To secure an anchor — a recovery planning approach to ambiguity in perceptual anchoring

Lars Karlsson [a,*], Abdelbaki Bouguerra [a]
Mathias Broxvall [a] Silvia Coradeschi [a]
Alessandro Saffiotti [a]

[a] *Center for Applied Autonomous Sensor Systems Dept. of Technology, Örebro University Fakultetsgatan 1, S-70182 Örebro, Sweden E-mail: {lars.karlsson, abdelbaki.bouguerra, mathias.broxvall, silvia.coradeschi, alessandro.saffiotti}@tech.oru.se*

An autonomous robot using symbolic reasoning, sensing and acting in a real environment needs the ability to create and maintain the connection between symbols representing objects in the world and the corresponding perceptual representations given by its sensors. This connection has been named *perceptual anchoring*. In complex environments, anchoring is not always easy to establish: the situation may often be ambiguous as to which percept actually corresponds to a given symbol. In this paper, we extend perceptual anchoring to deal robustly with ambiguous situations by providing general methods for detecting them and recovering from them. We consider different kinds of ambiguous situations. We also present methods to recover from these situations based on automatically formulating them as conditional planning problems that then are solved by a planner. We illustrate our approach by showing experiments involving a mobile robot equipped with a color camera and an electronic nose.
Keywords: planning, cognitive robotics, perceptual anchoring

[*] Corresponding author: Lars Karlsson

1. Introduction

Autonomous systems embedded in the physical world typically incorporate two different types of processes: high-level cognitive processes, that perform abstract reasoning and generate plans for actions; and sensory-motoric processes, that observe the physical world and act on it. These processes have different ways to refer to physical objects in the environment. Cognitive processes typically (although not necessarily) use symbols to denote objects, like ‘b1’. Sensory-motoric processes typically operate from sensor data that originate from observing these objects, like a region in a segmented image. If the overall system has to successfully perform its tasks, it needs to make sure that these processes “talk about” the same physical objects. We call this the anchoring problem [13].

An important challenge in anchoring is to resolve situations where the sensors detect several objects that are consistent with the symbolic description of a desired object. In order for an autonomous system to function robustly when encountered with such ambiguous situations, it needs to reason and act in a way that allows it to distinguish between the perceived object and determine which one is the correct one. We have first investigated ambiguity in anchoring in [10] and [9]. In this paper, we take a more general approach and analyze different types of ambiguity that can make the anchoring process fail. We then propose a general approach to automatically detect and isolate these failures, and to automatically generate a conditional plan to recover from the failure when possible. The plan involves finding out relevant information about the objects involved until the correct object is identified. Our approach also includes the ability to replan in case new objects that can serve as candidates are found. This paper is an extension of [9] with the additions of a probabilistic world model and replanning for new objects. This leads to more robust recovery: our recovery planner does not need to find a plan that works all the time, but just with a certain probability, and if new, relevant objects are discovered, the plan can be updated to take these into account.

As an additional contribution, we extend the anchoring framework to handle symbolic descriptions including relations among objects, like "the red ball near the yellow can". Relations are an important aspect in the description and recognition of objects and they contribute to enrich the anchoring process by making more complex anchoring cases possible.

The motivation for using recovery planning is the possibility to handle complex situations in an optimal way in terms of, e.g., the number of observation and movement actions required. Due to the combinatorial nature of the problem with different types of failures, involving different numbers of objects characterized by different properties and (possibly nested) relations, the number of recovery situations and plans grows exponentially and it becomes infeasible to use a hand-coded approach.

There are a number of systems that use planning-based techniques to perform recovery. However, most systems that take this approach (e.g., [19, 5,32,34,22]) focus on the external state of the world, looking for discrepancies between the observed state and the expected one. In comparison, our work focus on the inability to acquire the perceptual information needed to make a certain decision, in particular about how to anchor a specific symbol. The problem of perception is still unsolved in the general case and unreliable in most practical situations. By performing recovery at a higher cognitive level, we can increase the robustness of the system in face of imperfect perception, and handle cases which are inherently ambiguous even with a perfect perception system.

Several works have addressed the problem of planning for perceptual actions. Perception planning has been studied as a means for gathering better visual information [26,3], for achieving safer landmark-based navigation [27,20], for performing tasks that involve sensing actions [21,25], and for generating image processing routines [4]. None of these works, however, deal with the problem of recovery.

