Path Clearance

Maxim Likhachev
Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104
maximl@seas.upenn.edu

Anthony Stentz
The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
axs@rec.ri.cmu.edu

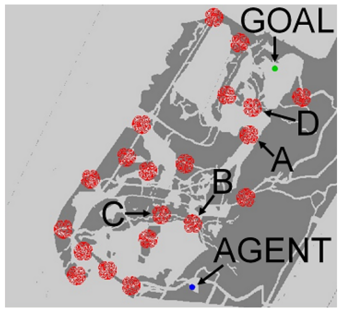
In military scenarios, agents (i.e., troops of soldiers, convoys, and unmanned vehicles) may often have to traverse environments with only a limited intelligence about the locations of adversaries. We study a particular instance of this problem that we refer to as *Path Clearance* problem. In Path Clearance, an agent has to navigate to its goal as quickly as possible without being detected by an adversary. When picking a path to follow, the agent does not know the precise locations of adversaries. Instead, it has a list of their possible locations, each associated with the probability of containing an adversary. Any of these locations can be sensed by either the agent itself at close enough distance (provided the agent has a capability of long-range sensing) or by one of the scouts (if they are available). If no adversary is present at a sensed location, the agent can then safely traverse through it. Otherwise, the agent has to take a detour.

The challenge in solving the Path Clearance problem is to figure out what path should the agent pick, when should the agent sense for adversaries and, finally, what adversaries should scouts sense in order to minimize the overall cost such as time and risk before the agent reaches its goal. This translates into a well-defined but challenging planning with incomplete information problem.

The example in figure 1 demonstrates the path clearance problem. In this example, there are no scouts. Figure 1(b) shows the traversability map of the satellite image of a 3.5km by 3km area shown in figure 1(a). The traversability map is obtained by converting the image into a discretized 2D map where each cell is of size 5 by 5 meters and can either be traversable (shown in light grey color) or not (shown in dark grey color). The large circles (e.g., A, B, C, D, and others) are *possible* adversary locations and their radii represent the sensor range of adversaries (100 meters in this example). The radii can vary from one location to another. The locations can be specified either manually or automatically in places such as narrow passages. Each location also comes with a probability of containing an adversary (50% for each location in this example): the likelihood that the location contains an adversary. The probabilities can vary from one location to another.

The path the agent follows may change any time the agent senses a possible adversary location (the sensor range of the agent is 105 meters in our example). A planner, therefore, needs to reason about possible outcomes of sensing and generate a plan (policy) that dictates which path the agent should take as a function of the outcome of each sensing. For the agent to act efficiently (on average), the generated policy should minimize the *expected* cost such as traversal distance. Unfortunately, such planning problem falls into the category of planning with incomplete information about the environment and with sensing and more generally falls into a broader category of planning for Partially Observable Markov Decision Processes (POMDPs) [4]. Planning optimally for POMDPs, in general, and planning with incomplete information and with sensing, in particular, is known to be intractable [16], [2]. In addition, the size of a typical environment is several kilometers wide while its traversability is highly non-uniform making the size of the problem large and cost function complex. Finally, the Path Clearance problem becomes even more challenging to solve when there are multiple scouts available. Planning in this case involves both large-scale planning under uncertainty as well as coordination of multiple scouts.

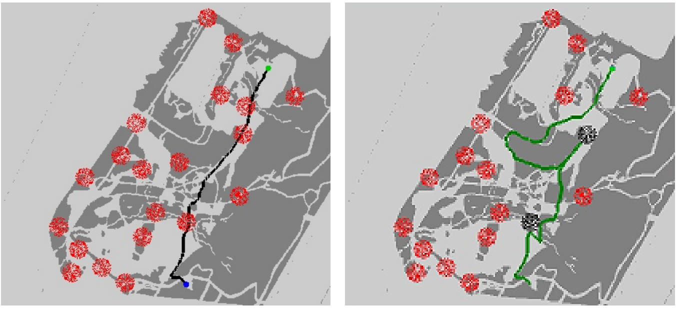
(a) 3.5 by 3.0 km satellite image



(b) traversability map

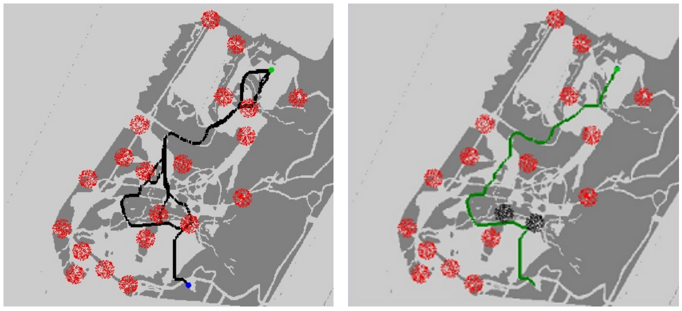
Fig. 1. Path Clearance without scouts

This article presents a survey of our work on scalable and suitable for real-time use approaches to solving the Path Clearance problem. In particular, in the first part of the article, we show that the Path Clearance problem exhibits clear preferences on uncertainty. It turns out that these clear preferences can be used to develop an efficient algorithm, called PPCP (Probabilistic Planning with Clear Preferences) [14]. The algorithm is anytime, converges to an optimal solution under certain conditions and scales well to large-scale problems. We briefly describe the PPCP algorithm and show how it can be used to solve the Path Clearance problem when no scouts are present [12]. In the second part of the article, we show several strategies for how to use the PPCP algorithm in case multiple scouting UAVs are available [10], [13]. The experimental analysis shows that planning with PPCP results in a substantially



(a) planned path (b) actual path of the agent

Fig. 2. Solving the Path Clearance problem with freespace assumption



(a) generated plan (b) actual path of the agent

Fig. 3. Solving the Path Clearance problem with PPCP

smaller execution cost than when ignoring uncertainty, and employing scouts can decrease this execution cost even further.

