Rescue Path Planning in Uncertain Adversarial Environment

Jean Berger, Khaled Jabeur, Abdeslem Boukhtouta, Adel Guitouni, and Ahmed Ghanmi

Abstract— **Efficient vehicle path planning in hostile environment to carry out rescue or tactical logistic missions remains very challenging. Most approaches reported so far relies on key assumptions and heuristic procedures to reduce problem complexity. In this paper, a new model and a hybrid genetic algorithm are proposed to solve the rescue path planning problem for a single vehicle navigating in uncertain adversarial environment. We present a simplified mathematical linear programming formulation aimed at minimizing traveled distance and threat exposure. As an approximation to the basic problem, the user-defined model allows to specify a lower bound on the optimal solution for some particular survivability conditions. Hard problem instances are then solved using a novel hybrid genetic algorithm relaxing some of the common assumptions considered by previous path construction methods. The algorithm evolves a population of solution combining genetic operators with a new stochastic path generation technique, providing guided local search, while improving solution quality. The value of the problem-solving approach is shown for simple cases and compared to an alternate heuristic.**

I. Introduction

AUTONOMOUS and assisted intelligent systems such as robots and vehicles are increasingly used to conduct civilian and military missions in hazardous or hostile environments. Typical missions include tactical logistic support to sustain combat units, civilian/combat search and rescue operations, emergency management, and intelligence surveillance and reconnaissance to name a few. Given the high multi-dimensional cost of losing tactical assets a key challenge faced by such mission lies on efficient path planning in adversarial uncertain environments.

The classical single agent (e.g. vehicle, robot) path planning problem is usually defined as follows: Given an operational environment representation, compute a path plan allowing an agent to travel between two locations labeled as the source and destination sites respectively, to carry out a specific mission. The path must be collision-free (or feasible) and satisfies certain optimization criteria. In other words, path planning is generating a collision-free path with physical obstacles and optimizing it with respect to some criteria.

The rescue path planning problem is a resource and risk constrained shortest path problem (SPP) in adversarial environments [4], [12], [1], [17], [10] the building block of many navigation-based complex problems. Non-linearity characterizing problem complexity introduced by a new composite measure of performance coupling rescue agent survivability and path length makes problem-solving very challenging.

Various problem-solving techniques have been developed to tackle this problem from classical Dijkstra Bellman-Ford method variants [9] to metaheuristics. A recent survey of the main proposed methods may be found in [14], [15]. However, approaches proposed so far mostly rely on key assumptions such as deterministic threat distribution (number and positions) known in advance, threat-free path solution construction, use of predefined/preprocessed path segments and/or intermediate points to minimize risk exposure when traveling between threat locations. These assumptions are mainly exploited to reduce computational complexity. Accordingly, Voronoi diagrams –like approaches defining various polygonal cell shapes to enable spatial decomposition and approximate and limit suitable paths over the geographical search area are used extensively. A deterministic real-time dynamic feasible trajectory generation algorithm for unmanned aerial vehicles (UAVs) flying through a sequence of waypoints is introduced in [2]. As similarly reported in [11], the algorithm builds a path using segments and then smooth them to provide a feasible solution. A deterministic two step path-planning algorithm for UAVs is also proposed in [3]. A suboptimal path is first obtained by generating a graph based on Voronoi polygons. The graph is then exploited to improve the solution by solving a set of nonlinear ordinary differential equations through simulation.

Deterministic approaches have been mostly considered for non-adversarial environments. UAVs entering into a threat zone subject to potential detections or attacks may nevertheless be more realistic. Accordingly, recent path planning publications on UAVs mainly proposed probabilistic methods to handle uncertainty. Non-deterministic threat locations further allow exploiting "soft" threat/obstacle, moving away from threat-free path solutions. In that case, attractive paths may possibly include likely threat location visits. Dogan [5] uses continuous threat probability distribution, but relies on the computation of an upper bound on threat risk level to solve the path planning problem. In other respect, intermediate solution points

Manuscript received January 31, 2010.

J. Berger, A. Boukhtouta, K. Jabeur and A. Guitouni are affiliated with Defence Research and Development Canada – Valcartier, 2459 Pie-XI Blvd North, Val-Bélair, Quebec, G3J 1X5, Canada, (emails, jean.berger@drdc-rddc.gc.ca, abdeslem.boukhtouta@drdc-rddc.gc.ca, adel.guitouni@drdc-rddc.gc.ca).

A. Ghanmi is affiliated with the Centre for Operational Research & Analysis, Defence Research and Development Canada, Canadian Department of National Defence, 101 Col By Dr, Ottawa, ON, K1A 0K2 (email: Ahmed.Ghanmi@forces.gc.ca)

arbitrarily defined by a user or a heuristic problem-solver have also been used to prune the search space. Similarly to deadline or resources requirements imposed on a path, minimum survivability is a task-dependent constraint that can be further exploited in exploring solutions as the cost of losing assets may be an important consideration. A probabilistic path planning approach involving threat zones and conflicting objectives such as travel time, probability of escape, and first detection expected time is presented in [14]. A UAV path planning approach based on a probabilistic map reflecting threat zones derived from surveillance data is alternatively reported in [9]. The latter can be considered an extension from [3] to the stochastic case. A real–time route planner based on evolutionary computation and applicable to multiple UAVs [18] handles different constraints such as minimum route leg length, flying altitude, and maximum turning angle. In that case, vehicle probability of survival is based on threat exposure. Knowledge based genetic algorithm (GA) for solving robot path planning problem is developed in [16]. The proposed procedure relies on a problem-specific rather than a classical GA method.