A problem which has similarities to anchoring recovery is active object recognition using computer vision [15,6,1]. This problem typically has three ingredients: (1) an object that has been encountered and that needs to be recognized (2) a mobile sensor platform (e.g. a robot with a camera) and (3) a set of models of the objects of interest typically derived from images from different views of the modeled objects. The problem then is to determine which object model corresponds to the encountered object. In order to do that, the sensor platform can be moved to obtain images from other positions. The problem is how to do that in order to optimize the discriminatory effect of the images, and typically some notion of information entropy is used. Compared to anchoring recovery planning, there are three important differences. First, active object recognition typically involves a single object, as opposed to several objects in anchoring recovery. Second, object recognition is typically concerned with classifying an object, whereas anchoring is concerned with the identity of the object (not just the class). Third, active object recognition uses an object model in or derived from the perceptual (image) domain to match against the encountered object, whereas anchoring recovery uses a symbolic description of the requested object that is matched against the different perceived candidates.

In the next section, we give a brief reminder of perceptual anchoring. In section 3 we analyze different types of ambiguity, and explain how the ambiguity can be detected and dealt with. In section 4 we show how the ambiguous situation can be modeled in a planner and a recovery plan generated automatically for those cases that can be resolved. In section 5, finally, we demonstrate our technique by presenting a series of experiments run on a mobile robot.

2. Perceptual Anchoring

Anchoring is the process of creating and maintaining the correspondence between symbols and sensor data that refer to the same physical objects. In our work, we use the computational framework for anchoring defined in [11]. In that framework, the symbol-data correspondence for a specific object is represented by a data structure called an **anchor**. An anchor includes pointers to the symbol and sensor data being connected, together with a set of properties useful to re-identify the object, e.g., its color and position. These properties can also be used as input to the control routines.

Consider for concreteness a mobile robot equipped with a vision system and a symbolic system including a planner and a knowledge base. Suppose that the planner has generated the action 'GoNear(b1)', where the symbol 'b1' denotes an object which is described in the knowledge base as 'a tall green gas bottle'. The 'GoNear' operator is implemented by a sensori-motor loop that controls the robot using the position parameters extracted from a region in the camera image. In order to execute the 'GoNear(b1)' action, the robot must make sure that the region used in the control loop is exactly the one generated by observing the object that the planner calls 'b1'. Thus, the robot uses a functionality called **Find** to link the symbol 'b1' to a region in the image that matches the description 'a tall green gas bottle'. The output of Find is an anchor that contains, among other properties, the current position of the gas bottle. While the robot is moving, a functionality called **Track** is used to update this position using new perceptual data. Should the gas bottle go out of view for some time the **Reacquire** functionality would be called to update the anchor as soon as the gas bottle is in view again.

The description 'a tall green gas bottle' is obtained from the knowledge base of the robot, which typically will include causal knowledge in the form of planning operators, a topological map of the environment, information about different objects, and possibly also other forms of knowledge like object taxonomies [8]. It is in symbolic terms — 'tall', 'green' and 'gas bottle' are all symbols. The vision system, on the other hand, produces percepts. A percept is an aggregation of relevant information from e.g. a region in an image, and contains a number of attribute-value pairs such as average color (represented as Red-Green-Blue or Hue-Saturation-Value) and shape (represented as the distance from the center to the border of the region relative to a fixed set of angles). In order to relate the symbols used in the descriptions to the attributes of the percepts, one need to specify a grounding relation. For instance, the symbol 'green' is interpreted as the attribute 'average color' belonging to a specific subspace of the RGB or HSV color space.

More details on perceptual anchoring can be found in [11,12,13].

2.1. *Matching*

A central ingredient in all the anchoring functionalities is the *matching* between the symbolic description given by the planner and the attributes of percepts generated by the sensor system. This is needed to decide which percepts to use to create or update the anchor for a given symbol. Matchings between a symbolic description and a percept can be *partial* or *complete* [12].

Definition Given a description σ and a percept π, we say that π *fully matches* σ if each symbolic property in σ is matched by the corresponding attribute in π according to the grounding relation. π *partially matches* σ if there is some property in σ that is not observed in π, but all properties in σ that corresponds to some observed attribute π are matched. Otherwise π *does not match* σ.

For example, consider the description "a gas bottle with a yellow mark". A gas bottle in an image where no mark is visible provides a partial match, since the mark might not be visible from the current viewpoint. Another example of partial match is the case when several cups are identical from the point of view of vision, but they can be distinguished by using the sense of smell. A percept is said to be a complete anchoring candidate for a symbol if it fully matches the symbolic description of the symbol, and a partial anchoring candidate if it partially matches the description.

2.2. Definite and indefinite descriptions

The symbolic descriptions used in the anchoring process can be either definite or indefinite. A description is *definite* when it denotes a unique object, for instance "the cup in my office". Linguistically one uses in this case the article "the". An *indefinite* description requires that the object corresponds to the description, but not that it is unique, for instance "a red cup". Definite descriptions are especially challenging when an object is conceptually unique, but its perceptual properties do not characterize it unequivocally, for instance "the cup that I have seen before". This is a common event in the Reacquire functionality when more than one object matches the description of a previously seen object (in Reacquire, descriptions are always definite). An example of this situation is shown later in this paper.

