# Planning for Landing Site Selection in the Aerial Supply Delivery

Aleksandr Kushleyev
ESE Department
University of Pennsylvania

Brian MacAllister
CIS Department
University of Pennsylvania

Maxim Likhachev
Robotics Institute
Carnegie Mellon University

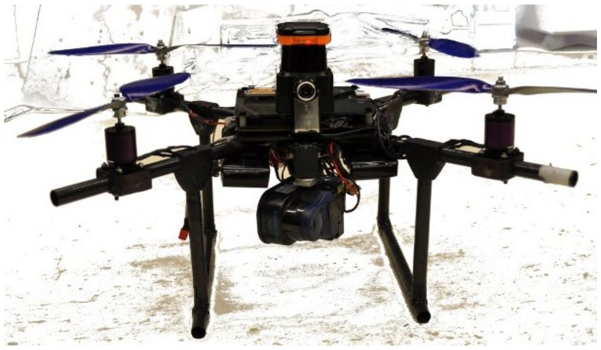
*Abstract*— **In the aerial supply delivery problem, an unmanned aircraft needs to deliver supplies as close as possible to the desired location. This involves choosing, flying to, sensing, and landing at a safe landing site that is most accessible from the goal. The problem is often complicated by the fact that the availability of these landing sites may be unknown before the mission begins. Therefore, the aircraft needs to compute a sequence of actions that will minimize the expected value of the objective function. The problem of computing this sequence corresponds to planning under uncertainty in the environment. In this paper, we show how it can be solved efficiently via a recently developed probabilistic planning framework, called Probabilistic Planning with Clear Preferences (PPCP). We show that the problem satisfies the Clear Preferences assumption required by PPCP, and therefore all the theoretical guarantees continue to hold. The experimental results in simulation demonstrate that our approach can solve large-scale problems in real-time while experiments on our custom quad-rotor helicopter provide a proof of concept for the planner.**

## I. INTRODUCTION

Autonomous aerial robots can sometimes be tasked to deliver supplies to various remote locations. Their high speed and operating altitude offer advantages over ground robots, which have to drive through possibly cluttered environments. However, UAVs need to take off and land, often requiring special equipment and algorithms for choosing a landing site that is good for landing *and* is easiest to reach from the desired goal location for the supplies. Much work has been done previously on the analysis of the ground terrain in search for good landing sites [1], [2], [3]. Similarly, there has been work done on the control aspect of landing robustly at a selected landing site [4], [5]. This paper studies a different aspect of the landing site selection problem: the problem of planning a sequence of flights and sensing actions in order to find and land at the best landing site as quickly as possible.

Figure 2 demonstrates a sample scenario for the planning problem we are considering and shows the steps required for completing the task. The helicopter (Figure 1(b)) needs to deliver a payload to a location marked $GOAL$ in Figure 2(b). Based on prior terrain information and its analysis, the helicopter identifies 7 possible landing sites, shown in the same figure. These possible landing sites could have been also provided by a human operator. The task of the planner is to decide where the helicopter should fly, sense, and potentially land in order to minimize the expected sum of the flight time and the cost of reaching the landing site from the designated goal location for the supplies.

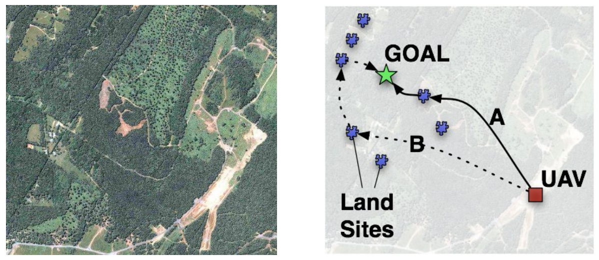
To compute this plan, one could use assumptive planning techniques [6], [7], [8] by assuming that all possible landing sites are valid, flying to the landing site that minimizes the objective function, and then landing there if it is good, or flying to the next one otherwise. While fast to plan, these approaches do not minimize the expected cost and can result in unnecessary flight time and landing at a suboptimal site.


(a) Autonomous quad-rotor aircraft, fully built by our group, is used as a test bed before transitioning to a larger aircraft

(b) Tandem aircraft, built by Dragonfly Pictures, Inc.,

Fig. 1. Aircrafts involved in the experiments. (a) shows our current test platform. (b) shows the target platform for our autonomy software

In this paper, we present a planning approach to computing plans that do minimize the expected cost. In particular, while our planning problem corresponds to an instance of planning with missing information and, in general, is hard-to-solve, we show that it exhibits clear preference on uncertainty. This makes it solvable via a recently proposed approach PPCP (Probabilistic Planning with Clear Preferences) [9], which uses these preferences to decompose the problem into a series of easy-to-solve graph searches. This decomposition allows PPCP to solve large scale planning under uncertainty problems in anytime fashion while providing rigorous theoretical guarantees on the solution quality. Our experimental analysis in simulations shows that our approach can solve problems with up to 15 possible landing sites in under a second on average. Experiments on our custom quad-rotor helicopter



(a) satellite image (b) start, goal, candidate land sites

Fig. 2. Landing site selection problem. (a) shows a sample satellite image that can be used for extracting candidate landing sites shown in (b). Trajectories A and B are only two out of $O(c^n)$ possible paths with no loops for $n$ sites.

