Online Goal Recognition by Mirroring in Continuous Domains

Mor Vered, Gal A. Kaminka and Sivan Biham

Abstract

Goal recognition is the problem of inferring the (unobserved) goal of an agent, based on a sequence of its observed actions. The prevalent approach to goal recognition relies on a dedicated *plan library*, which represents all known plans to achieve known goals and facilitates efficient inference. Such methods are inherently limited to the knowledge represented in the library. In contrast, we advocate *goal mirroring*, an online recognition approach that does not rely on a plan library, but instead uses a black-box planner to generate recognition hypotheses. This paper contributes: (i) an online goal mirroring algorithm and novel heuristic for continuous spaces, empirically evaluated in two challenging domains; (ii) an exploration of the factors impacting recognition performance; and (iii) a comparison to prior methods.

1 Introduction

Goal recognition is the problem of inferring the (unobserved) goal of an agent, based on a sequence of its observed actions [Hong, 2001; Blaylock and Allen, 2004]. It is a fundamental research problem in artificial intelligence, closely related to plan, activity, and intent recognition [Schmidt *et al.*, 1978; Carberry, 2001; Sukthankar *et al.*, 2014]. The problem has many applications in continuous environments, e.g., for recognizing intended gestures and sketches [Sezgin and Davis, 2005], or for recognizing intended navigational goals [Zhu, 1991].

The prevalent approach to goal recognition relies on a dedicated *plan library*, which represents all known ways to achieve known goals [Sukthankar *et al.*, 2014]. Recognition methods—sometimes applicable to continuous domains—vary in the expressiveness of the representation and efficiency of the inference algorithms used. While powerful when the plans are known, this does not work when the observations come from an unknown plan to achieve a known goal. An additional difficulty is raised when adding goals to the set of recognizable goals, as plans for them need to be inserted in the library, in order to be recognized. Notable exceptions focus on library-free recognition, where a planner is used as a black box, to dynamically generate plans that are matched against the observations, eliminating or ranking recognition hypotheses [Ramírez and Geffner, 2009, 2010]. However, these inspiring approaches target discrete domains only, and are inefficient for online recognition, where observations are incrementally revealed.

This paper advocates *goal mirroring*, an *online* goal recognition approach which works in continuous domains. Like [Ramírez and Geffner, 2010], goal mirroring uses a planner as a black box, to generate recognition hypotheses. However, it is designed to efficiently handle incremental, continuous observations. It also uses a different ranking heuristic than earlier work, which we show is significantly superior in continuous domains.

We describe goal mirroring in detail, and report on extensive experiments over hundreds of goal-recognition problems, in two challenging continuous domains (navigation goals, sketch recognition). We empirically explore two key factors in recognizer performance: the planner used, and the ranking heuristic. Finally, we contrast the recognition results to those achievable with library-based methods and show that *goal mirroring*, utilizing no plan library, can recognize plans just as successfully.

2 Related Work

Prevalent approaches to plan- and goal- recognition rely on a dedicated *plan library* as the basis for the recognition process [Sukthankar *et al.*, 2014]. The plan library efficiently represents all known plans to achieve known goals; methods vary in the representation and inference algorithms used: action decomposition graphs [Kautz and Allen, 1986; Avrahami-Zilberbrand and Kaminka, 2005], Bayesian networks [Charniak and Goldman, 1993; Albrecht *et al.*, 1997], hidden Markov model variants [Blaylock and Allen, 2004; Bui, 2003], conditional random fields [Liao *et al.*, 2007; Vail *et al.*, 2007; Liang *et al.*, 2015], grammars [Pynadath and Wellman, 2000; Geib and Goldman, 2011], etc. The use of a library limits recognition capabilities to known plans. If the observations are of an unknown plan, even leading to a known goal, library-based methods fail. Also, when we wish to add to the set of recognizable goals, we need

to also insert plans for recognizing the new goals into the library (e.g., manually or by learning), in order for the goals to be recognized.

Some prior investigations have begun to explore alternative methods. Hong [2001] uses a specialized representation and online algorithm to generate possible goals without a plan library. Ramírez and Geffner [2009, 2010] advocate *plan recognition by planning*, where an unmodified planner is used as a black box to generate recognition hypotheses that match the observations. A heuristic comparison between the generated plans and an optimal plan that ignores the observations is used to probabilistically rank the hypotheses. Both of these approaches work in discrete domains only (e.g., using PDDL-capable planners), where there is no uncertainty in the observations, and observed actions are discretely defined. The latter may also be inefficient in online recognition, where observations are incrementally revealed.

