In the Book Chapter of *"Cooperative Control: Models, Applications and Algorithms"*, Kluwer, edited by S. Butenko, R. Murphey, and P. Pardalos.

Chapter 6

PATH PLANNING FOR UNMANNED AERIAL VEHICLES IN UNCERTAIN AND ADVERSARIAL ENVIRONMENTS*

Myungsoo Jun
Sibley School of Mechanical and Aerospace Engineering
Cornell University
Ithaca, NY 14853-7501, USA
mj73@cornell.edu

Raffaello D'Andrea
Sibley School of Mechanical and Aerospace Engineering
Cornell University
Ithaca, NY 14853-7501, USA
rd28@cornell.edu

Abstract One of the main objectives when planning paths for unmanned aerial vehicles in adversarial environments is to arrive at the given target, while maximizing the safety of the vehicles. If one has perfect information of the threats that will be encountered, a safe path can always be constructed by solving an optimization problem. If there are uncertainties in the information, however, a different approach must be taken. In this paper we propose a path planning algorithm based on a map of the probability of threats, which can be built from *a priori* surveillance data. An extension to this algorithm for multiple vehicles is also described, and simulation results are provided.

Keywords: Unmanned aerial vehicles, path planning, probability map, uncertain adversarial environments, optimization

*Research sponsored by AFOSR Grant F49620-01-1-0361

1. Introduction

Autonomous robots and vehicles have been used to perform missions in hazardous environments, such as operations in nuclear power plants, exploration of Mars, and surveillance of enemy forces in the battle field. Among these applications is the development of more intelligent unmanned aerial vehicles (UAVs) for future combat in order to reduce human casualties. One of main challenges for intelligent UAV development is path planning in adversarial environments.

Path planning problems have been actively studied in the robotics community. The problem of planning a path in these applications is to find a collision-free path in an environment with static or dynamic obstacles. Early work focused on holonomic and non-holonomic kinematic motion problems with static obstacles without considering system dynamics. Despite many external differences, most of these methods are based on a few different general approaches: roadmap, cell decomposition, and potential field (Latombe, 1990). When moving obstacles are involved in planning problems, the time dimension is added to the configuration space (Erdmann and Lozano-Perez, 1987) or state-space of the robot (Fraichard, 1999), and planning is termed motion planning or trajectory planning instead of path planning. Research has recently been performed in motion planning that takes into account dynamic constraints, called *kinodynamic planning* (LaValle and Kuffner, 1999, Hsu et al., 2000). All of the aforementioned path or motion planning methods focus on obstacle avoidance issues.

In the UAV path planning problem in adversarial environments, the objective is to complete the given mission — to arrive at the given target within a prespecified time — while maximizing the safety of the UAVs. We can consider adversaries as obstacles and employ similar methods to those used for robot path planning. The main difference between robot path planning and UAV path planning is that a UAV must maintain its velocity above a minimum velocity, which implies that it cannot follow a path with sharp turns or vertices. There has been research on path planning for UAVs in the presence of risk — see Bortoff, 2000, Chandler et al., 2000, McLain and Beard, 2000, Zabarankin et al., 2001, and the references therein. The first three approaches are similar. They decompose path planning into two steps: First, a polygonal path is generated from the Voronoi graph by applying Djiktra's algorithm, which is the same as the roadmap and A^* search approaches in robot path planning; the initial polygonal path is then refined to a navigable path by considering the UAV's maneuverability contraints (see Chandler et al., 2000, for example) or by using the dynamics of a set of virtual masses in a virtual force field emanating from each radar site (see Bortoff, 2000, for example), which is similar to the potential field approaches in robotics. By refining this process, one can produce paths without vertices. The main problem with these methods is that the effects of uncertainties in the locations of the radar sites are not considered. Other potential problems are the effects of the decoupling assumption, as stated by the authors; specifically, the value of the cost function after refining may not be minimal, and moving along the Voronoi graph yields suboptimal trajectories. In Zabarankin et al., 2001 an optimization problem is solved, where the solution minimizes the integrated risk along the path with a constraint on the total path length. The result is not guaranteed to be optimal if there are uncertainties in the information, such as the locations of the radar sites, etc.. In addition, the computational load grows quickly as the number of radar sites increases.

Uncertainties in the information lead naturally to consider probability models. In Hespanha et al., 2001 a probabilistic map of radar sites are constructed and a safe UAV path is generated by solving a minimization problem. The probability map was constructed using a likelihood function which was defined by considering radar range and other radar characteristics, and by using Bayes' rule. It was assumed, however, that radar sites remain at fixed locations. The algorithm also requires m way-points for trajectory planning, and thus the resulting path is a minimum risk path only among the paths which include the way-points.