A new problem model and a hybrid GA are proposed to solve the rescue path planning problem in uncertain adversarial environment. The model disregards some commonly known assumptions about the problem, while providing a simple approach to further mitigate computational complexity. The current work is primarily aimed at optimizing a user-defined weighted sum objective function using a new approach, rather than characterizing the optimal Pareto front defined by non-dominating solutions to a constrained multi-objective problem [7]. This is primarily motivated by an intrinsic relative objective predominance order for the problem at hand and assumed prior knowledge by the decision-maker of relative weights. The simplified mathematical linear programming formulation is aimed at minimizing traveled distance and threat exposure. The bi-objective minimization problem model captures user-defined preferences and non-deterministic threat level exposure moving away from precise threat locations prior knowledge requirement typically reported. As an approximation to the basic problem, the user-defined model allows to specify a lower bound on the optimal solution for some particular survivability conditions. In the current setting, a probabilistic adversarial environment with unknown threat locations is assumed. We're using discretization and propose a pre-processing procedure to explicitly capture threat risk exposure caused by multiple geographically distributed adversarial sensors/ information sources or enemy assets. No particular conditions on threat risk model are assumed. Threat level exposure map is built from threat presence probability distribution and the conditional risk (e.g. probability of detection or being destroyed) posed on the survivability of a fly-by rescuing agent to ultimately generate threat level exposure map to be exploited in the decision model. Building on the notion of "soft" threat, the model assumes agents traveling across high-risk zones if a real benefit can be expected, rather than confining search to collision or risk–free solutions. Minimum survivability is explicitly considered as well, as it is reasonable to assume a survivability threshold to successfully or satisfactorily complete a rescue path planning task. The proposed approximate model takes advantage of this information to specify a lower bound on the optimal solution of the original problem, in addition to the lower bound that can be obtained through Lagrangean relaxation of the approximate linear integer programming model formulation. Assuming these bounds available from computation, they could provide further guidance to search heuristic methods exploring the solution space for improved solution. Except possibly for hard problem instances, it is believed that the simplified model could reduce computational complexity while exploiting available problem-solving technology provided by the integer linear programming community. The real possibility to tackle larger problem sizes unlikely to be easily addressed through the classical problem model can be contemplated. Due to problem size, risk distribution, constrainedness or some minimal survivability range requirements, a new hybrid genetic algorithm is proposed to solve the problem and its hard problem instances. The approach consists in evolving a population of solutions combining classical recombination and mutation operators with a new local random path generation procedure through natural selection, providing guided local search, while improving solution quality. The procedure is used to recombine both crossing and non-crossing paths while taking advantage of parent solution structure, or mutate a solution by locally modifying a path segment.

The remainder of the paper is outlined as follows. The description of the single vehicle rescue path planning problem in uncertain adversarial environment is first given in Section II. Then, a simplified new linear programming formulation is introduced as an alternative to the original problem model. A new hybrid GA is then depicted in Section III. Preliminary results are presented in Section IV, and a conclusion is given in Section V.

II. PROBLEM DESCRIPTION

The simplest rescue path planning problem in uncertain adversarial environment involves an agent vehicle planning a trajectory over a certain area avoiding obstacles and threats in order to rescue or 'extract' stranded individuals while minimizing a combination of traveled distance and threat level exposure (or an equivalent survivability measure) subject to a variety of temporal, itinerary and/or survivability constraints such as deadline or survivability threshold. The agent is assumed to move from a base station (source site s) to a rescue service point (destination site d), and then move out of the danger zone as shown in Figure 1.

The adversarial environment is pictured as a grid or lattice composed of N cells. Cells are colored using a spectrum of shades to depict various treat levels (or survivability) ranging from white for safe cells, to red for physical obstruction or zero survivability cells. Modeled as a directed graph $G(V,A)$, the grid specifies a set of vertices or nodes V

representing individual cells, connected to one another (center to center) through a set of arcs or directed edges (i,j) A. Arcs define neighborhood relationship or possible agent moves between adjacent cells i and j. Typically, a non-boundary node i has 8 outgoing arcs and 8 incoming arcs. The rescuing agent is assumed to move one step at a time.

Envisioned as a discretized probabilistic threat risk exposure or survivability map representation, the grid is initially built from the probabilistic threat occupancy map describing the environment. Accordingly, each cell l is characterized by a probability of threat presence $P_T(l)$ estimated from a previous search task. Threat risk exposure (e.g. agent destruction or detection risk contribution $P_{kill/det}(l,j)$ in cell j from threat presence in cell l) for an agent traveling over a given cell is then derived from the threat map as follows:

$$risk_{\ j} = 1 - \prod_{l \in V} \{1 - p_{kill\ /\ \det}(l, j) p_T(l)\} \quad (1)$$

$$= 1 - p_{sj} \quad (2)$$

where p_{sj} is the vehicle's probability to survive cell j's visit.

