Proc. of the 17th European Conf. on Artificial Intelligence (ECAI)
Riva del Garda, Italy. August 2006.

Situation Assessment for Sensor-Based Recovery Planning

Abdelbaki Bouguerra and **Lars Karlsson** and **Alessandro Saffiotti**[1]

Abstract. We present an approach for recovery from perceptual failures, or more precisely anchoring failures. Anchoring is the problem of connecting symbols representing objects to sensor data corresponding to the same objects. The approach is based on using planning, but our focus is not on the plan generation per se. We focus on the very important aspect of situation assessment and how it is carried out for recovering from anchoring failures. The proposed approach uses background knowledge to create hypotheses about world states and handles uncertainty in terms of probabilistic belief states. This work is relevant both from the perspective of developing the anchoring framework, and as a study in plan-based recovery from epistemic failures in mobile robots. Experiments on a mobile robot are shown to validate the applicability of the proposed approach.

1 Introduction

Mobile robots and other autonomous systems acting in the real world face the problem of uncertainty, and uncertainty can cause even the best laid plans to fail. Therefore, the ability to recover from failures is vital for such systems. AI planning is one of the tools used to recover from failures. Recovery is viewed as a planning task where the initial state of the planner is the unexpected state encountered by the agent, and the goal is to reach a nominal state where the plan can be continued. Most robotic systems employing planning [9, 1] focus largely on generation and repair of plans. However, having a good planner is not sufficient, as a planner fed with the wrong initial state will produce an inadequate plan. Consequently, a functionality for situation assessment is also needed. By situation assessment we mean the process of identifying what went wrong, and the creation of a description that reflects what is known about the real world state.

In this paper, we study the problem of situation assessment for recovering from anchoring failures in mobile robotics using sensor-based planning [3]. Anchoring is the problem of establishing the correspondence between a symbol and a percept. A type of anchoring failure is when one cannot determine which percept corresponds to a given symbol due to ambiguity: there are several plausible percepts. This requires the robot to generate a description of the current situation and generate a plan that disambiguates that situation. However, the approach in [3] does not quantify uncertainty, so there is no way to distinguish between more likely and less likely candidates. All candidates are considered equally important.

The main contribution of this paper is an approach for automatic situation assessment for recovery planning in terms of probabilistic belief states. The recovery module can assign probabilities to what perceived object should be anchored to the given symbol, and it can generate plans that disambiguate the current situation with a threshold probability instead of plans that always have to succeed. Finally, it can choose to anchor an object to a symbol with a degree of certainty; absolute certainty is not required. In fact there are situations where certain relevant properties of an object cannot be determined, and absolute certainty about which is the correct object is not attainable. Besides, the robot might have only a limited amount of time for planning. The approach in [3] did not permit this.

Besides probabilities, there are two other contributions relative to [3]. The first one is the use of background knowledge to fill in missing information about perceived objects in terms of conditional probabilities. The second is the handling of new candidate objects that are perceived at run-time. We do so by making a closed world assumption during plan generation (by considering only those relevant objects perceived so far), and we use a monitor process to check that the closed world assumption is not violated. When it is violated, a new plan that takes the new objects into account is generated.

This work is relevant both from the perspective of developing the anchoring framework, and as a case study in the more general problem of sensor-based recovery.

This paper is organized as follows: in the next section we review the concept of perceptual anchoring and the problem of ambiguity. In section 3, we present our method for automatically recovering from ambiguous anchoring situations; this is the central part of the paper. Section 4 presents 3 different experiments intented to test our approach, and section 5 provides some discussion.

2 Overview of Perceptual Anchoring

Anchoring is the process of establishing and maintaining the correspondence between the perceptual data and symbolic abstract representation that refer to the same physical object [4]. Intelligent embedded systems using symbolic representations, such as mobile robots, need to perform some form of anchoring in order to achieve their tasks. Consider a mobile robot, equipped with a vision system and a symbolic AI planner, trying to find dangerous gas bottles in a building at fire. In order to execute the ‘`GoNear(bottle1)`’ action, where the symbol ‘`bottle1`’ refers to an object described as ‘a green gas bottle’, the robot must make sure that the vision percept it is approaching is the one of the object whose identity is ‘`bottle1`’.

