The Workshops of the Thirty-Second
AAAI Conference on Artificial Intelligence

Online Goal Recognition as Reasoning over Landmarks

Mor Vered,[1] **Ramon Fraga Pereira,**[2] **Maurício Cecílio Magnaguagno,**[2]
Felipe Meneguzzi,[2] **Gal A. Kaminka**[1]
[1] Bar-Ilan University, Israel
[2] Pontifical Catholic University of Rio Grande do Sul, Brazil
{veredm,galk}@cs.biu.ac.il
{ramon.pereira, mauricio.magnaguagno}@acad.pucrs.br
felipe.meneguzzi@pucrs.br

Abstract

Online goal recognition is the problem of recognizing the goal of an agent based on an incomplete sequence of observations with as few observations as possible. Recognizing goals with minimal domain knowledge as an agent executes its plan requires efficient algorithms to sift through a large space of hypotheses. 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 and substantially more accurate than the state-of-the-art.

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, Fox, and Kautz 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 2010; Sohrabi, Riabov, and Udrea 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, Oren, and Meneguzzi 2017a). This assumption does not hold 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 which one is the final observation.

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, Kaminka, and Biham 2016; Vered and Kaminka 2017) and minimizing the number of goal hypotheses to be computed using landmarks computed once during run-time (Pereira and Meneguzzi 2016; Pereira, Oren, and Meneguzzi 2017b), and a landmark-based heuristic (Pereira, Oren, and Meneguzzi 2017a). To generalize over discrete and continuous domains we adapt the notion of planning landmarks (Hoffmann, Porteous, and Sebastia 2004) to comprise its original planning semantics as well as continuous spatial domains and develop a new and efficient algorithm to generate spatial landmarks. Since our approach can use any type of PDDL (McDermott et al. 1998) planning algorithm or path planner (Sucan, Moll, and Kavraki 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 encompassing 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 show superior efficiency and generally superior recognition quality over the state of the art.

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 (Sukthankar et al. 2014). While often efficient, these

methods are limited to recognizing goals for which plans are a-priori known, and encoded in the plan library.

Plan recognition based on domain-theories removes the reliance on a plan-library, assuming the any valid sequence of actions is a possible plan. For example, *plan recognition as planning* (PRP) (Ramírez and Geffner 2009; 2010; Sohrabi, Riabov, and Udrea 2016) uses a planner to generate plan hypotheses dynamically, based on the domain-theory and the observations. More efficient offline approaches (Pereira and Meneguzzi 2016; Pereira, Oren, and Meneguzzi 2017a) avoid 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, Porteous, and Sebastia 2004), and thus provide strong evidence for recognizing these goals,

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) use a bayesian framework to calculate goal likelihoods by marginalizing over possible actions and generating state transitions. Martin et al. (2015) take an extreme approach, using significant offline computation to eliminate all online planner calls by pre-computing cost estimates.

In contrast, Vered et al. (2016; 2017) present an online PRP algorithm for continuous spaces, using off-the-shelf motion planners to estimate goal likelihoods. Like other PRP methods, they rely on repeated calls to 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.

Combining Landmarks and Planning

We now introduce an online goal recognition approach that combines the online use of a planner, and landmark information—computed once—for increased efficiency and accuracy. We start by adapting the formalization of Pommerening and Helmert (2015) to accommodate both discrete and continuous planning problems in the same formal framework. We then develop a baseline online recognition algorithm; algorithms to extract landmarks from continuous spaces; and a combined recognition algorithm that uses both a planner and landmarks in the recognition process.