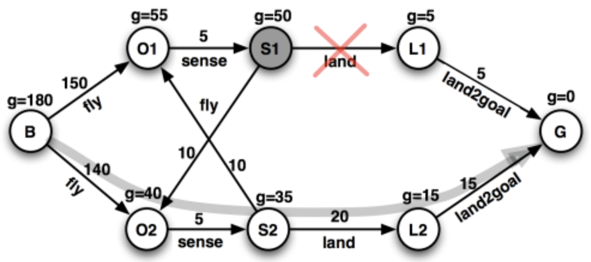


Fig. 3. Backward search on a fully known environment from start state $B$ to goal $G$ with two possible landing locations. Location 1 is known to be bad (shown in gray), and 2 is known to be good (shown in white). Transitions are denoted with arrows and costs are marked next to each transition. Only one path to the goal is possible and is shown with the thick arrow.

(Figure 1(a)) show that the approach is indeed feasible to run on a real system. To the best of our knowledge, the proposed approach is the first one to investigate the problem of planning a sequence of landing site sensing operations with the goal of minimizing the expected cost before landing.

## II. PROBLEM FORMULATION

### A. Task

The desired autonomous behavior can be summarized as follows: an unmanned helicopter (also referred to as robot), equipped with GPS, lidar and elevation maps, must safely deliver a certain payload to the designated location. It has to choose a landing location, fly a collision free trajectory, and land. This process involves a combination of planning, perception, and motion control, however the main focus of the paper is on planning an efficient landing site selection.

### B. Problem Representation

**Scenario 1: Fully known environment**. Consider first a scenario, in which the full environment is known to the robot. We represent the problem with a fully deterministic concise graph $G_d$ containing states $s_i$ which can be of one of five types: $B$ (robot is at the base), $O_i$ (robot is hovering over a candidate landing site $i$), $S_i$ (the robot is still hovering over site $i$ and has performed the sensing action to confirm the ability to land there), $L_i$ (the robot has landed at site $i$) and $G$ (the payload has been delivered to the destination).

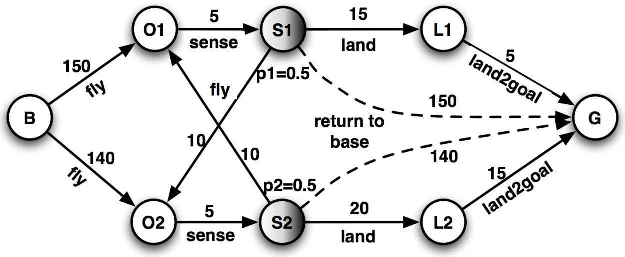


Fig. 4. Sample environment with missing information. The status of land sites 1 and 2 is no longer known as ground truth (shown with gradient-filled states).

Transitions are defined for a pair of states and an action such that $s_j = succ(s_i, a)$, where $s_j$ is the successor of $s_i$ if action $a$ is executed at $s_i$. Corresponding transition costs $c(s_i, a)$ are known a priori and depend only on the current state $s_i$ and deterministic action $a$ and are independent of past transitions (Markov property). Figure 3 shows the full graph of such environment along with the optimal solution, which may be trivially obtained by observation or running backward A*. The g-values represent the optimal cost-to-goal from any state.

The construction of graph $G_d$ reflects the sequence of events that is required for successful completion of the mission. The robot starts out at the base, flies to one of the candidate landing sites and senses it to confirm the ability to land. If the outcome of sensing action is positive (site is clear for landing), the helicopter can either land or fly to a different location, otherwise, flying is an only option. In this example, we assume that the sensing action is deterministic and reveals the ground truth about the status of the site.

If the robot is eventually able to find a good site and land, we assume that the payload can be delivered to the final destination by other means (e.g. human picks it up). Without missing information, the problem of selecting the optimal flight policy is quite simple and is trivially solved by a forward or backward search on the full state space. We chose the backward option in this example so that it directly relates to our next scenario (the direction is reversed and the search is performed from the goal to the start state).

**Scenario 2: Effect of missing information**. In reality, it may not be possible to guarantee that any particular site is available for safe landing. For example, there may be a chance that some external event interferes with the landing procedure. In our context, the sensing action reveals the true state of the site, which is no longer known a priori. Therefore, the problem formulation has to be modified slightly, making the sensing action stochastic. In particular, for each possible landing location, we assign a binary variable, whose true value represents the ground truth, and it simply tells whether or not the helicopter is capable of successfully landing at a particular location. Regardless of what may actually prevent it from doing so, this implies that the flight plan must be created without knowing for sure whether any particular site will accomondate safe landing. In practice, a number of sources may introduce such uncertainty such as possible exposure to enemy and recent changes to terrain due to military activity such as bombing, inaccurate prior maps, and other factors. Figure 4 shows the modified environment, reflecting the unknown state of sites 1 and 2. Note that an additional action is present, allowing the robot to return to the base if no landing site is available for landing.

