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 a distance close enough (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 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 3.5×3 km area shown in Figure 1(a). The traversability map is obtained by converting the image into a discretized two-dimensional (2-D) map, where each cell is 5×5 m in size and can either be traversable (shown in light gray color) or not (shown in dark gray

© U.S. AIR FORCE PHOTO/MASTER SGT. ROBERT W. VALENCA

Digital Object Identifier 10.1109/MRA.2009.932525

Path Clearance

Traversing Environments with Uncertainty in Adversary Locations

BY MAXIM LIKHACHEV AND ANTHONY STENTZ

1070-9932/09/$25.00©2009 IEEE

color). The large circles (e.g., A, B, C, D, and others) are the possible adversary locations, and their radii represent the sensor range of adversaries (100 m 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 m in our example). A planner, therefore, needs to reason about the 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 the 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) [1]. Planning optimally for POMDPs, in general, and planning with incomplete information and with sensing, in particular, is known to be intractable [2], [3]. In addition, the size of a typical environment is several kilometers wide while its traversability is highly nonuniform, making the size of the problem large and its 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.

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 probabilistic planning with clear preferences (PPCP) [4]. The algorithm is anytime usable, 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 [5]. In the second part of the article, we show several strategies for how to use the PPCP algorithm in case multiple scouting unmanned aerial vehicles (UAVs) are available [6], [7]. The experimental analysis shows that planning with PPCP results in a substantially 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 also has a tendency to be placed in places such 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 penalties 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 [8]–[10]. In particular, it often 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 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 recomputes 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 (and 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.

To avoid these situations, one may also try to set a cost function that penalizes the traversal of possible adversary locations.

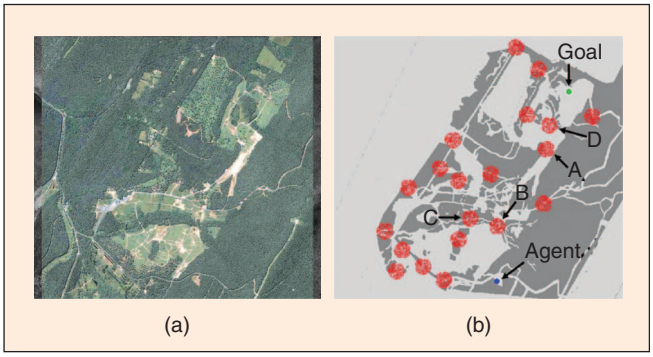

Figure 1. *Path clearance without scouts. (a) 3.5 × 3.0 km satellite image. (b) Traversability map.*

In path clearance, an agent has to navigate to its goal as quickly as possible without being detected by an adversary.

Although 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 s 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 to 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 m, while the length of the path traversed by the agent that makes the freespace assumption [Figure 2(b)] is 4,922 m.

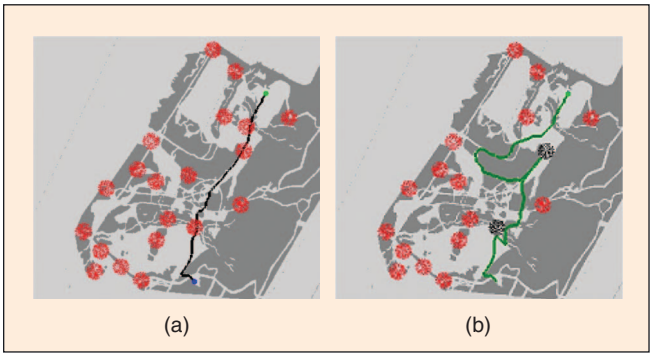

Figure 2. *Solving the path clearance problem with freespace assumption. (a) Planned path. (b) Actual path of the agent.*

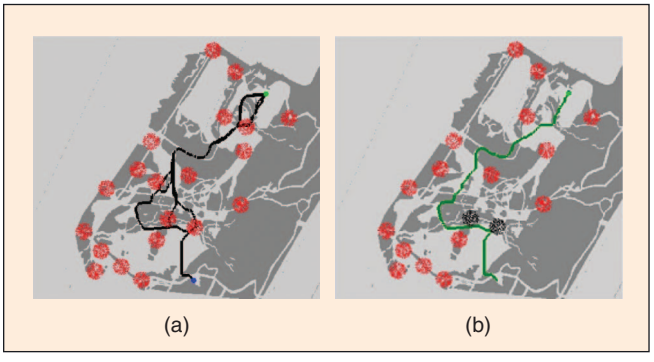

Figure 3. *Solving the path clearance problem with PPCP. (a) Generated plan. (b) Actual path of the agent.*

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 POMDPs [1]. Planning optimally for POMDPs, in general, and planning with incomplete information and with sensing, in particular, is known to be intractable [2], [3]. Various approximation techniques have been recently proposed instead [11]–[21].

