AIAA 2002-3455

EVOLUTIONARY APPROACHES TO PATH PLANNING THROUGH UNCERTAIN ENVIRONMENTS

David Rathbun, Ph.D. *
University of Washington, Seattle, WA 98195-2400

Brian Capozzi, Ph.D. †
Metron Aviation, Inc., Herndon, VA 20170

ABSTRACT

Evolutionary algorithms (EA) have been successfully used to compute near-optimal paths through obstructed, dynamically changing environments. The locations of the obstacles that form the obstructions in these environments may only be known with limited accuracy. Explicitly accounting for this uncertainty can result in the survival of "best" paths which differ from those that would be favored in a purely deterministic environment. In this paper, we consider the application of evolution-based path planning to the motion of an unmanned air vehicle (UAV) through a field of obstacles at uncertain locations. Specifically, we focus on the "cost function" utilized by the evolutionary algorithm to judge the likelihood of a given path successfully traversing the uncertain environment. We first show a method for computing a cost function based on the exact probability of intersection of the vehicle with an obstacle. A more computationally tractable approximation technique for this cost function is then derived. Both cost functions are compared to the weighted graph search technique found in much of the literature on path planning.

INTRODUCTION

The number of applications that are considering the use of intelligent un-manned air vehicles (UAV's) is increasing dramatically. These applications include areas such as reconnaissance, search and rescue, and weather observation. However attrition rates for UAV's used in such applications can be high, particularly when the vehicle must operate autonomously through an environment that is different from that which had been envisioned by the human operators.

The use of UAV's will become more effective and cost efficient as the vehicles achieve a greater ability to intelligently modify their internal plan based on changes in the environment. One of the areas in which greater autonomy could benefit the application is in the area of path planning. These applications require a UAV that can autonomously plan its path to achieve a set of goals given an estimate of the environment through which it flies. More specifically, they require a vehicle that can modify its path enroute based on new data about that environment, while still making a best effort at achieving the mission goals.

A number of authors have suggested algorithms for autonomous path planning. These include methods such as potential fields [1], graph search methods like A* and D* ([2], [3]), and evolutionary algorithmic (EA) techniques ([4], [5]).

In this paper we will focus our attention on methods based on evolutionary algorithmic (EA) techniques. The EA-based methods have a number of desirable features. One is their ability to consider nonlinear performance metrics. This includes such things as time of arrival constraints, nonlinear vehicle motion capabilities, and number of passes over a desired target. Another desirable feature is that they do not require complete re-planning when environmental parameters change. At any time, the algorithm retains a broad range of solutions to draw upon, which provide a good set of initial conditions for solving the "nearby" problem.

The path planning algorithm (EA-based or otherwise) will be creating paths for motion through a given environment. In a real-world scenario, there will be uncertainty in the parameter estimates that describe the environment. For a tactical scenario, this could be uncertainty about the exact location of obstacles and radar detection threats to the UAV. In other scenarios, this could be the future locations of other vehicles. The path plan-

*Research Associate, Department of Aeronautics and Astronautics, email: rathbun@veribox.net.

†Systems Analyst, Aviation Research and Development, email: capozzi@metsci.com, AIAA Professional Member

ner must avoid these obstacles, even though their locations are not fully known.

A fully capable planner must consider not only the best estimate of the obstacles in its environment, but also the uncertainty associated with that estimate. An object with an accurate estimate of its location can be passed very closely. An object with a more uncertain estimate might be better off being avoided. An object with an extremely large uncertainty should have relatively small effect on the outcome of the planning algorithm.

One approach to handling environmental uncertainty is the Markov localization of Fox ([6], [7]). For EA-based methods, uncertainty handling was presented in a simple fashion by Latourell [8].

In this paper, we will consider the application of an evolutionary algorithm based search method for path planning through an environment that contains a number of obstacles, while explicitly accounting for the uncertainty in the estimates of the location of those obstacles. In section 2, we give the path planning problem formulation and show the structure of the EA-based planner. We will show how the solution can be expressed in terms of a function that returns the probability that a given path will intersect a single obstacle. In section 3, we present a method for calculation of the exact probability of intersection. In section 4, we develop a method for approximating the exact solution, but in a more computationally efficient manner. In section 5, we will relate the use of this approximate solution to the graph based search methods for handling uncertain environments. In section 6, we will present simulation examples of an EA-based path planner using the derived approximation techniques.

