# A Switching Planner for Combined Task and Observation Planning

**Moritz Göbelbecker**
Albert-Ludwigs-Universität Freiburg, Germany
goebelbe@informatik.uni-freiburg.de

**Charles Gretton and Richard Dearden**
University of Birmingham, United Kingdom
{c.gretton,R.W.Dearden}@cs.bham.ac.uk

## Abstract

From an automated planning perspective the problem of practical mobile robot control in realistic environments poses many important and contrary challenges. On the one hand, the planning process must be lightweight, robust, and timely. Over the lifetime of the robot it must always respond quickly with new plans that accommodate exogenous events, changing objectives, and the underlying unpredictability of the environment. On the other hand, in order to promote efficient behaviours the planning process must perform computationally expensive reasoning about contingencies and possible revisions of subjective beliefs according to quantitatively modelled uncertainty in acting and sensing. Towards addressing these challenges, we develop a *continual planning* approach that *switches* between using a fast satisficing "classical" planner, to decide on the overall strategy, and decision-theoretic planning to solve small abstract subproblems where deeper consideration of the sensing model is both practical, and can significantly impact overall performance. We evaluate our approach in large problems from a realistic robot exploration domain.

## Introduction

A number of recent integrated robotic systems incorporate a high-level *continual planning* and execution monitoring subsystem [Wyatt *et al.*, 2010; Talamadupula *et al.*, 2010; Kraft *et al.*, 2008]. For the purpose of planning, sensing is modelled deterministically, and beliefs about the underlying state are modelled qualitatively. Both Talamadupula *et al.* and Wyatt *et al.* identify continual planning with probabilistic models of noisy sensing and state as an important challenge for future research. Motivating that sentiment, planning according to accurate stochastic models should yield more efficient and robust deliberations. In essence, the challenge is to develop a planner that exhibits speed and scalability similar to planners employed in existing robotic systems—e.g., Wyatt *et al.* use a satisficing *classical* procedure—and which is also able to synthesise relatively efficient deliberations according to detailed probabilistic models of the environment.

This paper describes a *switching* domain independent planning approach we have developed to address this challenge. Our Planner is *continual* in the usual sense that plans are adapted and rebuilt online in reaction to changes to the model of the underlying problem and/or domain—e.g., when goals are modified, or when the topological map is altered by a door being closed. It is integrated on a mobile robot platform that continuously deliberates in a stochastic dynamic environment in order to achieve goals set by the user, and acquire knowledge about its surroundings. Our planner takes problem and domain descriptions expressed in a novel extension of PPDDL [Younes *et al.*, 2005], called *Decision-Theoretic (DT)PDDL*, for modelling stochastic decision problems that feature partial observability. In this paper we restrict our attention to problem models that correspond to deterministic-action goal-oriented POMDPs in which all actions have non-zero cost, and where an optimal policy can be formatted as a finite horizon contingent plan. Moreover, we target problems of a size and complexity that is challenging to state-of-the-art sequential satisficing planners, and which are too large to be solved directly by decision-theoretic (DT) systems.

Our planner *switches*, in the sense that the base planning procedure changes depending on our robot's subjective degrees of belief, and on progress in plan execution. When the underlying planner is a fast (satisficing) *classical* planner, we say planning is in a *sequential* session, and otherwise it is in a DT session. A *sequential* session plans, and then pursues a high-level strategy—e.g., go to the kitchen bench, and then observe the cornflakes on it. A DT session proceeds in a practically sized *abstract* process, determined according to the current sequential strategy and underlying belief-state.

We evaluate our approach in simulation on problems posed by object search and room categorisation tasks that our indoor robot undertakes. Those feature a deterministic task planning aspect with an active sensing problem. The larger of these problems features 6 rooms, 25 topological places, and 21 active sensing actions. The corresponding decision process has a number of states exceeding $10^{36}$, and high-quality plans require very long planning horizons. Although our approach is not optimal, particularly as it relies on the results of satisficing sequential planning directly, we find that it does nevertheless perform better than a purely sequential replanning baseline. Moreover, it is fast enough to be used for real-time decision making on a mobile robot.