Related Research

The Path Clearance problem is closely related to the problem of planning for a robot navigating in a partially-known (or unknown) environment: the robot needs to reach its goal but it is initially uncertain about the traversability of some (or all) of the areas of the environment. The difference is that in the path clearance problem, detecting an adversary blocks a large area resulting in a long detour. An adversary location has also a tendency to be placed in such places that it blocks the whole path and the agent has to backup and choose a totally different route. As a result, the detours can be much costlier than in the case of navigation in a partially-known environment, even when the amount of uncertainty is much less. Finally, there may also be penalty for discovering an adversary by the agent, because it involves approaching the adversary and therefore increases the risk of being discovered itself. Nevertheless, approaches to planning for a robot navigating a partially-known environment are also applicable to planning for the Path Clearance problem.

Assumptive planning. To avoid the computational complexity, a robot operating in a partially-known environment often performs assumptive planning [15], [9], [22]. In particular, it often just follows a shortest path under the assumption that all unknown areas in the environment are free unless the robot has already sensed them otherwise. This is known as a freespace assumption [9]. The robot follows such path until it either reaches its goal or senses new information about the environment. In the latter case, the robot re-computes and starts following a new shortest path under the freespace assumption.

The freespace assumption is also applicable to the Path Clearance problem when no scouts are present. The agent can always plan a path under the assumption that no adversary is present unless sensed otherwise. This makes the Path Clearance problem a deterministic planning problem. It can therefore be solved efficiently. The fact that the agent ignores the uncertainty about the adversaries, however, means that it risks having to take long detours, and the detours in the Path Clearance problem tend to be longer than in the problem of navigation in a partially-known environment as we have just explained.

For example, figure 2(a) shows the path computed by the agent that uses the freespace assumption. According to the path, the agent tries to go through the possible adversary location A (shown in figure 1(b)) as it is on the shortest route to the goal. As the agent senses the location A, however, it discovers that the adversary is present in there (the circle becomes black after sensing). As a result, the agent has to take a very long detour. Figure 2(b) shows the actual path traversed by the agent before it reaches its goal.

In order to avoid these situations, one may also try to set a cost function that penalizes the traversal of possible adversary locations. While it typically reduces the number of times the path requires the agent to try to traverse through a possible adversary location, it still does not avoid the situations with long detours, and moreover, may generate paths with long detours that are not even necessary.

To deal with this problem properly, the planner needs to find a plan that minimizes the expected cost. Thus, figure 3(a) shows the plan returned by PPCP after it converged in about 30 seconds for our example. Every place where the plan branches out corresponds to where the agent senses a possible adversary location and chooses to go through it if no adversary is detected, or take a detour otherwise. In contrast to planning with freespace assumption, the plan produced by PPCP makes the agent go through the area on its left since there are a number of ways to get to the goal there and therefore there is a high chance that one of them will be available. The length of the actual path traversed by the agent (figure 3(b)) is 4,123 meters while the length of the path traversed by the agent that makes the freespace assumption (figure 2(b)) is 4,922 meters.

Planning with Incomplete Information and Sensing. Both planning for Path Clearance and planning for a robot navigating a partially-known environment are instances of planning with incomplete information about the environment and with sensing and fall into a broader category of planning for Partially Observable Markov Decision Processes (POMDPs) [4]. Planning optimally for POMDPs, in general, and planning with incomplete information and with sensing, in particular, is known to be intractable [16], [2]. Various approximations techniques have been recently proposed instead [5], [6], [1], [8], [18], [19],

[17], [21], [3], [24], [20].

The problem of planning for Path Clearance (as well as navigation in partially-known environments) is a narrower one than solving a general POMDP. For one, it assumes that the only uncertainty is the uncertainty about the actual location of adversaries. There is no uncertainty in the actions of the agent. It also assumes that sensing is perfect. We can therefore develop planning algorithms that take advantage of these special properties.

Perhaps, the most relevant approach to planning with incomplete information and sensing is the algorithm in [7], developed specifically for the problem of robot navigation in a partially-known terrain. Similarly to our definition of clear preferences, their planner has taken advantage of the idea that the cost of the plan if a cell is free can not be larger than the cost of the plan if the cell is occupied. Based on this idea, they proposed a clever planner that is capable of finding optimal policies much faster than other optimal approaches. It is not clear, however, how this approach can be generalized to the Path Clearance problem. Most importantly, the approach to solving the Path Clearance problem we present in this article avoids dealing with the exponentially large belief state-spaces altogether. This allows us to solve very efficiently and without running out of memory large environments with a large number of adversaries.

Path Clearance without Scouts

PPCP Algorithm

While in general, decision-theoretic planning that takes into account the uncertainty about the environment is very hard to solve, it turns out that many such problems exhibit a special property: one can clearly identify beforehand the best (called *clearly preferred*) values for the variables that represent the unknowns in the environment. For example, in the problem of navigation in partially-known environments, it is always preferred to find out that an initially unknown location is traversable rather than not. In a very similar problem of robot navigation in office-like environments with uncertainty about whether some doors are open or not, it is always preferred to find out that a door is open. The same property holds for the path clearance problem: there are also clear preferences for the values of unknowns. The unknowns are m binary variables, one for each of the m possible adversary locations. The preference for each of these variables is to have a value false: no adversary is present.

Mathematically, clear preferences can be defined as follows. A clearly preferred value b ("best") of an unknown variable u is such a value that for any belief state X - a state that includes the current state of the agent as well as its current probability distribution over the values of unknowns variables - and action a that senses (directly or indirectly) the value of some unknown variable u, there exists a successor belief state X' at which the value of u is known to be a clearly preferred value (i.e., $u = b$) and:

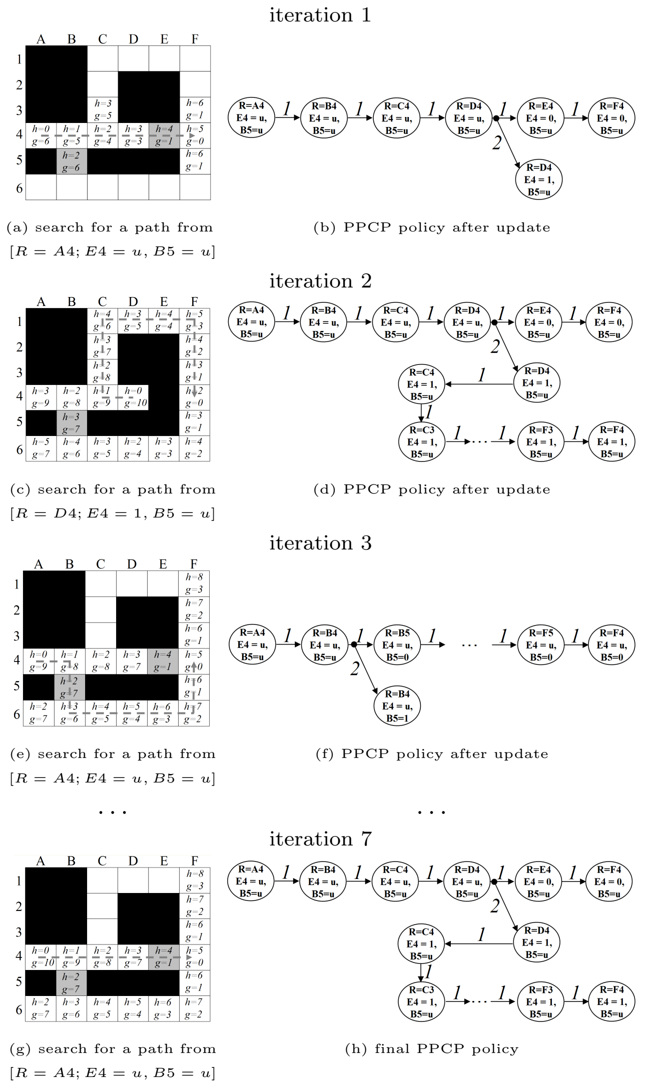
$$X' = \text{argmin}_{Y \in succ(X,a)} c(X, a, Y) + v^*(Y), \qquad (1)$$

$c(X, a, Y)$ is the cost of executing action a at state X and ending up at state Y, and $v^*(Y)$ is the expected cost of executing an optimal policy at the belief state Y. In the Path Clearance problem, a belief state is composed of an x, y position of the agent and m probability values, each representing the probability of a possible adversary location containing an adversary. Since sensing is assumed to be perfect, each of these probabilities can either be an initial probability of containing an adversary, or 0 (if sensing of the location indicated no adversary), or 1 (if sensing of the location indicated that an adversary was present). Every time an agent senses some location u, a clearly preferred value is for the location not to contain an adversary. The corresponding successor belief state will have $P(u = b) = 1$ and will satisfy equation 1.

PPCP (Probabilistic Planning with Clear Preferences) [11] is a recently developed algorithm that scales to large problems with a significant amount of uncertainty by exploiting a prior knowledge of clear preferences. It constructs and refines until convergence a policy by running a series of A*-like deterministic searches. By making a certain approximating assumption about the problem, PPCP keeps the complexity of each search low and independent of the amount of the missing information. Each search is extremely fast, and running a series of fast low-dimensional searches turns out to be much faster than solving the full problem at once since the memory requirements are much lower. While the assumption the algorithm makes does not need to hold for the found policy to be valid, it is guaranteed to be optimal if the assumption holds. In the problem of robot navigation in a partially-known environment, PPCP was also shown to nearly always return an optimal policy in the environments small enough to be solved with methods guaranteed to converge to an optimal solution [11].

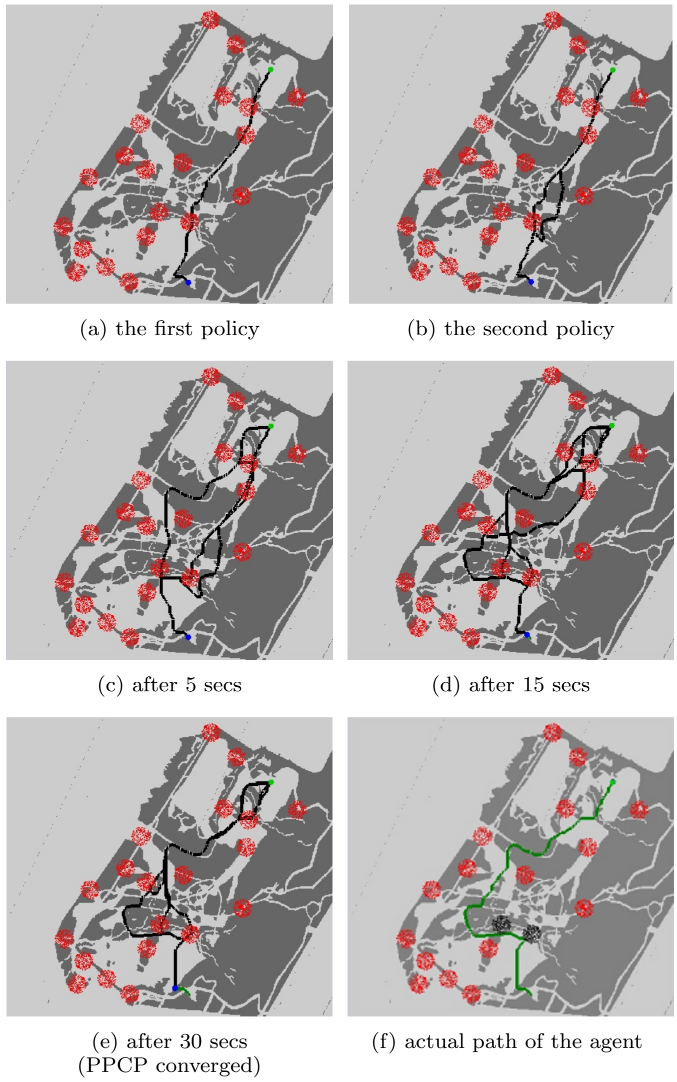
Figure 4 shows a simple example of PPCP planning a policy for a robot navigating in a partially-known environment. Initially, the robot is in cell A4 and its goal is cell F4. The status of cells B5 and E4 (shaded in grey) is initially unknown to the robot. For each of these cells though, the probability of containing an obstacle is 0.5. In this example, we restrict the robot to move only in four compass directions. Whenever the robot attempts to enter an unknown cell, we assume that the robot moves towards the cell, senses it and enters it if it is free and returns back otherwise. The cost of each move is 1, the cost of moving towards an unknown cell, sensing it and then returning back is 2. The goal of the planner is to construct a policy that makes the robot reach the cell $R = F4$ with a minimal expected cost. Figure 4(h) shows the policy generated by PPCP. It specifies the path the robot should follow after each outcome of sensing operation.

