Covert Robotics: Covert Path Planning in Unknown Environments

Mohamed Marzouqi and Ray A. Jarvis
Intelligent Robotics Research Centre
MONASH UNIVERSITY
Victoria, 3800, Australia
{mohamed.marzouqi, ray.jarvis}@eng.monash.edu.au

Abstract

This paper is an extension to our research on hidden path planning: Covert Robotics. A new method is introduced here that allows a mobile robot in initially unknown environments to plan a covert path to a specified goal location in the absence of information about possible hostile sentries within the same environment. The aim is to balance between discovering a path to the goal and minimizing the robot's exposure to the open and unknown spaces, as possible sentries might be anywhere. The presented approach is based on the concept of updating, during navigation, each location in the environment map with an estimated visibility cost value that represents the degree of exposure to the entire environment as can be judged with current map knowledge. The visibility cost at each location is integrated with its distance cost to the goal to plan and maintain the covert path. Promising results have been shown when testing the planning technique on a simulated environment. A number of test cases are presented.

1 Introduction

Covert Robotics is an interesting research problem which we have presented earlier in [Marzouqi and Jarvis, 2003; Marzouqi and Jarvis, 2004]. The aim of the research is to create autonomous mobile robots with the ability to accomplish different types of navigation missions covertly, i.e. without being observed by one or more, possibly hostile, sentries operating within the same environment. A covert path is defined here as a path with the minimum exposure to being observed by sentries anywhere in the free-space of the environment.

There are many applications where covert robotics systems can be useful, such as in police and military fields for surveillance, spying, and crime prevention missions. Covert missions are usually dangerous and can put human life at risk. Robots can be used to minimize such risk. Moreover, robust covert robots may perform better than humans can do, especially when real time and complex decision making is required.

Finding an optimal path between two points has been given a lot of attention in the robotics field, where the goal is to minimize distance, time or power. On the other hand, there has been very little work on planning covert paths although demand is increasing for robots to participate in high risk situations where the covert behavior is required. In [Teng et al., 1993; Ravela et al., 1994], a research has been conducted for covert navigation in natural terrain environments, where a robot vehicle makes use of the terrain elevation to maintain its covertness given a known sentry location. Another work has been done in [Birgesson et al., 2003] where a robot navigates in an unknown environment to a specified destination while taking advantage of the discovered obstacles to stay hidden from a stationary known observer.

We have presented in [Marzouqi and Jarvis, 2003] a solution that allows a mobile robot to navigate covertly from one point to another in a known, static environment where sentries' locations are unknown and they might be moving around. Obstacles are used as the only means to stay unobservable, they being assumed opaque and higher than all agents' sight levels.

The basic idea was to assign, prior to navigation, a visibility cost to each free-space location. As sentries' locations are unknown, it is assumed there could be a sentry standing at each free-space location. Given that, the visibility cost at a point is the number of sentries visible to it (i.e. its exposure to the entire environment). Using the Distance Transform algorithm [Jarvis, 1984], a known shortest collision-free path planning algorithm (described in section 2.1), the distance costs to the goal at each point are integrated with the visibility costs to find the global covert path.

In this paper, the research on covert robotics has been extended to deal with unknown environments as well as unknown sentries' locations. Given an initially unknown, static environment, a robot needs to balance between discovering a short path to a specified goal location and minimizing its exposure to the open and unknown spaces in the environment where a possible sentry might be standing. A similar approach to [Marzouqi and Jarvis, 2003] has been used here with an assumption that unknown spaces are free of obstacles (until proven otherwise) to provide a worst case situation. During navigation, each location is updated with an estimated visibility cost given the current map knowledge as it is accumulated. The global covert path is maintained accordingly.

The final results have shown the effectiveness of this approach to find covert paths in unknown environments. However, using this approach with unknown environments is very time consuming as the visibility costs need to be updated continuously as the robot gets new information about the environment's occupancy state. This problem has been addressed using a fast updating method which allows a reasonable computation time for a practical use. The time complexity of the approach is described in section 3.

In the next section, the Distance Transform algorithm is described as an important relevant tool, followed by a detailed description of the approach. The planning technique was tested using a simulator. A number of test cases are presented in section 4 that show promising results.