EA PLANNING

In order for an unmanned air vehicle (UAV) to move autonomously between desired locations, it needs to be able to determine a path that does not interfere with any of the obstacles that may be in the region that the vehicle transverses. In general, the vehicle must also be able to consider a number of additional constraints on the path planning problem, such as motion across desired target regions, speed and maneuverability constraints, or time of arrival requirements. However, for the purposes of analyzing the behavior of a search algorithm relative to an uncertain environment, we will consider only the ability to arrive at a goal location while avoiding obstacles.

We define the simple search problem as determination of an algorithm to find a path that is; [A] continuous from the start location to the goal location, [B] has a length less than some maximum value, and [C] maximizes the probability of reaching the goal without intersecting an obstacle. Obstacles in the environment will be expressed as; [A] a known size and [B] a probability density field describing the location of the center. We will present the discussion in a two dimensional (2-D) context. We will mention the three dimensional (3-D) problem only when it requires something beyond a simple extension of the 2-D concepts.

For implementation on-board the vehicle, this resultant path planning algorithm would be run in an iterative fashion. A solution would be found, some initial segment of the path would be run, and then new estimates of the obstacle locations would be formed based on observations at the new location. The next iteration would then be run, using as much information as possible from the previous solution about the best path. In effect, we are computing a "near term" path that puts the UAV at a location to give a better probability of achieving "long term" success, though we don't know a-priori what the true value of that measure of success will be.

We will start here with a very brief description of an evolutionary algorithm (EA) based method for solving this path planning problem. Those who desire a more in-depth description of EA or it's application to path planning through environments with fully known obstacles should see ([9],[10]).

The design of the EA-based path planning algorithm starts with: [A] a way to encode a path as a finite sequence of numbers (the genetic material), and [B] a (non-linear) cost function that scores the relative value of a candidate solution (the fitness). The algorithm is initialized with a grouping of encoded paths (the population). See figure 1. The algorithm is run in a loop. At each step (or generation), additional paths (members) are first added to the population by modifying current members (a mutation) or by mixing two or more current members (reproduction). Then the cost function is used to score all members of the larger population. Based on those scores, the population is reduced back to its initial size. Those paths that score well are likely to be retained, those that score poorly are likely to be dropped.

As the iterations progress, a well designed EA algorithm will produce a population of diverse solutions that score well against the problem requirements. The disadvantage in using a path planner based on an EA formulation is that the population may not progress to a truly optimal solution (the solution is optimalizing, but not optimal). This is often outweighed by both the robustness of the algorithm to changes in the environment as a result of the large diverse population, and by the ability to handle a wide variety of complex constraints in the cost function.

From here on, we will limit our focus to the cost function of the EA-based path planner. For the EA-based path planner to work well, its cost function must give a good

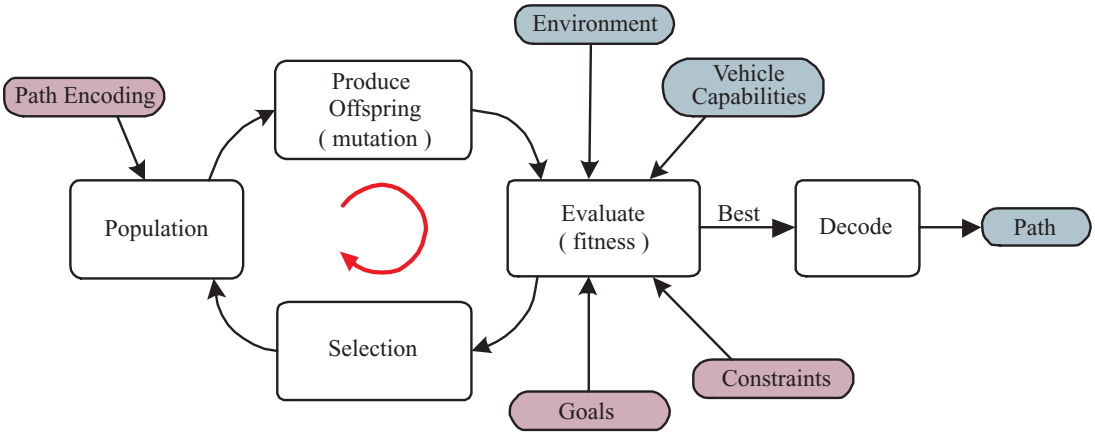


Figure 1: Overview of Evolutionary Search

measure of how well a candidate path P_i meets the requirements of the problem. For the simple problem addressed here, we will model the cost function as a linear combination of the path length and the probability of intersection with an obstacle:

$$J_i = a_1\, G\,(P_i) \;+\; a_2\, F\,(P_i) \;+\; a_3\, B_1\,(P_i) \qquad (1)$$

where $G(P)$ represents a measure of the proximity of the path end point to the goal location, $F(P)$ is the distance by which the path exceeds the maximum path length, $B_1(P)$ represents the probability that the path intersects an obstacle, and a_1, a_2, and a_3 are relative weighting factors. The EA-based path planner attempts to find a path that minimizes this cost function.