Online Goal Recognition Using Planning

We define the goal recognition problem R as a quintuple $\langle s_0, W, G, O, M\rangle$. 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)) and $s_0 \in W$ the agents' initial state; G is a set of $k \geq 1$ goals $g_1, \ldots, g_k$; each goal $g_j \in W$, a partial state in W; The *ordered* set of observations O is noisy and potentially incomplete (Sohrabi, Riabov, and Udrea 2016) and is defined for a subset of W. $M = \langle \mathcal{V}, \mathcal{O}\rangle$ is a domain model defined by a set of variables $\mathcal{V}$ and a set of operators $\mathcal{O}$ over states where states are defined over a set of variables $\mathcal{V}$ with finite domains (Pommerening and Helmert 2015). Partial states s map a subset of variables of s, $vars(s)$ to values in their domain so that $vars(s) \subseteq \mathcal{V}$. Consistent partial states are such that all their shared variables have the same value, i.e. s and s' are consistent iff $s(V) = s'(V)$ for all $V \in vars(s) \cap vars(s')$. We use the $s \models s'$ as a shorthand for the relation that states that s is consistent with s'. States are such that they contain all variables so that $vars(s) = \mathcal{V}$. Facts are variable/value pairs (i.e. they state what is the current value of a variable) $V \mapsto v$. The finite set of operators $\mathcal{O}$ contains state transformation operations $o = \langle pre(o), \mathit{eff}(o), cost(o)\rangle$ with preconditions, effects and cost. An operator o is applicable in state s if $pre(o)$ is consistent with s (i.e. $s \models pre(o)$). The set of operators induces a transition function γ that maps variable assignments and operators into new variable assignments. Applying o to s results in a new state s' that is consistent with $\mathit{eff}(o)$ and agrees with all variables not in the effect (i.e. $s' \models \mathit{eff}(o) \cup (s/vars(\mathit{eff}(o)))$). In this paper, we assume that $cost(o)$ is 1 for discrete domains, and the euclidean distance between the value of the position variables in the precondition and the effect of a move action.

A planning problem $\Pi = \langle s_0, g, W, M\rangle$ may have a number of solutions $\pi = \langle o_1, \ldots, o_n\rangle$ with $o_i \in \mathcal{O}$ such that the sequential application of o_i starting from initial position, s_0, leads to a state $s_n \models g$. A plan π^* is *optimal* if $\sum_{o_i \in \pi^*} cost(o_i)$ is minimal. Thus, M denotes a generative model for *intentional plans* for the pairs $\langle s_0, g_i\rangle \; \forall g_i \in G$, when transitioning to plan recognition we assume that the states in the observation O are consistent with valid states generated by plans generated by M. 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 cost, measured by concatenating the observations, against the cost of the hypothesized plan trajectory to reach goal g. Note that though some work on plan recognition as planning assumes observations to be actions, without loss of generality, these observations can be converted to sequences of states using the model M and taking the last known state s and applying the effects of operator $o \in \mathcal{O}$ matching the observation $o \in O$.

We build on *goal mirroring*, an online goal recognition approach first described in (Vered, Kaminka, and Biham 2016), and applied to recognition in continuous domains and generalize this algorithm to admit discrete domains as well. For each goal $g_k \in G$ in a problem R, goal mirroring compares the costs of two plans: an *ideal plan* denoted i_k, and an *observation-matching plan*, denoted m_k. 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_k is an optimal plan, computed once from

the initial state s_0 to each goal $g_k \in G$. The observation-matching plan m_k is constructed for each new observation such that it always visits the states induced by all the observations thus far, and then optimally reaches the goal. Ramirez and Geffner (2009, Theorem 7) show that necessarily, a goal g_k for which the two plans, m_k and i_k , 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_k and m_k are, the higher the likelihood of g_k. The plan hypothesis m_k is constructed for each new observation by concatenating two parts. First, a plan prefix m^- 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 $\emptyset$). As shown in (Masters and Sardina 2017) this may be generated only once for all possible goal trajectories. Second, a plan suffix m_k^+ generated by a motion planner, from the last state of the prefix (after incorporating the observations), to the goal state, g_k. Most computation takes place here by calling the planner.

The plan prefix m^- and suffix m_k^+ are handled slightly differently in continuous and discrete domains. In continuous domains, plans and observations are both trajectories in $\mathbb{R}^n$. Thus updating m^- 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^- to the new observation o. The suffix trajectory is generated by calling a motion planner to generate a plan from the end point of o to a goal g_k. In discrete domains, the synthesis of the sequence m_k^+ by calling a symbolic planner involves updating the initial known state by successively updating the variables of s_0 so that they are consistent with the variables in each observation in m^-.

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, Porteous, and Sebastia 2004). Landmarks are often partially ordered based on the sequence in which they must be achieved. 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; 2017a) show 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. (2017a) matches the observations against this list. This heuristic marks a landmark as *achieved* when facts in the observation match a landmark. The heuristic then 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, we can translate the same idea into recognition in continuous domains. In such domains, landmarks can be defined as areas surrounding goals, as illustrated in Figure 1 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.