Inspired by these investigations, *goal mirroring* uses a planner (as a black box) to generate plan and goal hypotheses on-the-fly. It departs from earlier work by addressing continuous domains, providing an efficient algorithm for online recognition and a different ranking heuristic, which we show improves recognition results.

3 Goal Recognition

We begin by giving a clear definition of the goal recognition problem (Section 3.1) proceeded by Section 3.2 with an in-depth portrayel of our the mirroring algorithm.

3.1 Problem Definition

We define R, the *goal recognition* problem in continuous domains as a tuple $R = \langle W, G, T, T_o, O, M \rangle$. $W \subseteq \mathbb{R}^n$ is the world in which the recognition problem is contained. This includes the familiar work area in $\mathbb{R}^n$ as defined in standard motion planning [LaValle, 2006]. For robot poses on flat ground, for instance, W is the space of possible positions and angle in each position (i.e., defined over $\mathbb{R}^3$). It may also include additional dimensions, such as velocity, color (to capture drawings), etc.

G is a set of $k \geq 1$ goals $g_1, \ldots, g_k$; each goal $g_j \subset W$ represents a subset of the space, e.g., a point location, a polygon drawn in some color, a trajectory, etc. T limits the duration interval $[0, T]$ in which the observed agent was active, whether observed during this time or not.

The set of observations O is defined for a subset $T_o \subseteq [0, T]$. T_o may include *specific* times in which observations were made, or *continuous* intervals of time in which observations were made. We define the set of observations $O : T_o \to 2^W$ as a mapping such that for any observation time $t \in T_o$, there exists $O(t) \subset W$, i.e., each observation is of a specific subset of the work area, e.g., a point, an edge, a trajectory segment which includes velocity, etc. Note that this permits both discrete and continuous observations.

Given the problem R, the task is to choose a specific goal $g \in G$ that *best matches* the observations O. We formulate this intuition by including M, a set of *plans*, in the definition of R, where at least one of the plans is assumed to account for the observations in O. Formally, a plan m_g^i, indexed by a goal $g \in G$, with $i \geq 1$, is a mapping $m_g^i : [0, T] \to 2^W$ from a time stamp $t \in [0, T]$ to a subset of the work area W, such that $m_g^i(1) = g$. In other words, a plan is a procedure that incrementally generates subsets of W, until the final subset at $t = 1$ is one of the goals $g \in G$. Intuitively, we are describing a plan by its effects on the world. This general definition of a plan avoids the question of the mechanisms by which effects are generated, and thus necessarily admits different approaches to representing the plan set M (as we discuss below).

A matching between a specific plan $m \in M$ and the observations O is defined by the *matching error* $e(m, O) = \int_{t \in T_o} D(O(t), m(t))$ for a distance metric D, such that $D \geq 0$ for all $t \in T_o$ (i.e., for all observations), and specifically $D = 0$ if $O(t) = m(t)$. m is said to *perfectly match* the observations if for any $t \in T_o$, $m(t) = O(t)$, in which case $e(m, O)$ is 0.

A solution to the goal recognition problem R minimizes $e(m_g, O)$; it is a member of the subset $S_R \subseteq G$ minimizing $e(m_s, O)$: $S_R = \{s | \operatorname{argmin}_{s \in G} e(m_s, O)\}$.

In general, this condition (minimizing $e(m_g, O)$) is *necessary*, but *not sufficient*. Any number of potential plans, especially in continuous domains, may perfectly match the observations, but differ in the unobserved parts. In general, a plan can perfectly match the observations, and then still achieve any goal g. For instance, in navigation goal recognition, suppose we observe points leading to a goal in the north. A path planner may generate a path that goes through all the observed points, and then doubles back to the south. Such a path will perfectly match all observations, but add an arbitrary suffix for any goal g.

This necessary—but insufficient—condition can be understood as a result of the abductive nature of plan-recognition; reasoning to the best explanation out of a potentially infinite set of explanations cannot be done without defining the necessary condition that allows to filter out non-explanations. This is what the condition above does. A second—separate—condition must define sufficiency; we discuss this in Section 3.2.

Online goal recognition. In this paper, we specifically address the online version of this problem, where the set O is revealed incrementally. Specifically, we set $t = 0$, and increment it until $t = T$. For every value of t, we may induce a goal recognition problem $R(t) = \langle W, G, t, T_o^t, O^t, M^t \rangle$, where T_o^t, O^t, M^t are defined as the respective subsets of T_o, O and M induced over the duration $[0, t]$. We denote the solution of $R(t)$ by $S_R(t)$. The objective of the online goal recognition problem is to minimize $t \in [0, T]$ such that $S_R(t) = S_R$.