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 [22], developed specifically for the problem of robot navigation in a partially known terrain. Similar 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 cannot be larger than the cost of the plan if the cell is occupied. On the basis of 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

Although decision-theoretic planning that takes into account the uncertainty about the environment is very hard to solve in general, 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 unknown 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

$$X' = \arg\min_{Y \in \mathrm{succ}(X,a)} c(X, a, Y) + v^*(Y). \qquad (1)$$

$c(X, a, Y)$ 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 (1).

PPCP [23] 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 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. Although 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 [23].

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 gray) 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 toward the cell, senses it and enters it if it is free, and returns back otherwise. The cost of each move is 1, and the cost of moving toward 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 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 moving 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)].

The path the agent follows may change any time the agent senses a possible adversary location.

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 was 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 seventh 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 [4] 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 5 s. 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 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 5 s 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 m/s) and the current policy generated by PPCP after 15 s since the agent was given its goal. Figure 5(e) shows the position of the agent and the policy

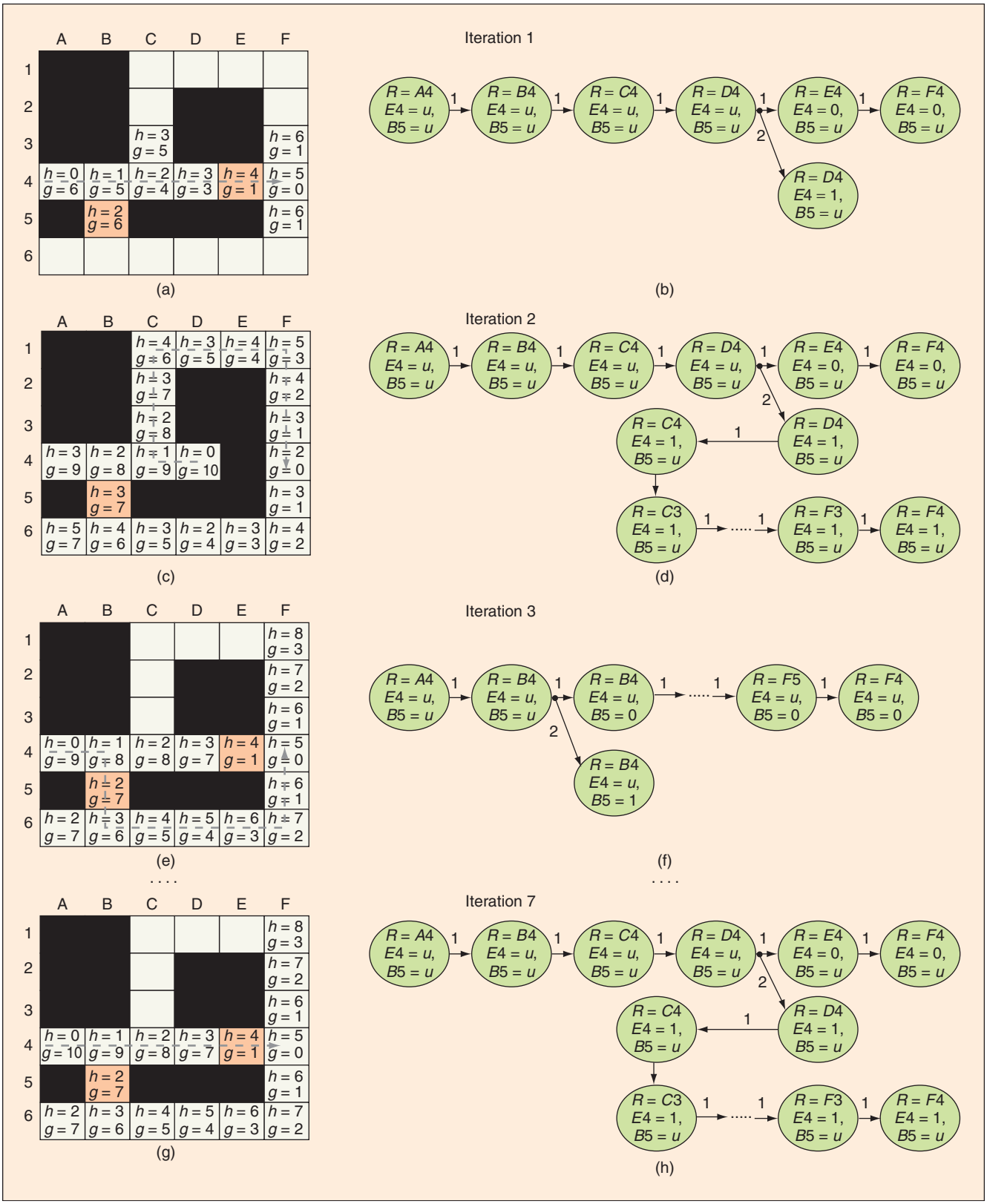

Figure 4. *Example of how PPCP operates. (a) Search for a path from* $[R = A4; E4 = u, B5 = u]$. *(b) PPCP policy after update. (c) Search for a path from* $[R = D4; E4 = 1, B5 = u]$. *(d) PPCP policy after update. (e) Search for a path from* $[R = A4; E4 = u, B5 = u]$.*(f) PPCP policy after update. (g) Search for a path from* $[R = A4; E4 = u, B5 = u]$. *(h) Final PPCP policy.*

the PPCP has generated after 30 s. 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, because 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 m long while the length of the trajectory traversed by the agent that plans with freespace assumption [Figure 2(b)] is 4,922 m.

Experimental Study

In [4], 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 an 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.

The path clearance problem exhibits clear preferences on uncertainty, and can therefore be solved efficiently by probabilistic planning with clear preferences.

We used randomly generated fractal environments that are often used to model outdoor environments [24]. 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. Figure 6(a) and (b) show typical environments that were used in our experiments. The lighter colors represent more easily traversable areas. All environments were of size 500×500 cells, with the size of each cell being 5×5 m.

The test environments were split into two groups. Each group contained 25 environments. For each environment in 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 Group II, we set up ten possible adversary locations. The coordinates of these locations, however, were chosen so 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 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

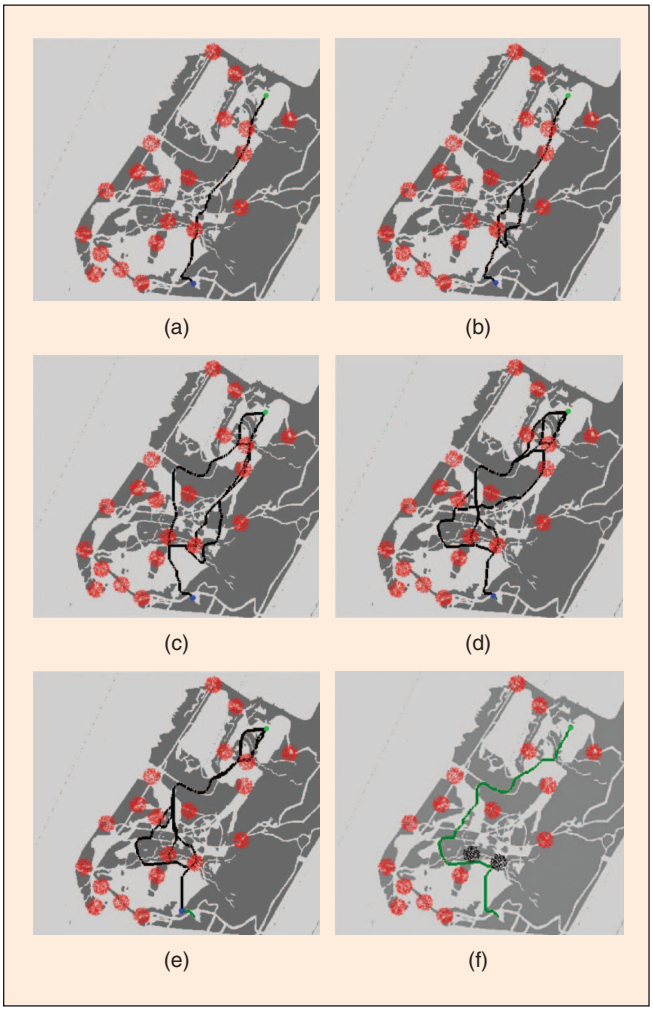

Figure 5. *Applying PPCP to path clearance. (a) The first policy. (b) The second policy. (c) After 5 s. (d) After 15 s. (e) After 30 s (PPCP converged). (f) Actual path of the agent.*