In this paper we propose a path planning method for UAVs by using a probability map. The approach in this paper is similar to those in Bortoff, 2000 and Chandler et al., 2000 in that we decompose the problem into two steps — first the generation of a preliminary polygonal path by using a graph, and then a refinement of the path. Our approach differs in that it is based on a map of the probability of threats, and it does not use a Voronoi graph to find a preliminary path. The nodes and links of the graph are based directly on the probability map. We also consider the effects of moving threats, changes in the probability map, and multiple vehicles, and perform an analysis of the effects of refinement on the initial path.

This paper is organized as follows: Section 2 contains some basic definitions in graph theory. Section 3 describes how to build an occupancy probability from measured data and how to calculate a probability of risk. The problem formulation is stated in Section 4. Section 5 provides a method for generating a weighted digraph from the probability map, which enables us to solve the original problem with a shortest path algorithm. The path planning algorithm is described in Section 6, and the extension to multiple vehicles in Section 7. Simulation results are provided in Section 8. Discussion and future problems are found in Section 9.

2. Graph Theory

In this section we will briefly cover some basic material on graph theory. The reader is referred to Bertsekas and Gallager, 1992 for more details. We define a *graph*, $G = (\mathcal{N}, \mathcal{A})$, to be a finite nonempty set $\mathcal{N}$ of *nodes* and a collection $\mathcal{A}$ of distinct nodes from $\mathcal{N}$. Each pair of nodes in $\mathcal{A}$ is called a *link* or an *arc.* A *walk* in a graph G is a sequence of nodes $(n_1, n_2, \cdots, n_l)$ such that each of the pairs (n_1, n_2), (n_2, n_3), $\ldots$, (n_{l-1}, n_l) are links of G. A walk with no repeated nodes is a *path.* A walk $(n_1, n_2, \cdots, n_l)$ with $n_1 = n_l$, $l > 3$, and no repeated nodes other than $n_1 = n_l$ is called a *cycle.* A graph is *connected* if for each node i there is a path $(i = n_1, n_2, \cdots, n_l = j)$ to each other node j.

A *directed graph* or *digraph* $G = (\mathcal{N}, \mathcal{A})$ is a finite nonempty set $\mathcal{N}$ of nodes and a collection $\mathcal{A}$ of *ordered* pairs of distinct nodes from $\mathcal{N}$; each ordered pair of nodes in $\mathcal{A}$ is called a *directed link.* The definitions of *directed walks*, *directed cycles*, and *connectedness* are analogous to those for graphs.

3. Probability Map

The basic concept in building the probability map in this paper is similar to the grid-based occupancy maps in map learning methods for autonomous robots in Elfes, 1987, Moravec, 1988, Thrun, 1998 and Yamauchi and Langley, 1997. Occupancy values for each grid cell are determined based on sensor readings and by applying the conditional probability of occupancy using Bayes' rule. These values are determined by the sensor characteristics, the location of the sensors, the measurement methods, etc..

Assume that the region $\mathcal{R}$ for the mission is given and $\mathcal{R}$ is composed of n cells. Define the following events:

$$\begin{aligned}
\mathcal{D}_i &= \text{event that the UAV in cell } i \text{ is detected} \\
&\quad\ \text{by the adversary} \\
\mathcal{E}_i &= \text{event that the adversary is in cell } i \\
\mathcal{S}_i &= \text{event that the UAV in cell } i \text{ is neutralized} \\
&\quad\ \text{by the adversary}
\end{aligned}$$

Denote the complement of event $\mathcal{A}$ by $\overline{\mathcal{A}}$. If X_i represents the i-th sensor reading, the probability of occupancy of cell i can be expressed as (Thrun, 1998)

$$P(\mathcal{E}_i|X_1,\cdots,X_k) = \\ 1 - \left(1 + \frac{P(\mathcal{E}_i|X_1)}{1-P(\mathcal{E}_i|X_1)} \prod_{j=2}^{k} \frac{P(\mathcal{E}_i|X_j)}{1-P(\mathcal{E}_i|X_j)} \frac{1-P(\mathcal{E}_i)}{P(\mathcal{E}_i)}\right)^{-1} \quad (1)$$