The symbolic description of an object with symbol o is denoted $Desc(o)$. A percept π consists of information about an object derived from sensor data (e.g. a video image), such as estimates of shape, color and other properties. These estimates are often uncertain; for instance, the vision module we use assigns degrees to the values of properties and to the classification of objects.

[1] Center for Applied Autonomous Sensor Systems - AASS; Örebro University; Sweden; Web: http://www.aass.oru.se; Email: {abdelbaki.bouguerra,lars.karlsson}@tech.oru.se; asaffio@aass.oru.se

In order to decide which percept π to anchor to a given symbol o, one needs to match $Desc(o)$ against the properties of π. The result is either *no* match, a *partial* or a *complete* match [5]. A percept π is said to be completely matching o if all the properties of $Desc(o)$ match those of π, and partially matching if it at least a property of the description $Desc(o)$ cannot be determined to match its counterpart in π due to uncertainty.

Another important distinction is whether a symbol o is making a definite reference, to exactly one object in the world ("the red ball"), or an indefinite reference, to any matching object ("a red ball").

2.1 Relational properties

There are situations when it is relevant to describe objects not just in terms of their properties but also their relations to other objects. An example is "the green garbage can that is near the red ball and the blue box". We consider the object that needs to be anchored, in the example "the green can", as the *primary object* and the other objects related to it, in the example "the red ball" and "the blue box", as *secondary objects* [3]. In this paper we use in particular binary relations and we allow for descriptions with several nested relations.

Definition 1 *Let O denote the set of object symbols. A relational description of an object $o \in O$ having m binary relations ($R_{k;1\leq k\leq m}$) with m secondary objects ($o_{k;1\leq k\leq m}$) is denoted $Rel_{desc}(o)$ and it is defined recursively as:*

$$Rel_{desc}(o) =_{def} Desc(o) \bigcup_{1\leq k\leq m} \{R_k(o, o_k)\} \cup Rel_{desc}(o_k)$$

Anchoring a symbol with a relational description, involves considering the secondary objects too. They become new objects to anchor. In practice, we always limit the depth of the relational description.

Example: In a world configuration there are three objects $O = \{o_1, o_2, o_3\}$ such that o_1 is a red ball, o_2 is a blue box, and o_3 is a cup whose color is green. We have also observed that o_1 is *near* o_2, and o_3 is *on* o_2. The relational description of o_1 is given by:
$Rel_{desc}(o_1) = \{(shape\ o_1 = ball), (color\ o_1 = red)\} \cup$
$\{(near\ o_1\ o_2), (shape\ o_2 = box), (color\ o_2 = blue)\} \cup$
$\{(on\ o_3\ o_2), (shape\ o_3 = cup), (color\ o_3 = green)\}$

2.2 Anchoring with Ambiguities

There are situations where the anchoring module cannot create or maintain an anchor for a specific symbol from its percepts because of the presence of ambiguity, i.e. it cannot determine what anchoring candidate to choose. In order to detect and resolve ambiguities, we need to consider what candidates there are to anchor the given symbol, as well as the symbols of related objects.

Definition 2 *A relational anchoring candidate for an object o having m binary relations ($R_{k;1\leq k\leq m}$) with m secondary objects ($o_{k;1\leq k\leq m}$) is represented by a list $(\pi_0, (\pi_{1,1}\ldots), (\pi_{2,1}\ldots), \ldots, (\pi_{m,1}\ldots))$ such that π_0 is a candidate percept for the primary object o, and for each secondary object o_k, a (possibly empty) list $(\pi_{k,1}\ldots)$ of all candidate percepts satisfying $Rel_{desc}(o_k)$ and relation $R_k(o, o_k)$.*

Notice that the same definition applies recursively to relational anchoring candidates for secondary objects. In fact, a relational anchoring candidate can be easily represented using an `and-or` tree where the `and` nodes represent the relations and the `or` nodes represent the candidate percepts satisfying the relation of the parent node.

A relational anchoring candidate is *completely matching* if π_0 completely matches $Desc(o)$ (the primary object), and for each secondary object o_k there is only one (definite case) or at least one (indefinite case) candidate percept π_{kj} completely matching $Desc(o_k)$. The definite/indefinite distinction here refers to whether the secondary object symbol is definite/indefinite.

A relational anchoring candidate is *partially matching* if for some o_k there is no completely matching percept π_{kl}. A special case is when there is no π_{kl} at all, i.e. the candidate list for o_k is empty.