In principle, it is possible to naively solve the online goal recognition problem by repeatedly calling an offline goal recognizer with the problem $R(t)$, as t increases

and the latest new observation $O^t(t)$ is made available. However, this is quite inefficient, as the growing set O^t is processed anew with every call. Thus the challenge is to determine an efficient solution.

3.2 Online Goal Mirroring

Goal mirroring is an approach to goal recognition in continuous domains, inspired by mirroring processes hypothesized to exist in primate brains [Rizzolatti, 2005]. The mirror neuron system gives humans the ability to infer the intentions leading to an observed action using their own internal mechanism. Therefore, the key to *goal mirroring* lies in the use of a planner in the recognition process. Instead of maintaining a library of plans (the set M), a planner is used to dynamically generate plans $m \in M$ as needed, without the need for explicit representation of the library M. This raises two key challenges.

First, the planner needs to generate a plan m_g that agrees with the observations, in order to minimize the matching error $e(m_g, O)$ defined above. It therefore needs to *fold the observation history* into m_g. For STRIPS-like discrete planners, Ramírez and Geffner [2010] have shown an elegant way to do this, by changing the domain theory used by the planner. But in continuous spaces, e.g., by most motion planners, this cannot be done. We therefore use an alternative, which approximates an ideal m_g that includes the observations, by appending a prefix plan (composed of the observations) and a suffix plan (composed of a new generated plan, from the latest observed state to the goal g). We denote the suffix plan m'_g.

The second challenge is that the plan m_g must also meet a *sufficiency condition*. Thus generating a plan m_g that agrees with the observations is no guarantee that the goal g is the intended goal. This is especially true in continuous domains, where infinite paths can pass through observed points.

To address this, goal mirroring is biased towards rationality. It uses the planner to also generate $\bar{m}_g$, an *ideal plan* that ignores all observations, and simply reaches g from the *initial* observed state, denoted $O(\emptyset)$. If the cost $cost(m_g)$—where m_g is made from the observations thus far and the suffix plan m'_g—greatly exceeds the $cost(\bar{m}_g)$, we rank g lower (or eliminate it from S_R). The underlying assumption here is that the ideal plan is optimal; if the observed plan is far from the ideal plan, then the agent must not be rational, and is likely pursuing an alternative goal altogether.

Algorithm 1 integrates these insights for online goal mirroring. It accepts as input a recognition problem R and a *planner* to be used as a black box. It then works as follows. First, In lines 1–2 we call the planner to compute the plan $\bar{m}_g$ (initial state $O(\emptyset)$ to each goal g, ignoring observations). This avoids recomputation of $\bar{m}_g$ with every loop iteration.

The loop in lines 3–7 is comprised of two steps. The first step (lines 4–6), centers on approximating the cost of an ideal plan m_g (which folds the observations), from the cost of the prefix (maintained by Δ), and the cost of a suffix plan m'_g.

The second step then ranks the goal hypotheses. For each goal, line 7 assigns a score, which is the ratio of the costs of $\bar{m}_g$ and the approximated m_g. As differences between them grow, the ratio of the costs decreases, resulting in a lower score. In line 8, we transform these scores into probabilities $P(G|O)$ via the normalizing factor $\eta = 1/\sum_{g \in G} score(g)$.

Algorithm 1 ONLINE GOAL MIRRORING $(R, planner)$

1: **for all** $g \in G$ **do**
2: $\quad \bar{m}_g \leftarrow planner(W, g, O(\emptyset))$
3: **for** $t = 0$ to T **do**
4: $\quad \Delta \leftarrow cost(O^t)$
5: $\quad$ **for all** $g \in G$ **do**
6: $\quad\quad m'_g \leftarrow planner(W, g, O^t(t))$
7: $\quad\quad score(g) \leftarrow cost(\bar{m}_g)/(\Delta + cost(m'_g))$
8: $\quad P(G|O(t)) \leftarrow \eta \cdot score(g)$

The choice to use this ratio in line 7 is not arbitrary. There exists evidence that human estimates of intentionality in action are heavily biased towards motion that is efficient (or rational), in the sense of preferring hypotheses that do not deviate from the optimal, rational plan from the initial state to the goal state. A cost ratio between this plan ($\bar{m}_g$ if the planner is optimal) and the observed plan m_g is known to agree with human judgement of intentions [Bonchek-Dokow and Kaminka, 2014].