2 Algorithms Involved

2.1 Distance Transform Algorithm

This subsection explains briefly the Distance Transform (DT) based path planning algorithm [Jarvis, 1984] in order to understand its usage later in this approach. Basically, the DT algorithm finds the optimal collision-free path between two points in a known, stationary environment. The core idea of this algorithm has been represented in [Barraquand et al., 1989] as a numerical potential field technique known as Wavefront algorithm.

To find the shortest path between two points using the DT algorithm, the environment is represented by a two dimensional map of exact cell decomposition that each of its cells representing either a free-space or an obstacle. The distance cost for each cell is derived based on propagating the distance costs of the surrounding cells. The cell representing the destination point is given a distance cost equal to zero. All other cells are initially set to a very high value. The process is constructed as a two pass traversal of the 2D map array: a forward raster order pass (left to right, top to bottom), and a reverse raster order pass (right to left, bottom to top).

In each pass, each free-space cell (obstacles are skipped) is assigned the value of one greater than the least value of the four neighbors previously visited on that pass, this assignment occurs only if the new value is less than the previous value at that cell. Both passes are repeated until no further changes to cell assignment occur. The final distance cost at each cell represents the minimum number of cell steps to the destination's cell. The optimal path can be found from any starting point by looking at its 8-neighbor cells and moving to the one with the lowest distance cost. This process is repeated until the destination's cell is reached.

The DT algorithm is used as well to plan a path to a goal in the case of an initially unknown environment. This is possible if the robot is equipped with a sensor that incrementally updates the environment map with obstacles' locations. An assumption is made that considers the unknown areas to be free-spaces unless discovered otherwise. The planned path needs to be updated only when an obstacle is found in the previously planned path.

2.2 Covert Path Planning Method

The process of finding a covert path in an unknown environment is described in this subsection. The approach is presented briefly in Algorithm 1. A visibility and an occupancy maps are created and updated as the robot navigates towards the goal. The maps are represented as two-dimensional arrays of exact cell decomposition. Both maps have the same size.

```
- While (goal is not reached yet)
{
 - Store the new seen obstacles in a list
 - For (each new found obstacle)
   {
     - Update the visibility map using the fast update method
     - Add the obstacle to the occupancy map
   }
 - Update the global covert path
 - Move one step in the created path
}
```

Algorithm 1. The covert path planning approach (pseudo code)

Initially, all cells in the environment occupancy map are assumed to be unoccupied by obstacles (in free-space state) until proven otherwise. This assumption leads to initially set the visibility costs of all cells in the visibility map to a maximum cost (i.e. the total number of cells which is the map's width x its height). The cost at each cell decreases as more cells are known to be occluded by new found obstacles. Assuming all cells are free-spaces initially is important as it is the worst case scenario in term of covertness. Subsequently, the robot will get a covert discovering behavior as it will utilize the discovered obstacles to minimize its exposure to the areas that are assumed or known to be free-spaces.

This approach mainly works with indoor (bounded) environments that its size is even known or can be guessed (by giving it the largest expected size). The shape of the environment is assumed to be rectangular. The approach can work as well with outdoor environments if virtual borders are specified where sentries are assumed to be operating only within those borders. In this case, the covert path will be planned within the bounded area.

The process goes through a loop of subtasks while the current robot's location is not the goal destination. The subtasks are: updating the visibility and occupancy maps, updating the global covert path, and moving one step in that path. The robot is assumed to have a panoramic vision system (i.e. 360 degree of vision) to obtain information about obstacles' locations. At the start of each loop, the new seen *obstacle cells* are stored in a created list. Each obstacle cell is used to update the visibility map before updating its location in the occupancy map. As mentioned earlier, a fast method is used to update the visibility map. This method depends on identifying and recalculating the only cells whose visibility cost might be affected by a discovered obstacle cell. This method is described as follow:

To update the visibility map for a new found obstacle cell, we look from the obstacle location towards each *map border cell*. The number of cells along the imaginary line connecting the obstacle and a border cell are counted. An array of the same size as the environment map is created where the counted number is stored at the corresponding border cell's location. Adjacent lines might share one or more cells, it is important not to count these cells more than once by marking the counted cells. During the counting process, the free-space cells that are visible to the obstacle cell (not occluded by other known obstacles) are stored in a list. These cells are the only ones whose visibility costs might be affected by this obstacle cell.