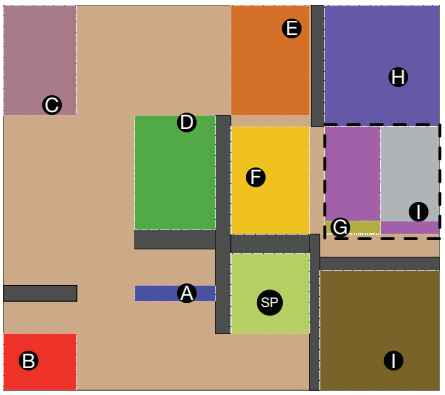

Figure 1: Landmarks for Cubicles environment.

Algorithm 1 Online Goal Recognition With Landmarks.

Require: $L \leftarrow$ EXTRACTLANDMARKS(s_0, W, M, G)
Require: $R = \langle s_0, W, G, O, M \rangle$
1: **function** CONTONLINELANDMARKS(R, L)
2: **static** $FL \leftarrow \emptyset$
3: **static** $activeFl \leftarrow \emptyset$
4: **static** $PruneG \leftarrow \emptyset$
5: **for all** $o_i \in O$ **do**
6: $activeFL, FL \leftarrow$ ACHIEVELANDMARK$(o_i, L, activeFL, FL)$
7: **for all** $g_k \in G$ **do**
8: **if** $l_k \in FL$ **then**
9: $PruneG \leftarrow PruneG + g_k$
10: **else**
11: $PruneG \leftarrow PruneG - g_k$
12: **for all** $g_k \in G \cap PruneG$ **do**
13: $P(g_k|O) \leftarrow$ RANK(g_k)
14: **return** P

The two characteristic operations between observations and landmarks in continuous environments are: testing whether a landmark has been observed, and differentiating between what constitutes an *active* landmark, and, how to identify a landmark that has recently been active but no longer fits the last observation (an *achieved* landmark). Algorithm 1 uses landmarks and the notions above for online recognition as follows. First, we pre-compute the landmarks for the problem and provide these (cached landmarks) to the algorithm at every execution. As we continually update our goal ranking by successive calls to CONTONLINELANDMARKS(R, L), we maintain a number of data structures between the calls (identified as static in Lines 2-4). We maintain FL, the *ordered* set of fact landmarks that have been *achieved*. We also need to remember the currently *active* landmark against which we will com-

Algorithm 2 Achieve landmark in continuous domains.

1: **function** ACHIEVELANDMARK($o_j, L, activeFL, FL$)
2: **if** $o_j \cap activeFL = \emptyset \wedge activeFL \neq \emptyset$ **then**
3: $FL \leftarrow FL \cup activeFL$ ▷ $activeFL$ is *passed*, i.e. a *fact* landmark
4: $activeFL \leftarrow \emptyset$
5: **else if** $o_j \cap activeFL = \emptyset \wedge activeFL = \emptyset$ **then**
6: **for all** $l_m \in L$ **do**
7: **if** $o_j \cap l_m \neq \emptyset$ **then**
8: $activeFL \leftarrow l_m$ ▷ Found an active landmark
9: **return** $activeFL, FL$

pare every additional observation, $activeFl$ and to maintain $PruneG$, the group of goals that has been pruned out during the recognition process. We maintain this group of pruned goals in case of backtracking in which we will have to re-introduce a recently pruned goal into G. For every incrementally revealed observation, o_i, we check if this observation has caused any landmarks to be activated or achieved via the ACHIEVELANDMARK (Algorithm 2) function in Line 6.

Because landmarks are necessary conditions to each goal, we can use the last ordered landmark associated with a goal to be the threshold, which provides evidence that an agent is pursuing a certain goal. Now that we can mark landmarks as achieved, we use them to infer that a certain goal can be removed from further consideration for recognition, as it has been passed. Thus, each goal g_k has a corresponding landmark $l_k \in L$ as the area that contains (i.e. is necessary for) that goal position. We then check these landmarks l_k for each goal g_k, and, if it has been passed we prune g_k out or reinstate it if necessary in Lines 9–11.Finally, in Line 13, we iterate over all unpruned goals ranking 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.

Analogously to the discrete case, we match observations to landmarks, by intersecting observations (points) with each landmark. However, unlike the discrete case where, once an observation causes a landmark to be achieved, the next observation will no longer be equal to the landmark (i.e., the next step will go out of the landmark), in the continuous case this may not be so. We may see several consecutive 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 activated. We therefore define $activeFL$ as the currently *active* landmark while FL marks the landmarks that have been active but are now *achieved*.