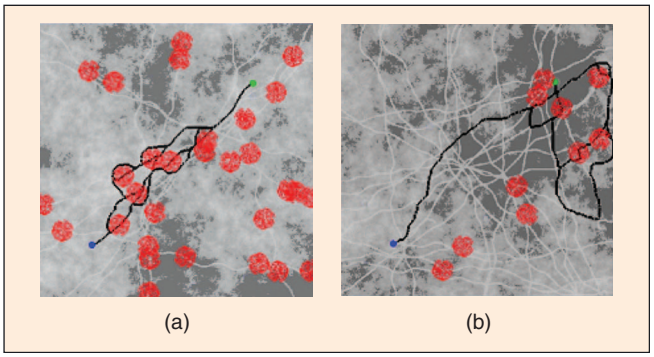

Figure 6. *The example of environments used in testing and the plans generated by PPCP for each. (a) Typical Group I environment. (b) Typical Group II environment.*

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 of 0.1–0.9.

In all of the experiments, the agent was moving and was given 5 s to plan while traversing the 5 m 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 1 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 Freespace 2 and Freespace 3 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 to bias the paths generated by the planner with freespace assumption away from going through possible adversary locations. The results are averaged over eight runs for each of the 25 environments in each group. For each run, the true status of each adversary location was generated at random according to the probability having an adversary in there.

The figure shows that planning with PPCP results in considerable execution of 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 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, 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 state space, 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. Although these strategies are heuristics, they are simple, efficient, scalable to a 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.

Table 1. The overhead in execution cost of navigating using planning with freespace assumption over navigating using planning with PPCP.

	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
Freespace 2	0.5	4.9	4.8	17.0
Freespace 3	2.1	4.3	0.0	12.7

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 to have the 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 the 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 toward 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 recomputes K possible adversary locations that the scouts need to sense and reassigns 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 recomputes 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 s 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 [Figure 7(c) and (d)]. In both of these locations, adversaries were detected (the locations turned black). Once this information is passed to the agent, PPCP recomputes a plan. The new plan, as shown in Figure 7(d), no longer directs the agent toward the locations B and C, since they are already known to contain adversaries.

At the same time, one of the scouts flies toward 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 3,795 m, which is 328 m 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 [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 in such a way 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 follows:

$$s = \arg\min_{s'} E\{c(\pi^*(s' \text{is known})) + \text{cost of sensing}(s')\}.$$

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 rewritten as follows:

$$s = \arg\max_{s'} E\{c(\pi^*) - c(\pi^*(s' \text{ is known})) - \text{cost of sensing } (s')\}, \quad (2)$$

The idea behind information value-driven use of scouts is to send the scouts to the possible adversary locations that maximize the value of information.