To find the new visibility cost for an affected free-space cell, the number of cells occluded by the obstacle cell is subtracted from its current visibility cost. The number of the occluded cells is the sum of the values stored in the occluded border cells. An example is shown in Fig. 1 and 2 describing that in details. In Fig. 1, only one obstacle cell is known so far which is represented by the letter 'O'. The cell represented by 'C' is an affected cell whose visibility cost needs to be recalculated. For clarity, the cells occluded by O from the viewpoint from C are in grey. The cells in dark grey are the occluded border cells. The sum of the values each of these border cells holds will be subtracted from C's current visibility cost.

As the occluded border cells are a connected set of cells, they can be determined by locating their two edges, 'A' and 'B'. The line equation (y = m x + b) is used here to find the two intersections of the $\overline{CO}$ line with the map's border. However, as O is a cell that does not represent an absolute point location, two neighbors to the obstacle are chosen to find each intersection. The choice depends on the location of O to C (see Fig.3). For the case in Fig. 1, the upper neighbor 'U' is used to find A, and the right neighbor 'R' is used to find B. It should be mentioned here that using this fast method does not guarantee finding the exact number of occluded cells; however, this inaccuracy does not affect the quality of the generated covert path as it is the ratio difference between cells' visibility costs is what matters.

In Fig. 2, another obstacle cell 'O2' is found that also affects the visibility cost at C. It is noticeable that some occluded cells by O2 have been considered earlier with O. To prevent counting those cells again when updating the visibility cost at C, it is important that each free-space cell remembers the last value at each occluded border cell. When a new obstacle cell occludes a border cell that was occluded by other obstacle before, then the difference between the current and the remembered value at that border cell is what will be subtracted from the cell's visibility cost. For that purpose, a 3D memory array is initially created. Its dimensions are the height, the width, and the perimeter of the environment map.

To apply that in our example, the values at the occluded cells in Fig. 1 are copied to its corresponding location in the memory array for C. To find the new visibility cost for C given O2 in Fig. 2, what will be subtracted from C's visibility cost is the sum of the difference between the new value and the memorized value at each occluded border cell. The new values at the occluded cells are copied to the memory array, erasing the previous ones.

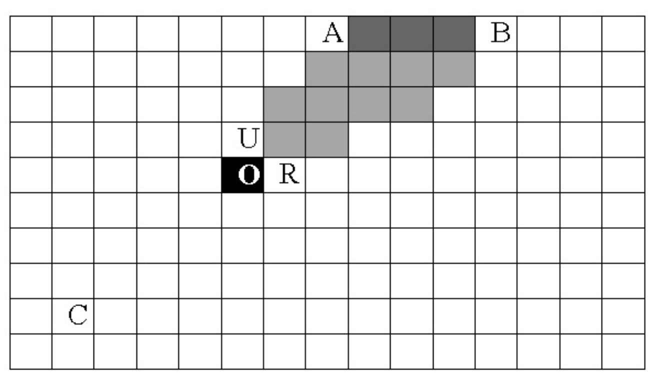

Figure 1. The total number of cells occluded by O from C is the sum of all stored values in the border cells between A and B.

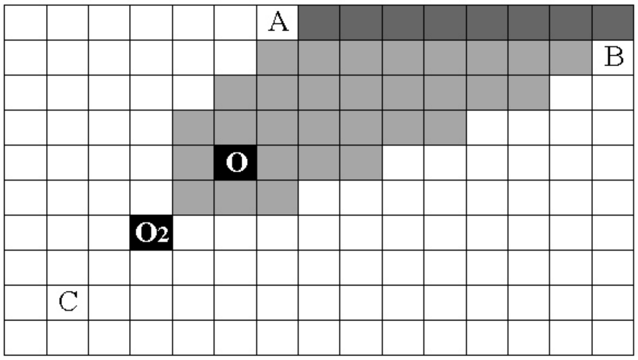

Figure 2. What will be subtracted from C's visibility cost given O2 is the sum of the difference between the current value and the memorized value at each occluded border cell.

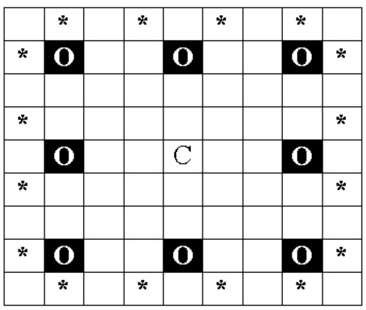

Figure 3. The stars represent the selected neighbors for each obstacle O, given the location of the affected free-space cell C.

