Online Goal Recognition Combining Landmarks and Planning

Mor Vered†, Ramon Fraga Pereira‡, Maurício Cecílio Magnaguagno‡,
Gal A. Kaminka† and **Felipe Meneguzzi‡**
†Bar-Ilan University, Israel
‡Pontifical Catholic University of Rio Grande do Sul, Brazil

Abstract

Goal recognition is the problem of recognizing the goal of an agent based on an incomplete sequence of observations. Recognizing goals with minimal domain knowledge as an agent executes its plan requires efficient algorithms to sift through a large space of hypotheses. In this paper, we develop an online approach to recognize goals in both continuous and discrete domains using a combination of goal mirroring and a generalized notion of landmarks adapted from the planning literature. Extensive experiments demonstrate the approach is more efficient than state-of-the-art online recognition, and substantially more accurate.

1 Introduction

Goal recognition is the problem of recognizing the goal of an agent based on an incomplete sequence of observations. Real-world applications include human-robot interaction [Wang *et al.*, 2013], intelligent user interfaces [Blaylock and Allen, 2004; Hong, 2001], and recognizing navigation goals [Liao *et al.*, 2007]. Most approaches to goal recognition rely on a plan library describing the plans assumed known by the agent being observed to achieve its goals [Sukthankar *et al.*, 2014]. While these approaches can be computationally efficient, they require substantial domain knowledge, and make strong assumptions about the preferences of observed agents. A different approach is *plan recognition as planning* [Ramírez and Geffner, 2009; Sohrabi *et al.*, 2016] whereby a planner is used in the recognition process to generate recognition hypotheses as needed, eliminating the need for a plan library. This approach has shown that it is possible to perform effective plan and goal recognition using only a domain-theory describing actions in the environment as domain knowledge. However, such approaches are computationally expensive as they require multiple executions of a planning algorithm to compute alternative ways in which the observed agent can achieve a goal. Furthermore, most existing approaches assume all observations, even if noisy or incomplete, are received at once (*offline*) at the end of their execution [Pereira and Meneguzzi, 2016; Pereira *et al.*, 2017b]. This assumption fails in many realistic environments, where one must recognize goals *online*. In online recognition, observations are provided incrementally, and the objective is to recognize the goal as soon as possible, without knowledge that the final observation is received.

In this paper, we develop an efficient approach for online goal recognition as planning that generalizes over both discrete (STRIPS style) and continuous (navigation) domains. Our approach achieves substantial runtime efficiency by reducing the complexity of the problems sent to an underlying planning algorithm using an online goal mirroring technique [Vered *et al.*, 2016] and minimizing the number of goal hypotheses to be computed using landmarks once during runtime, and a landmark-based heuristic [Pereira and Meneguzzi, 2016; Pereira *et al.*, 2017a]. To achieve the generalization over discrete and continuous domains we adapt the notion of planning landmarks [Hoffmann *et al.*, 2004] to comprise its original planning semantics as well as continuous spatial domains and develop a new and efficient algorithm to generate hierarchically ordered spatial landmarks. Since our approach can use any type of PDDL [McDermott *et al.*, 1998] planning algorithm or path planner [Sucan *et al.*, 2012], we can leverage current and future advances in efficiency in such algorithms.

This paper makes three key contributions: (a) a novel goal recognition approach, for both discrete and continuous domains; (b) an online approach to efficiently recognize goals early in the observed agent's plan execution; (c) a novel notion of landmarks that covers discrete and continuous domains and an algorithm to generate such landmarks. We evaluate the resulting approach empirically over hundreds of recognition problems in classical and motion planning domains. The results demonstrate superior efficiency and generally superior recognition quality over the state of the art.

2 Background and Related Work

Library-based goal recognition assumes the existence of a library of plans leading to known goals. Such methods include probabilistic inference [Bui, 2003; Avrahami-Zilberbrand and Kaminka, 2007], grammar-based approaches [Pynadath and Wellman, 2000; Geib and Goldman, 2009; Sadeghipour and Kopp, 2011; Geib, 2015; Mirsky and Gal, 2016], and others (see [Sukthankar *et al.*, 2014] for a larger enumeration). While often efficient, these methods are limited to recognizing goals for which plans are a-priori known, and encoded in the plan library. Nonetheless, the majority of these methods

can be used for both online and offline recognition, whereby observations may be incrementally revealed or given a-priori to the recognition process.

Plan recognition based on domain-theory removes the reliance on a plan-library, instead using the domain description in the recognition process. For example, *plan recognition as planning* (PRP) [Ramírez and Geffner, 2009; Sohrabi *et al.*, 2016] uses a planner to generate plan hypotheses dynamically, based on the domain-theory and the observations. A more efficient approach [Pereira and Meneguzzi, 2016; Pereira *et al.*, 2017b] avoids planning altogether, instead generating planning landmarks from the domain theory prior to recognition. Such landmarks are actions (or state properties) that *must* be included in plans that achieve specific goals [Hoffmann *et al.*, 2004]. When observed, landmarks provide strong evidence for recognizing these goals (indeed, we use them here), and all of these methods work offline.