**Legal Actions**. Initially, the robot is in state $B$ and its goal is $G$. There are two candidate landing sites and their status is given by the hidden variables $L1$ and $L2$. We use $L_i = u$ if the status is unknown, $L_i = 0$ if the site is not available for landing, and $L_i = 1$ if landing is possible at site $i$. Initially, the values of two variables $L_1$ and $L_2$ are not known to the robot and remain unknown until the robot actually senses the sites. The robot can get to the goal by landing (if possible) at either land site or returning to the base if neither locations are suitable for landing. The legal actions are assumed as follows: from $B$, the helicopter can only fly to either $O_1$ or $O_2$; from $O_i$ the robot performs a stochastic sensing action, whose outcome will depend on the true value of $L_i$. Sensing gets the robot to the state $S_i$ with $L_i$ set to its true value. Only if $L_i$=1, the robot has an option of landing at $i$, but either value of $L_i$ allows flying to a different location $O_j$ where $i \neq j$ (in other words, the robot is not forced to land if the site is good). After landing, the mission is complete and the payload is delivered to the goal with an additional known cost (referred to as $land2goal$ cost). If $L_i = 0 \forall i$ then helicopter can return to the base via an extra action (marked with dotted lines in Figure 4), which also completes the mission.

**Consequences**. The representation in Figure 4 no longer describes the full state space of the problem (as opposed to graph $G_d$ in Figure 3 for the first example). Since there are now two unknowns, each state must be augmented with two independent variables having values $u$, 0, or 1 (corresponding to unknown, bad, good). In addition, we introduce stochastic actions and sensing now has two possible outcomes according to the probability of success $p_i$. In Figure 5, graph $G_b$ reflects the new problem structure, showing the belief state space of the problem. Each belief state consists of an observable part (location of the robot $R \in \{B, Oi, Si, Li, G\}$) and a hidden part (true status of the unknown variables $Li \in \{u, 0, 1\}$). As the robot transitions from one state to another, variable $R$ is updated accordingly and $Li$ is updated with the result of sensing action at site $i$, only if sensing actually occurs. We assume perfect sensing, that is, the sensing action reveals the true value of the corresponding unknown. Figure 5 shows the full belief state space and the optimal policy for probability of sensing success $p_{1,2}$=0.5. It is clear how much the state space increased after only adding two unknowns - in fact, the size of the belief state space is exponential in the number of sites (exact formula for our formulation is $(3\times n+2)\times 3^n$).

## III. PPCP ALGORITHM

In this section, we briefly describe how the landing site selection problem with our formulation can be efficiently solved using the PPCP algorithm. We state the required assumptions and later show that they hold for our formulation. A more detailed description of the algorithm can be found in [9].

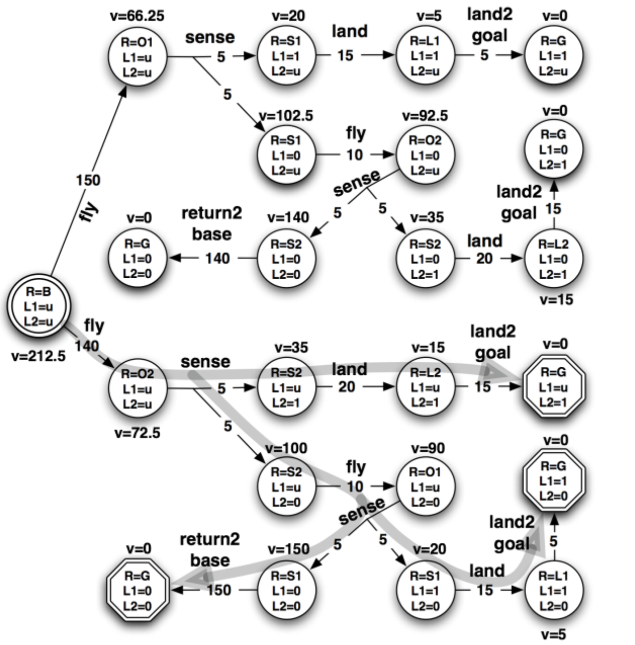


Fig. 5. Graph $G_b$ (full belief state space). Optimal policy for $p_{1,2} = 0.5$ is marked with the thick arrows. V-values show the expected cost-to-goal from any state.

The PPCP algorithm is well-suited for planning in the environments that are only partially-known. The algorithm assumes that the true environment itself is fully deterministic and can be modeled as a graph. Thus, in the landing site selection problem, if we knew precisely which sites are good and bad, then there would be no uncertainty about any robot actions: both sensing and move actions would be deterministic actions. In reality, the robot is uncertain which sites can provide safe landing conditions. As a result, there are two possible outcomes of a sensing action: the land site is *confirmed* at the possible landing location that was sensed and land site is *rejected*. PPCP assumes perfect sensing, so once the robot has sensed a particular location, it then knows with full certainty whether a site is suitable for landing or not.

The goal of the planner is to construct a policy that reaches any state at which $R = G$ while minimizing the expected cost of execution. Figure 5 shows a sample policy generated by PPCP. The policy specifies the path which the robot should follow after each outcome of sensing action. All branches of the policy end at a state whose $R = G$, in other words, the robot is at its goal location.

The main requirement for using the PPCP algorithm is that the problem must exhibit clear preferences on unknown information. What this means is that for any stochastic transition, it must be always clear which value of the missing variable results in the best outcome. In other words, there must exist a clear preference for the actual values of the missing information. Mathematically, for any unknown variable $h$, the preferred value exists and is equal to $b$ if for any state $X$ in which $h$ is unknown and action $a$, which reveals the value of $h$, there exists a (best) successor state $X'$ which has $h$=$b$ and satisfies the requirement

$$X' = argmin_{Y \in succ(X,a)} c(S(X), a, S(Y)) + v^{\star}(Y)$$

where $S(X)$ represents the part of $X$ that is always observable ( [9]).