where π^* is 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

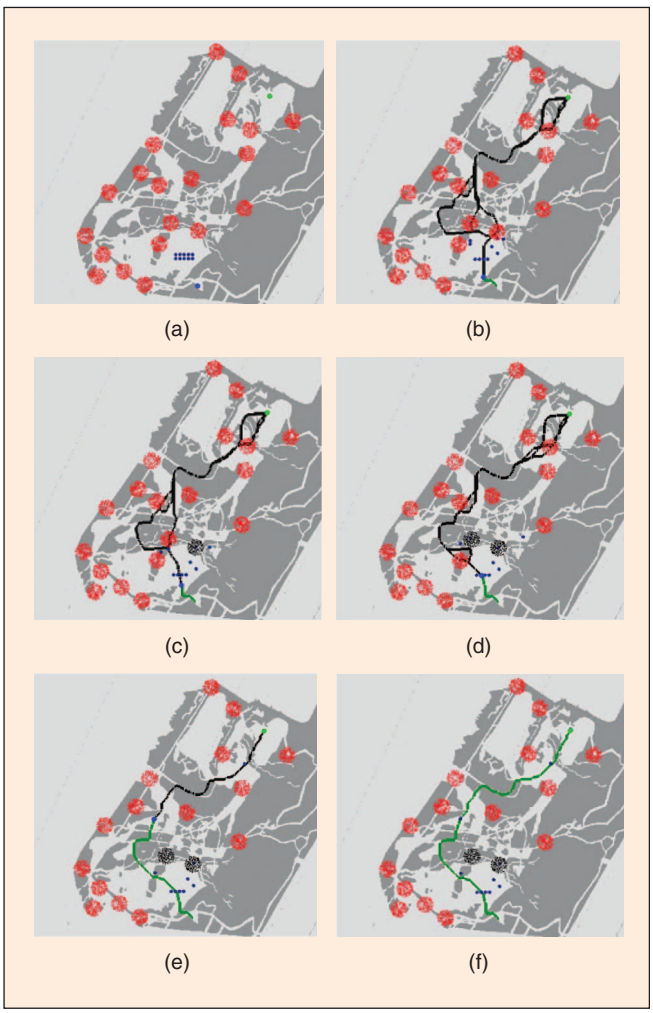

Figure 7. *Path clearance with PPCP planning and likelihood-driven use of ten scouts. (a) Initial configuration. (b) Plan after convergence (30 s). (c) Location B sensed (adversary detected). (d) Location C sensed (adversary detected). (e) Location D sensed (no adversary present). (f) Agent reaches its goal.*

The path clearance problem becomes even more challenging to solve when there are multiple scouts available.

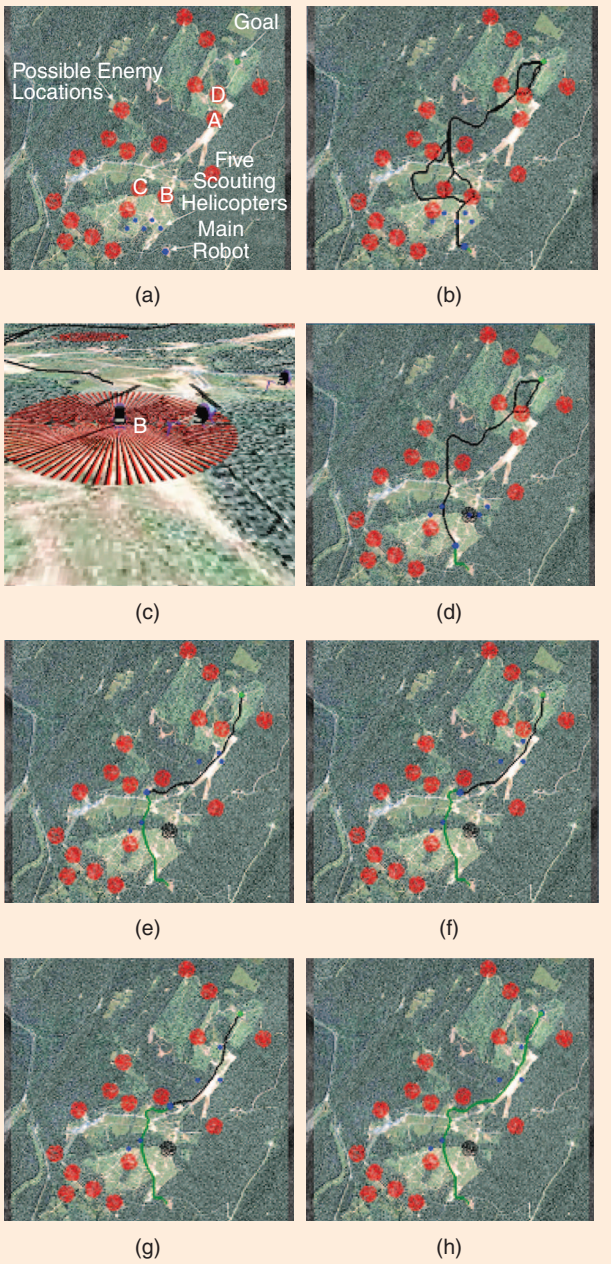

Figure 8. *Path clearance with PPCP planning and information value-driven use of five scouts. (a) Initial configuration. (b) Plan after convergence (30 s). (c) Scouting helicopters on mission. (d) Locations B and C sensed (adversary detected in B). (e) Location A sensed. (f) Scout moves toward D. (g) Location D sensed. (h) Agent reaches its goal.*

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 [7]).