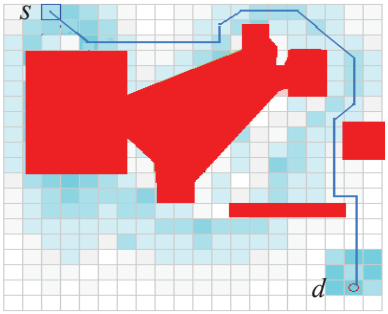

Fig. 1. A rescuing mission for a Combat Search and Rescue scenario. A rescue vehicle must travel from a source node s (base station) to a destination site d to perform extraction in a hostile (graded color shades) environment.

A. Model Formulation

The basic rescuing path planning problem $\mathcal{P}$ consists in minimizing traveled distance and threat risk exposure measures. Given user preferences α on threat risk exposure over traveled distance, a vehicle must move from a source node s to a destination site d over a N-cell grid environment characterized by a survivability distribution P_s, progressively constructing a path, by making decisions x_{ij} on visiting arcs (i,j). The objective is to minimize a user-defined weighted sum bi-objective function F, subject to itinerary (deadline/path length) and minimum survivability constraints S_m. The ideal solution would be to find the shortest and safest path separating source and destination sites subject to various constraints. $\mathcal{P}$ can be formally stated by a nonlinear integer programming formulation with resource as follows:

$\mathcal{P}roblem\ \mathcal{P}$: $\mathcal{P}(F, \alpha, N, L_{max}, S_m, P_s)$

F: objective function

α : user-defined threat exposure relative weight

N: number of cells over the grid

$L_{0\ max}$: user-defined maximum distance

S_m: minimum survival probability (survivability threshold)

P_s: survival probability distribution over the N-cell grid

$$\underset{\{x_{ij}\}}{Min} \quad F = (1-\alpha)L + \alpha(1-S) \quad (3)$$

where

$$L = \frac{\sum_{(i,j)\in A} d_{ij} x_{ij} / L_{\min} - 1}{L_{\max} / L_{\min} - 1}, \quad 0 \le L \le 1 \quad (4)$$

$$S = \prod_{(i,j)\in A} p_{sj}^{x_{ij}} \quad \text{-- survivability} \quad (5)$$

Subject to constraint set C:

$$\sum_{j \in V} x_{sj} = 1 \quad \text{path starts at source node } s \quad (6)$$

$$\sum_{j \in V} x_{jd} = 1 \quad \text{path ends at destination node } d \quad (7)$$

$$\sum_{i \in V} x_{ij} \le 1 \quad (i,j) \in A \quad \text{(at most one visit per node)} \quad (8)$$

disjoint subtours elimination:

$$t_i^s + d_{ij} - K(1 - x_{ij}) \le t_j^s \quad (i,j) \in A, K \in \aleph \quad (9)$$

flow constraint/continuity:

$$\sum_{j \in V} x_{ij} - \sum_{j \in V} x_{ji} = 0 \qquad \forall i \in V \setminus \{s,d\} \quad (10)$$

itinerary constraint/ deadline:

$$\sum_{(i,j)\in A} d_{ij} x_{ij} \le L_{\max} \quad (11)$$

survivability threshold:

$$-\sum_{(i,j)\in A} x_{ij} \ln(p_{sj}) \le -\ln(S_m) \qquad S_m \in]0,1] \quad (12)$$

$$x_{ij} = 0 \quad \forall (i,j) \notin A \quad (13)$$

$$x_{ij} \in \{0,1\} \quad (14)$$

$$0 \le \alpha \le 1$$

In (3)-(5), x_{ij} is a binary decision variable representing the agent's positive (x_{ij} =1) or negative (x_{ij} =0) decision to travel along arc (i,j), d_{ij} the distance separating centroids from cells i and j, L_{min} ($\sqrt{2N}$) and L_{max} (user-defined or $2\sqrt{N}$) are respectively the minimal (along the diagonal, disregarding risk level) and maximal possible traveled distance from s to d, and S the overall vehicle survivability on path completion. It is noteworthy to mention that L_{min} and L_{max} could possibly be further refined by solving simpler $\mathcal{P}$–based shortest path problem instances beforehand, leading to L_{min}=$MAX(\sqrt{2N}, L^*(\mathcal{P}(\alpha=0)))$ and L'_{max} = $MIN(L_{max}, 2\sqrt{N})$,

$L^*(\mathcal{P}\ (\alpha=0, S_m=1))$ for the shortest 'fully secure' path. $\mathcal{P}(\alpha=0, S_m=1)$ can be solved in polynomial time using Dijkstra's algorithm on the 'safe' subnetwork from *G*. The bi-objective function (3) to be minimized is subject to constraint set (6)-(14). Constraints (6) and (7) ensure a path to start at cell *s* and ends in cell *d* respectively. Constraint (8) specifies that a cell must be visited at most once to eliminate unnecessary visits in building the shortest path. Disjoint subtours (loop) elimination are reflected in (9) which for a move along arc (*i,j*) requires that service at *j* at time t_j^s is temporally compatible with the service at *i* at time t_i^s (*K* is a large positive constant). Flow continuity is imposed by (10), while itinerary constraint associated with resource limitations or path completion deadline is captured in (11). Minimum path survivability requirement is reinforced by constraint (12). Finally, (13) and (14) specify graph compatibility and decision constraints respectively.