PPCP exploits the existence of clear preferences to scale up to very large problems. In particular, PPCP constructs and refines a plan by running a series of deterministic A*-like searches. Furthermore, by making an approximating assumption that it is not necessary to retain information about the variables whose values were discovered to be clearly preferred values, PPCP keeps the complexity of each search low (exponentially smaller than the size of the belief state-space) 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 and deterministic searches can often be many orders of magnitude faster than probabilistic planning techniques. While the assumption PPCP makes does not need to hold for the algorithm to converge, the returned plan is guaranteed to be optimal if the assumption does hold.

## IV. APPLICATION OF PPCP TO LANDING SITE SELECTION

### A. Clear Preferences

In the context of landing site selection, we claim that the preferred outcome of a sensing action always exists. Consider the following for the proof. Let $X_g$ and $X_b$ be the good and bad successors of any state $Y$ after performing a sensing action. Let $v^{\star}(X_b)$ be the cost of executing the optimal policy at $X_b$. Similarly, we define $v^{\star}(X_g)$ for $X_g$. Also, let $A(X_b)$ and $A(X_g)$ be the sets of possible further actions at the corresponding states. According to our formulation, $A(X_g) = \{A(X_b), land\}$. This is true since if the outcome of sensing action is good, the further action set is augmented by the ability to land. Therefore, the robot at $X_g$ still has an option of following $X_b$'s optimal policy and $v^{\star}(X_g) \leq v^{\star}(X_b)$, making $X_g$ the preferred outcome. This is a consequence of the fact that knowing that a landing site is good does not invalidate $X_b$'s (or, in fact, any other state's) optimal policy.

### B. Operation

The PPCP algorithm operates in anytime fashion. It quickly constructs an initial policy and then refines it until convergence. At the time of convergence, the policy it obtains is guaranteed to minimize the expected execution cost under certain conditions (described in [9]). Figure 6 demonstrates how PPCP solves the landing site selection in a partially-known environment given in Figure 4.

PPCP repeatedly executes deterministic searches that are very similar to the A* search [10], an efficient and widely-known search algorithm. During each iteration PPCP assumes some configuration of unknown variables and performs search in the corresponding deterministic graph. Thus, the first search in Figure 6 assumes that both unknown sites are good and finds a path that goes straight to the goal (Figure 6(a)). (The shown $g$-values are equivalent to those maintained by the A* search). PPCP takes this path and uses it as an initial policy for the robot (Figure 6(b)). The first policy, however, has only computed a path from the preferred outcome of sensing site 1, corresponding to the site being available for landing. The state $[R = S_1; L_1 = 0, L_2 = u]$, on the other hand, has not been explored yet. The second search executed by PPCP explores this state by finding a path from it to the goal. This search is shown in Figure 6(c). During this search cell site 1 is assumed to be unavailable, same as in the state $[R = S_1; L_1 = 0, L_2 = u]$. The found path is incorporated into the policy maintained by PPCP (Figure 6(d)).

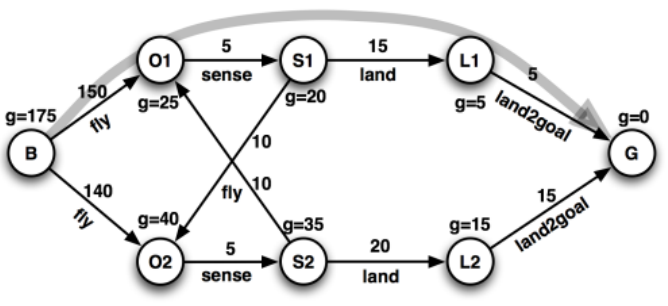
In the third iteration, PPCP tries to find a path from the start state to the goal again (Figure 6(e)). Now, however, it no longer generates the same path as initially in (Figure 6(a)). The reason for this is that it has learned that the cost of attempting to reach the goal through site 1 is higher than what it initially thought to be. It now appears that going through site 2 is cheaper, resulting in the new PPCP policy, shown in Figure 6(f). This policy, however, has an unexplored outcome state again, namely, state $[R = S_2; L_1 = u, L_2 = 0]$. This becomes the state to explore in the next iteration.

PPCP continues to iterate in this manner until convergence. In this example, it converges on the 5th iteration. The full belief state space with the final policy are shown in Figure 5. In this example the final policy 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 site the robot has discovered to be good (see [9] for more details).

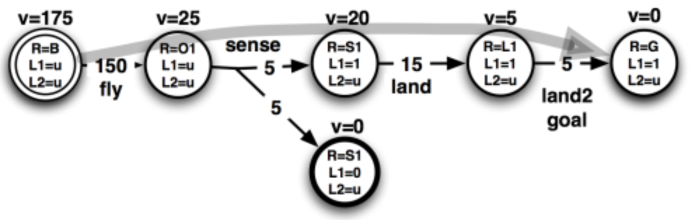
### C. Memory

The PPCP algorithm exhibits memoryless property - it does not remember the preferred outcomes of stochastic actions. This enables it to efficiently solve large-scale problems with uncertainty without traversing the whole belief state space, however the solution may be suboptimal if the true optimal policy relies on remembering preferred outcomes. We can never know whether the optimal policy requires memory, but our implementation allows to vary the number of remembered preferred outcomes (memory slots), trading off with execution speed and usage of computer memory.