After updating the visibility and the occupancy maps, the DT algorithm is used to find and update the covert path to the goal that depends on the information collected so far. The process of constructing DT array here is similar to the way described in section 2.1 where each non-obstacle cell is assigned a value of one greater than the least value of its neighbors, but with the difference here of adding to that the visibility cost of that cell [Marzouqi and Jarvis, 2003]. Therefore, each cell in the resulting array will contain a number that can be represented by (1):

$$\text{Cost} = \text{least 8-neighbors cost value} + \text{Distance cost} + \text{Visibility cost} \quad (1)$$

The final "visibility-distance cost" value at each cell can be described as the visibility of the cell adjusted by both the visibility and the distance costs to the goal's cell. The characteristics of the generated path are inherited from both the visibility and the distance costs, where there is a kind of a balance between finding a short path and a covert path to the goal. In [Marzouqi and Jarvis, 2003], a modification to the cost equation in (1) has been described that controls this balance by introducing coefficients 'a' and 'b' as in (2). The balance moves toward a short path when increasing 'a', and towards a covert path otherwise.

$$\text{Cost} = \text{least 8-neighbors cost value} + a \times \text{Distance cost} + b \times \text{Visibility cost} \quad (2)$$

In the last subtask, the robot moves one step in the created path before repeating the whole process. Allowing more than one step should decrease the overall computation time. On the other hand, this may affect negatively the covertness of the path.

3 The Time Complexity

The time complexity of this approach is mainly affected by updating the visibility map and the global path after each robot step. The computation time is proportional to the resolution of the environment map. The time complexity of continuously updating the visibility map in the previously described method is O(B x C), where B is the number of obstacle cells the robot has seen through its path, and C is the total number of the free-space cells. This method is much faster than the conventional way where the visibility cost at each cell is recalculated by counting the number of non-occluded cells at each direction.

Moreover, updating the visibility map and the global covert path after each robot's steps might not be necessary. Visibility map updating is skipped if no new obstacles have been seen. This leads as well to skip the process of updating the global path, as the same previous path will be generated.

An improvement has been added to the DT algorithm to increase its speed in our approach. As described in section 2.1, the DT algorithm finds the visibility-distance cost at all non-obstacle cells. Moreover, this cost is recalculated for some cells more than once before each cell gets the minimum possible value. The improvement restricts the calculations of the visibility-distance cost to some cells only that are expecting to be part of the final path.

The DT process is conducted using a flooding technique instead of the raster passes, where the cost is distributed starting from the goal location and towards the rest of the connected free-space. The pseudo code in Algorithm 2 represents the modified DT algorithm.

Describing the modification in details, a list is created that initially includes the goal location only. The first cell in the list is expanded by calculating the visibility-distance cost of its neighbor cells that did not get a cost value yet. The new calculated cells are added to the list by sorting them using the insertion sort algorithm. This algorithm should keep the added cells sorted according to their cost value from low to high. This is an important step as the first element in the list should have the lowest visibility-distance cost. Once the robot's location gets its cost value the process is ended and no more cells need to be calculated. At this point finding a path to the goal is guaranteed by moving steepest descent. The modified algorithm generates the exact path with faster processing speed.

-Initiate DT map with all cells value =-1
-Give the goal location value of 0
-Create a list
-Add the goal location to the list
-While (Robot location = = -1)
{
-get the first element in the list
-expand the first element cost to the neighbor cells with cost=-1 (neighbor cost= element cost + distance cost + the neighbor visibility cost)
-add the expanded cells to the list using insertion sort alg.
}

Algorithm 2. A fast DT algorithm for covert path planning

Fig. 3 shows an example. The robot start location is represented by 'R', the goal is represented by 'G', and the obstacles are in black. Given an example from [Marzouqi and Jarvis, 2003] where the environment is initially known, the visibility cost of each free-space cell is calculated which is its exposure to the entire environment. The grey area represents the visibility costs where a cell gets darker as it becomes more hidden and vice versa.

The modified DT algorithm has been applied to find the global covert path. The dotted areas represent the only cells that got a visibility-distance cost. It is clear how the cost was expanded from the goal and mostly through the low visibility route until reaching the robot. A covert path has been generated accordingly. The worst case scenario in this method is when the robot's location is that last cell to get its cost value, however, this still faster than the conventional DT algorithm as each cell will be assigned only once.