The probability that the UAV in cell i is neutralized by the adversary is

$$\begin{aligned} P(\mathcal{S}_i) &= P(\mathcal{S}_i|\mathcal{D}_i)P(\mathcal{D}_i) + P(\mathcal{S}_i|\overline{\mathcal{D}}_i)P(\overline{\mathcal{D}}_i) \\ &= P(\mathcal{S}_i|\mathcal{D}_i)P(\mathcal{D}_i) \end{aligned}$$

where we assume that the probability that the UAV is neutralized when it is not detected by the adversary is 0. Define $p_i \overset{\triangle}{=} P(\mathcal{E}_i|X_1,\cdots,X_k)$. Probability $P(\mathcal{D}_i)$ can be expressed as

$$\begin{aligned} P(\mathcal{D}_i) = {} & P\left(\mathcal{D}_i|(\mathcal{E}_i|X_1,\cdots,X_k)\right) P(\mathcal{E}_i|X_1,\cdots,X_k) + \\ & P\left(\mathcal{D}_i|(\overline{\mathcal{E}}_i|X_1,\cdots,X_k\right) P(\overline{\mathcal{E}}_i|X_1,\cdots,X_k) \\ = {} & P\left(\mathcal{D}_i|(\mathcal{E}_i|X_1,\cdots,X_k)\right) p_i + P\left(\mathcal{D}_i|(\overline{\mathcal{E}}_i|X_1,\cdots,X_k)\right)(1-p_i) \end{aligned}$$

We thus have

$$\begin{aligned} P(\mathcal{S}_i) = P(\mathcal{S}_i|\mathcal{D}_i) \{ & P\left(\mathcal{D}_i|(\mathcal{E}_i|X_1,\cdots,X_k)\right) p_i + \\ & P\left(\mathcal{D}_i|(\overline{\mathcal{E}}_i|X_1,\cdots,X_k)\right)(1-p_i)\}. \end{aligned} \quad (2)$$

We assume that we can estimate or calculate the probabilities $P(\mathcal{S}_i|\mathcal{D}_i)$, $P\left(\mathcal{D}_i|(\mathcal{E}_i|X_1,\cdots,X_k)\right)$, $P\left(\mathcal{D}_i|(\overline{\mathcal{E}}_i|X_1,\cdots,X_k)\right)$ and p_i from *a priori* information of the adversary.

The probability $P(\overline{\mathcal{S}})$ that the UAV is NOT neutralized by the adversary when it follows path $(a_1, a_2, \cdots, a_l)$ can be expressed as

$$P(\overline{\mathcal{S}}) = \prod_{i\in(a_1,a_2,\cdots,a_l)} P(\overline{\mathcal{S}}_i), \quad (3)$$

where $P(\overline{\mathcal{S}}_i) = 1 - P(\mathcal{S}_i)$. The objective is to find the path $(a_1, a_2, \cdots, a_l)$ that maximizes the probability $P(\overline{\mathcal{S}})$ when cell a_1 is the origin of the mission and cell a_l is the target of the mission.

4. Problem Formulation

Assume that the region of interest consists of n cells, and that the shape of the cells is given. See Figure 6.1, for example. Let $1 \le O \le n$ be

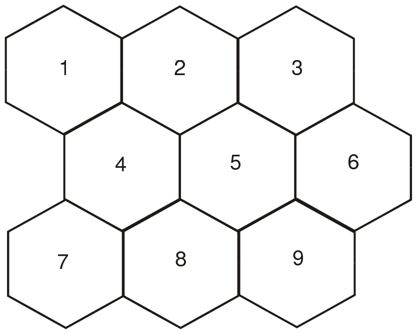

Figure 6.1. The region consists of 9 hexagon cells.

the cell which contains the origin of the mission, and $1 \le T \ne O \le n$ the cell which contains the target. Define a sequence of cells $(a_1, a_2, \cdots, a_l)$, where $2 \le l \le n$ and $a_i \ne a_j$ for all $1 \le i \ne j \le l$, to be *a path from cell a_1 to cell a_l* if cell a_i and a_{i+1} are adjacent to each other for all $1 \le i \le l-1$. The problem can be stated as follows:

Problem 1 (Minimum Risk Path Problem) *Find a path from cell O to cell T such that*

$$\prod_{i\in(O,a_2,\cdots,T)} P(\overline{\mathcal{S}}_i) \ge \prod_{i\in(O,b_2,\cdots,T)} P(\overline{\mathcal{S}}_i) \tag{4}$$

for all paths $(O, b_2, \cdots, T)$.

Since the logarithm is a monotonically increasing function of its argument, the above expression is equivalent to:

$$\sum_{i\in(O,a_2,\cdots,T)} (-\log(P(\overline{\mathcal{S}}_i))) \le \sum_{i\in(O,b_2,\cdots,T)} (-\log(P(\overline{\mathcal{S}}_i))) \tag{5}$$

5. Conversion to a Shortest Path Problem

This section describes several methods for generating a digraph from the probability map that was built based on Section 3. After generating the digraph, the Minimum Risk Path Problem is converted to a shortest path problem. The second part of this section describes the Bellman-Ford algorithm which will be used to find the shortest path.