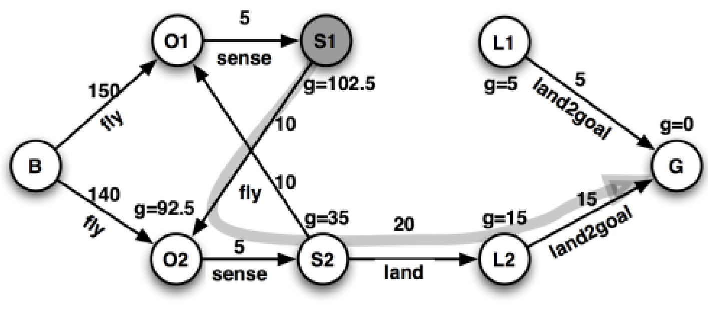
In order to implement the k-slot memory, the observable part of the belief states must be augmented with k variables. This results in substantial increase in the size of the projected search space. For example, with 10 landing sites and one memory slot, the number of states is increased by a factor of 11 (all permutations of the memory, including $empty$ value). Therefore, k must be chosen very carefully and for



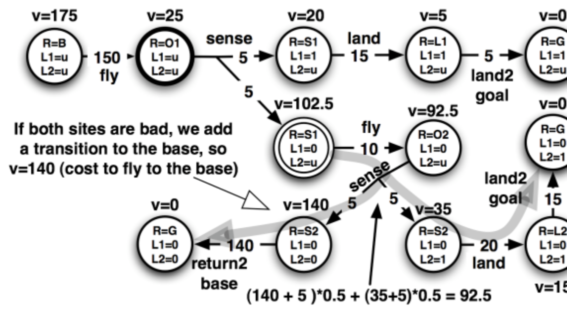
(a) search for a path from $[R = B;\ L_1 = u;\ L_2 = u]$



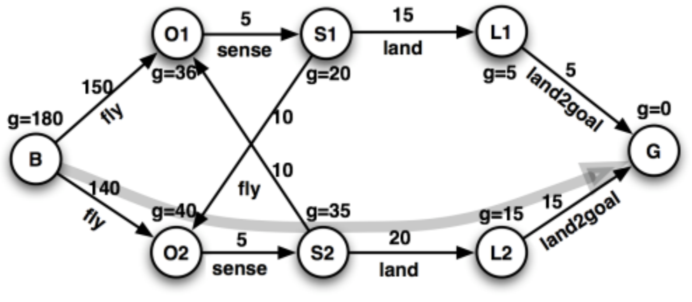
(b) PPCP policy after update



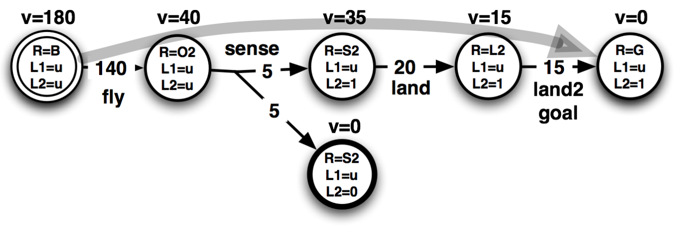
(c) search for a path from $[R = S1;\ L_1 = 0;\ L_2 = u]$



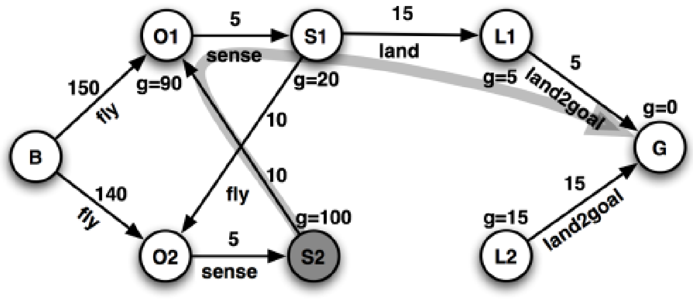
(d) PPCP policy after update



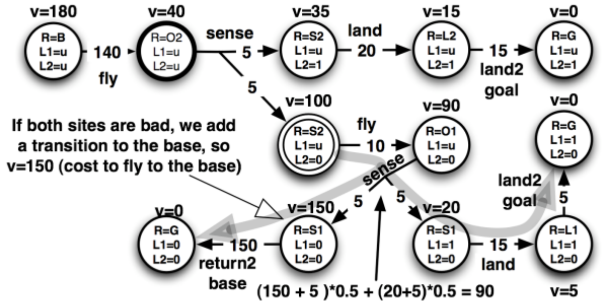
(e) search for a path from $[R = B;\ L_1 = u;\ L_2 = u]$



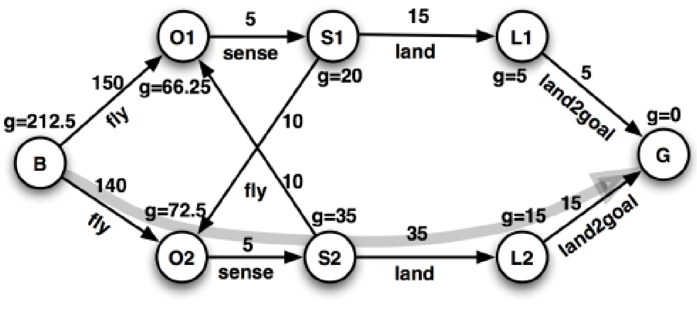
(f) PPCP policy after update



(g) search for a path from $[R = S2;\ L_1 = u;\ L_2 = 0]$



(h) PPCP policy after update



(i) search for a path from $[R = B;\ L_1 = u;\ L_2 = u]$

Fig. 6. Solution of the example problem using PPCP algorithm. All the steps, needed for convergence to the optimal solution, are shown and the final policy appears in Figure 5.

a very good reason, otherwise, depending on the problem, it is quite possible that this change would not help improve the produced policies.