If we can assume that the location of each obstacle is independent of the location of all other obstacles, we can express the probability of intersection as,

$$B_1\,(P_i) = 1 - \prod_j \{1 - B_2\,(P_i, O_j)\} \qquad (2)$$

where $B_2(P_i, O_j)$ is the probability that the i^{th} path, P_i, intersects with the j^{th} obstacle, O_j.

The rest of this paper will focus on methods for computing $B_2(P_i, O_j)$, and how those methods relate to the performance of an EA-based path planner.

PROBABILITY OF INTERSECTION

The true probability that a vehicle following along path P will intersect (hit) an obstacle O can be defined mathematically as follows:

For an obstacle of radius R_o and a vehicle of radius R_v, the vehicle will hit an obstacle with a center location of $C_x(O)$ if at any length s along its path P,

$$|P\,(s) - C_x\,(O)\,| < D \qquad (3)$$

where the "intersection distance" D is defined as

$$D = R_o + R_v \qquad (4)$$

This can thought of as either the vehicle getting too close to the obstacle, or, alternatively, that the location of the obstacle gets too close to the path.

So if we define a region in space, $\mathbf{X}$, as all points within a distance D from the path P,

$$\mathbf{X} = \{\mathbf{x} :\; |\mathbf{x} - P\,(s)\,| < D, \text{for all } s\} \qquad (5)$$

then if $C_x(O)$ is contained in $\mathbf{X}$, the vehicle will hit the obstacle. See figure 2. Note that, as pictured, if the minimum turn radius of the vehicle is less than D, the determination of the boundaries of region $\mathbf{X}$ is not a simple matter.

However, for an uncertain environment, the location of the center of the obstacle, $C_x(O)$, is not known. Rather, it is described by the probability density function $\rho(\mathbf{x})$,

$$\rho\,(\mathbf{x}) \simeq \frac{\text{probability}[\,\mathbf{x} - \epsilon < C_x(O) < \mathbf{x} + \epsilon\,]}{2\epsilon} \qquad (6)$$

The probability that the vehicle will hit the obstacle is the integral of the density function over the area where an intersection is possible:

$$B_2\,(P_i, O_j) = \int_X \rho \;\mathrm{d}x\,\mathrm{d}y \qquad (7)$$

Because of the form of the solution, for any path more complex than a straight line passing in the vicinity of an obstacle, a simple measure of the geometry of the interaction could be a very bad approximation to the true probability. Consider a simple measure of the probability

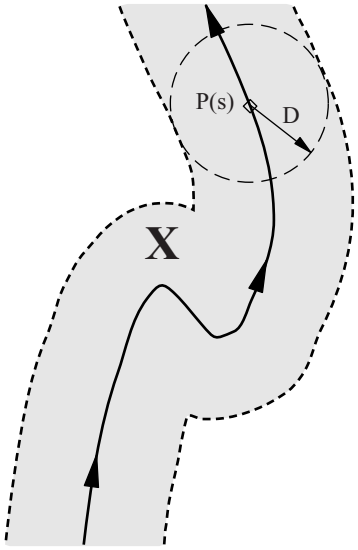


Figure 2: Obstacle Intersection Geometry

of intersection such as one based on the closest approach of the path to the expected value of the location of the obstacle center:

$$B_2\left(P_i, O_j\right) = \text{Func}\big(d\big) \tag{8}$$

$$d = \min[\ |P_i\left(s\right) - \eta_j|,\ \text{all}\ s\ ]$$

$$\eta_j = E[\ C_x\left(O_j\right)\ ]$$

where $E[\,]$ is the expectation operator.

Figure 3 shows two paths, P_1 and P_2, that pass an obstacle at the same closest approach distance d. For a simple density field $\rho(\mathbf{x})$ and an approach distance D smaller than the radius of the density field, the true probability of intersection for path P_1 is much greater than for path P_2.

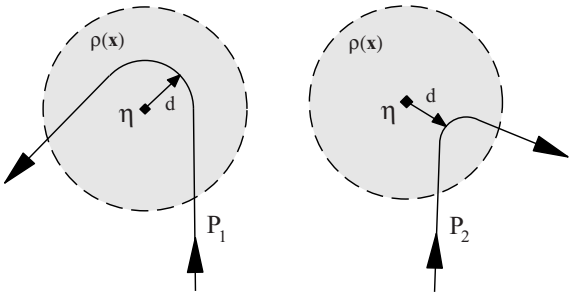


Figure 3: Failure of Simple Estimation Methods

The difficulty with the exact formulation of the probability as given in equation 7 is the ability to compute the result in a closed form manner. We know of no closed form solution of the integral for a path that consists of a piecewise continuous set of path segments (such as would be produced from an EA-based path planner).