3. Anchoring with Ambiguities

The matching process described in the previous section provides complete and partial anchoring candidates for a symbol. The anchoring module detects the presence of ambiguity on the basis of the number of complete and partial anchoring candidates, and whether the description involved is definite or indefinite. Table 1 summarizes the cases that can occur.

In cases 1 and 2 no anchoring candidates have been found fully matching the symbolic description. In case 1 the recovery module can try to recover by making a search. In case 2 temporary anchors are created for each of the partial candidates and these anchors are returned to the recovery module. A recovery plan is constructed and executed aiming at observing the missing properties of the object. If the situation is successfully disambiguated, the planner informs the anchoring module about which of the candidate perceived objects should be used for anchoring.

Case 3 is the ideal case where just one complete candidate is present. The anchoring module selects that percept.

In cases 4 and 5 at least one complete candidate for the symbol is present. If the symbolic description is indefinite any one of these complete candidates can be selected for anchoring the symbol. If the description is definite the presence of several candidates can constitute a problem. In case 4 where a complete candidate and partial ones are present a cautious approach consists of constructing and executing a recovery plan aiming at observing the missing properties of the partial candidates in order to rule them out. However the complete candidate could also be selected, this is why we have ok/fail as result. Case 5 constitutes a serious problem: as the matchings are full, the situation cannot be resolved by getting more perceptual information. Instead, the description has to be considered insufficient, and needs to be made

	# Matches		Definite		Indefinite	
Case	full	partial	result	action	result	action
1	0	0	Fail	Search	Fail	Search
2	0	1+	Fail	Observe	Fail	Observe
3	1	0	Ok	—	Ok	—
4	1	1+	Ok/Fail	-/Observe	Ok	—
5	2+	any	Conflict	—	Ok	—

Table 1

The cases that can occur during anchoring

more precise (how to do that is not addressed in this paper).

Finally, we should point to some particularly difficult situations: when important characteristics of the object have changed in an unpredictable way (e.g., the shape has been deformed); and when our percepts are not just uncertain but wrong (e.g., a reflection is seen as a mark). In such cases, we might get mismatches that should have been matches, and vice versa, which leads to an erroneous estimate of the situation and hence does not allow a correct recovery.

3.1. Relational properties and ambiguity

An interesting challenge in the treatment of ambiguity is represented by situations where an object can be described not only by its properties, like color and shape, but also by its relations to other objects. By considering relations, we may be able to resolve cases where the known properties of the object are not sufficient to distinguish it from other similar 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", *secondary objects.* In this paper we consider in particular binary relations and we allow for descriptions with several relations.

The anchoring process handles relational cases by considering the relation as an additional property of the primary object. In the previous example, being "near the red ball" is an additional property of the object besides being a garbage can and being green. Clearly a relational property has the additional complexity that an anchor needs to be found also for the secondary object. The anchoring process for the secondary object is the same as the one for the primary object: the secondary object can be described as definite or indefinite and it can have complete, partial or no anchoring candidates.

In practice we first consider all possible candidates for the primary object based on the non-relational properties in its description. Then for each of these candidates we try to find anchors for all secondary objects on the basis of their descriptions and their relations to the primary object. A *relational anchoring candidate* is represented by a list $(\pi_0, (\pi_{1,1} \ldots), (\pi_{2,1} \ldots), \ldots)$ containing a candidate percept π_0 for the primary object and for each secondary object, a (possibly empty) list of all candidate percepts satisfying the expected relation.

We extend our definitions of matching to relational anchoring candidates as follows. A relational anchoring candidate is *completely matching* if, by applying the table of cases above, the result of anchoring is "OK" for the primary object candidate (viewed as a single-element list) and for each secondary object candidate list. For instance, in the example above, if we obtain an anchoring candidate (pi1 (pi2) (pi3)) where pi1 is a percept of a green garbage can, pi2 of a red ball, and pi3 of a blue box, and the latter two are both perceived as being near the can, then this constitutes a complete matching.

A relational anchoring candidate is *partially matching* if the result of anchoring is "fail" for at least one of the candidate lists, but for no object the result is "conflict". A "fail" indicates that an object is not seen or some of its properties are not perceived, for instance a mark is not visible. An example of a partial match would be (pi1 (pi2) ()). The planner can in this case select searching and observation actions.

Finally, the presence of a "conflict" indicates that one of the objects was described as unique, but more than one object was found corresponding to the description. For instance, given that the object to anchor was "the green garbage can near **the** red ball" a "conflict" would be present if a garbage can and two red balls both near the can were detected. In this case additional observation actions would not help and a more accurate description would be needed.