## Planning Language and Notations

We give an overview of the declarative first-order language DTPDDL, an extension of PPDDL that can express probabilistic models of the sensing consequences of acting, to quantitatively capture unreliability in perception. There are straightforward compilations from problems expressed in DTPDDL to flat (and factored) representations of the underlying decision process. Although similar to Bryce's POND input language, DTPDDL distinguishes itself by explicitly treating state and perceptual symbols separately, and by providing distinct declarations for operators (i.e, state model) and senses (i.e., observation model). In this last respect, DTPDDL admits more compact domain descriptions where sensing effects are common across multiple operators. In detail, DTPDDL has perceptual analogues of fluent and predicate symbols. For example, a simple *object search* domain would have:

```
(:functions
 (is-in ?v - visual-object) - location )
(:perceptual-functions
 (o-is-in ?v - visual-object) - location )
```

Where the first fluent symbol models the actual location of objects, and the second the instantaneous sensing of objects following application of an action with sensing consequences. To model sensing capabilities, we have operator-like "sense" declarations, with preconditions expressed using state and action symbols, and uniformly positive effects over perceptual symbols. For example, where *look-for-object* is the operator that applies an object detection algorithm at a specific place, an *object search* task will have:

```
(:sense vision :parameters
 (?r -robot ?v -visual-object ?l -location)
 :execution (look-for-object ?r ?v ?l)
 :precondition (and (= (is-in ?r) ?l) )
 :effect (and
  (when (= (is-in ?v) ?l)
   (probabilistic .8 (= (o-is-in ?v) ?l)))
  (when (not (= (is-in ?v) ?l))
   (probabilistic .1 (= (o-is-in ?v) ?l)))))
```

I.e., there is a 10% *false positive* rate, and 20% probability of a *false negative*. This representation allows us to represent actions that have multiple independent observational effects.

We now review the DTPDDL syntax for describing an initial state distribution, taken verbatim from PPDDL, in order to aid us in discussing our planner. That distribution is expressed in a tree-like structure of terms. Each term is either: (1) atomic, e.g., a state proposition such as $(= \ (\texttt{is-in box}) \ \texttt{office})$, (2) probabilistic, e.g., $(\texttt{probabilistic} \ \rho_1(T_1)..\rho_n(T_n))$ where $T_i$ are conjunctive, or (3) a conjunct over probabilistic and atomic terms. The root term is always conjunctive, and the leaves are atomic. For example, a simplified object search could have:[1]

```
(:init (= (is-in R2D2) kitchen)
 (probabilistic .8 (= (is-in box) kitchen)
                .2 (= (is-in box) office))
 (probabilistic .3 (= (is-in cup) office)
                .7 (= (is-in cup) kitchen)))
```

[1]In PDDL, $(\texttt{:init} \ T_1..T_n)$ expresses the conjunctive root of the tree – i.e., the root node $(\texttt{and} \ T_1..T_n)$. Also, we shall write $p$, rather than $(\texttt{and} \ p)$, for conjunctive terms that contain a single atomic subterm.

The interpretation is given by a *visitation* of terms: An atom is *visited* iff its conjunctive parent is visited, and a conjunctive term is visited iff all its immediate subterms are visited. A probabilistic term is visited iff its conjunctive parent is visited, and exactly one of its subterms, $T_i$, is visited. Each visitation of the root term according to this recursive definition defines a starting state, along with the probability that it occurs. The former corresponds to the union of all visited atoms, and the latter corresponds to the product of $\rho_i$ entries on the visited subterms of probabilistic elements. Making this concrete, the above example yields the following flat distribution:

| Probability | (is-in R2D2) | (is-in box) | (is-in cup) |
|---|---|---|---|
| .24 | kitchen | kitchen | office |
| .06 | kitchen | office | office |
| .56 | kitchen | kitchen | kitchen |
| .14 | kitchen | office | kitchen |

## Switching Continual Planner

We now describe our *switching* planning system that operates according to the continual planning paradigm. The system *switches* in the sense that planning and plan execution proceed in interleaved sessions in which the *base planner* is either *sequential* or *decision-theoretic*. The first session is sequential, and begins when a DTPDDL description of the current problem and domain are posted to the system. During a sequential session a serial plan is computed that corresponds to one execution-*trace* in the underlying decision-process. That trace is a reward-giving sequence of process actions and *assumptive* actions. Each *assumptive* action corresponds to an assertion about some facts that are unknown at plan time – e.g. that a box of cornflakes is located on the corner bench in the kitchen. The trace specifies a plan and characterises a *deterministic approximation* (see [Yoon *et al.*, 2008]) of the underlying process in which that plan is valuable. Traces are computed by a cost-optimising classical planner which trades off action costs, goal rewards, and determinacy. Execution of a trace proceeds according to the process actions in the order that they appear in the trace. If, according to the underlying belief-state, the outcome of the next action scheduled for execution is not predetermined above a threshold (here 95%), then the system switches to a DT session.