A numerical approximation to the integral is possible. The probability density function $\rho\left(\mathbf{x}\right)$ could be evaluated over a finite element mesh $\rho\left(x_i, y_j\right)$ of small area $\Delta x \cdot \Delta y$. The path could similarly be discretized into small segments $P\left(s_k\right)$ of length Δs. For each path segment $P\left(s_k\right)$, all probability grids $[x_i, y_j]$ that are less than distance D from the path segment are flagged. At the end of the evaluation, all marked grids are summed and multiplied by the grid area.

Figure 4 gives a depiction of this digital approximation for a probability density field that is non-zero over a finite radius σ. This digital approximation is the method used in this paper to evaluate the true probability. The difficulty with such a method is the computational load required for the evaluation. Each evaluation in 2-D space requires three nested loops; the path segments, the density function x grid, and the density function y grid. Evaluation in 3-D space requires four nested loops.

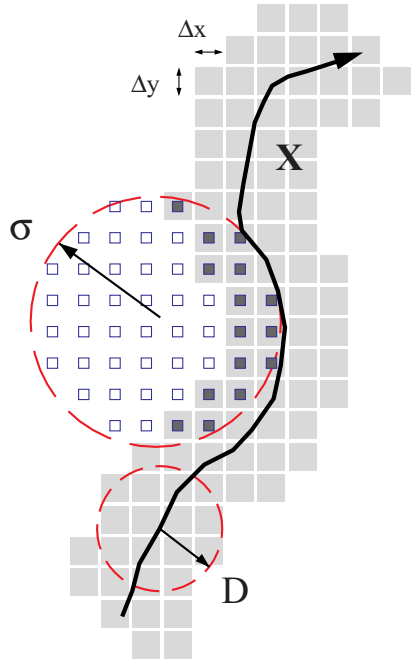


Figure 4: Digital Computation of Exact Solution

FIELD INTEGRAL APPROXIMATION TECHNIQUE

In this section we analyze a method for approximating the exact probability defined in the previous section in a

more computationally efficient manner. We will assume a continuous field $\beta()$ related to the probability density field for the location of the center of the obstacle. The probability that the vehicle hits the obstacle will be defined as the integral of the path length through the field.

$$B_2\left(P_i, O_j\right) \equiv \int_{P_i} \beta_j \, \mathrm{d}s \tag{9}$$

The question that needs to be answered is how to determine such a function, and to what degree it produces a good approximation of the exact probability.

Examine figure 5. The figure shows two paths which move past an obstacle. We assume that the probability density function for the location of the obstacle center is radially symmetric and non-zero over a finite radius σ. The distance at which an intersection between the vehicle and obstacle can occur $D = R_v + R_o$ is equal to σ. In 5(a) exactly half the density function is in the region $\mathbf{X}$ for that path, and so the probability is exactly 0.5. In 5(b), using similar arguments, the probability for that path must 0.75. Therefore, if using a field integration method, the amount contributed by path segment S_3 must be 0.25.

$$\int_{S_3} \beta \, \mathrm{d}s = 0.25 \tag{10}$$

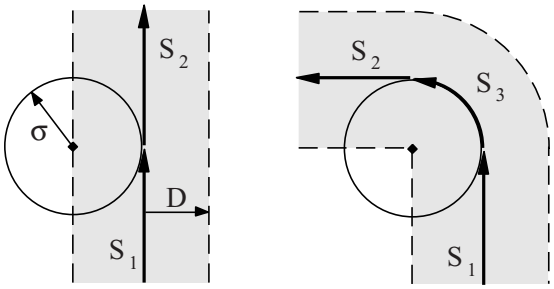


Figure 5: Thought Experiment

Therefore, we will define the field $\beta\left(r\right)$ as the value required to give the correct probability of intersection given a path of a single 360° revolution about the expected value of $C_x(O)$ at a distance of r. This results in,

$$\beta\left(r\right) = \frac{p\left(r+D\right) - p\left(r-D\right)}{2\pi r} \tag{11}$$

where $p(r)$ is the probability that $C_x(O)$ is within a distance r of the expected value (i.e. the probability distribution).