It should be noticed that if α=1, the problem may be reformulated as a shortest path problem, using alternatively the logarithm of the objective function:

$$\underset{\{x_{ij}\}}{Min} \quad -\sum_{(i,j)\in A} x_{ij} \log(\, p_{sj}\,) \tag{15}$$

A. Simplified Model

The simplified model consists in replacing *S* in equation (5) in the objective function by an approximate function $\widetilde{S}(S)$ having the property to linearize the transformed bi-objective function. The resulting approximate linear program formulation captures key elements of the original problem while being exposed to classical problem-solving techniques widely used in the Operations Research (OR) community.

$$\widetilde{S} = \frac{\ln(s/s_m)}{\ln(1/s_m)}(1-s_m)+s_m \quad for \quad s\in[s_m,1] \tag{16}$$

By substituting equation (5) in (16) the approximate survivability function becomes:

$$\widetilde{S} = 1-\frac{1-s_m}{\ln(s_m)}\sum_{(i,j)\in A} x_{ij}\ln(p_{sj}) \tag{17}$$

While $\widetilde{S}$ proves to be exact for $S = S_m$ and 1, the induced overestimation (relative error) is expressed as follows:

$$\varepsilon(s) = \frac{\Delta(s)}{s} = \frac{\widetilde{S}-s}{s} = \frac{\dfrac{\ln(s/s_m)}{\ln(1/s_m)}(1-s_m)+s_m-s}{s} \tag{18}$$

$$s_\varepsilon^* = e s_m^{\frac{1}{1-s_m}} \Rightarrow \varepsilon_{\max} = \frac{(s_m-1)\, s_m^{\frac{1}{s_m-1}}}{e\,\ln(s_m)} - 1 \tag{19}$$

The maximum relative error varies monotonically with S_m varying from 1% for S_m=0.75, to 6% for S_m=0.5, 10% for S_m=0.4, 15% for S_m=0.33, but 85% for S_m=0.1. Despite this limitation, solving the approximate model formulation provides a lower bound on the optimal objective function of the original problem over an acceptable range of minimum survivability requirements for a wide variety of applications. Ultimately, computed solutions for problem instances in which S_m<0.33 can be exploited by another heuristic method as an initial solution to improve solution quality.

Revisiting equation (3), the new linear integer program formulation consists in alternatively minimizing the objective function *G*:

$$\underset{\{x_{ij}\}}{MIN} \quad G = (1-\alpha)L + \alpha(1-\widetilde{S}) \tag{20}$$

subject to the same constraints set (6)-(14). From the new formulation and equations (3)-(5) we derive the following relations and bound estimation for F_{min} ($F(x_F^*)$) in (3):

$$G \le F = G + \alpha\varepsilon(s)S \tag{21}$$

$$G_{\min} \le F_{\min} \le F(x_G^*) = G_{\min} + \alpha\varepsilon(s_G^*)s_G^* \tag{22}$$

$$G_{\min} \le F_{\min} \le G_{\min} + \alpha\varepsilon(s_G^*)s_G^* \tag{23}$$

where x_G^* is the optimal solution of the simplified model with quality G_{min} and survivability S_G^*. Bounds on $L(x_F^*)$ and S_F^* can be similarly derived from (23) and exploited by a heuristic:

$$L_F^* \ge L_{\min} Max\left(1, 1+\left(\frac{L_{\max}}{L_{\min}}-1\right)\left(\frac{G_{\min}+\alpha S_m-\alpha}{1-\alpha}\right)\right) \tag{24}$$

$$S_m \le S_F^* \le S_{\max} = Min\left(1, \frac{1-G_{\min}}{\alpha}\right) \tag{25}$$

By virtue of its linearity, we believe the simplified linear model easier to solve, paving the way to address larger problem size. The linear model is also anticipated to provide good solution approximations for heuristic methods dealing with hard problem instances. However, constrained SPP is weakly NP-Hard [4]. In particular, complexity related to problem size, risk distribution, constrainedness imposed by path length and minimum survivability thresholds in very hostile environment may make the problem computationally very hard. Efficient heuristic methods are therefore required to compute near optimal solutions. In that regard, a new hybrid GA is presented in Section III.

III. Genetic Algorithm

A. Description

The proposed hybrid GA [6] builds complete path solutions in a single phase. It makes no restrictive assumptions, approximations or arbitrarily introduces intermediate cell visits biases, as previously proposed by some genetic–based methods to reduce computational complexity. The approach consists in evolving a population of path solutions over successive generations, combining classical recombination and mutation operators with a new local random path generation procedure through natural selection, providing guided local search, while improving solution quality. The procedure is used to recombine both crossing and non-crossing paths while taking advantage of parent solution structure, or mutate a solution by locally modifying a path segment. Population individuals are

represented as chromosomes explicitly encoding a path plan from *s* to *d,* expressed as a sequence of cell visits moving along related arcs from source *s* to destination *d*. Path length is defined dynamically.

1) Algorithm: The proposed algorithm can be summarized as follows:

Initialize Population
gen=0
Repeat for each new generation
    adjust individual's fitness value
    Evolve Population - build a new generation
        generate λ new offspring using genetic operators
            -selection, recombination, mutation
        evaluate fitness of new individuals
        eliminate the λ worst individuals of the expanded population
gen=*gen*+1
Until (solution convergence or *gen* = *gen_max*)
Return (best computed path plan from Population)

The algorithm first initializes a population of feasible and non-feasible path plan individuals. Solutions are constructed randomly, selecting visits based on a new random path generation procedure further described below. From source node *s*, unvisited (if possible) successor nodes are randomly selected until the destination node *d* is reached. The iterative problem-solving process evolving the population of path solutions is then initiated. The first step consists in adjusting the individual's fitness emerging from the last episode. The GA consists for each generation in expanding the population by λ offspring using genetic operations, and then removing the worst λ individuals to restore normal size. Recombination and mutation operators are sequentially applied with probability p_{Xover} and p_{Mut} respectively, until all λ new individuals are generated. Probability parameters are generally determined to balance search intensification and diversification. The process is repeated until a convergence criterion/condition is met.