Algorithm 2 is a general algorithm that evaluates whether landmarks are achieved in both discrete and continuous domains. Variable $activeFL$ either holds the currently active landmark, which means that the observations are within that landmark area, effectively making the goal corresponding to the landmark a leading goal hypothesis, or it could be $\emptyset$. Line 2 determines if a new incoming observation o_j has just caused an *active* landmark to become *achieved*. If o_j is not currently in the $activeFL$ area, **and** $activeFL$ is not empty, it means the observations have just left the $activeFL$ area and have therefore caused that landmark to be achieved. We can therefore add it to the FL set (Line 3) and re-initialize $activeFL$ (Line 4). However, if o_j is not currently in the $activeFL$ area, **and** $activeFL$ is *empty* we check if this new observation has caused any landmarks to be *activated*. In this case we check whether the observation is part of a landmark area and insert it into $activeFL$ (Lines 5– 8).

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 adapt the algorithm of Hoffman et al. in (2004) since it efficiently approximates landmark sets that are good enough for the domains we use. 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.

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. Though we define a landmark generation algorithm for continuous path planning domains, our approach should work with any notion of numeric landmarks, e.g. recent work on landmarks for hybrid domains (Scala et al. 2017).

Algorithm 3 extracts continuous landmarks using a world configuration W and the set of goals G, and maps each $g \in G$ to a rectangular area that represents a landmark position.[1] The landmark area for each goal starts as the outermost bounding box in the environment in Line 4. The algorithm iteratively scales it down by generating a horizontal or vertical line limit using the closest visible walls in 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, though at greater computational cost we could use more sophisticated notions of visibility for any polygonal obstacle through a visibility graph (Choset et al. 2005, Chapter 5). 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 in Line 13, discarding the original area. This partition separates rectangles with multiple goals into a partition analogous to a Voronoi diagram with rectangular areas (Aurenhammer 1991).

Figure 1 illustrates such landmark partition: the black

[1]Note that we include additional parameters to match the algorithm for extracting landmarks for discrete domains.

Algorithm 3 Landmark Extraction for continuous domains.

1: **function** EXTRACTLANDMARKS(s_0, W, M, G)
2: $landmarks \leftarrow []$ ▷ Initialize an empty dictionary
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** $rect \notin landmarks$ **then** $landmarks[rect] \leftarrow \emptyset$
9: $landmarks[rect] \leftarrow landmarks[rect] \cup goal$
10: **for all** $(rect, goals) \in landmarks$ **do**
11: **if** $|goals| > 1$ **then**
12: **for all** $g \in goals$ **do**
13: $landmarks[$MIDPOINTBOX$(g, goals)] \leftarrow g$
14: REMOVE($landmarks[rect]$) ▷ Remove $rect$ index from dictionary
15: **return** $landmarks$

lines represent walls; black dots represent goal candidates; and the different colored rectangles represent landmark areas. We can see the leftmost wall limiting the width of the landmark areas B and C of the two leftmost goals while the center wall limits their height. The dashed area including goals G and I exemplify a partition using the midpoints between these two goals from a square that initially contained both goals. Now that we can compute landmarks for both discrete and continuous domains, we proceed to employ them to perform online goal recognition.

Goal Mirroring with Landmarks

In general, PRP recognizers repeatedly call a planner during recognition, and this is exacerbated in online recognition, as the goal recognizer previously described calls the planner to compute a new plan suffix, m_k^+ with every observation, and for every goal $g_k \in G$. By combining goal mirroring and the evidence provided by landmarks, we exploit both the flexibility of a PRP approach and the efficiency of reasoning about landmarks. Specifically, we use the information conveyed by the landmarks as a pruning mechanism with which we may rule out hypotheses, reducing $|G|$ and therefore the number of calls to the planner and overall run-time.

We extensively modify the original *goal mirroring* algorithm to use landmark information as a pruning mechanism in Algorithm 4. For simplicity of the algorithm we assume agents do not backtrack and therefore eliminate the need to monitor the last achieved landmark and to maintain a separate set of pruned out goals. Like Algorithm 1, we assume a single cached computation of domain specific landmarks for all monitored goals, and the initialization of the previously introduced FL and $activeFL$ in Lines 2– 3. Additionally, as part of the *Goal Mirroring* algorithm we now calculate the ideal plan (minimum cost) from the initial position to each possible goal (Line 4).