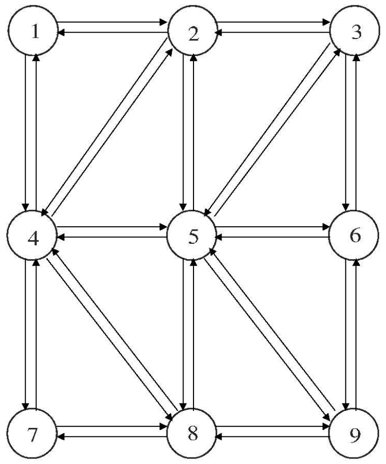

Figure 6.2. The digraph converted from the map of cells in Figure 6.1.

5.1. Defining Digraph

Define nodes to be cells in the probability map, that is, $\mathcal{N} = \{1, 2, \cdots, n\}$. Define d_{ij}, the weight of link (i, j), as follows:

$$d_{ij} = \begin{cases} -\log(P(\overline{\mathcal{S}_j})) & \text{if cells } i \text{ and } j \text{ are adjacent,} \\ \infty & \text{otherwise} \end{cases}. \tag{6}$$

For example, $d_{17} = \infty$ for the map in Figure 6.1 since cell 1 and 7 are not adjacent. By defining the weights of the links as in Eq. (6), we can convert the Minimum Risk Path Problem to a shortest path problem.

We may also want to consider path length and include a constant term in the link weight as a penalty:

$$d_{ij} = \begin{cases} -\log(P(\overline{\mathcal{S}_j})) + c & \text{if cells } i \text{ and } j \text{ are adjacent,} \\ \infty & \text{otherwise} \end{cases}. \tag{7}$$

where c is a constant real number.

With the above definition of nodes and links, the probability map is converted to a digraph. Figure 6.2 is the digraph converted from the probability map in Figure 6.1. The Minimum Risk Path Problem has been converted to the problem of finding the shortest path in a network-like routing problem. Our path planning problem is simpler than a network routing problem since we do not have to consider fairness, maximum resource utilization, etc..

We will obtain a preliminary path once the shortest path is found from the converted digraph. For example, if the shortest path from node 9 to node 1 in Figure 2 is $(9, 5, 2, 1)$, the path we should take will pass cells 9, 5, 2, and 1. With this in mind, we can refine the path by using conventional optimization with constraints on the UAV maneuverability.

5.2. Shortest Path Algorithm

There are many shortest path algorithms. The Djikstra algorithm was used in Chandler et al., 2000 to find the shortest path in a Voronoi graph. In this paper, we will use the Bellman-Ford algorithm. The Djikstra algorithm is computationally more attractive than the Bellman-Ford algorithm in the worst case: The worst case computation of the Djikstra algorithm is $O(n^2)$ whereas it is $O(n^3)$ for the Bellman-Ford algorithm. There are many problems, however, where the Bellman-Ford algorithm terminates in a very few number of iterations relative to the Djikstra algorithm. Generally, for non-distributed applications, the two algorithms appear to be competitive (Bertsekas and Gallager, 1992).

The main advantage of the Bellman-Ford algorithm is that the iteration process is very flexible with respect to the choice of initial estimates and updates. This allows an asynchronous, real-time distributed implementation of the algorithm, which can tolerate changes in the link lengths — changes in adversary probability in our problem — as the algorithm executes. We can thus update the link lengths without terminating and restarting the algorithm.

The Bellman-Ford algorithm can be briefly described as follows. Suppose that node 1 is the *target* node. We assume that there exists at least one path from every node to the target. A shortest walk from a given node i to node 1, subject to the constraint that the walk contains at most h links and goes through node 1 only once, is referred to as a *shortest*($\leq h$) *walk* and its length is denoted by D_i^h. By convention, we take $D_1^h = 0$ for all h. The Bellman-Ford algorithm maintains that D_i^h can be generated by the iteration

$$D_i^{h+1} = \min_j \left[d_{ij} + D_j^h \right], \quad \text{for all } i \neq 1 \tag{8}$$

starting from the initial conditions $D_i^0 = \infty$ for all $i \neq 1$. The assumptions for the distributed asynchronous Bellman-Ford algorithm are i) each cycle has positive length and ii) if (i, j) is a link, then (j, i) is also a link. Our problem satisfies these two assumptions, and thus we can use the distributed asynchronous Bellman-Ford algorithm.

6. UAV Path Planning

This section describes a path planning algorithm for UAVs by using the shortest path that was given from the digraph and the Bellman-Ford algorithm. The algorithm decomposes the problem into two steps — generation of a polygonal path from the graph, and a refinement of the path. This section also considers the effects of moving adversaries, changes in probabilities, and an analysis of the effects of path refinement.