A different heuristic, though motivated by the same principle, is suggested by Ramírez and Geffner [2010]: they propose looking at the *difference* in costs, i.e., a score inversely proportional to $|cost(m_{\bar{g}}) - [\Delta + cost(m'_g)]|$. We believe that a difference may be biased when dealing with larger cost values, where small differences may still be very large and skew results. For a contrast between the two heuristics see Section 4.

4 Evaluation

We empirically evaluate goal mirroring in two challenging continuous domains, over hundreds of goal recognition problems. In Section 4.1 we discuss the generality of the approach by contrasting implementations in different domains for recognizing 3D navigation goals, and hand-drawn sketches. Then we use the 3D navigation domain to evaluate the effects of the selection of the planner used in recognition , on the recognition results and runtime (Section 4.2) and the effects of different heuristic approaches (Section 4.3). We contrast goal mirroring with a method based on a plan library (Section 4.4) and draw lessons as to the advantages and disadvantages of goal mirroring. Finally, we discuss the sensitivity of the recognition approach, by contrasting results in easier and harder goal recognition problems (Section 4.5).

4.1 Two goal mirroring implementations

We implemented goal mirroring both for sketch recognition and navigation goal recognition.

Recognizing sketches of regular polygons. The task is to recognize 2D hand-drawn regular polygons. We had three people draw equilateral triangles, squares, pentagons, hexagons, septagons, and octagons, for a total of 18 recognition problems. Shapes were drawn in various scales and rotations. Naturally, hand drawings, even under ideal conditions, reflect quite a bit of inaccuracy.

The observations, of the edges incrementally added, were generated by using machine vision to analyze the drawings. Specifically, we used OpenCV to implement a Hough-transform edge detector[Duda and Hart, 1972] and to identify coordinates of the initial and last points in the drawing, marking the limits of the initial and current observed edge. To overcome scanning noise and drawing inaccuracy (which causes the Hough transform to generate multiple candidate edges) hierarchical clustering [de Hoon *et al.*, 2004] was used to filter through false edges giving us a correct estimate as to the actual number of edges.

For the mirroring process, we developed a shape-drawing planner, which takes a partial drawing (as an initial state), and a goal shape specification, and attempts to complete the drawing to the goal shape specification (or report failure if cannot be done, e.g., attempting to complete a 4-edge open polygon into a triangle). In this domain, goal hypotheses are ranked according to the ratio between the ideal internal angle size for the goal shape, and the measured internal angle, implied by each newly-added edge (Algorithm 1, line 7). Figure 3 (left), contrasts the results of this version of the sketch recognizer with a version of the recognizer which only rules out shapes inconsistent with the observations (i.e., rules out triangles if four edges have been observed).

3D Navigation Recognition. We implemented goal mirroring to recognize the goals of navigation in 3D worlds. Here, we used four off-the-shelf planners (RRT*, TRRT, RRTConnect, KPIECE1), available as part of the Open Motion Planning Library (OMPL [Şucan *et al.*, 2012]). We utilized these planners in the existing *cubicles* environment and the default robot(Figure 1). Here, the cost measure (Algorithm 1, lines 4 and 7) is simply the length of the path.

To generate goal recognition problems, we arbitrarily selected 11 points spread out relatively evenly over the environment(Figure 1-left). We then generated observed paths from each point to all others, for *a total of 110 goal recognition problems*. Each problem had between 20 and 76 observed points. Figure 3 contrasts the results of three versions of the navigation goal recognizer on an example problem of recognizing the navigation goal of a moving robot in the environment. Two of the versions utilize different planners and rank according to the heuristic and another *naive* version, which does not rank the observations or utilize a planner. Seeing as this is a continuous domain, and at all times, every one of the goals is possible, there is no elimination process and the results of this process are static; i.e. at all times, each of the goals has an equal chance of being chosen.

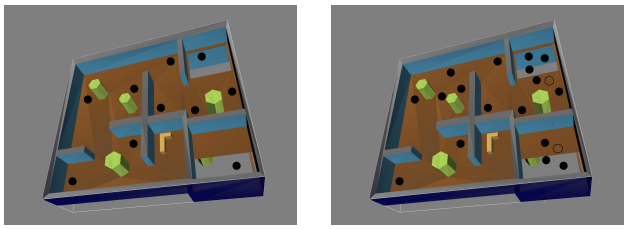


Figure 1: Left: Cubicles environment, robot and goals. Right: Cubicles environment, robot and cluster goals

Recognition results in two domains

We implemented goal mirroring in both domains, demonstrating the general applicability of our technique, and hoping to gain insight as to its strengths and weaknesses. To do this, across a wide variety of observation sequences, domains, and recognition problems, we need quantitative recognition performance measures that normalize for observation length, the size of the goal library G, and are neutral with respect to the domain. We define three such measures below, using example runs from the two domains to assist in the presentation.