Once each relational candidate has been classified this way, the entire situation is classified according to the table of cases based on the number of complete and partial relational anchoring candidates. For instance, if $a_1 =$ (pi1 (pi2) (pi3)) (complete) and $a_2 =$ (pi5 (pi6) ()) (partial) then we have case 4. In case of ambiguity the recovery module is invoked, which devises a plan to acquire additional information about primary and/or secondary candidate objects according to the matching results of the objects.

The above concepts are illustrated in the following example. This example will also be used throughout the next section.

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 and a red ball are seen, but no mark is visible. Applying our table of anchoring cases we find that this corresponds to case 3, one fully matching percept (ok), for the primary object and case 2, one or more partial matches (fail), for the secondary object. This implies that the entire relational candidate is a partial match. Applying the case table to our singleton set of relational candidates, we find that we have case 2, a partial match (fail). Thus, to be sure that the observed garbage can is the requested one, the red ball has to be observed further to test if it has any marks visible from other viewpoints.

Notice that we allow cases when the secondary object is not initially found due to, for instance, occlusion. The framework also allows for nested relations, for instance "the red ball near the blue box touching the green can". An additional case we have not considered yet is when the relational property per se is not observable, for instance it cannot be established if two objects are touching from the current view.

4. Recovery Planning for Anchoring

We propose an approach to actively recover from the recoverable cases above by automatically analyzing and encoding the ambiguous situation as a planning problem and generating a conditional recovery plan. In practise we use a recovery module using a conditional planner called PTLplan [24] which can use either probabilistic or possibilistic [17] uncertainty. PTLplan searches in a space of belief states, where a belief state represents the agent's incomplete and uncertain knowledge about the world at some point in time. A belief state can be considered to represent a set of hypotheses about the actual state of the world, for instance that a certain gas bottle has a mark on it or has not a mark on it. It can also assign probabilities or possibilities to these hypotheses. Actions can both have causal effects that change properties in the world, and observation effects that lead to a splitting up of a belief state into several new and more informative belief states. The latter leads to conditional branches in the plan. We omit further details of how plans are generated here, as our approach is not dependent on the particular planning algorithm. Actually, another planner for partially observable and probabilistic domains could in principle have been used instead (e.g. [31,30,23,2]).

A recovery situation in anchoring typically occurs when the robot is executing some higher-level plan and encounters one of the ambiguous but recoverable cases above. Such a situation is handled in five steps:

1. The problematic situation is detected and classified as above, and the top-level plan is halted.
2. The recovery module formulates an initial situation (belief state) by considering the properties of the requested object and of the perceived objects, and generating different hypotheses for which of the objects corresponds to the requested object. It also formulates a goal that the requested object should be identified if present.
3. The recovery module calls the planner with the belief state and the goal as input, and a plan is returned.
4. 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. 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. If recovery was successful, the top-level plan is resumed.

We now present steps 2, 3 4 and 5 in more detail.

4.1. Formulating the initial situations and goals

The initial situation (which specifies the initial belief state for the planner) and the goal are constructed differently depending of what case of ambiguity we have.

When there are one or more partially matching anchoring candidates, the agent needs to figure out which of them actually corresponds to the requested object s. Thus, the recovery module formulates a set of hypotheses that consist of the different ways s can be anchored, based on the known properties of s and its secondary objects and the observed properties of the perceived objects.

4.1.1. Initial situation for definite case

We start by presenting the procedure for definite anchors. We deviate from the Lisp-style syntax to a logic-like syntax in order to enhance readability. $\bigvee_i (p_i : d_i)$ represent a number of disjoint alternative hypothesis d_i with probabilities p_i. One can also use possibilities [17], in which case we obtain a possibility distribution over hypotheses instead (this is computed by interpreting the logical connectives as in possibilistic logic). We focus our presentation on the probabilistic case, where we assume probabilistic independence between properties of different objects.

1. One starts with a combined description d for the requested object and its related objects. For each anchoring candidate

   $$a_i = (\pi_{i,0}, (\pi_{i,1,1} \ldots), (\pi_{i,2,1} \ldots), \ldots),$$

   a description d_i of the perceived objects of a_i is computed. For object properties that are uncertain, a probability distribution over possible values is used. This distribution depend on the particular property (including our sensing model for observing that property), as well as on what a priori information we have about that property.

   Example The robot Pippi is looking for `g1`, described as:

$d =$ (and (shape g1 = garbage-can)
(near g1 b1 = t) (shape b1 = ball)
(mark b1 = t)).

Two anchoring candidates are found: $a_1 =$ (pi1 (pi2)) and $a_2 =$ (pi3 (pi4)). Their descriptions are:

$d_1 =$ (and (shape pi1 = garbage-can)
(near pi1 pi2 = t) (shape pi2 = ball)
(mark pi2 = (t 0.5) (f 0.5)))

$d_2 =$ (and (shape pi3 = garbage-can)
(near pi3 pi4 = t) (shape pi4 = ball)
(mark pi4 = (t 0.5) (f 0.5))).