Once each relational candidate for the desired object (primary) has been classified this way, the entire situation is handled. The anchoring module detects the presence of ambiguity on the basis of the number of complete and partial anchoring candidates, and whether the description is definite or indefinite [3]. In short, the definite case is ambiguous unless there is exactly one candidate, and it is completely matching (more than one complete match is considered a conflict). The indefinite case is ambiguous if there is no completely matching candidate. In case of ambiguity the recovery module is invoked to find a plan to acquire additional information about primary and/or secondary candidate objects if necessary.

Example The robot is presented with the symbolic description "*g1* is a garbage can near a red ball with a mark" and given the task to go near *g1*. To do this, the robot needs to anchor the symbol *g1*. Consider a situation where a garbage can π_0 and a red ball π_{11} are perceived, but no mark is visible. We have a situation where we have a singleton set $\{(\pi_0, (\pi_{11}))\}$ of relational candidates. There is one fully matching percept for the primary object, and one partial match for the secondary object: the mark was not observed. Consequently the entire relational candidate is a partial match, giving rise to ambiguity. Thus, to be sure that the observed garbage can is the requested one, the red ball has to be checked for marks from other viewpoints.

3 Recovering from Anchoring Failures

A recovery situation in anchoring typically occurs when the robot is executing some higher-level plan and encounters an ambiguous situation. Such a situation is handled in six steps:

1. Detection: the problematic situation is detected, and the top-level plan is halted.
2. Situation Assessment: the recovery module analyzes the problematic situation. If the situation is found to be recoverable, i.e. is due to uncertain perceptual information (see [3] for more details), it formulates a belief state consisting of a set of possible worlds with probabilities. This step is achieved by considering the properties of the requested object and of the perceived objects to generate different hypotheses for which of the perceived objects corresponds to the requested one.
3. Planning: the planner is called to achieve the goal of disambiguating the situation and anchoring the requested object.
4. Plan Execution: the plan is executed, and either the requested object is found and identified and can be anchored, or it is established that it cannot be identified.
5. Monitoring: if during the execution of the recovery plan, new perceived objects are encountered that completely or partially match the primary or a secondary object, then go back to step 2.
6. Resume: if recovery was successful, the top-level plan is resumed.

The authors in [3] describe how the recovery module builds the initial belief state, that encodes the different hypotheses of anchoring the desired object to one of the perceived candidates. In summary, it constructs one hypothesis, for each relational anchoring candidate, consisting of a positive (completely matching) description of the chosen candidates, and negative (non-matching) descriptions of the remaining candidates. Hypotheses with zero or two or more candidates may also be generated. These different hypotheses are then combined in a disjunction to generate a (crisp) belief state.

When we introduce probabilities, we need to be able to handle dependencies between different candidates in a graceful manner, in particular when the same percept appears more than once. Consequently, we have taken a reversed approach: instead of starting from the different candidates, we simply start from what is known about the different perceived objects, and generate a belief state from that. While doing this, we can also use background knowledge to fill in missing information about percepts. Next, we go through the possible worlds in the belief state and classify them according to what relational anchoring candidate they agree with (if any).

3.1 Background knowledge

Ambiguous situations typically involve partial matches of some perceived objects. This occurs when a certain property or relation is observed with uncertainty, or neither it nor its opposite was observed at all. Situation assessment involves considering how probable it is that those properties hold. There are two sources from which this information can be obtained.

First, the vision system (and other sensing systems) can provide us with matching degrees, which can serve as weights when assigning a probability distribution to a property or relation. For instance, if it could not be determined whether a perceived object π_1 was red or orange but red is matching better than orange, we might assert in our planner language `(color pi-1 = (red 0.6) (orange 0.4))`, which means $P(color(\pi_1) = red) = 0.6$ and $P(color(\pi_1) = orange) = 0.4$.

Second, we can use background knowledge expressed as conditional probabilities. This background knowledge is encoded as probabilistic assertion rules that can specify conditional and prior probability distributions over the values of uncertain properties of perceived objects. Probabilistic assertion rules are a part of a language used for the forward-searching planners [8, 2] to describe the results of the application of an action in a belief state. These action results may involve conditional probabilistic effects, and hence the same representation can also be used to encode background knowledge.