Measuring recognition results. Let us examine the recognizer output on a specific problem. The two figures in Figure 2 show the recognition results in both domains, each subfigure contrasting two different experiment conditions on the same recognition problem. In each figure, the X-axis marks the observations coming in incrementally. The Y axis measures the rank of the correct goal hypothesis among all the goals ranked by the recognizer, thus lower is better (rank 1 indicates that the correct goal was ranked as the top hypothesis). Naturally, this rank is only analyzed post-hoc. The recognizer does not have access to the ground truth during the run.

For instance, in Figure 2 (left), receiving 6–9 observations, the correct goal was ranked 5 (out of 10) by the TRRT-based recognizer; it was ranked 8 by the KPIECE1-based recognizer. As more and more observations come in, it is only natural that the recognition problem becomes easier and easier, and indeed towards the end of the observation sequence, the two recognizers converge to ranking the correct goal at the top of their ranking (i.e., rank 1). Likewise in the right figure, the two variant recognizers vary in how they rank the correct goal among all hypothesized goals, but show the same trend of converging towards the correct goal as more observations come in.

Such graphs can be drawn for any online recognition problem instance, to compare the performance of different recognizers. Recognizers may vary in three measures: (1) the time (measured by number of observations from the end) in which the recognizer converged to the correct hypothesis (including 0 if it failed); (2) the area under the curve drawn in this graph, which gives a measure

of the false-positive response (greater area means recognizer tended to rank the correct hypothesis lower, farther from top); and (3) the number of times they ranked the correct hypothesis at the top (i.e., rank 1), which indicates their general accuracy.

For example in Figure 2 (left), the TRRT-based recognizer converges at observation 46, i.e., top 19.3% of the observations, whereas the KPIECE1-based recognizer converges at observation 51, i.e., was slower to converge. Normalizing for the observation sequence length, to allow comparison across different recognition problems, we measure the normalized convergence of the TRRT recognizer at 19.3%, while the KPIECE1 recognizer is measured at 10.5%. Higher results indicate earlier convergence, thus better.

Measuring the area-under-curve (auc) gives us a measure of the uncertainty the planner encountered during the recognition process. Here, a lower value is considered better indicating that the recognizer was closer to the correct ranking along most of the process. For instance, in Figure 2 (right) it is clear that the auc for the *ranking* recognizer is smaller than for the *non-ranking* recognizer. We can again normalize to allow comparison between different recognition problems, even normalizing for the number of potential goals considered. We compute the ratio of the auc to the worst-case scenario, where a recognizer consistently ranked the correct hypothesis as the lowest rank (i.e., at rank=$|G|$). A smaller percentage indicates less false positives considered by the recognizer. To be consistent with the other measures (where a larger result is better), we consider the complementary normalized value (1-normalized auc). Note that results in the two domains can be contrasted here on the same scale, as the normalization process eliminates differences due to number of potential goals (10 in the navigation domain; 6 in the sketch domain). Figure 2 (left) shows the area under the curve for the TRRT planner will be significantly smaller than the area under the curve for the KPIECE planner signifying that TRRT was closer to the correct ranking along most of the process.

Finally, counting the amount of times the planner ranked the correct goal as the top hypothesis (rank=1) gives us an overall measure of the reliability of the recognizer. The more frequently the recognizer ranked the correct hypothesis at the top, the more reliable it is, hence a larger value is better. We again normalize using the length of the observation sequence. In Figure 2 (left), the TRRT recognizer ranked the correct goal at the top 29 times, significantly more than KPIECE which ranked it only 19 times. Here, TRRT is the better recognizer in this regard. For the shape recognition the results here are 2 times for the *ranking recognizer* vs. once for the *non-ranking recognizer*. Because the length of observation sequence is only 7 (the number of edges of a septagon, which was the shape being drawn), this proves to be very significant.