Here there is a probability distribution over the property mark, based on the probability of encountering a mark on an arbitrary garbage can, adjusted by the fact that one side of the garbage can has already been observed.

2. Next, for the description d_i of each achoring candidate two extra sets of descriptions d_i^+ and d_i^- are computed as follows.
a) Certain properties for certain perceived objects π_{ijk} in the candidate will be unspecified in d_i, implying a partial match. We can constrain all unspecified properties for such a π_{ijk} in terms of a formula d_{ijk}^+, such that $d_i \wedge d_{ijk}^+$ forms a description for π_{ijk} that matches the one in d. By constraining any of the properties not to match in terms of a formula d_{ijk}^-, one can likewise make it a mismatching description.
We also assign probabilities to these formulae:

$$p_{ijk}^+ = \prod_l P(\text{property } l \text{ of } \pi_{ijk} \text{ matches})$$

$$p_{ijk}^- = 1 - \prod_l P(\text{property } l \text{ of } \pi_{ijk} \text{ matches})$$

It is also possible to have conditional probabilities: in [7], we show how compute them using rules for coding indirect effects in our planning domain, provided they can be ordered as an directed acyclic graph.
Example For d_1 we get $d_{1,0}^+ = \top$ (for the garbage can, which already matches) and $d_{1,1,1}^+ =$ (mark pi2 = t) (for the ball, where the mark is unspecified) where $p_{1,1,1}^+ = 0.5$, and $d_{1,0}^- = \perp$ and $d_{1,1,1}^- =$ (mark pi2 = f) where $p_{1,1,1}^- = 0.5$, and similarly for d_2.
b) If there are several objects $\pi_{ij1}, \ldots, \pi_{ijk}$ partially matching related object number j in a candidate, one can combine those to get a match by making at least one of them match:

$$d_{i,j}^+ = \bigvee_{1 \le l \le k} (\alpha \cdot p_{ijl}^+ : d_{ijl}^+),$$

$$\text{with } p_{i,j}^+ = 1 - \prod_{1 \le l \le k} (1 - p_{ijl}^+)$$

where α is a normalization factor that makes the sum 1. One can make a mismatch by making all of them mismatch:

$$d_{i,j}^- = \bigwedge_{1 \le l \le k} d_{ijl}^- \text{ with } p_{i,j}^- = \prod_{1 \le l \le k} p_{ijl}^-.$$

Example In the first candidate a_1, there is only one perceived object pi2 that can be the related object (the ball, with description $d_{1,1,1}$). Hence, $d_{1,1}^+ = d_{1,1,1}^+$ with $p_{1,1}^+ = 0.5$ and $d_{1,1}^- = d_{1,1,1}^-$ with $p_{1,1}^- = 0.5$.
c) If there is no candidate percept for a specific related object, we assume that it has not been seen yet, as discussed below.
d) Each d_i^+ is a description of candidate i that completely matches d, by making the primary perceived object $\pi_{i,0}$ and one perceived object for each of the n related objects completely matching:

$$d_i^+ = \bigwedge_{0 \le j \le n} d_{i,j}^+ \text{ with } p_i^+ = \prod_{0 \le j \le n} p_{i,j}^+.$$

Likewise, each d_i^- contains the different ways in which either the primary perceived object or all candidate perceived objects for one related object can mismatch:

$$d_i^- = \bigvee_{0 \le j \le n} (\alpha \cdot p_{i,j}^- : d_{i,j}^-)$$

$$\text{with } p_i^- = 1 - \prod_{0 \le j \le n} (1 - p_{i,j}^-).$$

Example We get $d_1^+ = \top \wedge (\mathsf{mark\ pi2} = \mathsf{t})$ with $p_1^+ = 0.5$, and $d_1^- = \bot \vee (\mathsf{mark\ pi2} = \mathsf{f})$ with $p_1^- = 0.5$. Note that the $\top$ and $\bot$ are the positive and negative descriptions of the garbage can, and $(\mathsf{mark\ pi2} = \mathsf{t})$ and $(\mathsf{mark\ pi2} = \mathsf{f})$ are the positive and negative descriptions of the nearby ball.

3. Each hypothesis then consists of a matching description for one candidate and mismatching descriptions for the remaining ones. To each hypothesis is also added the statement $(\mathsf{anchor}\ s = \pi_{i,0})$ denoting that s should be anchored to the object anchored by $\pi_{i,0}$:

$$h_i = d_i^+ \wedge \bigwedge_{j \neq i} d_j^- \wedge (\mathsf{anchor}\ s = \pi_{i,0})$$

$$\text{with } p_i = p_i^+ \cdot \prod_{j \neq i} p_j^- .$$

There is also one hypothesis that no object matches:

$$h_0 = \bigwedge_j d_j^- \wedge (\mathsf{anchor}\ s = \mathsf{f})$$

$$\text{with } p_0 = 1/c \cdot \prod_j p_j^- ,$$

where c is a discount factor representing our subjective trust in the implicit assumption that the requested object actually is there.