During each iteration PPCP assumes some configuration of unknowns (unknown cells in this example) and performs search in the corresponding deterministic graph. Thus, the first search in figure 4 assumes that both unknown cells are free and finds a path that goes straight to the goal (figure 4(a)). (The shown g- and h-values are equivalent to



(a) search for a path from $[R = A4; E4 = u, B5 = u]$
(b) PPCP policy after update
(c) search for a path from $[R = D4; E4 = 1, B5 = u]$
(d) PPCP policy after update
(e) search for a path from $[R = A4; E4 = u, B5 = u]$
(f) PPCP policy after update
(g) search for a path from $[R = A4; E4 = u, B5 = u]$
(h) final PPCP policy

Fig. 4. Example of how PPCP operates



(a) the first policy
(b) the second policy
(c) after 5 secs
(d) after 15 secs
(e) after 30 secs (PPCP converged)
(f) actual path of the agent

Fig. 5. Applying PPCP to Path Clearance

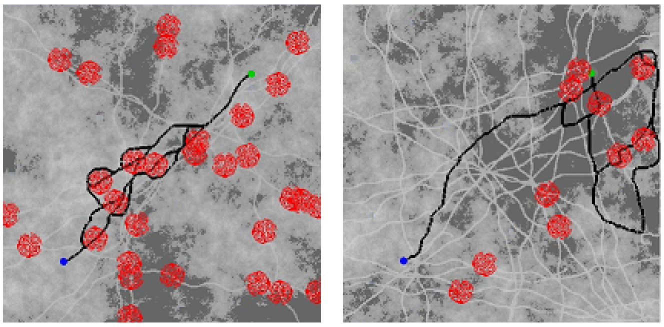
the g- and h-values maintained by the A* search.) PPCP takes this path and uses it as an initial policy for the robot (figure 4(b)). One of the actions on this policy, however, is move east from cell D4. The current policy has only computed a path from the preferred outcome state, the one that corresponds to cell D4 being free. The state $[R = D4; E4 = 1, B5 = u]$, on the other hand, has not been explored yet. The second search executed by PPCP, shown in figure 4(c), explores this state by finding a path from it to the goal. During this search cell E4 is assumed to be blocked, same as in the state $[R = D4; E4 = 1, B5 = u]$. The found path is incorporated into the policy maintained by PPCP (figure 4(d)).

In the third iteration, PPCP tries to find a path from the start state to the goal again (figure 4(e)). Now, however, it no longer generates the same path as initially (figure 4(a)). The reason for this is that it has learned that the cost of trying to traverse cell E4 is higher than what it initially thought to be. The cost of the cheapest detour in case cell E4 is blocked (found in figure 4(c)) is rather high. Consequently, in the current iteration PPCP finds another alternative policy that goes through cell B5. This now becomes the new policy in PPCP (figure 4(f)). This policy, however, has an unexplored outcome state again, namely, state $[R = B4; E4 = u, B5 = 1]$. This will become the state to explore in the next iteration.

PPCP continues to iterate in this manner and, on the 7th iteration, converges to the final policy shown in figure 4(h). In this example it is optimal: it minimizes the expected cost of reaching the goal. In general, PPCP is guaranteed to converge to an optimal policy if it does not require remembering the status of any cell the robot has successfully entered (see [14] for more details).

Application of PPCP to Path Clearance

Figure 5 shows the application of PPCP to the path clearance example in figure 1. Before the agent starts executing any policy, PPCP plans for five seconds. Figure 5(a) shows the very first policy produced by PPCP (in black color). It is a single path to the goal, which in fact is exactly the same as the path planned by planning with the freespace assumption (figure 2(a)). PPCP produced this path within few milliseconds by executing a single A*-like deterministic search. At the next step, PPCP refines the



(a) Typical group I environment (b) Typical group II environment

Fig. 6. The example of environments used in testing and the plans generated by PPCP for each.

policy by executing a new search which determines the cost of the detour the agent has to take if the first adversary location on the found path contains an adversary. The result is the new policy (figure 5(b)). PPCP continues in this manner and at the end of five seconds allocated for planning, it generates the policy shown in figure 5(c). This is the policy that is passed to the agent for execution. Each fork in the policy is where the agent tries to sense an adversary and chooses the corresponding branch.

Planning is interleaved with execution. Thus, while the agent executes the plan, PPCP improves it relative to the current position of the agent. Figure 5(d) shows the new position of the agent (the agent travels at the speed of 1 meter per second) and the current policy generated by PPCP after 15 seconds since the agent was given its goal. Figure 5(e) shows the position of the agent and the policy PPCP has generated after 30 seconds. At this point, PPCP has converged and no more refinement is necessary. Note how the generated policy makes the agent go through the area on its left since there are a number of ways to get to the goal and therefore there is a high chance that one of them will be available. Unlike the plan generated by planning under freespace assumption, the plan generated by PPCP avoids going through location A. Figure 5(f) shows the actual path traversed by the agent. It is 4,123 meters long while the length of the trajectory traversed by the agent that plans with freespace assumption (figure 2(b)) is 4,922 meters.

Experimental Study

In [14], we have compared planning with PPCP against several optimal approaches to planning in belief state-spaces. These experiments showed that, at least for the problem of navigation in a partially-known environment, PPCP returns optimal policies but does it orders of magnitude faster than the alternative approaches. The experiments have also shown that, in contrast to PPCP, the alternative approaches do not scale to real-size environments. The experiments in this section consider large-scale environments with large number of possible adversaries. As alternative to planning with PPCP, we therefore use planning with freespace assumption [9] which does scale to large environments. In particular, in our experiments we compared the cost of execution incurred by the agent planning with PPCP with the cost of execution incurred by the agent planning with freespace assumption.