Results. Figure 3 shows the results, in terms of the three criteria discussed, in the two domains. Figure 3(a) contrasts the results in the sketch recognition domain,

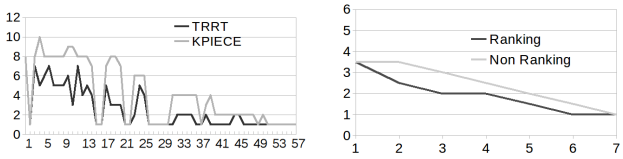
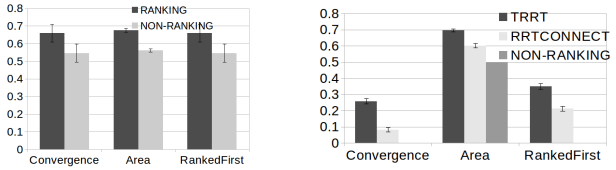


Figure 2: Left:TRRT- and KPIECE- based recognizer performance for one path containing 57 observations. Right: Performance of ranking and non-ranking recognizer observing a septagon being drawn.

when using the ranking ratio (line 8, Algorithm 1), and when not using any ranking (only filtering inconsistent hypotheses). In both cases, the shape-drawing planner described above was used. Figure 3(b) shows the results for non-ranking and ranking recognizers, but here also showing the results when using different planners as the basis for the recognition: TRRT vs RRTConnect. In both figures, higher columns denote improved results. Error bars mark the standard error.



(a) Sketched Shapes (b) Navigation Goals

Figure 3: Recognition results in two domains.

We draw several lessons from these results. First, it is clearly possible to successfully utilize goal mirroring in different domains, on challenging tasks. Second, the two critical factors that directly and significantly impact the recognition success are: the choice of the planner (TRRT is significantly better than RRTConnect), and the use of the ranking heuristic (in the sketch domain, the non-ranking procedure was able to produce some results; in the navigation domain, because no plan can ever be rules out in an online scenario, it fails completely). We explore these factors in the following sections, using the navigation domain (where we have a selection of planners).

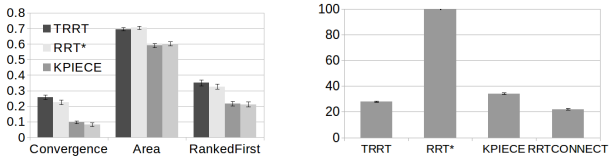
4.2 The effects of planner choice

We experimented with four off-the-shelf OMPL planners to evaluate their effect on recognition (thus, with a fixed ranking heuristic, as shown in Algorithm 1). KPIECE1 is a tree-based geometric planner that uses multiple levels to guide a frontier-based exploration of the continuous state space [Şucan and Kavraki, 2010]. The other three planners are from the RRT (Rapidly-exploring Random Trees) family [Nieto *et al.*, 2010]. Here, the tree is constructed incrementally from samples drawn randomly from the search space and is biased to grow towards large unsearched areas. The planners differ in their optimality guarantees: RRT* is an optimized planner that guarantees asymptotic optimality: it will utilize

the full duration of time allotted to it to generate incrementally improving solutions. TRRT only guarantees asymptotic *near*- optimality, preferring shorter solutions. RRTCONNECT and KPIECE1 provide no optimality guarantees whatsoever. By comparing the results of all of these planners we can get an assessment as to the power of our heuristic approach.

Figure 4(a) shows the recognition results when utilizing the various planners in goal mirroring in the 3D navigation domain. Each of the columns shows the mean recognition results over the same set of 110 recognition problems, with error bars marking the standard error. We see that TRRT and RRT* are clearly and significantly better *for goal recognition* than RRTConnect and KPIECE. Indeed RRT* and TRRT, which tend to produce paths closer to optimal, are nearly indistinguishable from each other in terms of recognition success (though TRRT is a bit better).

However, the two top planners differ from each other very much in run-time. Every call to the planners was limited to one second of run-time. But given that a planner is called with each new observation, for each one of the goals, the mean total time can grow very quickly. Figure 4(b) shows the mean running times of the planners over the entire set of 110 recognition problems (i.e., a higher value is worse). The results are shown as percentage of the total time made available to the planners. As specified, the RRT* planner uses the full time allotted to it and reaches 100%. Following is KPIECE, with 34% and then TRRT and RRTCONNECT with 28% and 22%. Indeed, we see that TRRT is the second quickest, and is beat only by RRTConnect.



(a) Recognition success. (b) Runtime (sec).

Figure 4: Comparison of planner performance.

Thus an one conclusion is that recognition results improve very much with the optimality of the planner utilized in goal mirroring. Moreover, in the 3D navigation domain, TRRT seems to offer a remarkable choice for this task: It produces good results, while being very fast.