Because online DT planning is impractical for the size of problem we are interested in, DT sessions plan in a small abstract problem defined in terms of the trace from the proceeding sequential session. This abstract state-space is characterised by a limited number of propositions, chosen because they relate evidence about assumptions in the trace. To allow the DT planner to judge assumptions from the trace, we add *disconfirm* and *confirm* actions to the problem for each of them. Those yield a relatively small reward/penalty if the corresponding judgement is true/false. If a judgement action is scheduled for execution, then the DT session is terminated, and a new sequential session begins.

Whatever the session type, our continual planner maintains a factored representation of successive belief-states. As an internal representation of the $(\texttt{:init})$ declaration, we keep a tree-shaped Bayesian network which gets updated whenever an action is performed, or an observation received. That

belief-state representation is used: (1) as the source of candidate determinisations for sequential planning, (2) in determining when to switch to a DT session, and (3) as a mechanism to guide construction of an abstract process for DT sessions.

## Sequential Sessions

As we only consider deterministic-action POMDPs, all state uncertainty is expressed in the (:init) declaration. This declaration is used by our approach to define the starting state for sequential sessions, and the set of assumptive actions available to sequential planning. Without a loss of generality we also suppose that actions do not have negative preconditions. For a sequential session the starting state corresponds to the set of facts that are true with probability 1. Continuing our example, that starting state is the singleton:

$$s_0 \equiv \{(= \texttt{(is-in R2D2) kitchen})\}.$$

To represent state assumptions we augment the problem posed during a sequential session with an *assumptive action* $\mathcal{A}^{\circ}(\rho_i; T_i)$ for each element, $\rho_i(T_i)$, of each probabilistic term from (:init). Here, $\mathcal{A}^{\circ}(\rho_i; T_i)$ can be executed if no $\mathcal{A}^{\circ}(\rho_j; T_j)$, $j \neq i$, has been executed from the same probabilistic term, and, either (probabilistic ..$\rho_i$ ($T_i$)..) is in the root conjunct, or it occurs in $T_k$ for some executed $\mathcal{A}^{\circ}(\rho_k; T_k)$. We also add constraints that forbid scheduling of assumptions about facts after actions with preconditions or effects that mention those facts. For example, the robot cannot assume it is plugged into a power source immediately after it unplugs itself. Executing $\mathcal{A}^{\circ}(\rho_i; T_i)$ in a state $s$ effects a transition to a successor state $s^{T_i}$, the union of $s$ with atomic terms from $T_i$, and of course annotated with auxiliary variables that track the applicability of assumptive actions. For example, consider the following sequential plan:

$\mathcal{A}^{\circ}(.8; (= \texttt{(is-in box) kitchen}))$;
$\mathcal{A}^{\circ}(.3; (= \texttt{(is-in cup) office}))$;
(look box kitchen); (look cup office);
(report box kitchen); (report cup office)

Applying the first action in $s_0$ yields a state in which the following facts are true:

$\{(= \texttt{(is-in R2D2) kitchen}), (= \texttt{(is-in box) kitchen})\}$

In the underlying belief-state, this is true with probability $0.8$. The assumed state before the scheduled execution of action (look box kitchen) is:

$\{(= \texttt{(is-in R2D2) kitchen}), (= \texttt{(is-in box) kitchen}),$
$(= \texttt{(is-in cup) office})\}$

Which is actually true with probability $0.24$ according to the underlying belief.

To describe the optimisation criteria used during sequential sessions we model $\mathcal{A}^{\circ}(\rho_i; T_i)$ probabilistically, supposing that its application in state $s$ effects a transition to $s^{T_i}$ with probability $\rho_i$, and to $s^{\perp}$ with probability $1 - \rho_i$. State $s^{\perp}$ is an added sink. Taking $\rho_i$ to be the probability that the $i^{th}$ sequenced action, $a_i$, from a trace of state-action pairs $\langle s_0, a_0, s_1, a_1, .., s_N \rangle$ does not transition to $s^{\perp}$, then the optimal sequential plan has value:

$$V^* = \max_N \max_{s_0, a_0, .., s_N} \prod_{i=1..N-1} \rho_i \sum_{i=1..N-1} \mathsf{R}(s_i, a_i),$$