For every incoming observation $o_j \in O$, we update the plan prefix m^- in Line 6 and then proceed to ascertain whether this observation has caused any landmarks to

Algorithm 4 Goal Mirroring With Landmarks.

Require: $R = \langle s_0, W, G, O, M \rangle$
Require: $L \leftarrow$ EXTRACTLANDMARKS(s_0, W, M, G)
1: **function** ONLINEGMWLANDMARKS(R, L)
2: **static** $FL \leftarrow \emptyset$
3: **static** $activeFL \leftarrow \emptyset$
4: **for all** $g_k \in G$ **do** $i_k \leftarrow$ PLANNER(s_0, M, g_k)
5: **for all** $o_j \in O$ **do**
6: $m^- \leftarrow m^- \cup o_j$
7: ACHIEVELANDMARK$(o_j, L, activeFL, FL)$
8: **for all** $g_k \in G$ **do**
9: **if** $l_k \in FL$ **then**
10: $G \leftarrow G - g_k$
11: **else**
12: $m_k^+ \leftarrow$ PLANNER(o_j, M, g_k)
13: $m_k \leftarrow m^- \oplus m_k+$
14: $rank_k \leftarrow cost(i_k)/cost(m_k)$
15: **for all** $g_k \in G$ **do**
16: $P(g_k|O) \leftarrow \eta \cdot rank_k$
17: **return** P

be achieved via the previously introduced ACHIEVELANDMARK function. If the observation has caused a landmark to be achieved, FL will be updated and we may use the existing fact landmarks to prune unlikely goals in Line 10, in which case we only call the planner to compute plans for those goals whose landmarks have been satisfied in the correct order and have not been passed (Lines 12–14).

We use the same ranking procedure as Vered and Kaminka (2017) in which the goals are ranked according to the ratio between the initially generated *ideal* plan and the newly generated plan hypothesis, which is comprised of a concatenation of the plan prefix and plan suffix (Lines 13– 14). Finally, the algorithm transforms these rankings into probabilities $P(g_k \mid O)$ via the normalizing factor $\eta = 1/\sum_{g_k \in G} rank(g_k)$ and returns these rankings in Lines 15–17. Mathematically, Algorithm 4 approximates $P(g \mid O)$ for all $g \in G$ using landmarks to rank probabilities, so that, when computing candidate goal probabilities in the bayesian framework of Ramírez and Geffner 2010, we compute $P(O \mid g) = cost(i_k)/(cost(m^-) + cost(m_k+))$. Since $P(g_k \mid O) = \eta \sum_{g_k \in G} P(O \mid g_k)$, our pruning step updates $P(g_k) = 0$ for all ruled-out goals thereby limiting the number of times we need to call the planner.

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.

Evaluation on Continuous and Discrete Domains

For 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

				Continuous Domains																	
				GOAL MIRRORING						GOAL MIRRORING WITH LANDMARKS						ONLINE RECOGNITION WITH LANDMARKS					
Domain (# problems)	$\|G\|$	$\|O\|$	$\|L\|$	*Time*	*PC*	*TPR*	*FPR*	*RF*	*CV*	*Time*	*PC*	*TPR*	*FPR*	*RF*	*CV*	*Time*	*PC*	*TPR*	*FPR*	*RF*	*CV*
Cubicles (220)	11.0	26.5	11.0	104.70	265.0	**100%**	100%	20.2%	21.8%	85.90	184.8	78.2%	61.1%	**24.3%**	**26.2%**	**0.020**	**0**	78.3%	**60.9%**	21.7%	15%