4.3 The effects of the ranking heuristic

We have seen in Figure 3 that in both domains, the use of the ranking heuristic we proposed earlier improves the results. The comparison between the two domains yields an interesting insight as to differences between types of recognition problems. In the sketch recognition domain, recognition problems can be ruled out, as soon as the number of observed edges is greater than the number of edges in a hypothesized goal shape. This is why the non-ranking recognizer, which chooses arbitrarily between all goals consistent with the observations, still manages to provide measurable results. But in the navigation domain, goals cannot be similarly ruled out from being considered. Even if the observed agent has already navigated and reached a target goal, it is always possible (though unlikely) that it is just taking a long way out towards a different (and much farther goal). Here, then, the ranking heuristic is critical.

We therefore want to contrast the use of the ranking heuristic we propose (ratio of costs), to that of [Ramírez and Geffner, 2010] (difference of costs). Figure 5 shows the results when applying the different heuristics to all four recognizers, based on the different planners. The figure shows eight columns for each recognition performance measure: four columns measuring the results of recognizers using the *ratio* heuristic (using the four planners), and four (marked G) for the same, but using the *difference* heuristic. The results show that in the navigation domain, the ratio heuristic is clearly superior. We believe that this is because paths (to different goals) can vary much in length. The difference heuristic compares absolute differences between paths of different lengths, while the ratio heuristic compares relative differences.

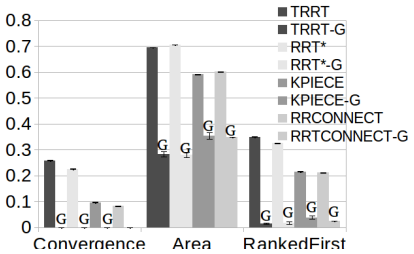


Figure 5: Comparison of different heuristics.

4.4 Mirroring vs. library-based methods

Mirroring has the principled advantage over library-based methods in terms of storage, and in being able to handle any arbitrary initial observed state; no need to add possible plans to a plan library. However, this is also a disadvantage, in principle: Mirroring does not utilize prior knowledge even when it can be made available. To evaluate this aspect, we contrast mirroring with recognition by a hidden Markov model (HMM), a popular library-based technique, often used as a standard for comparison (see Section 2). Testing the HMM on plans unknown to it is a valid, but futile exercise, where the superiority of goal mirroring would be obvious.

We therefore evaluate HMMs vs mirroring when the plans are known to the HMM. To do this, we first needed to discretize the navigation problem. We divided the 3D environment into a discrete robot-size cell grid. Each grid cell corresponded to a potential state in a hidden Markov model. We then used data from planner runs as data for training HMMs, one for each of the 110 recognition problems. In particular, since optimal generated paths seem to have done well (see above), we used 20

paths generated by the asymptotically-optimal RRT*. In this way we enhanced the HMMs prediction ability since most of the paths tend to be rather similar and very close to optimal, and also put it on the same playing field as our RRT* planner for future comparison. We had to change the path representation to contain the available states by mapping each of the points on every path to one of the cells of the grid. Thus the states and observations are both discussed in the same language. So we created a specialized HMM, trained on data proven to be useful, for each recognition problem. The HMM training and recognition were carried out using the standard MATLAB HMM package.

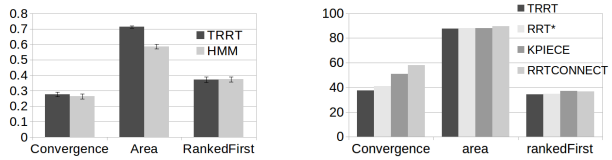
We contrasted the HMM performance with the results obtained by running TRRT with a time limit constraint of 10 seconds (Figure 6(a)). Even without any prior knowledge, mirroring is on-par with the HMM results, even better in the *area* measure.

Obviously, as more prior knowledge is available, this can change. Indeed, re-training the HMMs with 95 examples for each path, the HMM recognition results improved markedly. Our conclusion is that mirroring should be preferred when relatively less data is available, and when the number of possible plans is very large (or infinite, as in these two domains).

4.5 Sensitivity to recognition difficulty

Finally, we wanted to evaluate the sensitivity of the results shown above to the hardness of the recognition problems. We therefore added 9 goal points to the recognition problems in the navigation domain (e.g., now 19 potential goals in each recognition problem, up from 10). This means a total of 380 recognition problems (Figure 1,left) (Figure 1,right). Moreover, we added these extra 9 points specifically in close proximity to some of the preexisting 10 points, such that navigating towards any one of them appears (to human eyes) to be just as possible as any other.

Figure 6(b), shows the deterioration (%) in each of the recognition criteria, when running the recognizers using the different planners on the 380 harder problems. A lower result here is better, indicating less deterioration in performance. For the *area* and *ranked-first* measures, all planners deteriorated equally. However, TRRT proved more robust than others in the *convergence* measure.