## *D. Transition Costs and Probabilities*

The low-level details of the algorithm have been presented and now we must explain how all the transition costs and sensing probabilities are generated.

**Flight Cost** informs the planner how expensive it is for the helicopter to fly between different locations (i.e. base and landing sites). Very simply, Euclidean distance between the two locations may serve as a reasonable estimate. If a more accurate approach is desired, a helicopter motion planner, tuned for the particular aircraft, can be used, which will take into account the environment and create trajectories that avoid collisions.

**Sensing Cost** is associated with the procedure for evaluating the quality of the site. The helicopter must resolve any existing uncertainty by discovering the true values of all hidden variables, associated with the particular landing site.

**Quick-Sensing Cost** does not appear in the main formulation, however, if memory slots are enabled, and the helicopter's policy instructs it to come back to an already confirmed landing site, it may still be required to perform a last-minute check before the landing maneuver. This may require additional resources, therefore, has a cost.

**Landing cost** Computation of the landing cost is outside of the scope of this paper, but has been investigated quite thoroughly by [1], [2], [3] and others. In general, this cost depends on the helicopter's capabilities and various environmental factors like terrain roughness, wind direction, and weather conditions.

**Land2Goal Cost** summarized the efforts required to get the payload delivered to the desired location after the helicopter has landed. Since the landing location may not be right at the goal, some form of ground transport, manned or autonomous, might be required for completing the mission. The amount of additional efforts may also play a role in the policy through the land2goal cost.

**Return2Base Cost** If all the landing sites are actually unacceptable for landing, the only thing that the robot can do (according to our formulation) is come back to the base. This may be the same as the forward flight cost, or maybe different, if factors like wind direction play a significant role.

**Landing Site Probability**. In addition to the landing cost, it is critical to estimate the probability of each site being good. For example, a particular landing location may seem perfect from the terrain analysis point of view, but may not be so good if there is a chance of enemy appearing in the area. This probability value may reflect some aspects of the uncertainty in terrain as well, however, most importantly, it is able to incorporate environmental factors that are probabillistic in nature. These may include weather conditions, chance that the site is occupied by another vehicle or people, or probability of being exposed to enemy surveilance. The exact values of these quantities may never be known, but good approximations may be made based on additional sources of intelligence. Information from a number of such sources can be combined into one number that summarizes the probability of the availability of each particular landing site.

## *E. Theoretical Properties*

*Theorem 1: The policy returned by our memoryless planner is guaranteed to be cost-minimal (optimal) if there exists an optimal policy that does not require the helicopter to sense a landing site without landing at it when it is found to be a good landing site.*

The expected cost of the PPCP policy after full convergence, is also upper-bounded by the optimal policy found by any memoryless planner, assuming that PPCP assumptions hold. This means that while being very efficient, our planner cannot be beaten (cost-wise) by any memoryless planner. If one or more memory slots are used in the PPCP algorithm, the guarantees are even better. For k memory slots, the upper bound drops to the optimal policy found by any planner that can remember the last k good landing sites.

## V. EXPERIMENTAL ANALYSIS

### *A. Simulation Setup*

In order to show the advantage of using PPCP to solve the landing site selection problem, we have generated test environments for varying number of possible landing sites. The transition costs and site probabilities were randomly generated as follows: The base of the helicopter is assumed at (0,0) and landing site x-y locations, are generated randomly on a square region 100 by 100 meters, centered at coordinates (1000,1000). Goal position is fixed in the middle of that 100 by 100 square. The $fly$ cost is then computed as the Euclidean distance for each possible pair of landing sites and the base. $Sense$ cost is assumed to be random on the range [10..100] and the $land$ costs are generated as uniformly random variables on the range [50..150], modeling different landing conditions. $Land2go$ cost is assumed proportional to the distance to the goal and the proportionality constant is set to a value of 5. This means that we are assuming that traveling on land is five times more expensive than flying. For the experiments with memory, the $quicksense$ action is assumed to be free and carries no cost. The probability of each site site is also assumed to be a uniformly random variable from 0 to 1.

A memoryless PPCP planner, adapted to this problem, has been implemented according to our formulation and description in [9]. In addition, we augmented the planner with an ability to turn on one memory slot for performance evaluation. This very small addition resulted in a fairly large implementation overhead in keeping track of the actions and outcomes. In addition, a very simple assumptive planner has also been implemented, which ignores the probability information and assumes that all the landing locations are good. Its policy always instructs the robot to go towards the landing site with minimum sum of the $fly$, $sense$, $land$, and $land2goal$ costs. When the helicopter actually gets to

| Number of sites | 0 mem slot (msec) | 1 mem slot (msec) | Belief state space size |
|---|---|---|---|
| 5 | 0.6± 0.3 | 1.7± 0.9 | $4.1 \times 10^3$ |
| 10 | 32± 36 | 205± 290 | $1.9 \times 10^6$ |
| 15 | 560± 1000 | 15000± 11000 | $6.7 \times 10^8$ |