				Discrete Domains																	
				GOAL MIRRORING						GOAL MIRRORING WITH LANDMARKS						ONLINE RECOGNITION WITH LANDMARKS					
Domain (# problems)	$\|G\|$	$\|O\|$	$\|L\|$	*Time*	*PC*	*TPR*	*FPR*	*RF*	*CV*	*Time*	*PC*	*TPR*	*FPR*	*RF*	*CV*	*Time*	*PC*	*TPR*	*FPR*	*RF*	*CV*
Campus (15)	2.0	5.4	8.6	0.441	12.8	60.0%	21.3%	57.3%	41.3%	0.212	7.7	**96.4%**	**1.7%**	**96.4%**	**96.4%**	**0.065**	**0**	92.8%	3.5%	92.8%	92.8%
IPC-Grid (61)	8.3	21.8	10.2	10.36	209.1	**87.2%**	19.4%	36.6%	35.6%	3.29	71.2	55.6%	**10.5%**	**45.8%**	**41.5%**	**0.335**	**0**	59.4%	21.8%	32.6%	31.1%
Ferry (28)	7.5	24.2	28.5	55.24	179.5	83.1%	10.2%	59.2%	57.2%	7.98	35.4	**83.3%**	**3.1%**	**82.4%**	**82.1%**	**0.101**	**0**	82.4%	5.4%	72.5%	71.9%
Intrusion (45)	16.6	13.1	16.0	2.02	235.5	**100%**	7.2%	55.3%	55.3%	0.257	34.7	75.5%	**3.6%**	**67.1%**	**67.1%**	**0.127**	**0**	87.6%	3.9%	57.1%	55.1%
Kitchen (15)	3.0	7.4	5.0	0.141	25.4	70.1%	18.4%	44.6%	36.1%	0.07	20.0	77.6%	**17.9%**	**62.6%**	**58.3%**	**0.04**	**0**	**100%**	50%	23.9%	23.9%
Logistics (61)	10.4	24.4	16.1	53.82	199.3	**95.4%**	14.7%	26.9%	25.8%	14.39	49.6	61.7%	**6.7%**	**49.1%**	**48.4%**	**0.594**	**0**	56.1%	9.5%	40.5%	40.5%
Rovers (28)	6.0	24.9	19.8	*Timeout*	-	-	-	-	-	58.87	31.1	**76.8%**	**4.8%**	**76.2%**	**75.1%**	**0.867**	**0**	72.1%	8.5%	62.1%	62.1%
Satellite (28)	6.4	16.9	10.1	93.89	177.2	**100%**	33.8%	36.1%	36.1%	5.18	30.6	81.8%	9.4%	**72.8%**	**71.9%**	**1.09**	**0**	78.2%	**9.3%**	64.4%	64.1%

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

	Discrete Domains							
	ONLINE RECOGNITION WITH LANDMARKS							
Domain (# problems)	$\|G\|$	$\|O\|$	$\|L\|$	*Time*	*TPR*	*FPR*	*RF*	*CV*
Blocks-World (92)	20.2	20.3	21.0	**0.251**	**39.4%**	**3.9%**	**38.1%**	**37.2%**
Depots (28)	8.8	27.4	33.2	**0.812**	**49.5%**	**9.5%**	**32.1%**	**30.6%**
Driver-Log (28)	7.1	21.7	10.7	**0.574**	**51.8%**	**8.8%**	**43.7%**	**40.1%**
DWR (28)	7.2	51.8	45.0	**0.708**	**45.1%**	**7.9%**	**43.1%**	**33.5%**
Miconic (28)	6.0	35.5	25.5	**0.711**	**82.5%**	**9.7%**	**62.6%**	**61.2%**
Sokoban (28)	7.1	27.7	9.8	**0.772**	**58.1%**	**14.1%**	**36.0%**	**29.5%**
Zeno-Travel (28)	6.8	21.1	8.5	**1.23**	**70.8%**	**6.4%**	**61.3%**	**59.7%**

Table 2: Experimental results for discrete domains (large and non-trivial planning problems).

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, Moll, and Kavraki 2012)) along with the OMPL *cubicles* environment and default robot. 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.

For our discrete environments we used the openly available datasets (Pereira and Meneguzzi 2017) based on the ones developed by Ramírez and Geffner (2009; 2010).These datasets comprise 15 domain models with thousands of non-trivial and large goal recognition problems with optimal and sub-optimal plans. We evaluated our approaches in sub-optimal plans. Each goal recognition problem contains a domain description, initial state, set of candidate goals, a hidden goal, and an observation sequence representing a plan that achieves the hidden goal. For our discrete planner we used the JavaFF[2] implementation of Fast-Forward (Hoffmann and Nebel 2001). We ran the experiments for discrete environments with an 8GB memory limit on the JavaVM and a 2-minute time limit.

We evaluated our combined approach (GOAL MIRRORING WITH LANDMARKS) both in terms of improvement in efficiency and in terms of overall performance showing 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 our newly presented online recognition 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: the number of planner calls (PC) within the recognition process; and the overall time (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.