Some domain-theory methods address online recognition. Early seminal work by Hong [2001] uses a *goal graph* representation for online goal recognition, constructed from a domain theory and incoming observations; recognized goals are not probabilistically ranked. In contrast, Baker et.al [2005] present a bayesian framework to calculate goal likelihoods by marginalizing over possible actions and generating state transitions. Martin et al. [2015] take an extreme approach, investing in significant offline computation to completely eliminate online planner calls by pre-computing cost estimates.

In contrast, Vered et al. [2016] present an online PRP algorithm for continuous spaces, using off-the-shelf motion planners to estimate goal likelihoods. Like other PRP methods, they repeatedly call on a planner. We generalize their algorithm to also work in discrete domains, and show how to heuristically use landmarks, computed only once at run-time, to reduce the number of planner calls and improve accuracy. To do this, we generalize the notion of landmarks to motion planning in continuous spaces, and show how to use landmarks in an online fashion.

3 Combining Landmarks and Planning

We introduce an online goal recognition approach which combines the online use of a planner, and landmark information—computed once—for increased efficiency and accuracy. We define the online recognition problem and the baseline algorithm, originally developed by Vered et al. [2016], in Sec. 3.1. We then provide a general definition of landmarks (including in continuous domains), and their use for online recognition in Sec. 3.2. We develop an algorithm to extract landmarks in continuous spaces in Sec. 3.3 and finally, we introduce a combined algorithm in Sec. 3.4 that uses both a planner and landmarks in recognition.

3.1 Online Goal Recognition Using Planning

We refer to [Vered *et al.*, 2016] to define the goal recognition problem R as as a tuple $\langle W, G, O, M \rangle$. W is the set of possible states of the world (in discrete domains, this is implicitly represented by the domain theory; in continuous spaces, it is the standard motion planning work area [LaValle, 2006]); G is a set of $k \geq 1$ goals $g_1, \ldots, g_k$; each goal $g_j \in W$; The set of observations O is defined for a subset of W; and M is a set of *plans* where at least one of the plans is assumed to be consistent with the observations in O. Given the problem R, the task is to choose a specific goal $g \in G$ that *best matches* the observations O. Vered et al. [2016] define *best matches* as minimizing matching error, this refers to minimizing the difference between the accumulated distance, measured by the observations, against the hypothesized generated trajectory to reach goal g.

We build on *goal mirroring*, an online goal recognition approach described in [Vered *et al.*, 2016], and applied to recognition in continuous domains and generalize this algorithm to admit discrete domains as well. The goal mirroring algorithm works as follows. Assume we are given a set of goals G, and an initial state of the observed agent's plan, I. For each goal $g \in G$ goal mirroring compares the costs of two plans: an *ideal plan* denoted i_g, and an *observation-matching plan*, denoted m_g. In the 3D navigation domain *cost* is defined according to distance, as the distance reflects the "effort" needed to achieve the goal; and the optimality is determined according to cost alone. For continuous variables we consider cost to be the euclidean distance between observations whereas for discrete ones we consider cost to be the number of observations.

The ideal plan i_g is an optimal plan, computed once for each goal g, from the initial state I to the goal. The observation-matching plan m_g is constructed for each new observation such that it always passes through all the observations thus far, and then optimally reaches the goal. Ramirez and Geffner [2009, Theorem 7] show that necessarily, a goal g for which the two plans, m_g and i_g , have equal costs is a solution to the goal recognition problem. We use this to probabilistically rank the goals. The closer two costs for i_g and m_g are, the higher the likelihood of g.

The plan m_g is constructed for each new observation by concatenating two parts:

- A plan prefix m_g^- which is a concatenation of all observations received to date. This is very efficiently done by simply adding the latest observation to the current prefix (which is initially $\emptyset$).
- a plan suffix m_g^+ which is a new plan, issued by a motion planner, from the last state of the prefix (after incorporating the observations), to the goal state. The bulk of the computation takes place here using the planner.

The plan prefix m_g^- and suffix m_g^+ are handled differently in continuous and discrete domains. In continuous domains, plans and observations are both trajectories in $\mathbb{R}^n$. Thus updating m_g^- with a new observation o is a straightforward operation of adding the point o to a trajectory, or (if o is an observed trajectory segment) connecting the end-point of m_g^- to the new observation o. The suffix trajectory, connecting the end point of o to the goal g, is generated by a motion planner. In discrete domains, observations are of actions, not states. Thus m_g^- is not a trajectory through states, but rather an ordered sequence of actions. Adding an action to this is trivial. However, the synthesis of the sequence m_g^+ by calling a symbolic planner, is more involved. The initial state provided to the planner is computed by adding the effects of the observed

action o to the previous state. Thus alongside the sequence of actions in m_g^-, we keep track of the current state induced by ordered application of m_g^- to the initial state I.