Example: The following rule describes the conditional probability of a perceived object `?o` containing (`has`) a substance (milk, tea or nothing) given its shape (cup, bowl, or something else).

```
(forall (?o) (perc ?o)
   (cond
      ((shape ?o = cup)
       (has ?o = (milk 0.4)(Tea 0.4)(nothing 0.2)))
      ((shape ?o = bowl)
       (has ?o = (milk 0.2)(Tea 0.3)(nothing 0.5)))
      ((true)(has ?o = (Tea 0.1)(nothing 0.9)))))
```

This rule can be used when we see an object but do not know what it contains. For instance, percepts that have the shape of a cup are believed to contain either milk, or tea (with 0.4 probability), or nothing (with 0.2 probability). Notice how conditional probabilities are defined using the `cond` form which specifies conditional outcomes. It works like a LISP `cond` where each clause consists of a test (which may refer to uncertain properties) followed by a consequent formula. The `forall` assertion formula allows to iterate over elements of a certain type (here `perc` for percept), to execute an assertion formula (here the `cond` form).

Assertion formulas are applied to belief states, and belief states are probability distributions over sets of possible worlds (crisp states). The assertion formula is applied to each possible world in the belief state, resulting in new possible worlds. These new possible worlds and their probabilities constitute the new belief state(s). Note that the belief state explicitly represents a joint probability distribution over the uncertain properties and relations in the situation.

3.2 Situation Assessment

Once the recovery module is notified of the failure to anchor an object o due to ambiguity, it starts a situation assessment process aiming at creating a belief state representing the different states the world may actually be in, given present uncertainties. The situation assessment process takes a set of relational anchoring candidates as input, and is performed in four steps.

1) Initialization: a number of properties not related to the anchoring candidates are assessed, such as the position of the robot and the topology of the room.

2) Description generation: the descriptions $Desc(\pi_j)$ and $R_k(\pi_j, \pi_k)$ are computed for all the perceived objects π_j in the relational anchoring candidates. The properties and relations that are considered are those appearing in $Rel_{desc}(o)$ for the object o such that π_j is a candidate object for o. Uncertain properties are estimated in the manner described in section 3.1, by using matching degrees and background knowledge. The result of the first step is a belief state representing a joint probability distribution over all uncertain properties and relations of the perceived objects in the candidates.

3) Classification: the possible worlds of the belief state are partitioned into three different sets for the definite case and two sets for the indefinite case. For the definite case, one partition contains the possible worlds where there is a unique matching candidate. A second partition contains those worlds where there is a conflict due to the presence of more than one matching candidate, and a third set includes worlds where there is no matching candidate. Partitioning relies on the evaluation of three existential formulas derived from $Rel_{desc}(o)$. Those formulas test if there is exactly one, two or more, and no relational anchoring candidate that matches $Rel_{desc}(o)$, respectively. In these formulas, the object names in $Rel_{desc}(o)$ are replaced by existentially quantified variables.

When the recovery module deals with an indefinite case, the partitioning yields only the set where there is no matching candidate and the set where there is at least a matching candidate (in the indefinite case there is no situation of conflict).

Sometimes, one might want to provide more weight to the possible worlds where there is exactly one anchoring candidate that matches. After all, if the robot was ordered to go to "the container with milk" it might be reasonable to consider it likely that there is exactly one such object. Hence, at this step one may discount the probability of the possible worlds with none or too many matching candidates using a discount factor α and then renormalize the probability distribution of the possible worlds to sum to one.

4) Labeling: The formula `(anchor o` π`)` is added to each possible world where percept π is a fully matching candidate, and the formula `(anchor o null)` to the set of non-matching worlds (and conflict worlds for the definite case) to ascertain that the desired object o is not possible to anchor. The percepts π are those binding the variables in the existential formulas in step 3.

Example Assume that the plan executor could not anchor object *C1* described as the container with milk which is near the fridge, because there are two perceived container objects π_c and π_b near the fridge π_f: π_c is a green cup and π_b is a blue bowl. Let's also assume that the recovery module uses the probabilistic conditional rule from section 3.1.

In step one, it is asserted that the robot is at the entrance of the coffee room and so on.

There are two relational candidates: $(\pi_c\ (\pi_f))$ and $(\pi_b\ (\pi_f))$. Step 2 consists of computing the description of all the perceived objects appearing in the relational candidates and relations among them to build an initial belief state for the subsequent steps.