Performance Measures We used several complementary measures for a thorough evaluation of recognition performance. *True positive rate* (TPR) measures the number of

[2] https://github.com/Optimised/JavaFF

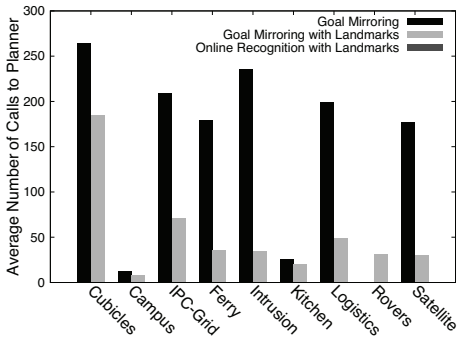

Figure 2: Comparison of average number of calls to planner.

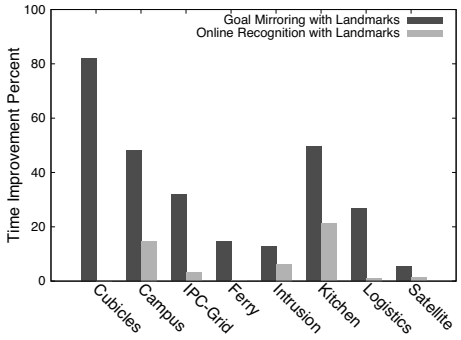

Figure 3: Comparison of runtime improvement.

times an approach recognized the correct goal as a possible hypothesis, i.e. didn't prune it out due to landmarks although didn't necessarily rank it as the chosen hypothesis. We measure mean average percent TPR over all recognition steps. Higher TPR values are better, indicating a measure of the reliability of the system. This value corresponds to 1-FNR (False negative rate). *False positive rate* (FPR) refers to how many goals were not pruned out due to landmarks. We measure FPR in percent out of overall goal number. Lower FPR values are better indicating more pruning and therefore a more efficient algorithm. *Ranked first* (RF) is the number of times the correct hypothesis was not only recognized as a possible hypothesis but also ranked first. *Convergence* (CV) to the correct answer indicates 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.

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 the correct goal hypothesis first. It proved just as reliable as ONLINE RECOGNITION WITH LANDMARKS in terms of *TFR* and *FPR*, however not as reliable as GOAL MIRRORING, which does not prune out goals at all, incurring no risk of overlooking the correct goal.

We see that the combined approach of GOAL MIRRORING WITH LANDMARKS achieves the overall best results in terms of *convergence* and *ranked first* in the discrete domains. However, unlike in the continuous domain, using the ONLINE RECOGNITION WITH LANDMARKS technique also provided good results, sometimes even better than the GOAL MIRRORING approach (Campus, Ferry domain, Logistics, Satellite). In the *Rovers* domain problem we see an instance where GOAL MIRRORING was unable to find a solution within the given time limit, however when utilizing landmarks, the GOAL MIRRORING WITH LANDMARKS approach was able to finish and provide better results than ONLINE RECOGNITION WITH LANDMARKS. This is due to the complex nature of the dataset and highlights the advantages of using landmarks as a pruning mechanism.

However, there were several instances where the dataset was so complex that both the GOAL MIRRORING and GOAL MIRRORING WITH LANDMARKS approaches failed. Due to the repeated calls to the planner these approaches timed-out without results. These problems were considerably more complex with a larger number of objects and instantiated actions. The results are summarized in Table 2, where 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 3. For the continuous domain, GOAL MIRRORING WITH LANDMARKS, incorporating 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 2 shows a comparison regarding the amount of times the planner was called within the recognition process for both continuous and discrete domains.

Conclusions

We developed an online approach to recognize goals in both continuous and discrete domains using a combination of goal mirroring and reasoning over a generalized notion of landmarks. Our formalization adapts work from the transition normal form (TNF) of Pommerening and Helmert (2015) without actually assuming problems are converted to it to avoid having to transform incomplete and noisy observations to suit the requirements of TNF. 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.

However, as our technique continually calls a planner within the recognition process, it is therefore limited from

recognizing very complex problems. Among some of the other limitations is its use of relatively simple landmarks for spatial domains, as well as the assumption that landmarks do not change over the course of the recognition, which would not be realistic for dynamically changing environments. Thus, we believe two important refinements should be the target of future work. First, we aim to refine the notion of spatial landmarks for more informative heuristics, such as the ones developed by Scala et al. (2017). Second, we aim to use techniques to compute landmarks incrementally so as to allow their online recomputation in dynamic domains.

References

Aurenhammer, F. 1991. Voronoi Diagrams - A Survey of a Fundamental Geometric Data Structure. *ACM Comput. Surv.* 23(3):345–405.