3.2 Online Goal Recognition Using Landmarks

In the planning literature, *landmarks* are facts (alternatively, actions) that must be true (alternatively, executed) at some point along all valid plans that achieve a goal from an initial state [Hoffmann *et al.*, 2004]. Landmarks are often partially ordered based on the sequence in which they must be achieved. Figure 1 shows an example of landmarks and their ordering for a Blocks-World example. The root node is the goal state, whereas leafs are facts of the initial state. Connected boxes represent facts that must be true together, i.e., conjunctive facts. For example, to achieve the fact `(on A B)`, immediately before the facts `(and (holding A) (clear B))` must be true, and so on.

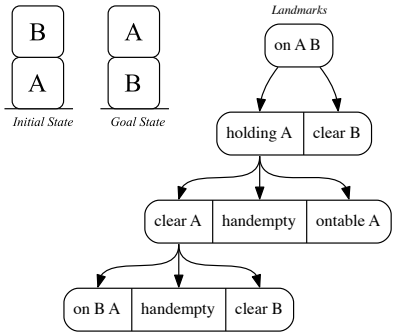

Figure 1: Blocks-World landmarks example.

Given their usefulness for planners and planning heuristics [Richter and Westphal, 2010], research has yielded multiple notions of landmarks [Porteous and Cresswell, 2002], including that of disjunctive landmarks. Pereira et al. [2016; 2017b] have shown that it is possible to carry out offline plan recognition by reasoning heuristically about landmarks. The key idea is to maintain a list of ordered landmarks associated with each goal, though partial overlaps are allowed. The *goal completion* heuristic from Pereira et al. [2017b] matches the observations against this list. This heuristic marks a landmark as achieved when facts in the preconditions and effects of an observation match a landmark. Then, the heuristic uses the ratio of the number of landmarks achieved to the total number of landmarks associated with the goal, inducing a ranking of the goals, used as a proxy for estimating $P(g|O)$.

In principle, it is possible to translate the same idea into recognition in continuous domains. In such domains, landmarks can be defined as areas surrounding goals, as illustrated in Figure 2 where black dots represent goals and the surrounding rectangles represent the continuous landmark areas. In this case, to reach a goal, the observed motion must intersect (go through) the corresponding landmark area. Naturally, we would prefer such areas to be maximal, but must maintain the restriction that landmarks cover only obstacle-free space, and do not intersect completely with other landmarks. In Section 3.3 we report on a novel algorithm for extracting such landmarks in 2D continuous domains.

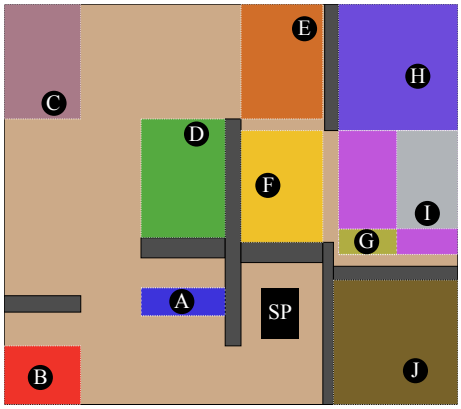

Figure 2: Landmarks for Cubicles environment.

In what follows, we define the characteristic operations between observations and landmarks: testing whether a landmark has been observed, what constructs an achieved landmark and, in continuous environments, how to distinguish a *passed* landmark (a landmark that has recently been achieved but but does not currently fit the last observations).

Online Recognition with Landmarks Algorithm 1 uses landmarks for recognition, in an online fashion and works as follows. First we maintain FL, the *ordered* set of fact (*achieved*) landmarks (Line 2). We will also need to remember the *last* achieved landmark to compare to every additional observation, $lastFl$ (Line 3) and to maintain $PruneG$, the group of goals that has been pruned out during the recognition process, it is necessary to maintain this group in case of backtracking in which we will have to re-introduce a recently pruned goal back into G.

For every incrementally revealed observation, o, we check if this observation has caused any landmarks to be achieved (Line 7). If so, we include this landmark in the set of achieved landmarks FL and update the last seen landmark (Lines 8 – 9). This differentiates from the last achieved landmark, $lastFL$, and the other landmarks in FL which may now be defined as *passed*. We then proceed to attempt to prune out goals (Line 12).