## DT Sessions

When an action is scheduled whose outcome is uncertain according to the underlying belief-state, the planner switches to a DT session. That plans for *small* abstract processes defined according to the action that triggered the DT session, the assumptive actions in the proceeding trace, and the current belief-state. Targeted sensing is encouraged by augmenting the reward model to reflect a heuristic value of knowing the truth about assumptions. In detail, all rewards from the underlying problem are retained. Additionally, for each relevant assumptive action $\mathcal{A}^{\circ}(\rho_i; T_i)$ in the current trace, we have a *disconfirm action* $\mathcal{A}^{\bullet}(\rho_i; T_i)$ so that for all states $s$:

$$\mathsf{R}(s, \mathcal{A}^{\bullet}(\rho_i; T_i)) = \begin{cases} \$(T_i) & \texttt{if } T_i \not\subseteq s \\ \hat{\$}(T_i) & \texttt{otherwise} \end{cases}$$

where $\$(T_i)$ (resp. $\hat{\$}(T_i)$) is a small positive (negative) numeric quantity which captures the utility the agent receives for correctly (incorrectly) rejecting an assumption. In terms of action physics, a disconfirm action can only be executed once, and otherwise is modelled as a self-transformation. We only consider *relevant* assumptions when constructing the abstract model. If $\tilde{a}$ is the action that switched the system to a DT session, then an assumption $\mathcal{A}^{\circ}(\rho_i; T_i)$ is *relevant* if it is necessary for the outcome of $\tilde{a}$ to be determined. For example, taking the switching action $\tilde{a}$ to be (look box kitchen) from our earlier sequential plan example, we have that $\mathcal{A}^{\circ}(.3; (= \texttt{(is-in cup)office}))$ is not relevant, and therefore we exclude the corresponding disconfirm action from the abstract decision process. Given $\tilde{a}$, we also include another once-only self-transition action $\mathcal{A}.\texttt{pre}(\tilde{a})$, a *confirmation action* with the reward property:

$$\mathsf{R}(s, \mathcal{A}.\texttt{pre}(\tilde{a})) = \begin{cases} \$(\texttt{pre}(\tilde{a})) & \texttt{if } \texttt{pre}(\tilde{a}) \subseteq s \\ \hat{\$}(\texttt{pre}(\tilde{a})) & \texttt{otherwise} \end{cases}$$

Here, $\texttt{pre}(\tilde{a})$ is the set of propositions that are the precondition of action $\tilde{a}$. Execution of either a disconfirmation or the confirmation action returns control to a sequential session, which starts anew from the underlying belief-state.

Turning to the detail of (dis-)confirmation rewards, in our integrated system these are sourced from a motivational subsystem. In this paper, for $\mathcal{A}^{\bullet}(\rho_i; T_i)$ actions we set $\$(x)$ to be a small positive constant, and have $\hat{\$}(x) = -\$(x)(1-\rho)/\rho$ where $\rho$ is the probability that $x$ is true. For $\mathcal{A}.\texttt{pre}(\tilde{a})$ actions we have $\hat{\$}(x) = -\$(x)\rho/(1-\rho)$.

In order to guarantee fast DT sessions, those plan in an abstract process determined by the current trace and underlying belief-state. The abstract process posed to the DT planner is constructed by first constraining as statically false all propositions except those which are true with probability 1, or which are the subject of *relevant* assumptions. For example, taking the above trace with assumptive action probabilities changed to reflect the belief-state in Fig. 1B, given switching action "(look box kitchen)" the underlying belief in Fig. 1B would determine a fully constrained belief given by Fig. 1A. Next, static constraints are removed, one proposition at a time, until the number of states that can be true with non-zero probability in the initial belief of the abstract process reaches a given threshold. In detail, for each statically-false proposition we

(A) Fully constrained belief

```
(:init (=(is-in R2D2)kitchen)
 (.6(=(is-in box)kitchen)))
```

(B) Underlying DTPDDL belief

```
(:init (=(is-in R2D2)kitchen)
 (.6(and(=(is-in box)kitchen)
      (.9(=(is-in milk)kitchen))
       .1(=(is-in milk)office))
  .4(and(=(is-in box)office)
      (.1(=(is-in milk)kitchen))
       .9(=(is-in milk)office)))
 (.6(=(is-in cup)office)
  .4(=(is-in cup)kitchen)))
```