Avrahami-Zilberbrand, D., and Kaminka, G. A. 2007. Incorporating observer biases in keyhole plan recognition (efficiently!). In *AAAI*.

Baker, C.; Saxe, R.; and Tenenbaum, J. B. 2005. Bayesian models of human action understanding. In *Advances in neural information processing systems*, 99–106.

Blaylock, N., and Allen, J. F. 2004. Statistical goal parameter recognition. In *ICAPS*, volume 4, 297–304.

Bui, H. H. 2003. A general model for online probabilistic plan recognition. In *IJCAI*, volume 3, 1309–1315.

Choset, H.; Lynch, K. M.; Hutchinson, S.; Kantor, G. A.; Burgard, W.; Kavraki, L. E.; and Thrun, S. 2005. *Principles of robot motion: theory, algorithms, and implementation*. MIT press.

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

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

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

Hoffmann, J.; Porteous, J.; and Sebastia, L. 2004. Ordered Landmarks in Planning. *JAIR* 22(1):215–278.

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

LaValle, S. M. 2006. *Planning Algorithms*. Cambridge University Press.

Liao, L.; Fox, D.; and Kautz, H. 2007. Hierarchical conditional random fields for gps-based activity recognition. In *ISRR*, Springer Tracts in Advanced Robotics (STAR). Springer Verlag.

Martin, Y. E.; Moreno, M. D. R.; Smith, D. E.; et al. 2015. A fast goal recognition technique based on interaction estimates. In *IJCAI*.

Masters, P., and Sardina, S. 2017. Cost-based goal recognition for path-planning. In *AAMAS*, 750–758. International Foundation for Autonomous Agents and Multiagent Systems.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. PDDL—The Planning Domain Definition Language. In *AIPS'98*.

Mirsky, R., and Gal, Y. K. 2016. SLIM: Semi-lazy inference mechanism for plan recognition. In *IJCAI*.

Pereira, R. F., and Meneguzzi, F. 2016. Landmark-Based Plan Recognition. In *ECAI*.

Pereira, R. F., and Meneguzzi, F. 2017. Goal and Plan Recognition Datasets using Classical Planning Domains. Zenodo.

Pereira, R. F.; Oren, N.; and Meneguzzi, F. 2017a. Landmark-Based Heuristics for Goal Recognition. In *AAAI*.

Pereira, R. F.; Oren, N.; and Meneguzzi, F. 2017b. Monitoring plan optimality using landmarks and domain-independent heuristics. In *The AAAI 2017 Workshop on Plan, Activity, and Intent Recognition*.

Pommerening, F., and Helmert, M. 2015. A normal form for classical planning tasks. In *ICAPS*.

Porteous, J., and Cresswell, S. 2002. 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)*.

Pynadath, D. V., and Wellman, M. P. 2000. Probabilistic state-dependent grammars for plan recognition. In *UAI-2000*, 507–514.

Ramírez, M., and Geffner, H. 2009. Plan recognition as planning. In *IJCAI*.

Ramírez, M., and Geffner, H. 2010. Probabilistic plan recognition using off-the-shelf classical planners. In *AAAI*.

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

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

Scala, E.; Haslum, P.; Magazzeni, D.; and Thiébaux, S. 2017. Landmarks for Numeric Planning Problems. In *IJCAI*.

Sohrabi, S.; Riabov, A. V.; and Udrea, O. 2016. Plan recognition as planning revisited. *IJCAI*.

Sucan, I. A.; Moll, M.; and Kavraki, L. E. 2012. The open motion planning library. *IEEE Robotics & Automation Magazine* 19(4):72–82.

Sukthankar, G.; Goldman, R. P.; Geib, C.; Pynadath, D. V.; and Bui, H., eds. 2014. *Plan, Activity, and Intent Recognition*. Morgan Kaufmann.

Vered, M., and Kaminka, G. A. 2017. Heuristic online goal recognition in continuous domains. In *IJCAI*.

Vered, M.; Kaminka, G. A.; and Biham, S. 2016. Online goal recognition through mirroring: Humans and agents. *The Fourth Annual Conference on Advances in Cognitive Systems*.

Wang, Z.; Mülling, K.; Deisenroth, M. P.; Amor, H. B.; Vogt, D.; Schölkopf, B.; and Peters, J. 2013. Probabilistic movement modeling for intention inference in human–robot interaction. *The International Journal of Robotics Research* 32(7):841–858.