$$p\left(r\right) = \int_0^r 2\pi \, y \, \rho\left(y\right) \, \mathrm{d}y \tag{12}$$

where, for $r < 0$, we define:

$$p\left(r\right) = -p\left(\, |r| \,\right) \qquad r < 0 \tag{13}$$

For simple types of paths, a closed from solution to the integration of equation 9 may be possible. For more complex types of paths, a digital approximation will once again be used. However, there is a computational advantage to the field integration method due to the fact that; [A] the required integration of $\rho\left(r\right)$ can be done off-line and closed from, since the area is simple and known a-priori, and [B] a discrete version of the algorithm requires only a single summation loop over the path segments.

$$B_2\left(P_i, O_j\right) = \prod_k \beta\left(\, |P_i\left(s_k\right) - \eta_j| \,\right) \Delta s \tag{14}$$

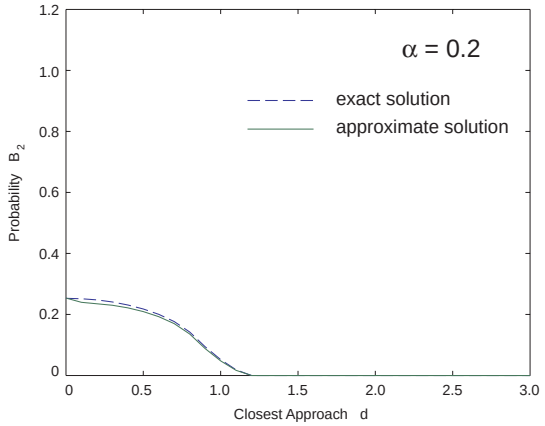
$$\eta_j = E[\, C_x\left(O_j\right) \,]$$

To examine how well the field integration method approximates the true solution, consider the case of a simple path, similar to figure 5(a), where the path is a straight line past the obstacle at a closest approach of d. The probability density function is uniform with a radius of $\sigma = 1$. The intersection distance D is set to $\alpha\,\sigma$. For this simple problem, a closed form solution for the exact probability can be found. Figures 6(a)-6(c) plot the exact solution and approximation as a function of the approach distance d for various values of α.

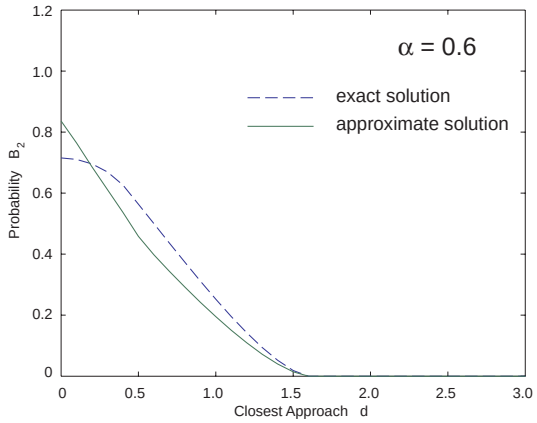
From figure 6(a)-6(c), we see that the field integration method can produce good results for many cases, but is not exact. The approximation deteriorates for larger values of α (smaller size of the probability density function), though it still gives correct results in regions of complete or non-existent intersection. The effect of this is to provide a good approximation for more uncertain obstacles, and to give an indication of hit/no hit for more certain obstacles.

We have found that, for our example problem of a vehicle passing an obstacle along a straight line, the field integration method is relatively insensitive to the size of the discritization interval. Good approximations are obtained for any path interval Δs of $\sigma/5$ or smaller. For probability density functions significantly more complex than the uniform and triangular density functions we have tested against, smaller discritization intervals Δs may be required to capture the behavior of the intersection probability.

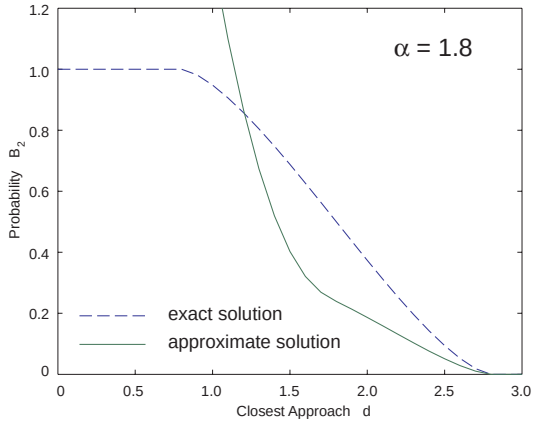
Since the field function $\beta\left(r\right)$ that we have defined gives the only values that produce a correct result for one type of path (a single revolution), but at the same time gives an incorrect result for another type of path (a straight line), we can make the claim that any method based on integrating or summing a static field without consideration of the history of the path cannot produce a correct result for all paths.