If the recovery module takes a cautious approach and wishes to ascertain that no more than one object is matching (in particular in case 3), it might also add hypotheses consisting of $d_i^+ \wedge d_j^+$ for each pair of candidates, and $(\mathsf{anchor}\ s = \mathsf{f})$, with $p_{i\&j} = 1/c \cdot p_i^+ \cdot p_j^+$.

Example One of the hypohteses, with pi1 matching b1, would be:

$h1 = (\mathsf{mark\ pi2} = \mathsf{t}) \wedge (\mathsf{mark\ pi4} = \mathsf{f}) \wedge (\mathsf{anchor\ b1} = \mathsf{pi1})$ with $p1 = 0.25$.

4. Finally, we combine our hypotheses according to

$$(\bigwedge_{1 \leq i \leq n} d_i) \wedge (\bigvee_{0 \leq i \leq n} (\alpha \cdot p_i : h_i))$$

where α normalizes the probabilites of our hypotheses to compensate for the fact that we have discounted the "no object" and "too many object" hypotheses. Recall that the d_i are our original unconstrained descriptions of the objects involved.

Example We get the following (incautious) set of hypotheses ($c = 2$):

h0: 0.2 (mark pi2 = f) $\wedge$ (mark pi4 = f) $\wedge$ (anchor b1 = f)

h1: 0.4 (mark pi2 = t) $\wedge$ (mark pi4 = f) $\wedge$ (anchor b1 = pi1)

h2: 0.4 (mark pi2 = f) $\wedge$ (mark pi4 = t) $\wedge$ (anchor b1 = pi3)

In addition, each of the two first hypotheses can be subdivided further into three different hypotheses regarding from where the mark can be detected: (mark-visible-from pi2 = r1_1) etc. These sub-hypotheses have a uniform probability distribution. The following information applies to all the hypotheses (i.e. is from d_1 and d_2):

(shape pi1 = garbage-can) (near pi1 pi2 = t)
(shape pi2 = ball)
(shape pi3 = garbage-can) (near pi3 pi4 = t)
(shape pi4 = ball)

To the above is added information about the topology of the room and other relevant background information.

4.1.2. Same object in several anchoring candidates

Finally, one has to consider the possibility that the same object may appear in several candidates (although at most once as a primary object). In such cases, when an anchoring candidate a_i involves a perceived object that also appears in some other anchoring candidate a_j, then the negative description d_j^- of the latter anchoring candidate has to be specially tailored when it is to be combined with d_i^+. The doubly appearing perceived object may either be expected to match the same description in the two anchoring candidates, in which case it cannot be included as a non-matching candidate object in d_j^-. Or it may be expected to match different descriptions in the two anchoring candidates, in which case it will be a non-matching candidate object in d_j^- with probability 1.0.

To complicate matters further, if the doubly appearing perceived object is only one of several candidate objects for a specific relation in d_i^+, then these constraints should only apply to the hypothesis where that particular candidate object matches.

4.1.3. Formulating the goal

The goal is achieved once a specific action (anchor s x) has been performed. It represents the decision to anchor the symbol s to some specific perceived object x (or to no object at all, if $x =$ f). This action has as a precondition that x is the only remaining anchor for s: (nec (anchor $s = x$)). Thus, all other candidate anchors have to be eliminated before the anchor action is applied.

4.1.4. Size of belief state for initial situation

The size of the belief state obtained above in terms of number of alternative states it contains can be estimated as follows. If we consider the descriptions of the anchoring candidates $d_1, ..., d_n$, then each one contains a number of objects that may contain some incompletely known properties, and it is these properties that generate alternative states in the belief state. If each incomplete property can have two alternative values (one matching the description and one the complement), and there are m_i such properties in d_i, then the number of alternative states $|d_i|$ generated by d_i is 2^{m_i}. Altogether, that gives us $\prod_{i=1,\dots,n} 2^{m_i}$ alternative states. Adding the positive and negative descriptions (combined to form alternative anchoring hypotheses) serves to eliminate some of the alternative states, so the above can be considered an upper limit.

4.1.5. The indefinite case

In the case with an indefinite anchor, one does not have to exclude other candidates when one candidate is proposed. Therefore each hypothesis simply consists of

$$h_i = d_i^+ \wedge (\text{anchor } s = \pi_{i,0})$$

and a no-matching-object hypothesis:

$$h_0 = \bigwedge_j d_j^- \wedge (\text{anchor } s = \text{f})$$

In order to obtain accurate probabilities for the different anchoring candidates, one may consider all possible combinations of matching and non-matching anchoring candidates (there is an exponential number). But our main concern is to get accurate probabilities for h_0. Therefore we have opted for the following approximation.