Fig. 7. Average planning times and standard deviations for memoryless PPCP and PPCP with one memory slot. Also, the size of the full belief state space is given for reference. PPCP planner finds the solution by computing a tiny fraction of this space.

| # PPCP (expected) | PPCP (actual) | Assumptive Planner |
|---|---|---|
| 1839.05 | 1839.16 | 1947.25 |

Fig. 8. Summary of the results over 2500 runs with 10 landing sites, comparing the expected cost of the PPCP policy, actual cost of traversing PPCP policy, and actual cost of following an assumptive planner

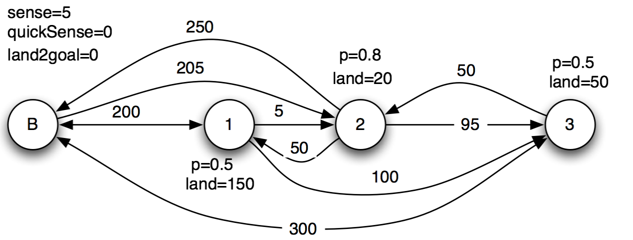


Fig. 9. Landing site selection scenario, where the optimal solution cannot be found without any memory slots. For this case, memoryless planner computes expected cost of 279, but adding one memory slot results in a lower cost of 278.5. The optimal policy is to go to Site1, sense it, and not land even if it is good. If it's good, then remember it and proceed to Site2. Sense Site2 and if it is good, land there. If Site2 is bad and Site1 was good, then it's less costly to go back and land there than trying Site3 (and possibly wasting more resources) because there is no need to sense Site1 again. Memoryless planner does not choose to go back to Site1, since the landing cost is large and it's probability is not very high.

the site, if it turns out bad, the same approach is used to find the next target location. This continues until a good land site is discovered, at which point, landing occurs.

In order to compare the performance of the two planners, for each random scenario, which also includes the probability of success for each site, we generate the ground truth according to those probabilities. We then evaluate each approach by simulating the result of the actions, with stochastic outcomes taking the value that is pre-determined by the ground truth. For PPCP, this means taking the final policy and evaluating it down the tree and accumulating cost until reaching the goal. All actions are now determined, since we generate and know the ground truth. In a similar manner, the simple planner is used to find the first good landing site and land. For each scenario, fifty random ground truth sets are evaluated in order to try different possible distributions of good landing sites.

## B. Simulation Results

**Search Performance**. The first set of results summarizes the planning times until full convergence for the memoryless and 1 memory slot operation. Table in Figure 7 summarizes the outcome of the experimental trials.

**Comparison with an assumptive planner**. Figure 8 summarizes the results after running 2500 planning scenarios. Even though the difference between PPCP and the assumptive planner is only 5%, the result is consistent with our expectations. For different scenarios, the difference may be much larger. In our case, most of the cost actually comes from the initial flight from the base to the first landing site. Therefore there is not much room for improvement if all the trajectories contain this high cost component.

**Memory Slot**. Remembering the preferred outcome is often not required for finding the optimal policy. However, we were able to construct an example in this context, for which allowing the planner to remember one previous preferred outcome (remembering one previous site which was sensed "good") actually improves expected cost of the policy. Consider the example shown in Figure 9. Note that the transitions are not symmetric - the cost of the return path may be different from the forward path due to various factors (i.e. wind direction). The difference in the expected costs of the two policies is quite small (0.5 out of 279) but this is just a result specific to this example - the main point is that memoryless planner would not find the optimal solution.

## C. Experimental Setup

**Quadrotor Helicopter Platform**. In order to demonstrate the real-world application of our work, we have designed and built an autonomous helicopter platform, shown in Figure 1(a). It is equipped with two complimentary
LIDAR sensors, which allow to perform true 3D localization and mapping. The aircraft is also capable of rotating the vertically-scanning LIDAR in the direction of interest. Ability to densely and accurately map the ground at the same time as the horizontal features is what distinguishes this platform from most similar aircrafts, carrying only a single LIDAR.

We have developed the software architecture all the way from state estimation, stabilization, and control to localization, mapping, terrain analysis, motion planning, and trajectory following (not discussed for the lack of space). In order to demonstrate the $Land2goal$ cost, a ground robot has been set up at the goal (shown in Figure 11(b)), which must drive to the actual landing location of the quad-rotor.

**Offline Computations**. A 3D map of the building was generated by manually flying the aircraft and postprocessing the collected data. Figure 10 shows the map which was provided to the PPCP planner to generate the flight policy and to a 4D helicopter motion planner to generate feasible collision-free trajectories. Landing costs and probabilities were picked to have reasonable values and $Fly$ and $Land2go$ cost were computed in the same way as in simulation. This prior information was provided to the robot, however the ground truth about the state of the landing sites was not disclosed. In fact, another quad-rotor helicopter was placed in the location of $Site1$, so that landing another aircraft is impossible.