(a) Goal mirroring vs HMM (b) Deterioration in recognition, hard problems

Figure 6: Additional experiment results.

5 Summary

We have presented *online goal mirroring*, a goal recognition approach for continuous domains that does not rely on a plan library, but instead uses a planner to generate recognition hypotheses that are continually matched against incremental observations. We evaluated goal mirroring in extensive experiments in two separate domains (navigation goals, sketch recognition) showing that goal mirroring is applicable to completely different domains. The experiments showed that two factors impact recognition success: the optimality of the planner used, and the hypothesis ranking heuristic. The experiments additionally demonstrated that mirroring can recognize plans as successfully as library-based methods.

References

D. Albrecht, I. Zukerman, and A. Nicholson. Bayesian models for keyhole plan recognition in adventure game. *User Modeling and User-Adapted Interaction*, 8(1–2):5–47, 1997.

Dorit Avrahami-Zilberbrand and Gal A. Kaminka. Fast and complete symbolic plan recognition. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-05)*, 2005.

Nate Blaylock and James Allen. Statistical goal parameter recognition. In *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS-04)*, pages 297–304, 2004.

Elisheva Bonchek-Dokow and Gal A Kaminka. Towards computational models of intention detection and intention prediction. *Cognitive Systems Research*, 28:44–79, 2014.

H. Bui. A general model for online probabilistic plan recognition. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-03)*, 2003.

S. Carberry. Techniques for plan recognition. *User Modeling and User-Adapted Interaction*, 11(1):31–48, 2001.

Eugene Charniak and Robert P. Goldman. A Bayesian model of plan recognition. *Artificial Intelligence*, 64(1):53–79, November 1993.

Michiel JL de Hoon, Seiya Imoto, John Nolan, and Satoru Miyano. Open source clustering software. *Bioinformatics*, 20(9):1453–1454, 2004.

Richard O Duda and Peter E Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972.

Christopher W Geib and Robert P Goldman. Recognizing plans with loops represented in a lexicalized grammar. In *AAAI*, 2011.

Jun Hong. Goal recognition through goal graph analysis. *Journal of Artificial Intelligence Research*, 15:1–30, 2001.

Henry A. Kautz and James F. Allen. Generalized plan recognition. In *Proceedings of the Fifth National Conference on Artificial Intelligence (AAAI-86)*, pages 32–37. AAAI press, 1986.

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

Bing Liang, Chong Chen, Ying-Hui Guan, and Xiao-Yu Huang. Estimating the missing traffic speeds via continuous conditional random fields. In *Web Technologies and Applications*, pages 35–43. Springer, 2015.

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.

Jorge Nieto, Emanuel Slawinski, Vicente Mut, and Bernardo Wagner. Online path planning based on rapidly-exploring random trees. In *Industrial Technology (ICIT), 2010 IEEE International Conference on*, pages 1451–1456. IEEE, 2010.

David V. Pynadath and Michael P. Wellman. Probabilistic state-dependent grammars for plan recognition. In *Proceedings of the 16th Annual Conference on Uncertainty in Artificial Intelligence*, pages 507–514, 2000.

Miquel Ramírez and Hector Geffner. Plan recognition as planning. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1778–1783, 2009.

Miquel Ramírez and Hector Geffner. Probabilistic plan recognition using off-the-shelf classical planners. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10)*, 2010.

Giacomo Rizzolatti. The mirror neuron system and its function in humans. *Anatomy and Embryology*, 210(5–6):419–421, 2005.

C. Schmidt, N. Sridhan, and J. Goodson. The plan recognition problem: an intersection of psychology and artificial intelligence. *Artificial Intelligent*, 11:45–83, 1978.

Tevfik Metin Sezgin and Randall Davis. Hmm-based efficient sketch recognition. In *Proceedings of the 10th International Conference on Intelligent User Interfaces*, pages 281–283. ACM, 2005.

Ioan A Şucan and Lydia E Kavraki. Kinodynamic motion planning by interior-exterior cell exploration. In *Algorithmic Foundation of Robotics VIII*, pages 449–464. Springer, 2010.

Ioan A. Şucan, Mark Moll, and Lydia E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, December 2012.

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

Douglas L. Vail, Manuela M. Veloso, and John D. Lafferty. Conditional random fields for activity recognition. In *Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-07)*, pages 1331–1338, 2007.

Qiuming Zhu. Hidden markov model for dynamic obstacle avoidance of mobile robot navigation. *Robotics and Automation, IEEE Transactions on*, 7(3):390–397, 1991.