6.1. Polygonal Path

After we obtain the shortest path from the digraph described in Section 5, we can build a polygonal path from the origin to the target in the original probability map. Suppose that the shortest path from the digraph is $(a_1, a_2, \cdots, a_l)$. This means that safest path in the probability map should lie within the cells $(a_1, a_2, \cdots, a_l)$, which is called a *channel* in the cell decomposition method. There are an infinite number of paths from the origin to the target which lie within those cells. One might say that it would be best to choose the one with the shortest path length. It is not easy to find the path with shortest path length, however, as this is generally a non-convex optimization problem since the set of cells $(a_1, \cdots, a_l)$ is generally not convex.

We propose the following simple method: Connect the origin in cell a_1 to the center point of cell a_2, connect the center points of cells a_i and a_{i+1}, $2 \le i \le l-1$, connect the center of cell a_{l-1} to the target. See Figure 6.3.

6.2. Path Refinement

In this section we describe how to refine the preliminary path of line segments generated by the Bellman-Ford algorithm. The main idea is similar to what was described in Chandler et al., 2000: make the vertices of the line segments *smooth* by tangential arcs of a circle of radius r_{min}, where r_{min} is the minimum radius that a UAV can execute.

The length d between the vertex point and the tangential point (see Figure 6.4) can be expressed as

$$d = r_{min} \cdot \tan\theta. \tag{9}$$

The valid size of a cell thus depends on r_{min}. If the shape of a cell is hexagon, as is shown in Figure 6.1, the minimum angle between vertices is $2\pi/3$. The minimum value of θ in Figure 6.4 is thus $\pi/6$. From Eq. (9)

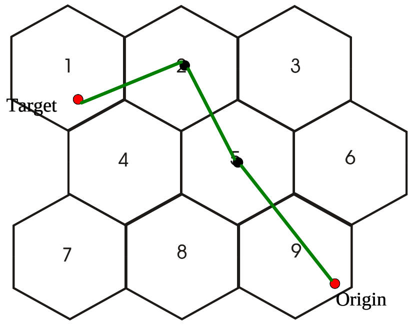

Figure 6.3. A rough path from the origin to the target when the shorted path of the digraph in Figure 2 is $(9, 5, 2, 1)$.

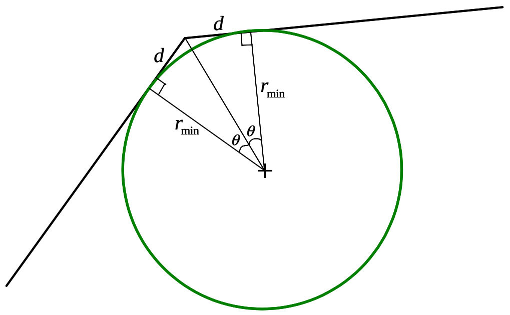

Figure 6.4. Smoothing the vertex of a preliminary path with an arc of a circle of radius r_{min}.

we have

$$\text{distance between the centers of two hexagons} \geq r_{min} \cdot \tan\frac{\pi}{6} = \frac{r_{min}}{\sqrt{3}}. \quad (10)$$

If this condition is satisfied, the refined path is guaranteed to lie within the predetermined cells, *with possible exceptions at the origin and target cells*. In the case of square cells, it can be easily noticed that the minimum value of θ is $\pi/4$.

7. Multiple UAVs

When planning for multiple vehicles, it may be desirable for the vehicles to take different paths. To illustrate this point, consider the following simplified scenario: Two paths are available, where the probability that an adversary will be encountered along path i is denoted p_i; if an adversary is encountered, the probability that a vehicle will be neutralized is equal to 1. Let $p_1 \leq p_2$. Let us plan the paths for two vehicles using two different strategies:

Same Path If both vehicles take the same path, the probability that both vehicles reach the target is simply $1 - p_1$. The probability that none of the vehicles reach the target is p_1.

Different Paths If the vehicles take different paths, the probability that both vehicles reach the target is $(1 - p_1)(1 - p_2)$, which is less than the Same Path scenario; we are providing the adversary more opportunities to neutralize at least one of our vehicles. Note, however, that the probability that none of the vehicles reach the target is $p_1 p_2$, which is less than p_1.

The point of this example is that if the success of a mission does not require that all the vehicles reach the target, the probability of success can be enhanced by requiring that the vehicles take different routes to the target. In particular, in an adversarial environment this may decrease the probability that all the vehicles are detected by the adversary at the same time and neutralized.

We propose a simple extension of the single UAV planning problem that results in different vehicle paths. The paths are planned sequentially by adding a penalty weight to the path of the previous vehicles. The cells traversed by one vehicle have thus a high penalty and the other vehicles will avoid these cells. The detailed algorithm is as follows:

i) Plan a path for vehicle 1 based on the initial probability map;

ii) Build a map by adding a penalty to the path generated for vehicle 1;

iii) Plan a path for vehicle 2 based on the newly generated map;

iv) Repeat the procedure for the other vehicles.