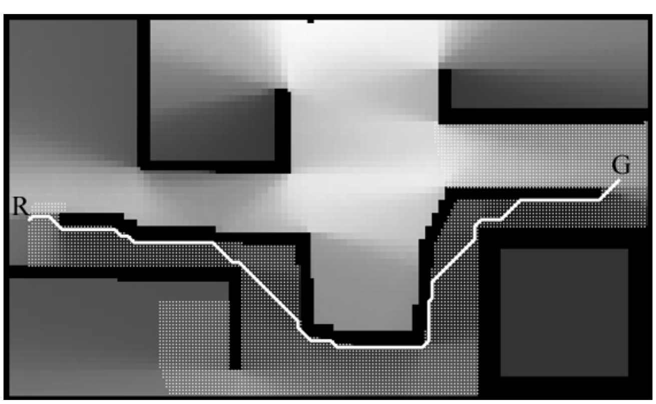

Figure 4. The visibility-distance costs are calculated for only 25% of the environment in this case, represented by the dotted areas.

Our covert path planning approach has been simulated and tested on a computer with processing speed of 1.5 GHz Mobile Pentium, and environment map resolution of 200x120. The recorded robot average motion speed was 18 cells per a second (given the example cases in the section 4). The processing time should change inversely with the map resolution, but proportionally with the processor speed which has reached 4 GHz nowadays.

4 Experiment

A simulator has been built to test the approach. The simulator inputs are the start location of the robot, its destination, and the environment's dimensions. The following figures, Fig. 5 to 8, show a number of test cases. The letters 'R' and 'G' represent the start and the destination locations, respectively. The obstacles are represented by the white dotted areas.

For comparison, the robot has followed two different paths to the goal. In the dashed path, the robot only concern about finding a short path without taking covertness into account (the coefficient 'b' in equation (2) is set to 0). The continuous path is the covert path (the coefficients are set so that 'b'>'a'). To evaluate the covertness of the generated paths, the visibility map of the environment is represented in different grey levels.

In each example, the average exposure of each path to the entire environment has been calculated and presented in Table 1. It is noticeable how the robot is making advantage of the obstacles it sees to reduce its exposure while navigating towards the goal. As one or more sentries might be at any locations, the robot has a higher chance to be less observable to those sentries in the covert path than in the short path.

Fig. 8 shows a situation where the robot is trying to find the best possible covert path. In this example, the robot has changed its covert path direction after it had discovered more about the environment at the point indicated by the arrow. The robot followed a more covert path accordingly.

Figure No.	Short path average exposure (%)	Covert path average exposure (%)
5	24	11
6	28	17
7	25	15
8	24	6

Table 1. The short and covert paths average exposure

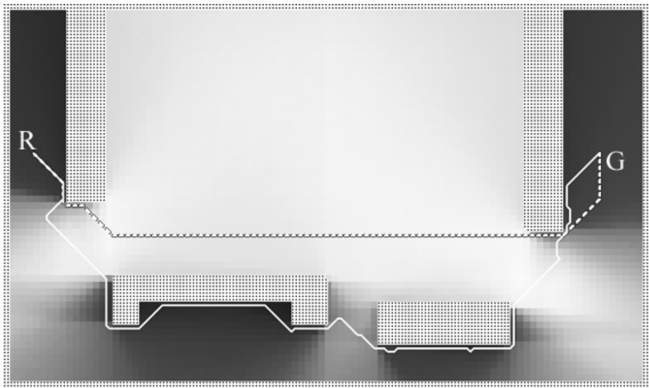

Figure 5. The average exposure of the short (dashed) path to the environment is 24%, while 11% in the covert (continuous) path.

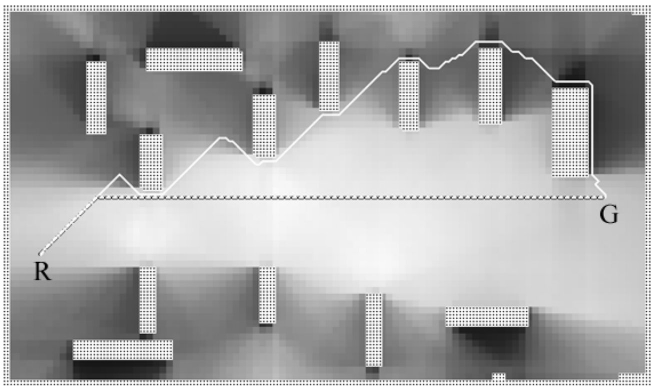

Figure 6. The average exposure of the short path to the environment is 28%, while only 17% in the covert path.