ACHIEVEDLANDMARK Analogously to the discrete case, we match observations to landmarks, by intersecting observations (points) with each landmark. We therefore define the method ACHIEVEDLANDMARK(O,L) as follows: For every landmark $l \in L$ the method checks if the landmark is achieved, i.e., a *fact landmark*, if for the observations o, $o \cap l \neq \emptyset$. If the intersection is non-empty, then the landmark is considered achieved and the method returns *true*, otherwise it returns *false*. If the landmark has indeed been achieved, we can then go on to update FL to include the achieved landmark l (Line 8).

However, unlike the discrete case, where the next observation will no longer be equal to the landmark (i.e., the next

Algorithm 1 ONLINE RECOGNITION WITH LANDMARKS.

```
1: function ContOnlineLandmarks(R, L)
2:   FL ← ∅
3:   lastFl ← ∅
4:   PruneG ← ∅
5:   for all o ∈ O do
6:     for all l ∈ L do
7:       if AchievedLandmark(o, l) then
8:         FL ← FL ∪ l
9:         lastFl = l
10:    for all g ∈ G do
11:      if l_g ∈ FL then
12:        if PruneGoal(l_g, o) then
13:          PruneG ← PruneG + g
14:        else
15:          PruneG ← PruneG − g
16:    for all g ∈ G ∩ PruneG do
17:      P(g|O) ← rank(g)
18:    return P
```

step will go out of the landmark), in the continuous case this may not be the case. We may see several observations, all in the same landmark area. Only once the observations *no longer* match the landmark we can mark it as *passed*. Thus continuous landmarks, in the form of areas in spaces, define an *inclusive* disjunction: multiple observations within an area cause the landmark to be marked achieved. We therefore define $lastFL$ as the last (maximal) element in FL, the last landmark achieved (Line 9).

PRUNEGOAL Given that we can now mark landmarks as achieved, we can consider when to use them in order to indicate that a goal g can be removed from further consideration for recognition, as it has been passed. Because landmarks are uniquely associated with each goal we can define for each goal, $g \in G$, a corresponding $l_g \in L$ as the landmark area that contains that goal position. Therefore, for every goal, we may check the corresponding landmarks l_g.

We define the method PRUNEGOAL(l_g, o) as follows. The method goes over all $l_g \in FL$, i.e., all achieved landmarks, if the last observation does not originate from l_g, i.e., $o \cap l_g = \emptyset$ we know that the landmark has been achieved but is not the current leading landmark, $lastFl$. Therefore we know that the landmark is *passed* and the method returns *true*, meaning we have already passed this goal and may proceed to prune it out (in Line 13). Otherwise, l_g has only been achieved but not passed, and the method returns *false*, so we cannot yet prune it out. In Line 15 we make sure to remove it from $PruneG$ in case of agent backtracking.

Finally we rank the goals (Line 16). Our ranking procedure iterates over all non eliminated goals and ranks them in decreasing order according to percentage of achieved landmarks. Consequently, the goals with the highest completion percentage will be ranked first and so on in consecutive order.

3.3 Extracting Landmarks in Continuous Space

We can use any one of a number of landmark extraction algorithms to extract landmarks in discrete environments. Here, we choose to use the algorithm proposed by Hoffman et al. in [2004] due to its runtime efficiency. This algorithm builds a tree in which nodes represent landmarks and edges represent necessary prerequisites between landmarks, thus representing the landmarks and their ordering. A node in this tree represents a conjunction of facts that must be true simultaneously at some point during the execution of a plan, and the root node is a landmark representing the goal state (Figure 1).

Since the interpretation of landmarks we rely on for plan recognition is that of bottlenecks in the state space, we try to partition a continuous space so that such bottlenecks become identifiable areas in the continuous space. Specifically, to extract landmarks in continuous environments we partition the area using the wall corners as references, to eventually identify pathways between individual "rooms" in the space.

The extraction of continuous landmarks algorithm receives the world configuration W and the set of goals G, and maps each $g \in G$ to a rectangular area that represents a landmark position. Each landmark area starts as the outermost bounding box for each goal, (Alg. 2 Line 3), and the algorithm iteratively updates it using each wall present in the world that is also visible as a horizontal or vertical limit, (Alg. 2 Line 6). We define visibility as there being no obstacles between the goal and the wall in question and assume that walls correspond to axis-aligned rectangles. If a single landmark area contains more than one goal, we partition this area again based on the midpoint between an arbitrary goal and the remaining ones to obtain new non-overlapping areas for each goal, (Alg. 2 Line 13), discarding the original area.

Figure 2 illustrates an example of such landmark partition: the black lines represent walls; black dots representing goal candidates; and the different colored rectangles represent landmark areas. We can see the leftmost wall limiting the width of the landmarks areas B and C of the two leftmost goals while the center wall limits their height. A repartition happens in the middle-right area to obtain areas G and I. Now that we can compute landmarks for both discrete and continuous domains, we proceed to employ them to perform online goal recognition.