This method can also be applied to the case of dynamic environments by updating the probability maps and by removing penalties in the links that the vehicles have not yet passed.

8. Simulation

We considered a region $\mathcal{R}$ consisting of 30×30 square cells. We allowed oblique moves through the vertex of the square connecting two centers in addition to vertical and horizontal moves. We generated random numbers with a uniform distribution in certain areas (5 areas in this simulation) and assigned them to $P(\mathcal{S}_i)$. In practice, probability

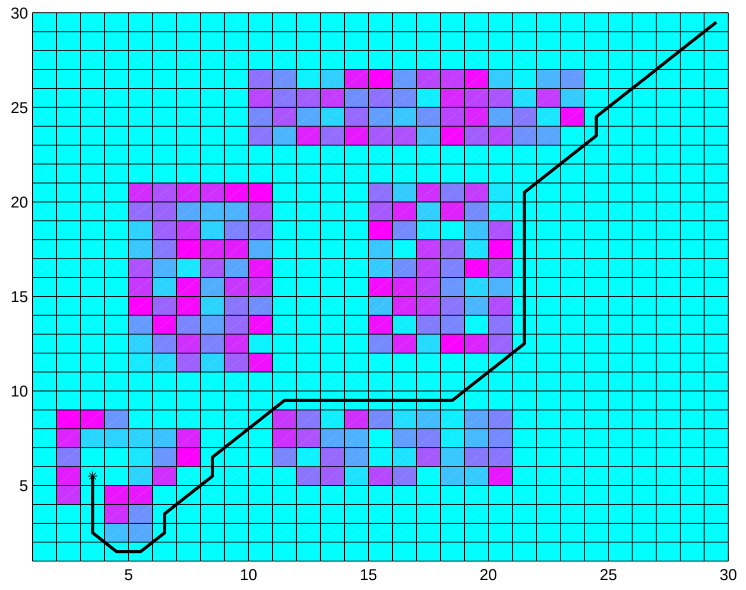

Figure 6.5. Simulation result: the path of the UAV when the path length is not penalized.

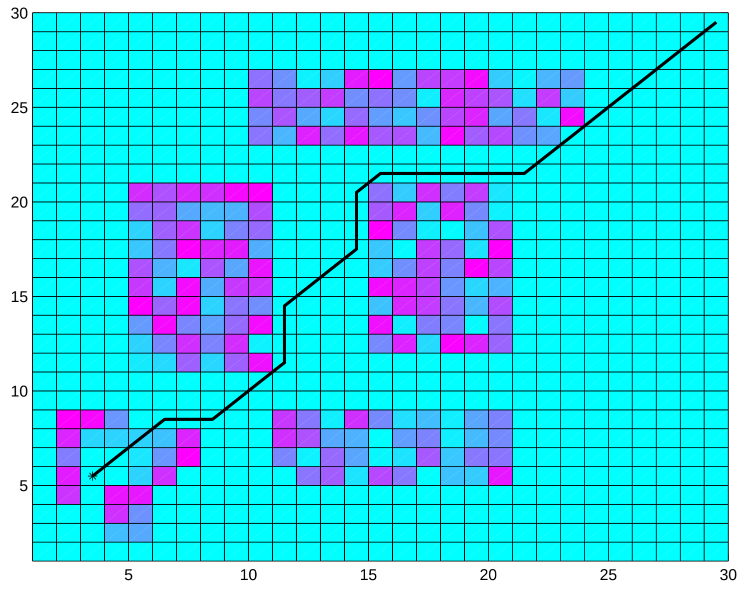

Figure 6.6. Simulation result: the path of the UAV when the path length is penalized.

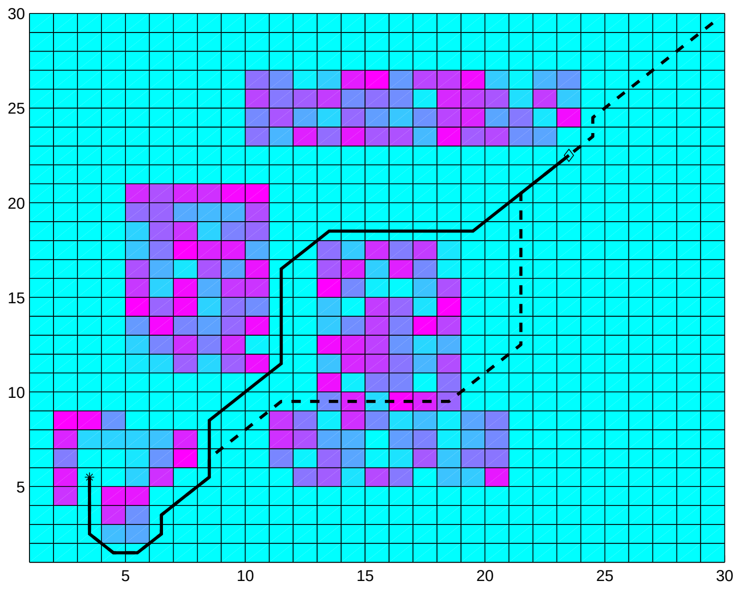

Figure 6.7. Simulation result: the path of the UAV when changes in the probability map were reported during path traversal.