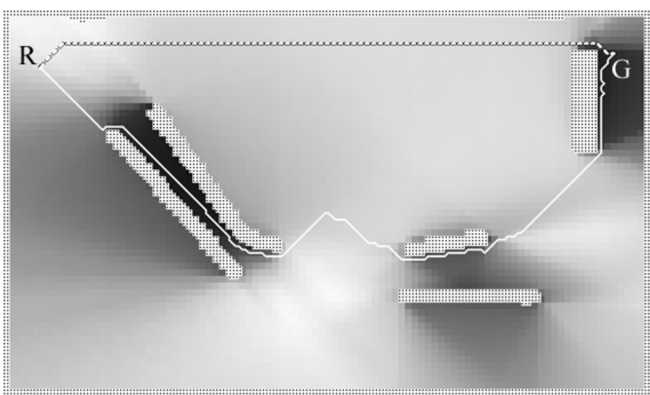

Figure 7. The average exposure of the short path to the environment is 25%, while only 15% in the covert path.

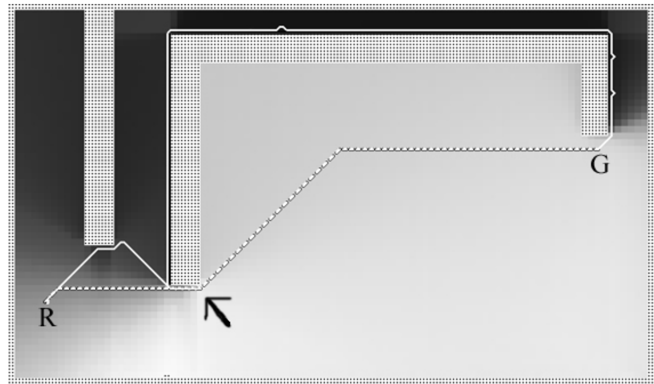

Figure 8. The average exposure of the short path to the environment is 24%, while only 6% in the covert path.

5 Conclusion and Future Work

In this paper, a covert path planning approach has been presented for the case of an initially unknown environment and unknown sentries' locations. The approach has shown its success with unknown environments as well as with known environments which was presented in [Marzouqi and Jarvis, 2003]. The conducted experiment in the previous section has shown a reasonable results and acceptable time complexity for a practical use.

This approach can be extended further to deal with the situation when some sentries' locations might be known or discovered during the path. In this case, a special covertness from those sentries and a general covertness from the unknown sentries should be taken into account at the same time.

Moreover, the approach can be modified easily to plan an overt path instead of a covert path. There are some applications where the robot needs to plan a path with maximum exposure to the environment for searching or exploring purposes. An investigation should be done on this side.

References

[Barraquand et al., 1989] J. Barraquand, B. Langlois, & J.C. Latombe. Numerical Potential Field techniques for Robot Path Planning, Report No. STAN-CS-89-1285, Department of Computer Science, Stanford University. 1989.

[Birgesson et al., 2003] E. Birgesson, A. Howard and G. Sukhatme. Towards Stealthy Behaviors, *IEEE/RSJ International Conference on Intelligent Robots and Systems* (IROS), Oct 2003, Las Vegas, pp 1703-1708.

[Jarvis, 1984] R. A. Jarvis, Collision-free path trajectory using distance transforms, Proc. *National Conference and Exhibition on Robotics*, Melbourne, August 1984.

[Marzouqi and Jarvis, 2003] Mohamed Marzouqi and Ray A. Jarvis, Covert path planning for autonomous robot navigation in known environments, Proc. *Australasian Conference on Robotics and Automation*, Brisbane 2003.

[Marzouqi and Jarvis, 2004] Mohamed Marzouqi and Ray A. Jarvis, Covert Robotics: Hiding in Known Environments, To appear in Proc. 2004 *IEEE Conference on Robotics, Automation and Mechatronics* (RAM), Singapore, December 2004.

[Ravela et al., 1994] S. Ravela, R. Weiss, B. Draper, B. Pinette, A. Hanson and E. Riseman. Stealth Navigation: Planning and Behaviors, *ARPA Image Understanding Workshop*, Monterey, CA, 1994, pp. 1093-1100.

[Teng et al., 1993] Y. Teng, D. DeMenthon and L. Davis. Stealth Terrain Navigation, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 1, 1993, pp.96-110.