The descriptions we obtain are a set Rel of two relations $\{(\text{near}\ \pi_c\ \pi_f), (\text{near}\ \pi_b\ \pi_f)\}$, and the following descriptions of the different perceived objects:

$$Desc(\pi_c) = \{(\text{perc}\ \pi_c), (\text{shape}\ \pi_c = \text{ cup}), (\text{color}\ \pi_c = \text{ green})\}$$
$$Desc(\pi_b) = \{(\text{perc}\ \pi_b), (\text{shape}\ \pi_b = \text{bowl}), (\text{color}\ \pi_b = \text{red})\}$$
$$Desc(\pi_f) = \{(\text{perc}\ \pi_f), (\text{shape}\ \pi_f = \text{fridge})\}$$

We assert these descriptions, apply the background knowledge in section 3.1 and obtain a belief state bs with four possible worlds $w_1, \ldots, w_4$ such that:

$$w_1 = \{(\text{has}\ \pi_c = \text{milk}), (\text{has}\ \pi_b = (\text{Tea}\ 3/8)(\text{nothing}\ 5/8))\}$$
$$w_2 = \{(\text{has}\ \pi_c = \text{milk}), (\text{has}\ \pi_b = \text{milk})\}$$
$$w_3 = \{(\text{has}\ \pi_c = (\text{Tea}\ 4/6)(\text{nothing}\ 2/6)), (\text{has}\ \pi_b = (\text{Tea}\ 3/8)(\text{nothing}\ 5/8))\}$$
$$w_4 = \{(\text{has}\ \pi_c = (\text{Tea}\ 4/6)(\text{nothing}\ 2/6)), (\text{has}\ \pi_b = \text{milk})\}$$

In addition, they all contain $Rel \cup Desc(\pi_c) \cup Desc(\pi_b) \cup Desc(\pi_f)$. The probability distribution over the possible worlds is:

$$p(w_1) = 0.4 \cdot 0.8 = 0.32;\ p(w_2) = 0.4 \cdot 0.2 = 0.08$$
$$p(w_3) = 0.6 \cdot 0.8 = 0.48;\ p(w_4) = 0.6 \cdot 0.2 = 0.12$$

We compute $p(w_1)$ as the joint probability of π_c containing milk and π_b not containing milk, and so on.

In step 3, the situation assessment module classifies the worlds according to the number of matching candidates. In worlds w_1 and w_4 there is one and only one matching candidate. In world w_2, there are two matching candidates, therefore we have a conflict. Finally in world w_3 there is no candidate object matching $Rel_{desc}(c1)$ i.e. the test formula for no matching candidates, $\neg\exists x_1, x_2((\text{has}\ x_1 = \text{milk}) \wedge (\text{near}\ x_1\ x_2) \wedge (\text{shape}\ x_2 = \text{fridge}))$, holds in w_3.

In step 4, the situation assessment module adds $(\text{anchor}\ c_1\ \pi_c)$ to w_1, $(\text{anchor}\ c_1\ \pi_b)$ to w_4, $(\text{anchor}\ c_1\ \text{null})$ to both w_2 and w_3.

3.3 Planning and Monitoring

3.3.1 Planning and Execution

Once the recovery module has created the belief state encoding the possible worlds, it passes it to the planner together with the goal of finding an anchor to the requested object. We use probabilistic conditional planners for stochastic partially observable domains with actions and sensing modeled in a POMDP-equivalent manner [8, 2].

The generated plans incorporate motion actions and sensing actions aiming at collecting more information about the unknown properties of the perceived objects. They are guaranteed to solve the planning problem (if it is solvable) with a certain probability of success.

The anchoring plan is then sent to the plan execution and monitoring module. Fig. 1 shows the global architecture of our mobile robot. The recovery module is part of the block labeled Executor/Monitor.