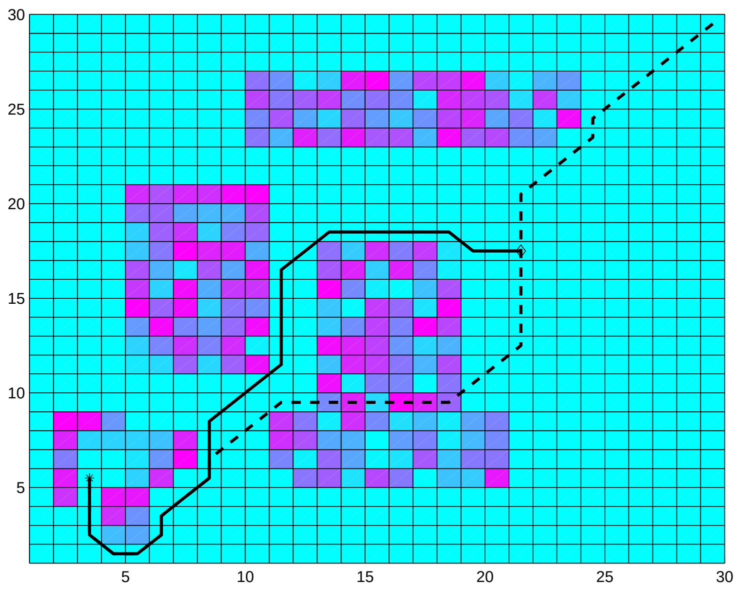

Figure 6.8. Simulation result: the path of the UAV when changes in the probability map were reported during path traversal.

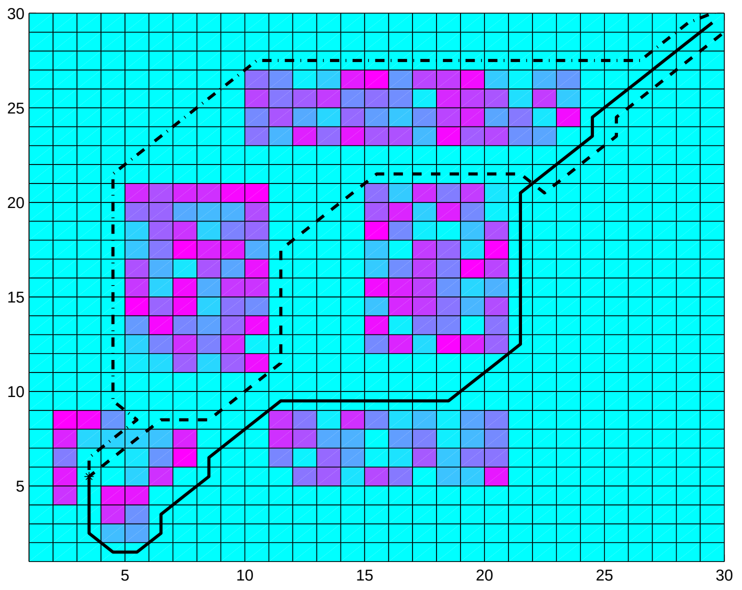

Figure 6.9. Simulation result: paths for multiple UAVs.

$P(\mathcal{S}_i)$ should be computed by using Eq. (2) with gathered information, measurement data, etc. The origin is cell $(30, 30)$ and the target is located at cell $(3, 5)$, which is marked by '$*$' in the plots. In the maps, darker regions represents higher value of $P(\mathcal{S}_i)$ (more dangerous areas).

The example in Figure 6.5 is the case when weight (6) was used. In this case, the path planning algorithm produced the safest path. The generated path takes a non-direct route for safety reason as can be seen in Figure 6.5. The second example in Figure 6.6 used the same probability map but adopted a different strategy: We used the weight in Eq. (7) with $c = 0.1$. As can be seen the generated path length is shorter than that in Figure 6.5, but the path included cells of non-zero probability.

The next scenario considered a dynamic situation. The movements of the adversaries or changes in the probabilities are reported to the UAV. We assumed that it was reported when the UAV was in cell $(23, 22)$, which is marked by '$\Diamond$' in Figure 6.7. The initial probability map is the same as the one in Figure 6.5 and Figure 6.6. We used the same weight as in the first example — without a penalty on path length. The dashed-line path in Figure 6.7 is the initial path, which is same as the one in Figure 6.5. After being reported the changes in the probabilities at the position marked by '$\Diamond$', the UAV took the solid-line path instead of the initial dashed-line path. Figure 6.8 shows the result when the

UAV was reported the changes in the probability map when it was in cell (21, 17).