2) Path generation procedure: Initial population generation and hybridized genetic operators builds upon a new constructive random path generation method to create or evolve individuals. Given two nodes *s* (source) and *t* (target), and a local search area *Z* enclosing these nodes, the proposed exploration procedure randomly search adjacent cells, building constructively a path segment solution over the local area, one cell at a time, until source node *s* gets successfully connected to target node *t*.

Candidate nodes are first split over three (possibly overlapping) sets from which a unique subset will be considered for selection. That unique subset determines the remaining eligible candidates. Pictured in Figure 2, the three classes of candidates are defined as follows: A zone 'τ', defining neighbor cells for which the distance to the target node, is less than the distance from the current node to the target; a zone 'S' where distance from a candidate to the source cell is less than the distance from the source to the current node of the path; and a zone '$S\tau$' for the remaining unvisited candidates. A candidate node may be a member of multiple sets. Random exploration of a unique subset is biased toward a predetermined user-defined probability distribution associated to the corresponding zones, namely, $p_\tau > ½$, p_S, and $1 - p_S - p_\tau$. That probability distribution drives path exploration either toward the target, away from the target or arbitrarily from/to the target respectively.

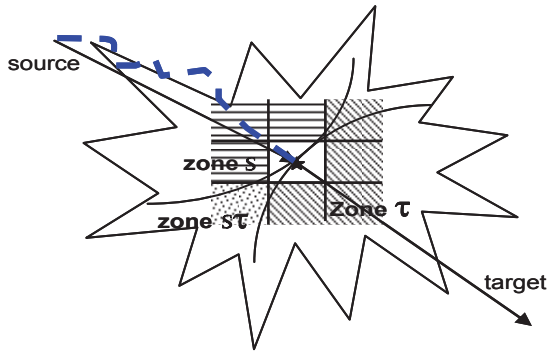

Fig. 2. Constructive random path generation procedure from source to target. The partial path is expanded from the current node (white cell) one cell at a time, choosing a successor. Successor selection is biased toward zones with highest corresponding exploration probability (p_τ, p_S, and $1 - p_S - p_\tau$).

It allows search diversification ($1 - p_\tau$) blindly avoiding myopic policies, identifies dead-end situations resorting to backtracking if needed, interrupts unfeasible path expansion (constraint violation) to explore an alternative route over a larger search area, and exercises pressure to eventually move toward the target ($p_\tau > ½$). In safe surroundings, a large p_τ generally emphasizes path length reduction. As p_τ depends on the risk probability distribution, one can simply pose $p_\tau = (1 + \bar{p}_{sZ})/2$, where $\bar{p}_{sZ}$ is the average survivability over the given search area *Z*. Once a unique subset of candidate successor nodes is identified, one of them is finally selected. Accordingly, next visit selection probability is proportional to a utility function expressed as a weighted sum of risk exposure and distance. Inspired from equation (3), a node utility relies on the combination of the distance separating a candidate (adjacent) node to the target cell using a direct subpath, and the relative risk incurred in migrating to the corresponding cell. A time-out is specified to bound search. As a result, a new path emerges.

The path generation procedure *Path_Gen* is summarized as follows:

Path_Gen (*s*: source, *t*: target, Z: search area, F, L_{max}, S_m, P_s):
$\mathbb{T}$=1; $\mathbb{T}$: deadline
$s,t \in Z$
Repeat
    Inititialization:
        Path.surv=1; Path.length=0
        Path={*s*}
        c = *s* -- current_node
        succ=*s*
        $\forall i \in Z, \forall k \in \{1..8\}$ (*k*: successor index)
            if ($p_{s\ i.succ(k)} > S_m \wedge i.succ(k) \notin$ Path $\wedge$
                $i.succ(k) \in Z$) *i.succ(k)* is admissible
            else
                i.succ(k) is non admissible
        constr_violated = FALSE

New path construction from *s* to *t*:

While (*not constr_violated* OR *succ* $\neq t$ OR $\mathrm{T}<\mathbb{T}$) do *set* = $\varnothing$

Candidate successors:

Split *c's* neighborhood $Neigh_c$ over 3 subsets

$$\tau = \{j \in Neigh_c \,|\, d_{jt} < d_{ct}\}$$

$$\mathbf{S} = \{j \in Neigh_c \,|\, d_{js} < d_{cs}\} \quad \mathbf{S}\tau = Neigh_c / (\tau \cup \mathbf{S})$$

Select from $Neigh_c$ a subset '*set*' containing admissible nodes based on probability distribution: $p_\tau > ½$, $p_\mathbf{S}$ and $1\text{-}p_\mathbf{S} - p_\tau$

Expandable path condition

If (*set* $\neq \varnothing$) then

$\forall i \in set$ -- potential cycle elimination

if ($i \in$ Path OR $p_{s_i} < S_m$ / Path.surv)

$c.succ(k_i)=i$ is non admissible *set* = *set* /{*i*}

Successor selection:

Succ = $i \in$ *set* with probability $p_{sel}(i)$

$$p_{sel}(i) = \frac{U(i)}{\sum_{i \in set} U(i)} \tag{26}$$

$$U(i) = p_{s\,i}^{\,1-\ln(Path.surv - S_m)} \exp\left(\frac{\max_{i \in set} d_{it} - d_{it}}{\max_{i \in set} d_{it} - \min_{i \in set} d_{it}} \right)$$

Path expansion: Path = Path $\cup \{succ\}$

Constraint satisfaction:

Path.length = Path.length + $d_{c\,succ}$

Path.surv= Path.surv*$p_{s\,succ}$

If (Path.surv<S_m OR Path.length> L_{max},)

constr_violated=true

c=*succ*

else *set* = $\varnothing$ -- dead-end and backtracking

Path.predecessor(*c*).*succ*(k_c)=*c* is non admissible

remove *c* from Path

c= Path.predecessor(*c*)

end while

$\mathrm{T}=\mathrm{T}+1$

if (T mod *period* = 0) expand *Z* – larger search area

Until (*succ*=*t* OR $\mathrm{T}=\mathbb{T}$) -- convergence

If (*succ* $\neq t$) Path = $\varnothing$

Return (Path)

From the first step of the genetic algorithm described earlier, an initial population of solutions is generated by repeatedly calling the path generation procedure along with random parameters (p_τ, $p_\mathbf{S}$) over a domain bounded by *s* and *d*, considering *s* as the source and *d* as the target node. Initial solutions could alternatively be generated randomly joining arbitrary intermediate zero-risk cells along the way from *s* to *d*, and then repeatedly use *Path_Gen* to fill the gaps. The path generation procedure is also combined to genetic recombination and mutation operations to connect partial paths or nodes in reconstructing or repairing solutions as presented in sub-sections *D* and *E*.

B. Fitness

Fitness reflects an individual's propensity to reproduce. Fitness is defined in terms of a weighted combination of the objective function and a penalty contribution reflecting constraint violations.

$$fitness_{Path} = F_{Path\{(i,j)\,|\,x_{ij}=1\}} + Penalty \tag{27}$$

$$Penalty = \lambda \left\{ u\left(\sum_{i \in Path} d_{i\,succ(i)} - L_{\max} \right) + u\left(\ln(S_m) - \sum_{j \in Path} \ln(p_{sj}) \right) \right\} \tag{28}$$

$$u(x) = \begin{cases} 1 & if\ x > 0 \\ 0 & otherwise \end{cases} \tag{29}$$

where λ is a large positive constant and u(x) is the step function. Other constraints described in (6)-(14) are implicitly satisfied during genetic operations.

C. Selection

In order to balance and control selection pressure for breeding purposes, fitness values are sorted out and normalized using a linear ranking scheme [13], scaling respective values based on rank. Individual selection for mating purposes is then ensured following a fitness-proportional scheme [6].

D. Recombination

This genetic operation recombines chromosomes from two selected parents in order to create a child. The proposed recombination operator X_path comprises a two-stage process, conditionally to crossing paths. First, if both parent individuals share at least one intersecting point, one of them is randomly selected as a crossover point to generate an offspring. A child is created by interchanging partial path segments, connecting together head and tail subpaths inherited from both parents as pictured in Figure 3. A second offspring can be similarly generated by swapping both parents. In contrast, a second stage might be required to breed non-intersecting path individuals. In this case, children are generated by selecting an early and late crossover points on respective parent solutions and then, interchanging partial path segments as shown in Figure 4. The resulting disjoint path components inherited from parent solutions are then reconnected using the path generation procedure introduced in Subsection *A* to build a feasible child path. Local search area is set to trade-off potential threat risk minimization along the way and run-time computation. The higher the local threat risk level, the larger the exploration area.

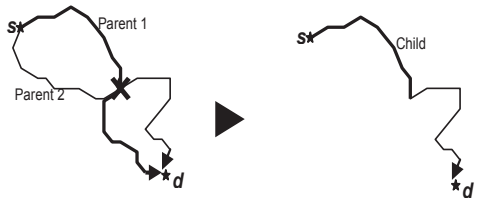

Fig. 3. Crossover operator X_path mating Parent 1 and 2 to generate a new child solution. Parent trajectories are shown to intersect at a cross-over point.

As a result, a new path emerges, subject to solution repair to remove redundant visits. In that case, loops are eliminated keeping the shortest path. Accordingly, all visits between the

first and the last occurrence of the first encountered node (from the source) having multiple occurrences are eliminated, and the process is repeated over the revised path, and so on until the path is cycle-free.

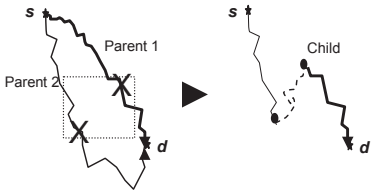

Fig. 4. Crossover operator X_path mating Parent 1 and 2 to generate a new child solution. Orphan segments inherited from parents are reconnected through their respective crossover points.

E. Mutation

Mutation is a natural evolution process modifying some individual's genes more or less frequently. The M_path_local_repair mutation operator randomly removes a path segment (a sequence of consecutive moves, at least two steps from the path endpoints) from a solution individual and locally builds an alternate partial path bridging the gap between the pending disconnected components as displayed in Figure 5. The path generation procedure described in 3.1 is used to fill that gap. As for the recombination operator, a complete path results from local search exploration over a limited region. Likewise, the emerging path is revisited to eliminate potentially redundant visits (loop).