(C) Partially constrained belief

```
(:init (=(is-in R2D2)kitchen)
 (.6(and(=(is-in box)kitchen)
      (.9(=(is-in milk) kitchen))
       .1(=(is-in milk)office))
  .4(and(=(is-in box)office)
      (.1(=(is-in milk)kitchen))
       .9(=(is-in milk)office)))
```

Figure 1: Simplified examples of belief-states from DT sessions.

compute the *entropy* of the relevant assumptions of the current trace *conditional* on that proposition. Let $X$ be a set of propositions and $2^X$ the powerset of $X$, then taking

$$\chi = \{ \bigwedge_{x \in X' \cap X} x \ \wedge \bigwedge_{x \in X \setminus X'} \neg x \mid X' \in 2^X \},$$

we have that $\chi$ is a set of conjunctions each of which corresponds to one truth assignment to elements in $X$. Where $p(\phi)$ gives the probability that a conjunction $\phi$ holds in the belief-state of the DTPDDL process, the entropy of $X$ *conditional* on a proposition $y$, written $\mathrm{H}(X|y)$, is given by Eq. 1.

$$\mathrm{H}(X|y) = \sum_{x \in \chi, y' \in \{y, \neg y\}} p(x \wedge y') \log_2 \frac{p(y')}{p(x \wedge y')} \qquad (1)$$

A low $\mathrm{H}(X|y)$ value suggests that knowing the truth value of $y$ is useful for determining whether or not some assumptions $X$ hold. When removing a static constraint on propositions during the abstract process construction, $y_i$ is considered before $y_j$ if $\mathrm{H}(X|y_i) < \mathrm{H}(X|y_j)$. For example, if the serial plan assumes the box is in the kitchen, then propositions about the contents of kitchens containing a box, e.g. $(=\,(\texttt{is-in milk})\texttt{kitchen})$, are added to characterise the abstract process' states. Taking a relevant assumption $X$ to be $(=\,(\texttt{is-in box})\texttt{kitchen})$, in relaxing static constraints the following entropies are calculated:

```
.47 = H(X|(=(is-in milk)office))
    = H(X|(=(is-in milk)kitchen))
.97 = H(X|(=(is-in cup)office))
    = H(X|(=(is-in cup)kitchen))
```

Therefore, the first static constraint to be relaxed is for `(=(is-in milk)office)`, or equivalently `(=(is-in milk)kitchen)`, giving a refined abstract belief state depicted in Fig. 1C. Summarising, if for Fig.1B the DT session is restricted to belief-states with fewer than 8 elements, then the starting belief-state of the DT session does not mention a "cup".

## Evaluation

We have implemented our switching approach in the MAPSIM environment [Brenner and Nebel, 2009], using DLIB-ML [King, 2009] for belief revision. Sequential sessions use a modified version of Fast Downward [Helmert, 2006], and DT sessions use our own contingent procedure. Since most of the problems we consider are much larger than any available DT planner can solve directly, for comparison purposes we also implemented a simple dual-mode replanning baseline approach. Here, when a switching action is scheduled for execution the DT session plans to a single entropy reduction action, whose execution can provide evidence regarding the truth value of a relevant assumption. Control is then immediately returned to a new sequential session.

We evaluate our approaches in robot exploration tasks from home and office environments. Spatially, these consist of *rooms* (office/kitchen/etc), and an underlying topological map over smaller areas of space, called *places*, and connectivity between those. The mobile *robot* and *visual objects* inhabit the topological places. Objects indicate the category of space they inhabit—e.g., spoons are likely to be in kitchens. By examining view cones at places for particular objects, the robot is able to: (1) categorise space at high (room) and low (place) levels, and (2) find objects for the user, exploiting information about object co-occurrence and room categories for efficiency. Also, in the presence of a person, the robot can ask about the category of the current room.

We compare switching to the baseline in several realistic tasks, with the number of rooms ranging from 3 (12-places, 16-objects, $|\text{states}| > 10^{21}$) to 6 (26-places, 21-objects, $|\text{states}| > 10^{36}$). We also compare those systems with near optimal policies computed using Smith's ZMDP for small 2 room problems (4-places, 3-objects, $|\text{states}| \simeq 5000$). Our evaluation considers 3 levels of reliability in sensing: *reliable* sensors have a .1 probability of a false negative, *semi-reliable* have a chance of 0.3 of false negative and 0.1 of false positive, and *noisy* sensors with probabilities of 0.5 and 0.2 respectively. Each object class is assigned one sensor model—e.g. cornflakes may be harder to detect than refrigerators. We performed several experiments with different levels of reliability for sensing the target object(s), while keeping sensing models for non-target objects constant.