3.4 Goal Mirroring with Landmarks

In general, PRP recognizers issue repeated calls to a planner in order to carry out the recognition. This is exacerbated in online recognition, as the goal monitoring recognizer described above, (Sec. 3.1), calls the planner to compute a new m_g^+ suffix with every observation, and for every goal. To address this challenge, we use the evidence provided by comparing observations to planning landmarks. By combining these two approaches we aim to exploit both the flexibility of a PRP approach and the efficiency of reasoning about landmarks.

To improve the efficiency, we can therefore use the information conveyed by the landmarks as a pruning mechanism with which we may rule out hypotheses. In this way, we may reduce $|G|$ and therefore reduce the number of calls to the planner and overall run-time.

Algorithm 2 EXTRACT CONTINUOUS LANDMARKS.

1: **function** EXTCONTINUOUSLANDMARKS(W, G)
2: $landmarks \leftarrow map$
3: **for all** $g \in G$ **do**
4: $rect \leftarrow$ BOUNDINGBOX(g, W)
5: **for all** $wall \in W$ **do**
6: **if** VISIBLEFROMCENTROID$(g, wall, W)$ **then**
7: $rect \leftarrow updateBoundingBox(rect, wall)$
8: **if** $\neg rect \in landmarks$ **then**
9: $landmarks[rect] \leftarrow \emptyset$
10: $landmarks[rect] \leftarrow landmarks[rect] \cup goal$
11: **for all** $(rect, goals) \in landmarks$ **do**
12: **if** $|goals| > 1$ **then**
13: **for all** $g \in goals$ **do**
14: $landmarks[midpointBox(g, goals)] \leftarrow g$
15: $remove(landmarks[rect])$
16: **return** $landmarks$

Algorithm 3 GOAL MIRRORING WITH LANDMARKS.

1: **function** ONLINEGMWLANDMARKS($R, planner$)
2: $L \leftarrow$ EXTCONTINUOUSLANDMARKS(W, G)
3: $FL \leftarrow FL \leftarrow \emptyset$
4: **for all** $g \in G$ **do**
5: $i_g \leftarrow planner(I, g)$
6: **for all** $o \in O$ **do**
7: **for all** $l \in L$ **do**
8: **if** ACHIEVEDLANDMARK(o, l) **then**
9: $FL \cup l$
10: $m_g^- \leftarrow m_g^- \cup o$
11: **for all** $g \in G$ **do**
12: **if** PRUNEGOAL(l_g, o) **then**
13: $G \leftarrow G - g$
14: **else**
15: $m_g^+ \leftarrow$ PLANNER(o, g)
16: $score(g) \leftarrow rank(cost(i_g), cost(m_g^-), cost(m_g'+))$
17: **for all** $g \in G$ **do**
18: $P(g|O) \leftarrow \eta \cdot score(g)$
19: **return** P

The original *goal mirroring* algorithm had to undergo several adjustments to be able to use landmark information as a pruning mechanism, see Alg. 3. For simplicity we assume *rationality* of the agent, in that sense there will be no backtracking and therefore no need to monitor the last achieved landmark and to maintain a separate set of pruned out goals.

The algorithm now begins with the run-time generation of domain specific landmarks for all monitored goals in Line 2, and the initialization of the previously introduced FL in Line 3. For every incoming observation $o \in O$, we must ascertain whether any of the conditions for the existing landmarks $l \in L$ have been met (Line 8). If the landmarks have been achieved we update FL and can now proceed to use the existing fact landmarks to prune unlikely goals (Line 13). In which case we will only call the planner to compute plans for those goals whose landmarks have been satisfied in the correct order and have not been exceeded (Lines 16–17). Finally, the rankings are transformed into probabilities $P(G|O)$ via the normalizing factor $\eta = 1/\sum_{g \in G} score(g)$ and the ranking is returned (Lines 18–20).

4 Experiments and Evaluation

We empirically evaluated our approach on both discrete and continuous environments, over hundreds of goal recognition problems while measuring both efficiency and performance.

4.1 Setup

As our continuous environment we used the domain of 3D navigation, where the target is to recognize navigational goals as soon as possible while the observations, i.e., observed agents' positions, are incrementally revealed. We used TRRT (Transition-based Rapidly-exploring Random Trees), an off-the-shelf planner that guarantees asymptotic near-optimality by preferring shorter solutions, available as part of the Open Motion Planning Library (OMPL [Sucan *et al.*, 2012]) along with the OMPL *cubicles* environment and default robot displayed in Figure 3. The yellow polygon representing the robot and the green polygons representing obstacles in the environment. Each call to the planner was given a time limit of 1 sec; and the cost measure being the length of the path. We set 11 points spread through the cubicles environments. We then generated two observed paths from each point to all others, for *a total of* 110×2 *goal recognition problems*. The observations were obtained by running the asymptotically optimal planner RRT* on each pair of points, with a time limit of 5 minutes per run.