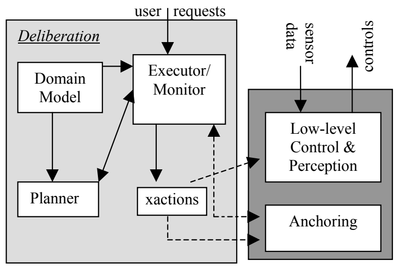

Figure 1. Plan execution and monitoring architecture

The actions such as (smell π_c) and (move-to pos2) are translated into executable perceptual and movement tasks (xactions in the figure) that are sent to the low-level control architecture for execution. The anchoring module provides the plan executor/monitor with the results of perceptual task, which allow the latter to keep an updated belief state and to monitor how the plan progresses. The domain model module in the figure stores planning domains (actions, axioms and predicates) as well as the different conditional rules for building the background knowledge.

3.3.2 Closed World Assumption Monitoring

The recovery plans are generated under the assumption that all relevant candidates have been observed, with the exception of the explicit "missing objects" used when there is no candidate at all. However, there may be additional candidate objects that are not visible from the initial position of the robot, but become visible as the robot moves around while executing its recovery plan. The anchoring system regularly checks for the appearance of new percepts matching the description of the requested object. If such a percept is detected, the assumption of knowing all relevant objects has been violated, and the current recovery plan is considered obsolete. Hence, a new initial belief state is produced, taking into account the previous perceived objects and the information we have gained of those, as well as the new perceived object, and a new recovery plan is generated. This new plan replaces the old one.

Replanning for new candidate objects is not only used for the unexpected discovery of a new object: it is also instrumental in the case where the robot was explicitly searching for a candidate, and found a partially matching one. In such a situation, one cannot jump to the conclusion that the correct object has been found, but must plan to find more information about the object in question (together with the other remaining candidate objects).

4 Experiments

In [3] it has been experimentally demonstrated that the ability to recover from anchoring failures can improve the robustness of the behaviors of a mobile robot. Here, we intend to demonstrate how our approach can handle three different scenarios where probabilities are required. Our experiments are intended to show the feasibility of our approach; not to evaluate its performance. The experiments were run on a Magellan Pro Research Robot equipped with a CCD camera. The recovery planning time was always less than two seconds.

Scenario 1 serves to demonstrate that we can handle cases were there is no recovery plan that is certain to succeed. Our robot is to approach a marked gas bottle (indefinite reference). From its initial position, it perceives two gas bottles π_1 and π_2, but no mark is perceived on them. The mark may be on one of four sides of each gas bottle. However, the presence of obstacles prevents observing them from all sides. The situation assessment produced a belief state consisting of four possible worlds $w_1, \cdots, w_4$ with a uniform probability i.e. $p(w_i) = 1/4; 1 \leq i \leq 4$. Each world reflects if π_1 or π_2 is having a mark, and if it had a mark, on which side it is. A plan was generated in which the robot is to move to the different accessible positions to look for a mark on the perceived gas bottles. When observation actions are performed, the probabilities of the possible worlds are updated accordingly. For instance, if no mark is seen on one side of π_1, the probability that π_1 is a match decreases. In one of the runs, the robot had not found a mark after observing π_1 from three sides, and π_2 from two sides. At this stage, the anchoring module decided to anchor the more likely candidate π_2 but with a low degree of certainty. The scenario was also run successfully under different configurations (in terms of the locations of the gas bottles, whether they were marked, and observability of the marks on them).

Scenario 2 concerns run-time handling of new objects. Again, our robot is to approach a gas bottle with a mark upon it (indefinite reference). This time, only one gas bottle π_1 is initially perceived. The situation assessment produced a belief state with two uniformly distributed possible worlds w_1 (π_1 is marked), and w_2 (π_1 is not marked). In this scenario, all sides are observable, therefore in w_1 there is a probability distribution for which side, of the gas bottle, has the mark. When the recovery plan is executed, and the robot has moved to a different position, it detects a second gas bottle initially occluded by the first one. The situation is reassessed (to include the new perceived gas bottle) and a new recovery plan is produced that includes actions to check both the old and the new gas bottle. Also, this scenario was successfully run under different configurations (which of the bottles was marked, and which side of the bottle the mark was on).

Scenario 3 involves using background knowledge to resolve an ambiguous situation, where planning time is very short. The robot has to achieve the same task as before, but in addition there is the background knowledge that 90% of brown gas bottles have a mark, and only 20% of green ones have a mark. From its initial position, the robot perceives two gas bottles: one green and one brown. Using the background knowledge, the situation assessment procedure notices that the brown gas bottle has a higher chance of matching the description and decides to anchor it directly.