Figure 6.9 shows path planning for multiple vehicles. The solid line is the path for vehicle 1, the dashed line for vehicle 2, and the dotted line for vehicle 3. Referring to the discussion in Section 7, the solid-line path was produced first, the dashed-line path second, and the dotted line path third.

9. Discussion and Future Research

Our approach is similar to the approximate cell decomposition methods in robot path planning in that it uses discretized cells for path planning. The differences are that i) cells in the cell decomposition method are recursively decomposed into smaller rectangles near the obstacles until some predefined resolution is attained, and thus the sizes of the cells are not same, and ii) the decomposition is performed in the configuration space, not in the two dimensional map, and thus the dimension of the cells is dependent on the degrees of freedom of the robot.

Cell decomposition in robot path planning is accomplished with the initial information on the locations and shapes of the obstacles. If the robot detects an unexpected obstacle inside the channel, a sensory-based potential field is used to guide the robot. In our method, we decompose the region into uniform cells, and change the values of probabilities when we detect unexpected changes during the mission. This can be done in a short time since the map is only two dimensional while the dimension of configuration space in the cell decomposition method is same as the degrees of freedom of the robot. Once the initial graph has been built, we do not have to rebuild the graph when we detect changes in the probabilities; we simply have to modify the weights of the pertinent links.

We did not consider a penalty for frequent accelerations and decelerations. Frequent accelerations and decelerations are undesirable as they require more fuel consumption. The number of accelerations and decelerations can be estimated by the number of heading angle changes. We can thus put a penalty in sharp heading angle changes. In order to capture this penalty, each node should have information of the previously traversed node, and the graph becomes a tree. This tree expands exponentially with the number of cells, which makes it difficult to plan paths in a real time environment. A future research direction is to find a fast method for path planning which considers sharp heading angle changes.

Our proposed algorithm for multiple UAVs is based on a simple heuristic: different paths are desirable, for the reasons outlined in Section 7.

The advantage of this approach is that it is very fast, and amenable to distributed computation. A more direct approach would explicitly optimize the probability that a given number of vehicles reach the target; it may be possible to simplify the resulting conditions to yield computational costs comparable to what has been presented in this paper, but with enhanced performance.

References

Bertsekas, D. and Gallager, R. (1992). *Data Network*. Prentice-Hall, Upper Saddle River, NJ.

Bortoff, S. A. (2000). Path planning for UAVs. In *Proc. of the American Control Conference*, pages 364–368, Chicago, IL.

Chandler, P. R., Rasmussen, S., and Pachter, M. (2000). UAV cooperative path planning. In *Proc. of AIAA Guidance, Navigation and Control Conference*, Denver, CO.

Elfes, A. (1987). Sonar-based real-world mapping and navigation. *IEEE Trans. on Robotics and Automation*, RA-3(3):249–265.

Erdmann, M. and Lozano-Perez, T. (1987). On multiple moving objects. *Algorithmica*, 2:477–521.

Fraichard, T. (1999). Trajectory planning in a dynamic workspace: A 'state-time space' approach. *Advanced Robotics*, 13(1):75–94.

Hespanha, J. P., Kizilocak, H. H., and Ateşkan, Y. S. (2001). Probabilistic map building for aircraft-tracking radars. In *Proc. of the American Control Conference*, pages 4381 –4386, Arlington, VA.

Hsu, D., Kindel, J. C., Latombe, J. C., and Rock, S. (2000). Randomized kinodynamic motion planning with moving obstacles. In *Proc. Workshop on Algorithmic Foundation of Robotics*, Hanover, NH.

Latombe, J. C. (1990). *Robot Motion Planning*. Kluwer Academic Press.

LaValle, S. M. and Kuffner, J. J. (1999). Randomized kinodynamic planning. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 473–479, Detroit, MI.

McLain, T. W. and Beard, R. W. (2000). Trajectory planning for coordinated rendezvous of unmanned air vehicles. In *Proc. of AIAA Guidance, Navigation and Control Conference*, Denver, CO.

Moravec, H. P. (1988). Sensor fusion in certainty grids for mobile robots. *AI Magazine*, pages 61–74.

Thrun, S. (1998). Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99:21–71.

Yamauchi, B. and Langley, P. (1997). Place recognition in dynamic environments. *J. of Robotic Systems*, 14:107–120.

Zabarankin, M., Uryasev, S., and Pardalos, P. (2001). Optimal risk path algorithm. Technical Report 2001-4, Department of Industrial and Systems Engineering, University of Florida.