Our evaluation examines DT sessions with initial belief-states admitting between 20 and 100 abstract states with non-zero probability. We run 50 simulations in each configuration, and have a timeout on each simulation of 30 minutes (1800 seconds)[2]. The continual planning times are reported in Figure 2, and the quality data in Figure 3. For each task, the goal is to find one or more objects and report their position to a user. Usually there is a non-zero probability that no plan exists, as the desired object might not be present in the environment. In these experiments we only allocate reward on the achievement of all goals, therefore we find it intuitive to report average plan costs and the success rates in problems that admit a complete solution (i.e., positive reward scaled by a constant factor). The exception occurs for items f and g of Figure 3, where we report expected discounted rewards (not plan costs).

We find that if sensing is reliable, then little is gained using DT sessions, as the greedy approach of the baseline is sufficient. As sensing degrades DT sessions prove more useful.

[2]All experiments were conducted on a 2.66GHz Intel Xeon X5355 using one CPU core.

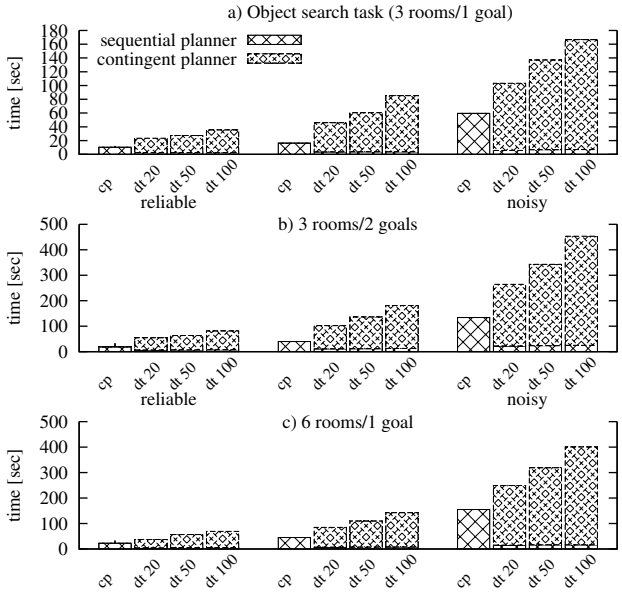


Figure 2: Average runtime

Here, time spent on DT planning increases steeply as the abstraction becomes more refined, which is compensated for by fewer planning sessions overall. More detailed abstractions lead to a better overall success rate, particularly for tasks $d$ and $e$. Speaking to the effectiveness of our entropy heuristic for abstraction refinement, we see relatively high success rates irrespective of the level of refinement. Comparing finally to the best ZMDP policy, although producing relatively costly plans, the continual planners performed quite well, especially in terms success rate. A key source of inefficiency here, is due to sequential sessions always being optimistic, and refusing to abandon the search.