(a) Large Distribution Radius



(b) Medium Distribution Radius



(c) Small Distribution Radius

Figure 6: Evaluation of Path Integration Method

COMPARISON TO GRAPH-BASED SEARCHES

The field integration method can be compared to methods used in graph based search algorithms. In the graph based search algorithms, the area that the vehicle may traverse is divided into a set of discrete nodes. Each node is assigned a weight, and the cost of the path is produced from a summation of the weights over the nodes traversed by the path. Searches for the "best" path are made using network optimization techniques such as Dijkstra's method, A^*, or D^*. Typically, these nodal weights are assigned as an estimate of the probability that the obstacle will occupy a given position. In the extreme, these nodal weights are assigned either a one or zero. This is particularly the case if the objective is simply to find an obstacle-free path through the environment.

Obviously, given the correct assignment of the node weights, the graph search method can be thought of as a discrete version of the field integration method. Therefore, a graph based search would have the same property as the field integration method, where the solution cannot be exact for all paths, regardless of the discretation size.

For a simple path planning problem where obstacle intersection is the only concern, a node weight such as the probability of intersection at that location may be sufficient to produce an obstacle free path. For a more complex set of requirements, where the probability of intersection needs to be directly compared against other constraints (such as a limited amount of fuel or power), it would be necessary to scale the node weights to produce an accurate estimate of the exact probability. The scaling would have to account for various values of the intersection distance D and the size of the grid spacing. Such scaling would enable the search algorithm to make explicit tradeoffs between potentially conflicting objectives.

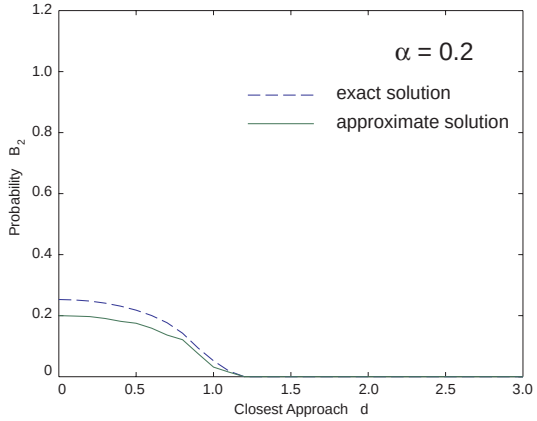
Figures 7(a)-7(c) repeat the tests of figures 6(a)-6(c) for a graph-based search. For purposes of this comparison, the node weights are set according to

$$W[n] = \frac{w\,(n)\,\Delta x}{2D} \tag{15}$$

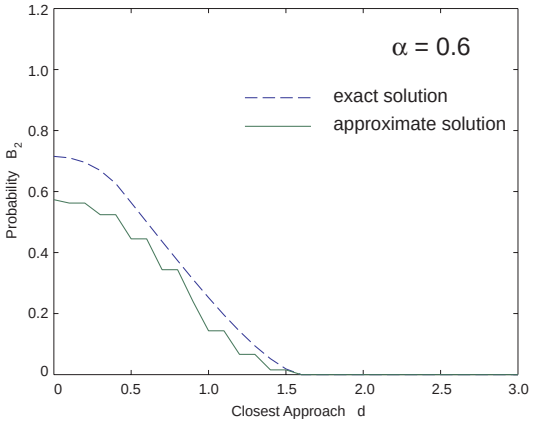
where $w\,(n)$ is the probability of intersection if the vehicle was at the node n location, and Δx is the distance between nodes. The grid size for this example was set to $\sigma \min\{1/5, \alpha/3\}$. This is comparable to the discritization size used in figures 6(a)-6(c), while still large enough to show the digital effects. Techniques are available for reducing the computational load of such graph search methods by adaptively changing the node spacing

Δx, but those techniques are beyond the scope of this paper.

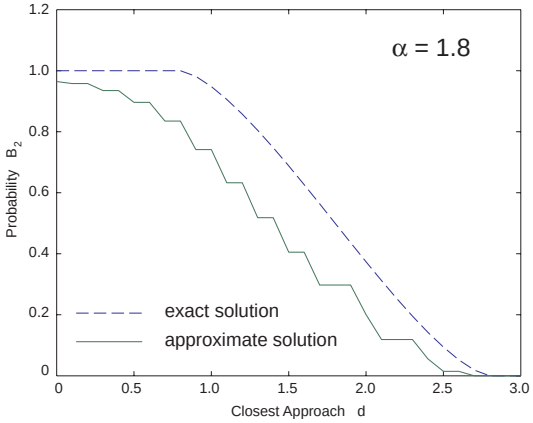
Comparing figures 6(a)-6(c) and 7(a)-7(c), we can see that there are slight differences between the field integration and graph search methods, but the results are similar. This is not unexpected, as both methods are based on similar principles. Other considerations, such as computational efficiency and the ability to accommodate time based probabilities may be important in distinguishing which method is more appropriate for a given problem. For application to an EA-base path planner, the field integration method provides certain flexibility of use that the graph based search does not. It is important to remember that, regardless of the discritization size, both methods are only approximations to the true probability of intersection.