We used randomly generated fractal environments that are often used to model outdoor environments [23]. On top of these fractal environments we superimposed a number of randomly generated paths in between randomly generated pairs of points. The paths were meant to simulate roads through forests and valleys and that are usually present in outdoor terrains. Figures 6(a,b) show typical environments that were used in our experiments. The lighter colors represent more easily traversable areas. All environments were of size 500 by 500 cells, with the size of each cell being 5 by 5 meters.

The test environments were split into two groups. Each group contained 25 environments. For each environment in the group I we set up 30 possible adversary locations at randomly chosen coordinates but in the areas that were traversable. Figure 6(a) shows a plan the PPCP algorithm has generated after full convergence for one of the environments in group I. For each environment in the group II we set up 10 possible adversary locations. The coordinates of these locations, however, were chosen such as to maximize the length of detours. This was meant to simulate the fact that an adversary may often be set at a point that would make the agent take a long detour. In other words, an adversary is often set at a place that the agent is likely to traverse. Thus, the environments in group II are more challenging. Figure 6(b) shows a typical environment from the group II together with the plan generated by PPCP. The shown plan has about 95% probability of reaching the goal (in other words, the agent executing the policy has at most 5% chance of encountering an outcome for which the plan had not been generated yet). In contrast to the plan in figure 6(a), the plan for the environment in group II is more complex - the detours are much longer - and it is therefore harder to compute. For each possible adversary location the probability of containing an adversary was set at random to a value in between 0.1 and 0.9.

In all of the experiments, the agent was moving and was given 5 seconds to plan while traversing 5 meter distance. This amount of time was always sufficient for planning with freespace assumption to generate a path. The PPCP planning, on the other hand, was interleaved with execution as was shown in figure 5. Thus, neither of the approaches required the agent to stop and wait for a plan to be generated.

Table I shows the overhead in the execution cost incurred by the agent that planned with the freespace assumption over the execution cost incurred by the agent that used PPCP for planning. The rows freespace2 and freespace3 correspond to making the cost of going through a cell that belongs to a possible adversary location twice and three times higher than what it really is, respectively. One may scale costs in this way in order to bias the paths generated by the planner with freespace assumption away from going through possible adversary locations. The results are averaged over 8 runs for each of the 25 environments in each group. For each run the true status of each adversary loca-

	Overhead in Execution Cost			
	Group I no penalty	Group II no penalty	Group I with penalty	Group II with penalty
freespace	5.4%	5.2%	35.4%	21.6%
freespace2	0.5%	4.9%	4.8%	17.0%
freespace3	2.1%	4.3%	0.0%	12.7%

TABLE I

The overhead in execution cost of navigating using planning with freespace assumption over navigating using planning with PPCP

tion was generated at random according to the probability having an adversary in there.

The figure shows that planning with PPCP results in considerable execution cost savings. The savings for group I environments were small only if biasing the freespace planner was set to 2. The problem, however, is that the biasing factor is dependent on the actual environment, the way the adversaries are set up and the sensor range of an adversary. Thus, the overhead of planning with freespace for the group II environments is considerable across all bias factors. In the last two columns we have introduced penalty for discovering an adversary. It simulated the fact that the agent runs the risk of being detected by an adversary when it tries to sense it. In these experiments, the overhead of planning with freespace assumption becomes very large. Also, note that the best bias factor for freespace assumption has now shifted to 3 indicating that it does depend on the actual problem. Overall, the results indicate that planning with PPCP can have significant benefits and do not require any tuning.

Path Clearance with Scouts

We now present two strategies for employing scouts, when available, in order to reduce the cost (e.g., time) it takes for the agent to reach its goal. An optimal coordination of scouts would involve running PPCP in a joint (agent and all scouts) state-space. The dimensionality of this statespace, however, is too high - exponential in the number of scouts - for keeping planning tractable. Instead we propose two strategies for coordinating scouts, both based on the idea of first running PPCP for the agent, and then using the policy generated by PPCP to schedule scouts. While these strategies are heuristics, they are simple, efficient, scalable to large number of scouts and can be applied to heterogenous scouts. Experimental results prove the benefits of these strategies. The first strategy is simpler but not as effective as the second one. Both strategies are equally efficient though.

Likelihood-Driven Use of Scouts

Because PPCP produces a policy for the agent, we can evaluate the likelihood of any possible adversary location being visited by the agent. The idea behind likelihood-driven use of scouts is have scouts navigate to and sense the locations that are most likely to be visited by the agent.

To be specific, suppose there are K scouts available. Our approach is to first run PPCP to produce a policy for the agent, as if there are no scouts available. That is, only agent itself can sense for adversaries. Once we obtain the policy for the agent, we find K possible adversary locations that have the highest probability of being visited by one or more paths on the policy. In other words, these are the locations that PPCP assumes the agent will sense on one of its branches in the policy. To select the ones that have the highest probabilities of being visited by the agent, we can perform a single pass over all the states in the policy in topological order starting with the state of the agent. During this pass we can propagate the probability of the agent visiting the state when following the policy according to the probability distribution of the policy action outcomes.

Once we compute these K possible adversary locations, we assign them to the nearest scouts. Each scout starts traveling towards its assigned adversary location and performs sensing when it reaches the location. While each scout travels, the agent executes its policy. PPCP is also being executed to improve the policy as we have described previously. Every time the agent changes its policy onto the policy generated by PPCP, it re-computes the K possible adversary locations that scouts need to sense and re-assigns them to the scouts. Also, every time one of the scouts performs sensing, the agent updates its knowledge about the adversaries, so that all the subsequent planning done by PPCP can take this information into account. The agent then re-computes the K possible adversary locations that the scouts should sense.

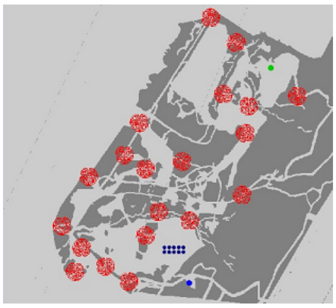
Figure 7 shows the operation of the algorithm. There are ten scouts available, shown by the smaller dots in a two-row formation in figure 7(a). In this example, the scouts are assumed to be aerial vehicles moving with the same speed as the agent. The PPCP planner starts planning a plan for the agent and after 30 seconds converges to the final plan shown in figure 7(b). While the agent follows the plan, the scouts are assigned to the possible adversary locations that are most-likely to be visited by the agent. As the figures show, at any point in time at most four scouts are used because this is the maximum number of adversary locations involved in the policy generated by PPCP.