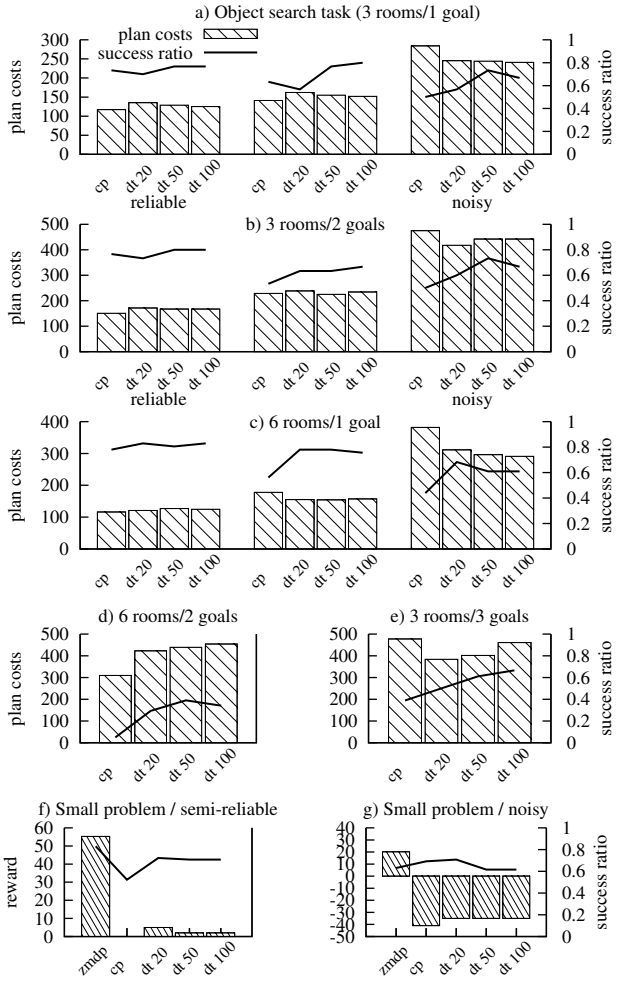


Figure 3: Average plan costs and number of successful runs.

## Related Work

Addressing task and observation planning specifically, there have been a number of recent developments where the underlying problem is modelled as a POMDP. For vision algorithm selection, Sridharan *et al.* (2010) exploit an explicitly modelled hierarchical decomposition of the underlying POMDP. Doshi and Roy (2008) represent a preference elicitation problem as a POMDP and take advantage of symmetry in the belief-space to exponentially shrink the state-space. Although we have been actively exploring the Doshi and Roy approach, those exploitable symmetries are not present in problems we consider due to the task planning requirement. Also, our approach is in a similar vein to *dual-mode* control [Cassandra *et al.*, 1996], where planning switches between entropy and utility focuses.

There has also been much recent work on scaling offline approximate POMDP solution procedures to medium-sized instances. Recent contributions propose more efficient belief-point sampling schemes [Kurniawati *et al.*, 2010; Shani *et al.*, 2008b], and factored representations with procedures that can efficiently exploit structures in those representations [Brunskill and Russell, 2010; Shani *et al.*, 2008a]. Offline domain independent systems scale to *logistics* problems with $2^{22}$ states [Shani *et al.*, 2008a], taking over an hour to converge, and around 10 seconds on average to perform each Bellman backup. Brunskill and Russell are able to solve problems with approximately $10^{30}$ states, by further exploiting certain problem features – E.g., problems where no actions have negative effects. Moving someway towards supporting real-time decision making, recent online POMDP solution procedures have been developed which leverage highly approximate value functions—computed using an offline procedure—and heuristics in forward search [Ross *et al.*, 2008]. These approaches are applicable in relatively small problems, and can require expensive *problem-specific* offline processing in order to yield good behaviours.

In the direction of leveraging *classical* systems/approaches for planning under uncertainty, the most highlighted system to date has been FFR$_a$ [Yoon *et al.*, 2007]; The winning entry from the probabilistic track of the 2004 International Planning Competition. In the continual paradigm, FFR$_a$ uses FF to compute sequential plans and execution traces. More computationally expensive approaches in this vein combine sampling strategies on valuations over *runtime variables* with deterministic planning procedures [Yoon *et al.*, 2008].

Also leveraging deterministic planners in problems that feature uncertainty, CONFORMANT-FF [Hoffmann and Brafman, 2006] and $T_0$ [Palacios and Geffner, 2009] demonstrate how conformant planning —i.e., sequential planning in unobservable worlds— can be modelled as a deterministic problem, and therefore solved using sequential systems. In this conformant setting, advances have been towards compact representations of beliefs amenable to existing best-first search planning procedures, and lazy evaluations of beliefs. We consider it an appealing future direction to pursue conformant reasoning during the sequential sessions we proposed. Most recently this research thread has been extended to contingent planning in fully observable non-deterministic environments [Albore *et al.*, 2009].

## Concluding Remarks

We have addressed a key challenge, specifically that of high-level continual planning for efficient deliberations according to rich probabilistic models afforded by recent integrated robotic systems. We developed a system that can plan quickly given large realistic probabilistic models, by switching between: (a) fast sequential planning, and (b) expensive DT planning in small abstractions of the problem at hand. Sequential and DT planning is interleaved, the former identifying a rewarding sequential plan for the underlying process, and the latter solving small sensing problems posed during sequential plan execution. We have evaluated our system in large real-world task and observation planning problems, finding that it performs quickly and relatively efficiently.

## References


[Albore *et al.*, 2009] Alexandre Albore, Héctor Palacios, and Héctor Geffner. A translation-based approach to contingent planning. In *IJCAI*, pages 1623–1628, 2009.

[Brenner and Nebel, 2009] Michael Brenner and Bernhard Nebel. Continual planning and acting in dynamic multiagent environments. *Journal of Autonomous Agents and Multiagent Systems*, 19(3):297–331, 2009.