The term $E\{\text{cost of sensing}(s')\}$ can be computed as the minimum expected cost of sensing s' across all available scouts. After we compute an estimate of $E\{c(\pi^*) - c(\pi^*(s' \text{ is known})) - \text{cost of sensing}(s')\}$ for each s', we pick s' that maximizes it and assign it to the scout which minimized the term $E\{\text{cost of sensing}(s')\}$. In the experiments, the scouts were helicopters and therefore the term $E\{\text{cost of sensing}(s')\}$ 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 the 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 there are five scouting helicopters now (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 toward 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 recomputed 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 toward the adversary locations that are on the current plan of the agent, but also toward 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 much faster route toward the goal by cutting through the location A. This is shown in Figure 8(e): the location A was cleared, and the new plan recomputed 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 recomputed by PPCP that cuts through location D [Figure 8(g)]. Figure 8(h) shows the final trajectory of the agent.

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 2 and 3 show the execution costs incurred by the agent that used different planning approaches. Table 2 is for the case when the speed of the scouts was the same as the speed

of the agent, whereas Table 3 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 ten 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 s to plan while traversing the 5 m 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 eight 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 2 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 an average be up to 3.0% worse. Although this 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 a 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 the 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 usable 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 nonscout agent (e.g., multiple ground troops and convoys that need to reach their goals without being discovered by adversaries).

The idea behind likelihood-driven use of scouts is to have the scouts navigate to and sense the locations that are most likely to be visited by the agent.

Table 2. The speed of the scouts is the same as the speed 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)

Numbers in parentheses give 95% confidence intervals.

Table 3. Scouts are four times faster than the agent.

	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)

Numbers in parentheses give 95% confidence intervals.

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.

Keywords

Path clearance, planning under uncertainty, planning with adversaries, multiagent coordination.

References

[1] B. Bonet and H. Geffner, "Planning with incomplete information as heuristic search in belief space," in *Proc. 6th Int. Conf. Artificial Intelligence Planning and Scheduling*, Breckenridge, CO, S. Chien, S. Kambhampati, and C. Knoblock, Eds. Menlo Park, CA: AAAI Press, 2000, pp. 52–61.

[2] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of Markov decision processes," *Math. Oper. Res.*, vol. 12, no. 3, pp. 441–450, 1987.

[3] C. Baral, V. Kreinovich, and R. Trejo, "Computational complexity of planning and approximate planning in the presence of incompleteness," *Artif. Intell.*, vol. 122, no. 1–2, pp. 241–267, 2000.

[4] M. Likhachev and A. Stentz, "Probabilistic planning with clear preferences on missing information," *Artif. Intell. J.*, vol. 173, no. 5–6, pp. 696–721, 2009.

[5] M. Likhachev and A. Stentz, "Goal directed navigation with uncertainty in adversary locations," in *Proc. Int. Conf. Intelligent Robots and Systems (IROS)*, 2007, pp. 4127–4134.

[6] M. Likhachev and A. Stentz, "Path clearance using multiple scout robots," in *Proc. Army Science Conf. (ASC)*, 2006.

[7] M. Likhachev and A. Stentz, "Information value-driven approach to path clearance with multiple scout robots," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2008, pp. 2651–2656.

[8] I. Nourbakhsh and M. Genesereth, "Assumptive planning and execution: A simple, working robot architecture," *Autonom. Robot. J.*, vol. 3, no. 1, pp. 49–67, 1996.

[9] S. Koenig and Y. Smirnov, "Sensor-based planning with the freespace assumption," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 1996, pp. 3540–3545.

[10] A. Stentz, "The focussed D* algorithm for real-time replanning," in *Proc. Int. Joint Conf. Artificial Intelligence (IJCAI)*, 1995, pp. 1652–1659.

[11] C. Boutilier and D. Poole, "Computing optimal policies for partially observable decision processes using compact representations," in *Proc. Nat. Conf. Artificial Intelligence (AAAI-96)*, Portland, Oregon., Menlo Park, CA: AAAI Press, 1996, pp. 1168–1175.