The probability for the non-matching hypothesis is $p_0 = 1/c' \cdot \prod_j p_j^-$ where c' is a discount factor (typically $c' < c$). For the matching hypotheses h_i, the p_i are normalized relative to $1 - p_0$. However, just doing that tends to exagerate the total probability of each particular candidate, so we also reduce the matching probabilities for properties of objects in a_i in the other matching hypotheses $h_j, j \neq i$ to keep the balance.

The anchor action in this case is (anchor-indef s), it has a precondition (nec (not (anchor $s =$ f))), and it works in the following way. Once we have observed that s does not have f as an option, we can be sure that there is at least one candidate object that is matching. But how to find those that actually match? There might still be candidates left that are undetermined. The solution is simply to query the belief state on each candidate: if (nec d_i^+) holds, then candidate a_i is completely matching.

Example We get the following set of hypotheses ($c = 1$):

h0: 0.250 (mark pi2 = f) ∧ (mark pi4 = f) ∧ (anchor b1 = f)
h1: 0.375 (mark pi2 = t) ∧ (mark pi4 = (t 0.333) (f 0.667)) ∧ (anchor b1 = pi1)
h2: 0.375 (mark pi2 = (t 0.333) (f 0.667)) ∧

(mark pi4 = t) ∧ (anchor b1 = pi3)

Note that if we would observe a mark on the ball pi2, the non-matching hypothesis *h0* would be eliminated, and d_1^+ = (mark pi2 = t) would be necessarily true in the belief state. Hence, b1 could then be anchored to pi1.

4.2. No candidate found

If no candidate has been found for the primary object (case 1) the recovery module hypothesize that the object is somewhere but is not visible from the current position. Therefore, the initial situation consists of a number of hypotheses concerning from what position the object can be found, each of the form (visible-from $s = pos$) where $pos =$ f signifies that the object is nowhere around.

The planning goal is formulated as (exists (?x) (nec (visible-from s = ?x))), which means that the agent has determined from what place the object is visible. A missing secondary object in an anchoring candidate is treated in a similar way, but here the relation involved is also part of the hypothesis.

When the plan eventually is executed and an object is found, it may lead to further planning, as described further down.

4.3. Generating the recovery plan

After the initial situation and the goal have been established, plan generation starts, using the initial belief state and goal and the set of available actions. The following action, for instance, is for looking for marks (and other visual characteristics) on objects.

```
(ptl-action
 :name (look-at ?y)
 :precond ( ((?p) (robot-at = ?p))
           ((?y) (perceived-object ?y)) )
 :results (cond
             ((and (mark ?y = t) (mark-visible-from ?y = ?p))
              (obs (mark! ?y = t)))
             ((not (and (mark ?y = t)
                        (mark-visible-from ?y = ?p)))
              (obs (mark! ?y = f))))
 :execute  ((aiming-at me ?y)(anchor-find ?y)))
```

In short, the precond part states that the action requires a perceived object ?y and a current position ?p. The result part states that if ?y has a mark, and if the robot looks at ?y from the position from which the mark is visible, then the robot will observe the mark (and thus know that there is a mark), and otherwise it will not observe any mark. The obs form is the way to encode that the agent makes a specific observation, and different observation results in different new belief states. In this case, there would be one belief state where the agent knows there is a mark, and one where it knows there isn't a mark on that side. If the agent keeps making observations, it can ideally eliminate anchoring hypotheses (signified by (anchor $s = x$)) until only one remains. It can then perform the action (anchor s x). Recall that the goal is to have done an anchor.

PTLplan is a progressive planner, so it starts from the initial belief state and adds actions until a belief state satisfying the goal is reached. When an action results in several new belief states with different observations, the planner inserts a conditional branching in the plan and continues planning for each branch separately. In order to search more efficiently, the planner can also eliminate belief states that invalidate a given temporal logic formula. The planner terminates when a plan has been found with a probability/necessity to reach a goal state that is above some given threshold.

The following plan is generated for looking for marks on a red ball pi2 from three different positions, starting from a fourth position:

```
((move r1_2) (look-at pi2)
 (cond
   ((mark! pi2 = f) (move r1_3) (look-at pi2)
    (cond
      ((mark! pi2 = f) (move r1_4) (look-at pi2)
       (cond
         ((mark! pi2 = t) (anchor g1 pi1) :success)
      ((mark! pi2 = f) (anchor g1 f) :fail)))
    ((mark! pi2 = t) (anchor g1 pi1) :success))
  ((mark! pi2 = t) (anchor g1 pi1) :success)))
```

Note how a conditional branching follows after each application of look-at: the first clause "(mark! pi2 = t/f)" of each branch is the observation one should have made in order to enter that branch, and the subsequent clauses are actions.