We evaluated our approaches in discrete environments using datasets provided by Ramírez and Geffner [2009; 2010][1]. These datasets comprise six domains with hundreds of non-trivial goal recognition problems. Each goal recognition problem contains a domain description, an initial state, a set of candidate goals, a hidden goal, and an observation sequence, representing a plan that achieves the hidden goal. To evaluate our approaches in discrete domains, we used JavaFF[2], a Fast-Forward [Hoffmann and Nebel, 2001] implementation in Java.

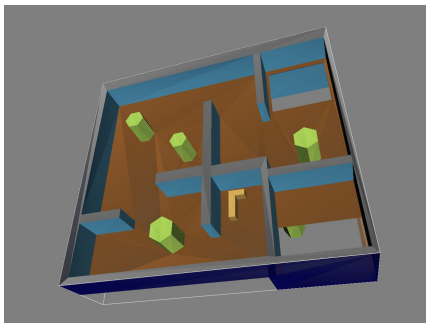

Figure 3: Cubicles environment.

[1] https://sites.google.com/site/prasplanning/
[2] https://github.com/Optimised/JavaFF

4.2 Evaluation on Continuous and Discrete Domains

We evaluated our combined approach, (GOAL MIRRORING WITH LANDMARKS) both in terms of improvement in efficiency and in terms of overall performance in order to show that the improvement in efficiency did not come at the expense of performance but rather improved it. We then contrasted the performance with the existing PRP approach (GOAL MIRRORING) and an approach utilizing only the landmarks for ranking and pruning out goals (ONLINE RECOGNITION WITH LANDMARKS).

Efficiency Measures We used two separate measures to evaluate the overall *efficiency* of our approach: (1) the amount of times the planner was called within the recognition process; and (2) the overall time (in sec.) spent planning. Both these parameters measure the overhead of the *PRP* approach of using the planner and while they are closely linked, they are not wholly dependent. While a reduction in overall number of calls to the planner necessarily results in a reduction in planner run-time, the total amount of time allowed for each planner run may vary according to the difficulty of the planning problem and therefore create considerable differences. Naturally, lower values are better.

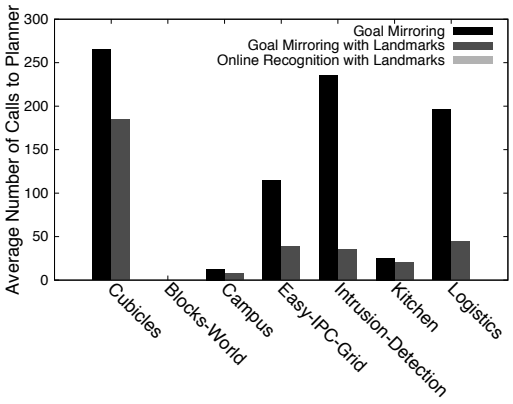

Figure 4: Average number of calls to planner comparison.

Performance Measures We use two measures of recognition performance : (1) convergence to the correct answer. We measure the time step in which the recognizer converged to the correct hypothesis from the end of the observation sequence (or 0 if it failed). Higher values indicate earlier convergence and are therefore better; and (2) the number of times they ranked the correct hypothesis at the top (i.e., rank 1), which indicates their general accuracy. The more frequently the recognizer ranked the correct hypothesis at the top, the more reliable it is, hence a larger value is better.

Results Table 1 shows the experimental results for both continuous and discrete domains across all criteria. For the continuous domain, the combined GOAL MIRRORING WITH LANDMARKS approach achieved the best performance with an improvement both in convergence and the amount of times the recognizer ranked correctly. The pure ONLINE RECOGNITION WITH LANDMARKS approach performed worst in terms of convergence but was on par with the GOAL MIRRORING approach in terms of correct rankings.

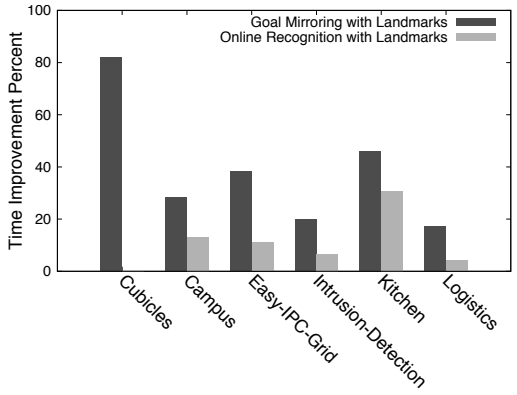

Figure 5: Time Improvement Percent comparison.