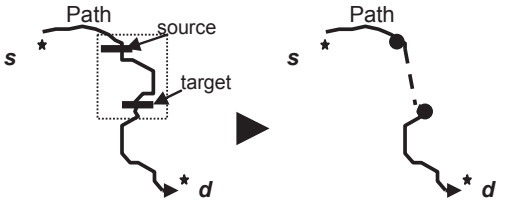

Fig. 5. Mutation: a short sequence of visits is randomly removed from an individual, and a mutated solution built by reconstructing an alternate link connecting the pending subpath components.

The mutator tends to reduce either path length or threat risk level while exploring the solution space. Adjustable parameters (local search area centered around the altered segment, p_J , p_S) can be dynamically set when calling the path generation procedure to further guide search or emphasize intensification/diversification. Path segment 'cardinality' or sequence size selection is based on a uniform probability distribution. The probability to select a l node path segment sequence starting from the i^{th} node ($\pi(i)$) is given by:

$$p(i,l) = p(l)\,p(i|\,l) \tag{30}$$

$$p(l) = 1/(L_{Path} - 2) \qquad 2 \le l \le L_{Path} - 1 \tag{31}$$

$$p(i|\,l) = \begin{cases} \dfrac{l - d^{H}_{\pi(i)\,\pi(i+l)}}{\sum_{k=1}^{L_{Path}-l} (l - d^{H}_{\pi(k)\,\pi(k+l)})} & 1 \le i \le L_{Path} - l \\ 0 & otherwise \end{cases} \tag{32}$$

$$d^{H}_{qr} = Max\left(|x_q - x_r|, |y_q - y_r|\right) \tag{33}$$

where L_{Path} is the path cardinality (number of nodes forming the path), (x_q,y_q) the position of node q, and d^{H}_{qr} the Hamming distance between cells q and r. Equation (32) reflects the selection bias toward promising node sequences likely to further reduce path length.

IV. Experimental results

A preliminary computational experiment has been conducted to show the feasibility of the proposed hybrid genetic approach to solve the rescue path planning in uncertain adversarial environment. Comparison with alternate methods reported in the literature remains very difficult as classical methods suffer from space complexity limitations, threat exposure models and underlying assumptions differ, and used data sets or benchmark are not published or openly made available. Alternatively, the proposed methodology consists in running the algorithm for a variety of parameters characterizing a set of specific problem instances for which the corresponding optimal solutions can be easily derived by simple visual inspection.

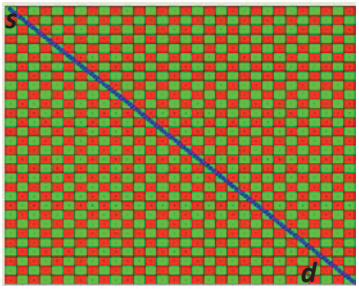

Fig. 6. A 30x30 "Checker board" threat risk map for a traveling agent from s to d. Light (dark) –colored cells refer to threat-free (hard obstacle/deadly) nodes. The computed path connecting s to d is optimal, following a safe straight line along the grid diagonal.

Simulation runs were executed using the following hybrid genetic algorithm parameters: Population size: 30; p_{Xover}= 0.8; p_{Mut}= 0.5, number of generations: 100. The performance of the problem-solving method is compared to an alternate heuristic, both coarsely implemented in visual basic on a 1 GHz PC. The heuristic method consists in repeatedly executing the local exploration procedure *Path_Gen* portrayed in Subsection III-A, for various parameters p_J, p_S, up to a maximum number of cycles (typically 40) and then, return the best computed solution.

Simulations have been conducted for 2 types of scenarios with α =0.7 for a k x k grid. The corresponding threat risk maps are shown in Figures 6-7 for k=30. Threat risk distributions for those problem instances depict a "Checker board" and a central obstruction structure respectively. In both cases, the rescuing agent must travel across the grid between opposite top corners labeled s and d. Light (dark) -colored grid locations refer to threat-free (deadly) cells for a visiting agent. The computed agent path solutions displayed clearly show to be near optimal as it can be validated by inspection. Typical run-time for the first scenario (checkerboard map) is approximately one minute, with a 14% gap from the optimal solution on average for a 40

simulation sample. Optimal solutions were found in a proportion of 10%. Run-time for the second scenario comparatively amounts to less than a minute and a half on average. Computational results proved similarly convincing for 50x50 grid problem instances, over a variety of (*s,d*) conditions.

Despite a disadvantageous 2-3 orders of magnitude run-time ratio, the hybrid GA outperforms the heuristic method in terms of solution quality, by at least 80% on average. It is believed that the use of a computer language more suitable for mathematical computation and code optimization might significantly reduce run-time execution.

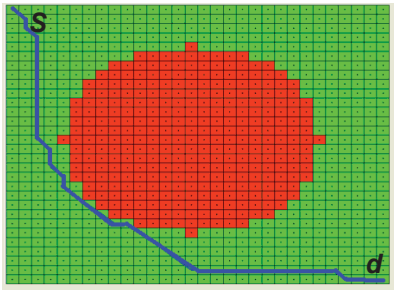

Fig. 7. A 30x30 single obstruction threat risk map for a traveling agent from *s* to *d*. Light (dark) –colored cells refer to threat-free (hard obstacle/deadly) nodes. The computed solution proves to be near optimal in moving around the central obstruction in safely connecting *s* to *d*.