The first locations that are being sensed are locations B and C (figures 7(c,d)). In both of these locations, adversaries were detected (the locations turned black). Once this information is passed to the agent, PPCP re-computes a plan. The new plan, as shown in figure 7(d), no longer directs the agent towards the locations B and C since they are already known to contain adversaries.

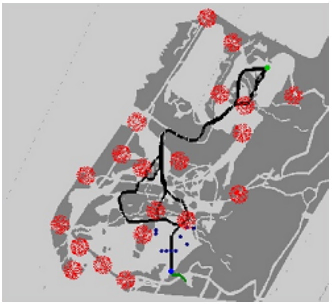
At the same time, one of the scouts flies towards the location D and senses it. The location turns out to be free of adversaries (figure 7(e)). Figure 7(f) shows the final trajectory of the agent. The total distance traversed by the agent is 3795 meters which is 328 meters shorter than in case of no scouts (figure 5(f)).

Information Value-Driven Use of Scouts

The likelihood-driven use of scouts is simple, very fast, and scales well to heterogeneous teams of scouts (e.g., a mix of aerial and ground robots). It can, however, be very greedy. In the example in figure 7 for instance, it was clearly worthwhile to send a scout to sense the location A

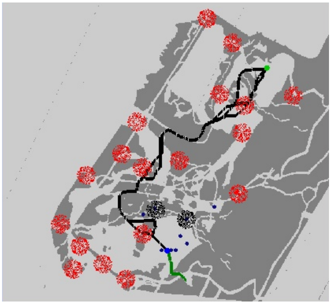


(a) initial configuration

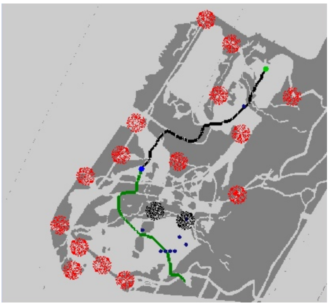


(b) plan after convergence (30 secs)

(c) location B sensed (adversary detected)



(d) location C sensed (adversary detected)



(e) location D sensed (no adversary present)

(f) agent reaches its goal

Fig. 7. Path Clearance with PPCP planning and likelihood-driven use of ten scouts

(the labels are shown in figure 1(b)). If it turned out to be empty, then the path to the goal via this location would have been the shortest possible route for the agent. The approach we present in this section - information value-driven use of scouts - is aimed at decreasing this greediness.

In brief, the approach can be summarized as the strategy of sending the scouts to those possible adversary locations that maximize the value of information which is defined as the expected decrease in the cost incurred by the agent before it reaches the goal less the cost of sensing. In other words, the scouts are sent to sense those locations, the knowledge about which would decrease the overall execution cost as much as possible. We don't have the exact values of the cost decreases - these would be very expensive to compute. Instead, we use estimates for these values computed as a by-product of running PPCP when planning for the agent.

To be specific, in this strategy, the next possible adversary location s to sense should be chosen such as to minimize the expected cost of executing an optimal plan for the agent given that the status of s is known, plus the expected cost of sensing s. Mathematically, it can be expressed as:

$$s = \text{argmin}_{s'} E\{c(\pi^*(s' \text{ is known})) + costofsensing(s')\}$$

In this equation, $\pi^*(s' \text{ is known})$ stands for the optimal plan for the agent that takes into account the fact that the status of s' is known, $c(\pi^*(s' \text{ is known}))$ is the cost of executing this plan. The expectation is taken over all possible configurations of possible adversary locations including s'. The equation can also be re-written as:

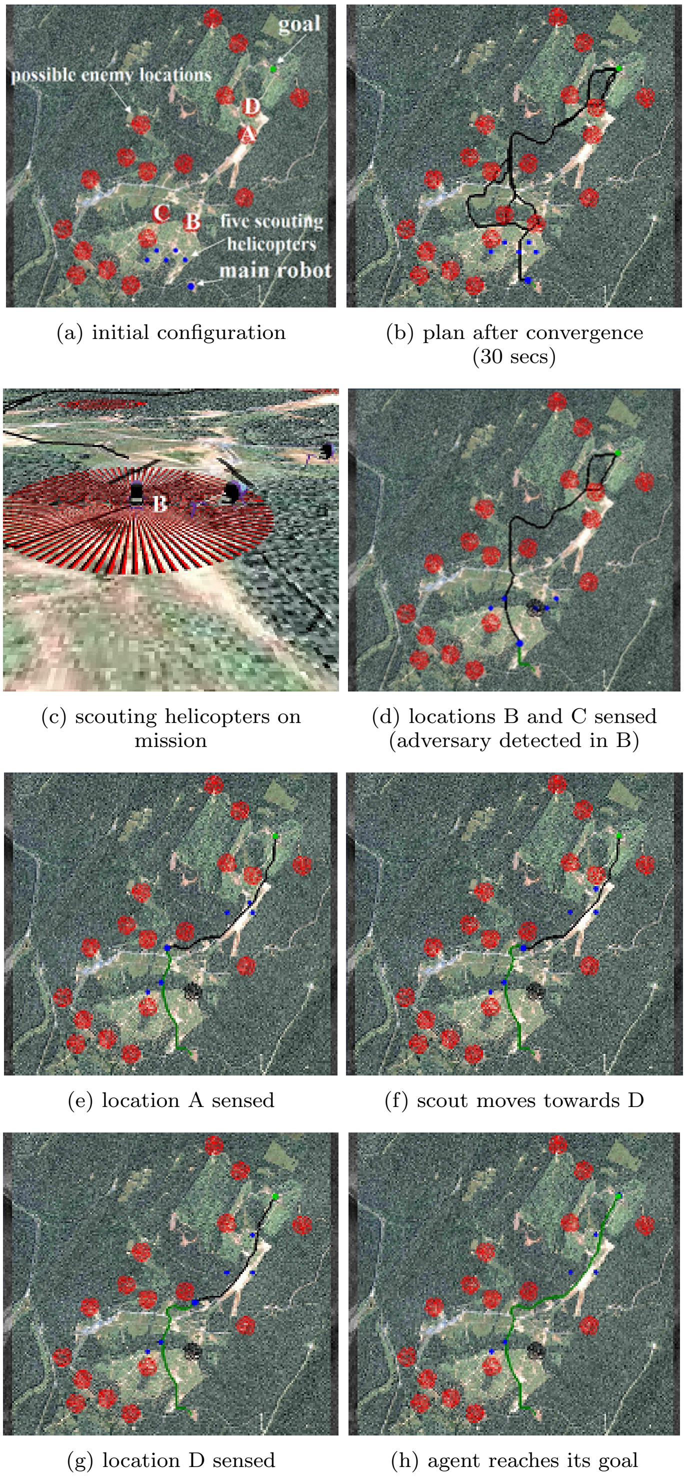
$$s = \text{argmax}_{s'} E\{c(\pi^*) - c(\pi^*(s' \text{ is known})) - costofsensing(s')\} \quad (2)$$