In the discrete domain, we see corresponding results. For domain problems: Campus, Easy-IPC-Grid, Intrusion-Detection, Kitchen and Logistics we see that the overall best results in terms of convergence and correct ranking of the chosen goal were achieved by the combined approach of GOAL MIRRORING WITH LANDMARKS. However, unlike in the continuous domain, we see that using the ONLINE RECOGNITION WITH LANDMARKS technique also provided good results, once even considerably surpassing that of the GOAL MIRRORING approach (Campus). For the domain problem of the Blocks-World we see that both GOAL MIRRORING and GOAL MIRRORING WITH LANDMARKS approaches failed to find solutions within the given, 60 sec. time limit. This dataset is considerably more complex than the rest, containing 20.3 goals for non-trivial stackable blocks problems. In this instance we see the strength of the ONLINE RECOGNITION WITH LANDMARKS approach, which does not employ a planner and therefore evades the considerable overhead calculations.

The improvement of run time over the baseline GOAL MIRRORING approach is presented in Figure 5. For the continuous domain, ONLINE RECOGNITION WITH LANDMARKS, introducing landmarks, reduces the run time to 80% while ONLINE RECOGNITION WITH LANDMARKS was by far the most efficient with a reduction to only 0.019% of the original GOAL MIRRORING runtime. For the discrete domain as well we see that ONLINE RECOGNITION WITH LANDMARKS more than doubles the reduction in run-time vs. the ONLINE RECOGNITION WITH LANDMARKS, which in itself reduces the run-time considerably to between 17%–46%. Figure 4 shows a comparison regarding the amount of times the planner was called within the recognition process for both continuous and discrete domains.

				Continuous Domains								
				Goal Mirroring			Goal Mirroring with Landmarks			Online Recognition with Landmarks		
Domain (# problems)	$\|G\|$	$\|O\|$	$\|L\|$	Time	Ranked First	Convergence	Time	Ranked First	Convergence	Time	Ranked First	Convergence
Cubicles (220)	10.0	26.5	11.0	104.7	20.23%	21.8%	85.99	**24.34%**	**26.23%**	**0.02**	21.7%	15%

				Discrete Domains								
				Goal Mirroring			Goal Mirroring with Landmarks			Online Recognition with Landmarks		
Domain (# problems)	$\|G\|$	$\|O\|$	$\|L\|$	Time	Ranked First	Convergence	Time	Ranked First	Convergence	Time	Ranked First	Convergence
Blocks-World (75)	20.3	8.4	15.6	*Time-out*	0%	0%	*Time-out*	0%	0%	**0.12**	**31.49%**	**30.68%**
Campus (15)	2.0	5.4	8.6	0.46	65.65%	50.22%	0.13	**96.44%**	**96.44%**	**0.06**	92.88%	92.88%
Easy-IPC-Gird (45)	7.7	13.2	11.8	3.71	44.62%	43.88%	1.42	**54.95%**	**52.65%**	**0.41**	40.18%	38.75%
Intrusion-Detection (45)	16.6	13.1	16.0	1.65	55.31%	55.31%	0.33	**67.11%**	**67.11%**	**0.11**	57.18%	55.15%
Kitchen (15)	3.0	7.5	5.0	0.13	52.22%	42.77%	0.06	**66.51%**	**51.11%**	**0.04**	23.91%	23.91%
Logistics (45)	10.0	18.6	18.7	7.96	27.71%	26.65%	1.36	**54.11%**	**53.41%**	**0.33**	43.11%	43.11%

Table 1: Experimental results for both continuous and discrete domains.

5 Conclusions

We developed an online approach to recognize goals in both continuous and discrete domains using a combination of goal mirroring and a generalized notion of landmarks. We have shown how to dynamically generate continuous and discrete landmarks and empirically evaluated the efficiency and performance of our approach over hundreds of experiments in both continuous and discrete domains; comparing our results to an existing PRP approach and a newly defined continuous landmark approach. We have shown that not only is our approach more efficient than the existing PRP recognizer but also outperforms both other approaches. As future work, we aim to refine the notion of spatial landmarks for more informative heuristics.

References

[Avrahami-Zilberbrand and Kaminka, 2007] Dorit Avrahami-Zilberbrand and Gal A. Kaminka. Incorporating observer biases in keyhole plan recognition (efficiently!). In *AAAI-07*, pages 944–949, 2007.

[Baker *et al.*, 2005] Chris Baker, Rebecca Saxe, and Joshua B Tenenbaum. Bayesian models of human action understanding. In *Advances in neural information processing systems*, pages 99–106, 2005.

[Blaylock and Allen, 2004] Nate Blaylock and James F Allen. Statistical goal parameter recognition. In *ICAPS*, volume 4, pages 297–304, 2004.

[Bui, 2003] Hung Hai Bui. A general model for online probabilistic plan recognition. In *IJCAI*, volume 3, pages 1309–1315, 2003.

[Geib and Goldman, 2009] Christopher W Geib and Robert P Goldman. A probabilistic plan recognition algorithm based on plan tree grammars. *Artificial Intelligence*, 173(11):1101–1132, 2009.