[Brunskill and Russell, 2010] E. Brunskill and S. Russell. RAPID: A reachable anytime planner for imprecisely-sensed domains. In *UAI*, 2010.

[Cassandra *et al.*, 1996] Anthony R. Cassandra, Leslie Pack Kaelbling, and James A. Kurien. Acting under uncertainty: Discrete bayesian models for mobile-robot navigation. In *IROS*, pages 963–972, 1996.

[Doshi and Roy, 2008] Finale Doshi and Nicholas Roy. The permutable POMDP: Fast solutions to POMDPs for preference elicitation. In *AAMAS*, 2008.

[Helmert, 2006] Malte Helmert. The Fast Downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.

[Hoffmann and Brafman, 2006] Jörg Hoffmann and Ronen I. Brafman. Conformant planning via heuristic forward search: a new approach. *Artif. Intell.*, 170:507–541, May 2006.

[King, 2009] Davis E. King. Dlib-ml: A machine learning toolkit. *J. Mach. Learn. Res. (JMLR)*, 10:1755–1758, December 2009.

[Kraft *et al.*, 2008] D. Kraft, E. Başeski, M. Popović, A. M. Batog, A. Kjær-Nielsen, N. Krüger, R. Petrick, C. Geib, N. Pugeault, M. Steedman, T. Asfour, R. Dillmann, S. Kalkan, F. Wörgötter, B. Hommel, R. Detry, and J. Piater. Exploration and planning in a three-level cognitive architecture. In *CogSys*, 2008.

[Kurniawati *et al.*, 2010] Hanna Kurniawati, Yanzhu Du, David Hsu, and Wee Sun Lee. Motion planning under uncertainty for robotic tasks with long time horizons. *The International Journal of Robotics Research*, 2010.

[Palacios and Geffner, 2009] Hector Palacios and Hector Geffner. Compiling uncertainty away in conformant planning problems with bounded width. *J. Artif. Intell. Res. (JAIR)*, 35:623–675, August 2009.

[Ross *et al.*, 2008] S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa. Online planning algorithms for POMDPs. *J. Artif. Int. Res. (JAIR)*, 32:663–704, July 2008.

[Shani *et al.*, 2008a] G. Shani, P. Poupart, R. Brafman, and S. E. Shimony. Efficient add operations for point-based algorithms. In *ICAPS*, pages 330–337, 2008.

[Shani *et al.*, 2008b] Guy Shani, Ronen I. Brafman, and Solomon Eyal Shimony. Prioritizing point-based pomdp solvers. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 38(6):1592–1605, 2008.

[Sridharan *et al.*, 2010] Mohan Sridharan, Jeremy Wyatt, and Richard Dearden. Planning to see: Hierarchical POMDPs for planning visual actions on a robot. *Artif. Intell.*, 174(11):704–725, 2010.

[Talamadupula *et al.*, 2010] K. Talamadupula, J. Benton, S. Kambhampati, P. Schermerhorn, and M. Scheutz. Planning for human-robot teaming in open worlds. *ACM Trans. Intell. Syst. Technol.*, 1:14:1–14:24, December 2010.

[Wyatt *et al.*, 2010] Jeremy L. Wyatt, Alper Aydemir, Michael Brenner, Marc Hanheide, Nick Hawes, Patric Jensfelt, Matej Kristan, Geert-Jan M. Kruijff, Pierre Lison, Andrzej Pronobis, Kristoffer Sjöö, Danijel Skočaj, Alen Vrečko, Hendrik Zender, and Michael Zillich. Self-understanding and self-extension: A systems and representational approach. *IEEE Transactions on Autonomous Mental Development*, 2(4):282 – 303, December 2010.

[Yoon *et al.*, 2007] Sung Wook Yoon, Alan Fern, and Robert Givan. FF-replan: A baseline for probabilistic planning. In *ICAPS*, pages 352–, 2007.

[Yoon *et al.*, 2008] Sungwook Yoon, Alan Fern, Robert Givan, and Subbarao Kambhampati. Probabilistic planning via determinization in hindsight. In *AAAI*, pages 1010–1016, 2008.

[Younes *et al.*, 2005] Håkan L. S. Younes, Michael L. Littman, David Weissman, and John Asmuth. The first probabilistic track of the international planning competition. *J. Artif. Intell. Res. (JAIR)*, 24:851–887, 2005.