In this equation, π^* stands for the optimal plan for the agent that assumes that the status of all possible adversary locations including s' is unknown. In general, the expected values of the quantities $c(\pi^*)$ and $c(\pi^*(s' \text{ is known}))$ are hard to compute since they require finding optimal policies. However, as explained previously, PPCP works by initially considering an optimistic plan (all possible adversary locations are free) and then using more and more informative estimates on the policies. We can use this property of PPCP to estimate the quantity $E\{c(\pi^*) - c(\pi^*(s' \text{ is known}))\}$ (details are in [13]).

The term $E\{costofsensing(s')\}$ can be computed as the minimum expected cost of sensing s' across all available scouts. After we compute an estimate of $E\{c(\pi^*) - c(\pi^*(s' \text{ is known})) - costofsensing(s')\}$ for each s', we pick s' that maximizes it and assign it to the scout which minimized the term $E\{costofsensing(s')\}$. In the experiments, the scouts were helicopters and therefore the term $E\{costofsensing(s')\}$ was computed as the time it takes for a scout to reach the center of the location s' which was proportional to the Euclidean distance between the two.

Figure 8 shows the operation of information value-driven use of scouts. Figure 8(a) shows the initial configuration. It is the same environment with the same set of possible adversary locations as in figure 7 with the only difference that now there are five scouting helicopters (shown as small dots, initially in a two-row formation). They are assigned to the possible adversary locations selected according to the information value-driven approach. Figure 8(c) shows how some of the helicopters fly towards adversary locations to sense them. Figure 8(d) shows that the location C turns out to be free of adversaries, whereas a scout did detect an adversary in the location B. The re-computed plan, also shown in figure 8(d), directs the agent to go through the location C.

The superiority of the information value-driven approach in comparison to likelihood-driven approach is reflected in the fact that scouts fly not only towards the adversary locations that are on the current plan of the agent, but also towards other locations that may potentially result in a faster route for the agent. One example that shows this is that even though the plan for the agent in figure 8(d) does not involve going through the location A, one of the helicopters is still flown to detect this location. If the location A turns out to be free, the agent will be able to follow a



(a) initial configuration

(b) plan after convergence (30 secs)

(c) scouting helicopters on mission

(d) locations B and C sensed (adversary detected in B)

(e) location A sensed

(f) scout moves towards D

(g) location D sensed

(h) agent reaches its goal

Fig. 8. Path Clearance with PPCP planning and information value-driven use of five scouts

much faster route towards the goal by cutting through the location A. This is shown in figure 8(e): the location A was cleared and the new plan re-computed by PPCP makes the agent go through it.

In a similar fashion, one of the helicopters is sent to sense the location D even though it is also not on the plan that the agent follows (figure 8(f)). After this location is cleared, a faster route for the agent is re-computed by PPCP that cuts through location D (figure 8(g)). Figure 8(h) shows the final trajectory of the agent.

	Group I		Group II	
	cost	overhead	cost	overhead
freespace, no scouts	5,602	8.9%(±2.6%)	4,595	14.2%(±3.8%)
PPCP, no scouts	5,351	4.1%(±1.8%)	4,405	10.9%(±3.6%)
PPCP, likelihood scouts	5,168	1.6%(±1.3%)	4,055	3.0%(±1.5%)
PPCP, info. value scouts	5,076	0.0%(±0.0%)	3,931	0.0%(±0.0%)

TABLE II

The speed of the scouts is the same as the speed of the agent. Numbers in parentheses give 95% confidence intervals.

Experimental Study

The experiments presented in this section compare the cost of execution incurred by the agent planning using several approaches. The experiments were performed on the same two groups of environments described previously (shown in figure 6), with the exact same setup of experiments. Once again, each group contained 25 environments.

Tables II and III show the execution costs incurred by the agent that used different planning approaches. Table II is for the case when the speed of the scouts was the same as the speed of the agent, whereas table III gives results for the case when the speed of scouts was four times faster than that of the agent. In both scenarios, however, the scouts are assumed to be aerial and therefore did not need to avoid obstacles on the ground. In all the experiments, there were 10 scouts.

In each of the tables, the first row corresponds to planning with freespace assumption and not utilizing scouts. The agent itself did sensing for adversaries. The second row corresponds to planning with PPCP but again without utilizing scouts. The third row corresponds to planning with PPCP and using a likelihood-driven strategy for scouts. Finally, the fourth row corresponds to planning with PPCP and utilizing scouts according to the information value-driven approach. Same as before, in all of the experiments, the agent was given 5 seconds to plan while traversing 5 meter distance.

The tables show execution costs, as well as the overhead in execution cost when planning with different approaches relative to the execution cost when planning with PPCP and using the information value-driven approach to scheduling scouts (the last rows of the tables). Each entry is an average over 8 runs for each of the 25 environments in each group (200 samples total). For each run the true status of each adversary location was generated at random according to the probability having an adversary in there.