These results illustrate the value of the approach in synergizing well-known genetic-based methods with a specialized problem-solving technique to potentially improve solution quality, in comparison to a single procedure considered independently. The ultimate benefit of the advocated hybrid approach lies in its attractive simplicity to harness existing problem-solving methods to naturally match or outperform them.

An alternate scenario involving hard obstacle configuration (threat risk distribution) and the computed path solution is depicted in Figure 8. The high quality of the emerging solution path is clearly shown.

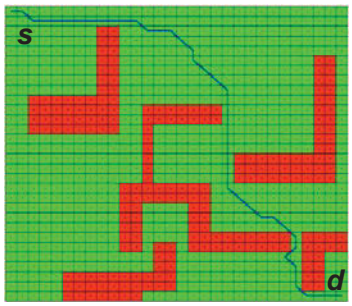

Fig. 8. 30x30 multi-obstructions threat risk map for a traveling agent from *s* to *d*.

V. Conclusion

A new approach to solve rescue path planning in adversarial uncertain environment modeled as a resource and risk –constrained shortest path problem has been presented. From a proposed integer programming optimization model combining traveled distance and threat exposure minimization objectives, a simplified approximate linear formulation was introduced. The solution to the approximate model provides a lower bound on the optimal path of the original problem for some survivability conditions. The challenge of computational complexity was further mitigated through the development of a hybrid GA mixing classical genetic operators with a new stochastic path construction technique. The hybrid approach provides additional search guidance in exploring the solution space.

Future work will generalize the approximate model to any threat risk conditions, defining the objective function over multiple intervals while arbitrarily refining the relative error bound induced on survivability. OR technology will be used to explore the computational limitations of the linear model.

References

[1] Anderson, E., Beard, R., and McLain, T. (2005). "Real-time Dynamic Trajectory Smoothing for Unmanned Air Vehicles", *IEEE Transactions on Control Systems Technology,* 13(3), 471-477.

[2] Beasley, J. and Christofides, N. (1989). "An Algorithm for the Resource Constrained Shortest Path Problem," *Networks*, 19(4), 379–394.

[3] Bortoff, S.A. (2000). "Path Planning for UAVs", *in Proc. of the American Control Conference,* Chicago, IL, 364-368.

[4] Carlyle, M., Royset, J.O. and Wood, R.K. (2007). "Routing military aircraft with a constrained shortest-path algorithm", *Military Operations Research*, to appear.

[5] Dogan, A. (2003). "Probabilistic approach in path planning for UAVS", I*EEE International Symposium on Intelligent Control*.

[6] Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning.* New York: Addison-Wesley.

[7] Hallam, C., Harrison, K.J., and Ward, J.A. (2001). "A multiobjective optimal path algorithm", *Digital Signal Processing*, 11, 133–143.

[8] Hespanha, J.P., Kızılocak, H.H., and Ate¸skan, Y.S. (2001). "Probabilistic map building for aircraft-tracking radars", in *Proceedings of the American Control Conference*, Arlington, VA, volume 6, 4381–4386.

[9] Jun, M. and D'Andrea, R. (2002). "Path planning for unmanned aerial vehicles in uncertain and adversarial environments", in S. Butenko, R. Murphey, and P. Pardalos, editors, *Co- operative Control: Models, Applications and Algorithms*, 95–111, Kluwer.

[10] Kohler E., Mohring R.H., Schilling H. (2005). "Acceleration of Shortest Path and Constrained Shortest Path Computation", S.E. Nikoletseas (Ed.): WEA 2005, LNCS 3503, 126–138.

[11] McLain, T., Chandler, P., Rasmussen, S., and Pachter, M. (2001). "Cooperative Control of UAV Rendezvous", *Proceedings of the American Control Conference*, Arlington, Virginia.

[12] Mitchell, J.S.B. and Papadimitriou, C.H. (1991). "The weighted region problem: Finding shortest paths through weighted planar subdivision", *Journal of the ACM*, 38, 18–73.

[13] Potvin, J.-Y. and Bengio, S. (1996). "The Vehicle Routing Problem with Time Windows Part II: Genetic Search", *INFORMS Journal on Computing*, 8, 165–172.

[14] Pfeiffer, B., Klamroth, K., Batta, R. and Nagi, R. (2005). "Path Planning for UAVs in the Presence of Threat Zones Using Probabilistic Modeling," submitted to *Operations Research*.

[15] Pfeiffer, B., Batta, R., Klamroth, K. and Nagi, R. (2008). "Probabilistic Modeling for UAV Path Planning in the Presence of Threat Zones," *Handbook of Military Industrial Engineering,* Editor: Adedeji Badiru, Taylor and Francis.

[16] Yanrong Hu; Yang, S.X. (2004). "A knowledge-based genetic algorithm for path planning of a mobile robot", *IEEE International Conference on Robotics and Automation*, 5, 4350 – 4355.

[17] Zabarankin, M., Uryasev, S., and Murphey, R. (2006). "Aircraft Routing under the Risk of Detection," *Naval Research Logistics*, 53(8), 728–747.

[18] Zheng, C. Li, L. Xu, F. Sun, F. Ding, M. (2005). Evolutionary Route Planner for Unmanned Air Vehicles, 21(4), *IEEE Transactions on Robotics,* 609–620.