(a) Large Distribution Radius



(b) Medium Distribution Radius



(c) Small Distribution Radius

Figure 7: Evaluation of Node Summation Method

EXAMPLE EA SIMULATION

In this section we present the results of simulations which demonstrate the use of the field approximation technique to compute probabilities of intersection to drive simulated evolution. The cost function utilized in these simulations is identical to that given in equation (1), namely a weighted combination of the probability of intersection with a function penalizing excessive path length.

For the purpose of demonstration, we choose a relatively simple problem domain in which a set of obstacles, each of known size, is distributed through the environment. The objective of the planner is to find a path through the obstacle field that reaches the goal location. The actual location of each of the obstacles is only known approximately. Uncertainty is explicitly modelled in terms of a probability density describing the obstacle center location. This density is assumed uniform in all directions, with a modelled radius of σ.

For the uniform obstacle probability density function, the distribution function used to construct the field as defined in equation 11, is given as,

$$p(r) = \begin{cases} r^2/\sigma^2 & r \leq \sigma \\ 1 & r > \sigma \end{cases} \tag{16}$$

The evolutionary model used for illustration is a genetic algorithm, where each individual in the population consists of a "string" of arbitrary length. Each "character" in the string represents a steering or speed command (e.g. speed up, slow down, turn right, etc.). The dynamics of the vehicle are represented through a simple 2-D kinematic model assuming constant acceleration and turn rate. The vehicle's maneuverability is constrained

via a limited turning rate (e.g. turn radius) and acceleration/deceleration. The vehicle's trajectory is modelled at a one-second resolution.

In figures 8-10 we present the results comparing three different cases using identical start location, goal location, expected obstacle locations, and known obstacle sizes. For each case, we only vary the size of the uncertainty in the estimate of the obstacle center location. For all three figures, the solid line represents the path from the EA-based search algorithm with the best fitness after a fixed number of generations. The thin lines are other (less fit) members of the population. The red circles represent the non-zero area of the probability density function. The gray circles represent the locations that could possibly be covered by the obstacle (radius $\sigma + D$). The white circles represent the actual location of the obstacles (unknown to the planning algorithm).

In figure 8, we show the route generated when the obstacle locations are assumed to be known nearly exactly (e.g. zero uncertainty). For this case, the red circles are small, since the obstacle locations are known with high certainty. Because of the high certainty, the algorithm chooses a path that passes through the small space between the obstacles which is known to be free of intersection.

In figure 9 we illustrate the effect of a medium sized uncertainty. In this case, the algorithm chooses a path that circles around all possible obstacle locations, since the probability that a shorter route between them (as in figure 8) would be obstacle free is too low.

In figure 10 we demonstrate the difference in trajectory obtained when a very high level of uncertainty is present. For this case, the algorithm chooses a nearly straight path to the goal. Since the (relatively small) obstacles could be nearly anywhere in the space, the probability that the straight line path is free of an obstacle intersection was high enough that it outweighed the added path length required to completely circle the region.