Table II shows that the overhead of *not* utilizing scouts while planning with freespace assumption can be up to 14.2%. This overhead goes even higher if the speed of the scouts is higher than the speed of the agent. The overhead of *not* utilizing scouts while planning with PPCP is also substantial (up to 10.9% when the speeds are the same and up to 13.5% when the scouts move faster). The difference between the two approaches to utilizing scouts (the last two rows in each table) is smaller. The execution cost of the agent utilizing scouts according to likelihood-driven approach can on average be up to 3.0% worse. While this

	Group I		Group II	
	cost	overhead	cost	overhead
freespace, no scouts	5,601	12.1%(±2.8%)	4,595	16.8%(±3.9%)
PPCP, no scouts	5,349	7.6%(±2.6%)	4,405	13.5%(±3.8%)
PPCP, likelihood scouts	4,988	1.6%(±0.7%)	3,927	2.9%(±1.6%)
PPCP, info. value scouts	4,902	0.0%(±0.0%)	3,819	0.0%(±0.0%)

TABLE III

Scouts are four times faster than the agent. Numbers in parentheses give 95% confidence intervals.

overhead may not seem to be very large, the overall behavior of scouts following the information value-driven approach is much more intelligent, and on the environments that were *not* randomly generated, such as the example in figure 1, the overhead of the likelihood-driven approach can be much higher. Unlike the likelihood-driven approach, the information value-driven approach is capable of taking advantage of the cases when sensing a possible adversary location that is not on the current plan of the agent can result in much faster route for the agent.

Conclusions

This article presented the techniques that we have recently developed to address the Path Clearance problem. Planning for Path Clearance is highly challenging since it involves both large-scale planning under uncertainty as well as coordination of multiple agents. As article describes however, the Path Clearance problem exhibits clear preferences on uncertainty. We have developed an efficient algorithm, PPCP, that takes advantage of the existence of clear preferences and can scale to large environments with large number of adversaries. The algorithm is anytime, converges to an optimal solution under certain conditions and scales well to large-scale environments. The article has also shown several strategies for how to use the PPCP algorithm in case multiple scouts are available. The important advantages of the presented strategies are that they are simple, efficient, scale to large environments and scale to large teams of heterogenous scouts. The experimental results demonstrated the scalability of the approaches and their benefits as compared to alternative approaches. In the future, it is important to investigate strategies for coordinating scouts using decentralized approaches and to develop planning algorithms that can scale to more than one non-scout agent (e.g., multiple ground troops and convoys that need to reach their goals without being discovered by adversaries).

Acknowledgments

This work was sponsored by the U.S. Army Research Laboratory, under contract Robotics Collaborative Technology Alliance (contract number DAAD19-01-2-0012). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government.

References

[1] Baral and Son. Approximate reasoning about actions in presence of sensing and incomplete information. In *Proceedings of International Logic Programming Symposium (ILPS)*, 1997.

[2] Chitta Baral, Vladik Kreinovich, and Raúl Trejo. Computational complexity of planning and approximate planning in the presence of incompleteness. *Artificial Intelligence*, 122(1-2):241–267, 2000.

[3] B. Bonet. An ϵ-optimal grid-based algorithm for partially observable markov decision processes. In *Proceedings of the International Conference on Machine Learning*, 2002.

[4] B. Bonet and H. Geffner. Planning with incomplete information as heuristic search in belief space. In S. Chien, S. Kambhampati, and C. Knoblock, editors, *Proc. 6th International Conf. on Artificial Intelligence Planning and Scheduling*, pages 52–61, Breckenridge, CO, 2000. AAAI Press.

[5] Craig Boutilier and David Poole. Computing optimal policies for partially observable decision processes using compact representations. In *Proceedings of the National Conference on Artificial Intelligence (AAAI-96)*, pages 1168–1175, Portland, Oregon, USA, 1996. AAAI Press / The MIT Press.

[6] Xavier Boyen and Daphne Koller. Tractable inference for complex stochastic processes. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 33–42, 1998.

[7] D. Ferguson, A. Stentz, and S. Thrun. PAO* for planning with hidden state. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, 2004.

[8] C. Guestrin, D. Koller, and R. Parr. Solving factored pomdps with linear value functions. In *Proceedings of the Workshop on Planning under Uncertainty and Incomplete Information*, 2001.

[9] S. Koenig and Y. Smirnov. Sensor-based planning with the freespace assumption. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1996.

[10] M. Likhachev and A. Stentz. Path clearance using multiple scout robots. In *Proceedings of the Army Science Conference (ASC)*, 2006.

[11] M. Likhachev and A. Stentz. PPCP: Efficient probabilistic planning with clear preferences in partially-known environments. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2006.

[12] M. Likhachev and A. Stentz. Goal directed navigation with uncertainty in adversary locations. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2007.

[13] M. Likhachev and A. Stentz. Information value-driven approach to path clearance with multiple scout robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[14] M. Likhachev and A. Stentz. Probabilistic planning with clear preferences on missing information. *Artificial Intelligence Journal (AIJ)*, 173(5-6):696–721, 2009.

[15] I. Nourbakhsh and M. Genesereth. Assumptive planning and execution: a simple, working robot architecture. *Autonomous Robots Journal*, 3(1):49–67, 1996.

[16] C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of Markov decision processses. *Mathematics of Operations Research*, 12(3):441–450, 1987.

[17] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for pomdps. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

[18] K.M. Poon. *A fast heuristic algorithm for decision-theoretic planning*. PhD thesis, The Hong Kong University of Science and Technology, 2001.

[19] N. Roy and G. Gordon. Exponential family pca for belief compression in pomdps. In *Advances in Neural Information Processing Systems*, 2002.

[20] Trey Smith and Reid G. Simmons. Heuristic search value iteration for POMDPs. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence (UAI)*, 2004.

[21] M.T.J. Spaan and N. Vlassis. A point-based POMDP algorithm for robot planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2399–2404, 2004.

[22] A. Stentz. The focussed D* algorithm for real-time replanning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1995.

[23] A. Stentz. Map-based strategies for robot navigation in unknown environments. In *AAAI Spring Symposium on Planning with Incomplete Information for Robot Problems*, 1996.

[24] R. Zhou and E. A. Hansen. An improved grid-based approximation algorithm for pomdps. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2001.