5 Discussion and Conclusions

We have presented a probabilistic approach to recovering from ambiguous anchoring situations, i.e. when one cannot determine which percept corresponds to a given symbol. The only previous approach [3] was non-probabilistic, and could not distinguish between more likely and less likely candidates, nor could it handle cases were the ambiguous situation could only partially be resolved.

Our approach (just like the previous one) builds on the steps detection, situation assessment, plan generation and execution. In addition, we have added monitoring for the appearance of new relevant objects. We have focused on the assessment step, where we have introduced the possibility to use matching degrees for percepts as well as background knowledge specifying conditional probabilities to determine the probability distributions for uncertain or unknown properties and relations.

The background knowledge is specified in terms of assertion formulas in the language of our planners. That has the advantage that they can be applied directly to the belief states that are to be used by the planners, but they do not offer a compact representation of uncertainty. Therefore, we are presently considering other representations for our situation assessment steps. Probabilistic relational models (PRMs) [6] generalize Bayesian Networks to represent first order knowledge i.e. knowledge involving classes of objects where some attributes in a specific class are uncertain and may depend on attributes of the same class or another one. Relational Bayesian Networks [7] are also a tool that uses Bayesian Networks to represent objects explicitly and relations among them. Finally, the Bayesian language BLOG [10] is a tool that can be used to create first order probabilistic models involving unknown number of objects. BLOG deals with possible worlds of different numbers of objects and uncertainty about their attributes. That might offer a way to reason and plan explicitly about unobserved but possibly present objects. However, we have favored handling that problem by interleaving planning and execution, a desirable characteristic for acting under uncertainty due to lack of information [11].

The particular problem we have addressed in this paper concerns the important problem of anchoring: if a mobile robot with symbolic cognitive layer is to interact with different objects in its environment, it will frequently need to anchor those objects. A robust anchoring mechanism is then a prerequisite for robust behaviors. But the problem is also interesting as a case of what can be called epistemic recovery: if the plans of a robot or other agent is stalled due to lack of information, how can that be handled? We believe our work has demonstrated the usefulness of probabilistic sensor-based planning, and the important role of situation assessment in such unexpected situations.

ACKNOWLEDGEMENTS

This work has been supported by the Swedish KK foundation and the Swedish research council.

REFERENCES

[1] M. Beetz, 'Runtime plan adaptation in structured reactive controllers', in *Proc. of the 4th Int. Conf. on Autonomous Agents*, pp. 19–20, (2000).
[2] A. Bouguerra and L. Karlsson, 'Hierarchical task planning under uncertainty', in *Proc. of the 3rd Italian Workshop on Planning and Scheduling*, (2004).
[3] M. Broxvall, S. Coradeschi, L. Karlsson, and A. Saffiotti, 'Recovery planning for ambiguous cases in perceptual anchoring', in *Proc. of the 20th National Conf. on Artificial Intelligence*, pp. 1254–1260, (2005).
[4] S. Coradeschi and A. Saffiotti, 'Anchoring symbols to sensor data: preliminary report', in *Proc. of the 17th National Conf. on Artificial Intelligence*, pp. 129–135, (2000).
[5] S. Coradeschi and A. Saffiotti, 'Perceptual anchoring of symbols for action', in *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence*, pp. 407–412, (2001).
[6] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer, 'Learning probabilistic relational models', in *Proc. of the 16th Int. Joint Conf. on Artificial Intelligence*, pp. 1300–1309, (1999).
[7] M. Jaeger, 'Relational Bayesian networks', in *Proc. of the 13th Conf. on Uncertainty in Artificial Intelligence*, pp. 266–273, (1997).
[8] L. Karlsson, 'Conditional progressive planning under uncertainty', in *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence*, pp. 431–438, (2001).
[9] S. Lemai and F. Ingrand, 'Interleaving temporal planning and execution in robotics domains.', in *Proc. of the 19th National Conf. on Artificial Intelligence*, pp. 617–622, (2004).
[10] B. Milch, B. Marthi, S. Russell, D. Sontag, D. L. Ong, and A. Kolobov, 'Blog: Probabilistic models with unknown objects', in *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence*, (2005).
[11] I. Nourbakhsh, 'Using abstraction to interleave planning and execution', in *Proc. of 3rd Biannual World Automation Congress*, (1998).