## D. Experimental Results

Even though a prior map was generated, the autonomous run started off with an empty map. The prior information

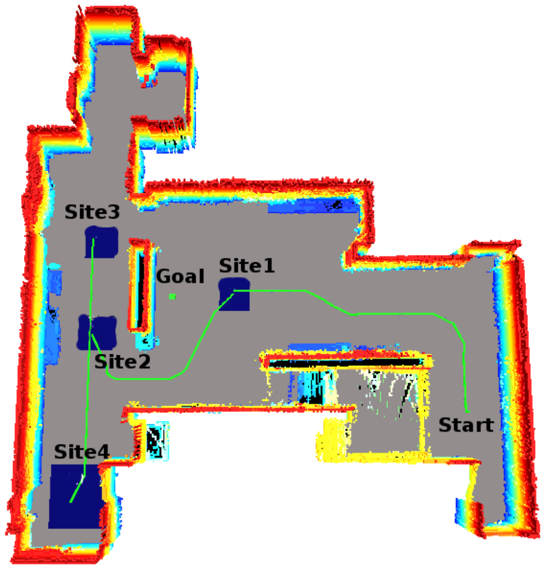
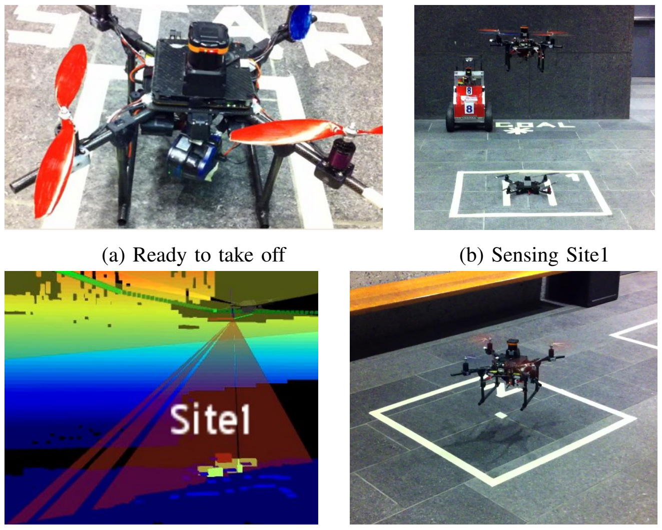


Fig. 10. Prior 3D map, landsites, and the PPCP policy (order = 1,2,3,4)



(a) Ready to take off (b) Sensing Site1

(c) Sensing Site1 (online 3D map) (d) Landing at Site2

Fig. 11. Stills from an autonomous run

was provided to the PPCP planner and the flight trajectories of the resulting policy were generated using a 4-dimensional (x,y,z,yaw) AD* Motion Planner [11] from the SBPL library [12], developped by our group as well.

The aircraft has autonomously navigated, according to the PPCP policy, from start to $Site1$, sensed an obstacle in the landing site and proceeded to $Site2$. After confirming that $Site2$ was clear, a landing command was issued. Figure 11(a) shows the quad-rotor a few seconds before start, (b,c) capture the aircraft in process of evaluating $Site1$ and (d) shows the robot just before landing on $Site2$

For our test configuration, the optimal policy was found in 0.2 milliseconds (2 milliseconds for 1 memory slot, but with the same result).

## VI. CONCLUSIONS

In this paper, we have studied one of the aspects of the aerial supply delivery problem. In particular, we have presented an approach to planning a sequence of flight and sense operations performed by UAV in order to select a landing site that minimizes the expected cost of flight, sense and access to the desired goal location. Our approach was based on showing that the problem exhibits clear preferences on uncertainty and therefore using PPCP algorithm to solve the problem by decomposing it into a series of deterministic graph searches that are fast-to-solve and require little memory. We describe the conditions under which the approach provides theoretical guarantees on the optimality of the returned policies. The experimental results in simulation show that our approach can solve problems with up to 15 possible landing sites in real-time while experiments on a physical quad-rotor we have recently built provide proof of concept. In addition, we are currently working on transitioning the software we have developed onto a larger (outdoor) unmanned aircraft, built by Dragonfly Pictures Inc, shown in Figure 1(b).

## VII. ACKNOWLEDGMENTS

This research was partially sponsored by the DARPA grant N10AP20011 and TATRC contract W81XWH-07-C-0043.

## REFERENCES

[1] S. Scherer, L. Chamberlain, and S. Singh, “Online assessment of landing sites,” in *AIAA*, 2010.
[2] N. Serrano, “A bayesian framework for landing site selection during autonomous spacecraft descent,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 5112–5117.
[3] T. Templeton, “Autonomous vision-based rotorcraft landing and accurate aerial terrain mapping in an unknown environment,” in *Technical Report University of California at Berkeley (UCB/EECS-2007-18)*, 2007.
[4] E. J. Sprinkle, J. and S. Sastry, “Deciding to land a uav safely in real time,” in *Proceedings of the American Control Conference*, vol. 5, 2005, p. 3506 3511.
[5] M. J. Johnson, A. and L. Matthies, “Vision guided landing of an autonomous helicopter in hazardous terrain,” in *Proceedings IEEE International Conference on Robotics and Automation*, 2005, p. 3966 3971.
[6] 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.
[7] I. Nourbakhsh and M. Genesereth, “Assumptive planning and execution: a simple, working robot architecture,” *Autonomous Robots Journal*, vol. 3, no. 1, pp. 49–67, 1996.
[8] A. Stentz, “The focussed D* algorithm for real-time replanning,” in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1995.
[9] 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.
[10] N. Nilsson, *Problem-Solving Methods in Artificial Intelligence*. McGraw-Hill, 1971.
[11] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, “Anytime search in dynamic graphs,” *Artificial Intelligence Journal (AIJ)*, vol. 172, no. 14, 2008.
[12] M. Likhachev, *SBPL graph search library*, http://www.cs.cmu.edu/∼maxim/software.html.