[Geib, 2015] Christopher Geib. Lexicalized reasoning. In *Proceedings of the Third Annual Conference on Advances in Cognitive Systems*, 2015.

[Hoffmann and Nebel, 2001] Jörg Hoffmann and Bernhard Nebel. The FF Planning System: Fast Plan Generation Through Heuristic Search. *JAIR*, 14(1):253–302, 2001.

[Hoffmann *et al.*, 2004] Jörg Hoffmann, Julie Porteous, and Laura Sebastia. Ordered Landmarks in Planning. *JAIR*, 22(1):215–278, 2004.

[Hong, 2001] Jun Hong. Goal recognition through goal graph analysis. *JAIR*, 15:1–30, 2001.

[LaValle, 2006] Steven M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[Liao *et al.*, 2007] Lin Liao, Dieter Fox, and Henry Kautz. Hierarchical conditional random fields for gps-based activity recognition. In *Robotics Research: The 11th International Symposium (ISRR)*, Springer Tracts in Advanced Robotics (STAR). Springer Verlag, 2007.

[Martin *et al.*, 2015] Yolanda E. Martin, Maria D. R. Moreno, David E Smith, et al. A fast goal recognition technique based on interaction estimates. In *IJCAI*, pages 761–768, 2015.

[McDermott *et al.*, 1998] Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. PDDL—The Planning Domain Definition Language. In *The Fourth International Conference on Artificial Intelligence Planning Systems 1998 (AIPS'98)*, 1998.

[Mirsky and Gal, 2016] Reuth Mirsky and Ya'akov (Kobi) Gal. SLIM: Semi-lazy inference mechanism for plan recognition. In *IJCAI*, 2016.

[Pereira and Meneguzzi, 2016] Ramon Fraga Pereira and Felipe Meneguzzi. Landmark-Based Plan Recognition. In *ECAI*, 2016.

[Pereira *et al.*, 2017a] Ramon Fraga Pereira, Nir Oren, and Felipe Meneguzzi. Detecting commitment abandonment by monitoring sub-optimal steps during plan execution. In *AAMAS*, pages 1685–1687, 2017.

[Pereira *et al.*, 2017b] Ramon Fraga Pereira, Nir Oren, and Felipe Meneguzzi. Landmark-based heuristics for goal recognition. In *AAAI*. AAAI Press, 2017.

[Porteous and Cresswell, 2002] J. Porteous and S. Cresswell. Extending Landmarks Analysis to Reason about Resources and Repetition. In *Proceedings of the 21st Workshop of the UK Planning and Scheduling Special Interest Group (PLANSIG '02)*, 2002.

[Pynadath and Wellman, 2000] David V. Pynadath and Michael P. Wellman. Probabilistic state-dependent grammars for plan recognition. In *UAI-2000*, pages 507–514, 2000.

[Ramírez and Geffner, 2009] Miquel Ramírez and Hector Geffner. Plan recognition as planning. In *IJCAI*, pages 1778–1783, 2009.

[Ramírez and Geffner, 2010] Miquel Ramírez and Hector Geffner. Probabilistic plan recognition using off-the-shelf classical planners. In *AAAI*, 2010.

[Richter and Westphal, 2010] Silvia Richter and Matthias Westphal. The LAMA Planner: Guiding Cost-based Anytime Planning with Landmarks. *JAIR*, 39(1):127–177, 2010.

[Sadeghipour and Kopp, 2011] Amir Sadeghipour and Stefan Kopp. Embodied gesture processing: Motor-based integration of perception and action in social artificial agents. *Cognitive Computation*, 3(3):419–435, 2011.

[Sohrabi *et al.*, 2016] Shirin Sohrabi, Anton V. Riabov, and Octavian Udrea. Plan recognition as planning revisited. *IJCAI*, pages 3258–3264, 2016.

[Sucan *et al.*, 2012] Ioan A Sucan, Mark Moll, and Lydia E Kavraki. The open motion planning library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, 2012.

[Sukthankar *et al.*, 2014] Gita Sukthankar, Robert P. Goldman, Christopher Geib, David V. Pynadath, and Hung Bui, editors. *Plan, Activity, and Intent Recognition*. Morgan Kaufmann, 2014.

[Vered *et al.*, 2016] Mor Vered, Gal A Kaminka, and Sivan Biham. Online goal recognition through mirroring: Humans and agents. *The Fourth Annual Conference on Advances in Cognitive Systems*, 2016.

[Wang *et al.*, 2013] Zhikun Wang, Katharina Mülling, Marc Peter Deisenroth, Heni Ben Amor, David Vogt, Bernhard Schölkopf, and Jan Peters. Probabilistic movement modeling for intention inference in human–robot interaction. *The International Journal of Robotics Research*, 32(7):841–858, 2013.