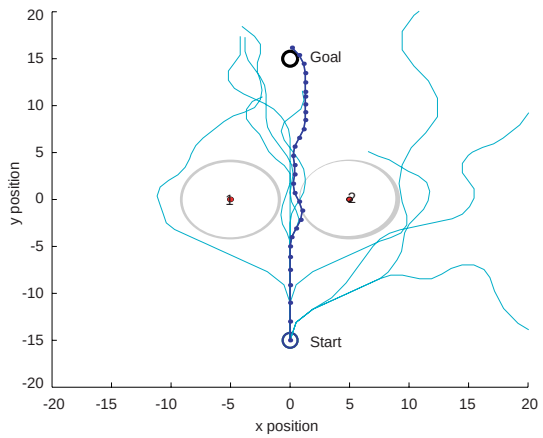


Figure 8: Simulation Results for Certain Obstacle Locations

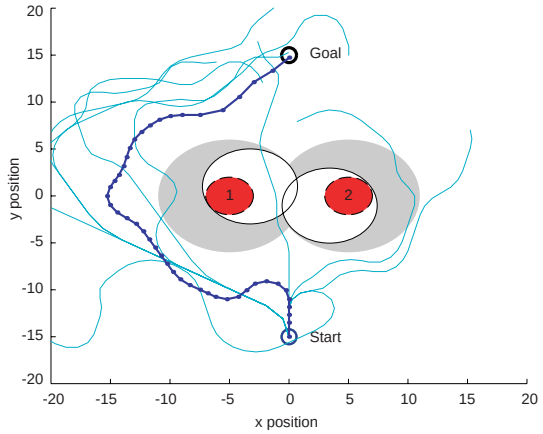


Figure 9: Simulation Results for Uncertain Obstacle Locations

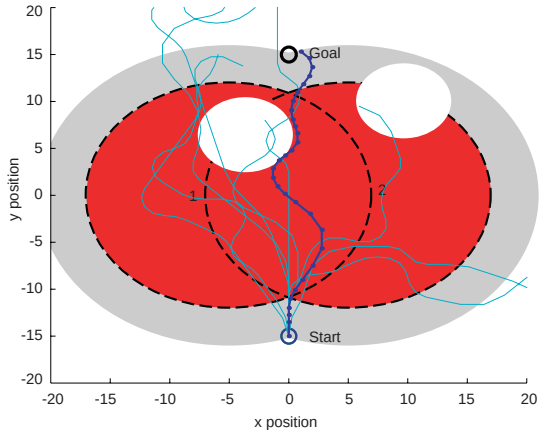


Figure 10: Simulation Results for Highly Uncertain Obstacle Locations

CONCLUSIONS

We have presented a simple framework for path planning using evolutionary algorithms that allows for consideration of uncertainty in estimates of the parameters describing obstacles in the environment. The planner was defined in terms of a function that produced the probability of intersection of a candidate path from the uncertainty specification of an obstacle.

We presented the theoretical solution for getting this probability of the intersection of the vehicle and the obstacle. Because this solution was based in the integration of a function over an arbitrary and nonlinear region,

closed form solutions are difficult and digital approximations are computationally intensive.

We compared the exact method with simpler methods based on either an integration or summation of a fixed field defined over the space of the environment. We showed that these methods cannot produce completely accurate results, but can give reasonable approximations for many scenarios. We showed that the approximation techniques had the most difficulty for tight radius turns or intersection distances of the same size as the obstacle probability density field.

We presented one method for defining an integration field for a radially symmetric probability density function that gave a good approximation to the exact solution.

ACKNOWLEDGMENTS

This research for this paper was completed under funding from the DARPA Mixed Initiative Control of Automa-Teams (MICA) project. The authors would like to thank both DARPA and our contracting agency, SPAWAR Systems Center San Diego, CA, for their support that made this work possible.

References

[1] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[2] J. S. Mitchell and D. M. Keirsey, "Planning strategic paths through variable terrain data," in *Proceedings of the SPIE Conference on Applications of Artificial Intelligence*, vol. 485, (Arlington, VA), pp. 172–179, 1984.

[3] A. Stentz, "Optimal and Efficient Path Planning for Partially-Known Environments," in *Proceedings of the 1994 International Conference on Robotics and Automation*, vol. 4, (Los Alamitos, CA), pp. 3310–3317, 1994.

[4] D. B. Fogel and L. J. Fogel, "Optimal Routing of Multiple Autonomous Underwater Vehicles through Evolutionary Programming," in *Proc. of the 1990 Symposium on Autonomous Underwater Vehicle Technology*, (Washington, D.C.), pp. 44–47, 1990.

[5] B. J. Capozzi and J. Vagners, "Evolving (Semi)-Autonomous Vehicles," in *Proc. of the 2001 AIAA Guidance, Navigation and Control Conference*, (Montreal, Canada), August 2001.

[6] S. Thrun, D. Fox, and W. Burgard, "A Probabilistic Approach to Concurrent Mapping and Localization for Mobile Robots," *Machine Learning and Autonomous Robots*, vol. 31, 1998.

[7] D. Fox, W. Burgard, and S. Thrun, "Markov localization for mobile robots in dynamic environments," *Journal of Artificial Intelligence Research*, vol. 11, 1999.

[8] J. Latourell, B. Wallet, and B. Copeland, "Genetic Algorithm to Solve Constrained Routing Problem with Applications for Cruise Missile Routing," in *SPIE Proceedings, Applications and Science of Computational Intelligence*, vol. 3390, pp. 490–500, 1998.

[9] J. Xiao *et al.*, "Adaptive evolutionary planner/navigator for mobile robots," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 18–28, April 1997.

[10] B. J. Capozzi, *Evolution-Based Path Planning and Management for Autonomous Vehicles*. PhD thesis, University of Washington, 2001.