[12] X. Boyen and D. Koller, "Tractable inference for complex stochastic processes," in *Proc. Int. Conf. Uncertainty in Artificial Intelligence (UAI)*, 1998, pp. 33–42.

[13] C. Baral and T. C. Son, "Approximate reasoning about actions in presence of sensing and incomplete information," in *Proc. Int. Logic Programming Symp. (ILPS)*, 1997, pp. 387–401.

[14] C. Guestrin, D. Koller, and R. Parr, "Solving factored POMDPs with linear value functions," in *Proc. Workshop Planning Under Uncertainty and Incomplete Information*, 2001, pp. 67–75.

[15] K. M. Poon, "A fast heuristic algorithm for decision-theoretic planning," Ph.D. thesis, The Hong Kong Univ. Sci. Technol., 2001.

[16] N. Roy and G. Gordon, "Exponential family PCA for belief compression in POMDPs," in *Advances in Neural Information Processing Systems*, S. Becker, S. Thrun, and K. Obermay, Eds. Cambridge, MA: MIT Press, 2002, pp. 1043–1049.

[17] J. Pineau, G. Gordon, and S. Thrun, "Point-based value iteration: An anytime algorithm for POMDPs," in *Proc. Int. Joint Conf. Artificial Intelligence (IJCAI)*, 2003, pp. 1025–1032.

[18] M. T. J. Spaan and N. Vlassis, "A point-based POMDP algorithm for robot planning," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2004, pp. 2399–2404.

[19] B. Bonet, "An ε-optimal grid-based algorithm for partially observable Markov decision processes," in *Proc. Int. Conf. Machine Learning*, 2002, pp. 51–58.

[20] R. Zhou and E. A. Hansen, "An improved grid-based approximation algorithm for POMDPs," in *Proc. Int. Joint Conf. Artificial Intelligence (IJCAI)*, 2001, pp. 707–716.

[21] T. Smith and R. G. Simmons, "Heuristic search value iteration for POMDPs," in *Proc. Int. Conf. Uncertainty in Artificial Intelligence (UAI)*, 2004, pp. 520–527.

[22] D. Ferguson, A. Stentz, and S. Thrun, "PAO* for planning with hidden state," in *Proc. 2004 IEEE Int. Conf. Robotics and Automation*, 2004, pp. 2840–2847.

[23] M. Likhachev and A. Stentz, "PPCP: Efficient probabilistic planning with clear preferences in partially-known environments," in *Proc. Nat. Conf. Artificial Intelligence (AAAI)*, 2006, pp. 860–868.

[24] A. Stentz, "Map-based strategies for robot navigation in unknown environments," in *AAAI Spring Symp. Planning with Incomplete Information for Robot Problems*, 1996, pp. 110–116.

Maxim Likhachev received his Ph.D. degree in computer science from Carnegie Mellon University (CMU) in 2005. He is a research assistant professor at the Computer and Information Science Department of the University of Pennsylvania. His research interests are primarily in planning for deterministic and probabilistic domains with applications to robotics. He develops planning methods that can be used in real time, can be analyzed theoretically, and are easy to use. Currently, one of the main thrusts of his research is solving complex high-dimensional planning problems with uncertainty using a series of highly efficient and easy-to-implement deterministic searches. He has applied his ideas to problems such as high-speed robot navigation in unknown and adversarial environments, coordination of multiagent systems, and motion planning of high-degree of freedom articulated robots.

Anthony Stentz received his B.S. degree in physics from Xavier University of Ohio in 1982 and M.S. and Ph.D. degrees in computer science from CMU in 1984 and 1989, respectively. He is a research professor at the Robotics Institute, CMU, and an associate director of the Robotics Institute's National Robotics Engineering Center. His research expertise includes autonomous vehicles, dynamic planning, multivehicle planning and coordination, perception for mobile vehicles, robot architecture, and artificial intelligence in the context of fieldworthy systems. He has authored more than 150 journal articles, conference and workshop papers, books, technical reports, and hold patents to his credit. He is a recipient of the Alan Newell Award for Research Excellence for his agricultural automation work and a NASA Board Award for developing software used on the Mars Exploration Rovers.

Address for Correspondence: Maxim Likhachev, Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104, USA. E-mail: maximl@seas.upenn.edu.