4.4. Plan execution

The anchoring plan is then executed: the actions such as (look-at pi2) are translated into executable perceptual and movement tasks (see field :execute in the definition of look-at above). The anchor action has a special role: it causes the symbol of the requested object to be anchored to a specific perceived object. The robot can then continue performing the task in its top-level plan that was interrupted.

4.5. Discovering new candidates

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.

	Experiments	#Anchors	Success
a	Find among 2 odors	11	82%
	Find among 3 odors	15	80%
	Find among 4 odors	21	76 %
	Find among 5 odors	25	76 %
b	Reacquire among 2 gas bottles	15	87%
	Reacquire among 3 gas bottles	10	80%
	Reacquire among 4 gas bottles	10	90%
c	Find can near occluded ball	10	80%
	Find can near ball with mark	15	93%
d	Multiple recoveries	24	79%
e	Discovery of new objects	11	73 %

Table 2
Experimental results

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 new 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).

5. Experimental Evaluation

To be able to experimentally evaluate the methods described above we have implemented and integrated them with a fuzzy behavior based system, the Thinking Cap [33], used for controlling a mobile robot called Pippi. Our primary sensor modality is through a vision system connected to the camera. As a second sensor modality we use a commercially available electronic nose [14] capable of distinguishing between a number of odors. See [29,28] for more information on how the electronic nose can be used together with PTLplan and anchoring.

We present several experiments that illustrate a variety of ambiguous situations and how they are handled. The results of the experiments are sum-

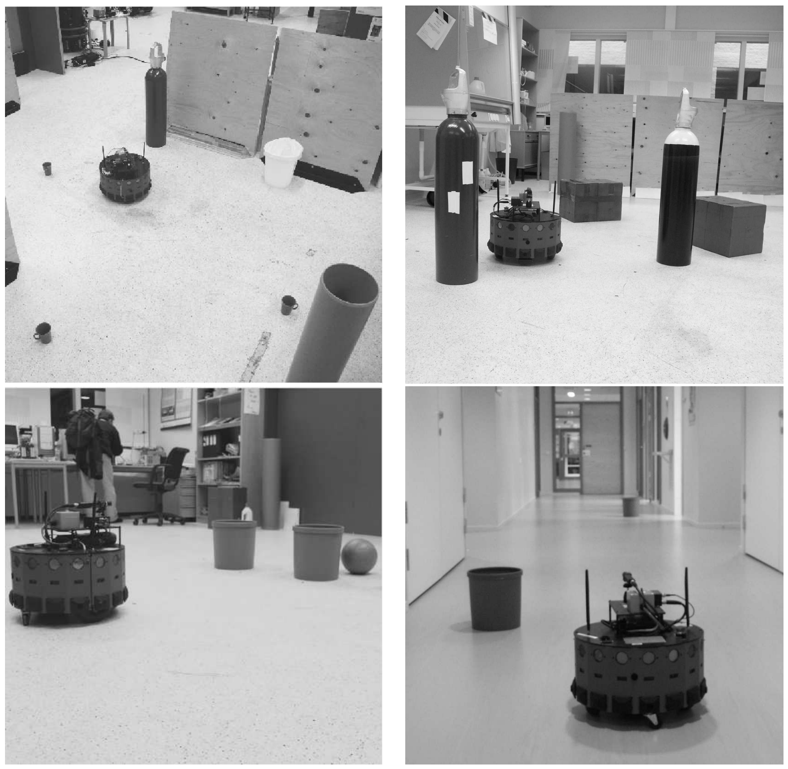

Fig. 1. Experimental setups, above: (a) complementary sensors, (b) displaced object, below: (c) relations, (d) multiple recoveries.

marized in Table 2. As performance measure we count the number of anchoring operations needed to achieve the top-level goal when performing a number of repetitions using the same setup. Note that we do not count any recursive anchoring operations performed during the recovery phase. For instance, in experiment (a) in the first setup with 2 odors, the robot was given the task to find a specific cup 11 times (each with a different configuration). Each of these times, the execution of the recovery included a number of additional anchoring operations (of the candidate objects from different positions), but these were not included. We compare the total number of successful operations with the total number of required operations for all repetitions of the respective setups. For the first three and fifth scenarios only one top-level anchoring operation is required per setup and the numbers thus also reflects the total number of runs. The fourth scenario was run a total of eight times and each run required anchoring three different objects. The top-level plan was of the form "move to p1, move near o1, move to p2, move near o2, move to p3, move near o3".

Most of the experiments where conducted with a possibilistic representation, but probabilistic planning was used in the experiments involving replanning (fifth scenario). There, additional flexibility was also achieved by online generation of observation points. The possibilities/probabilities used were based on the subjective estimates of the experimenters.

Our system has a success rate between 73% and 93% for these experiments, and most failures happened because the perceptual system delivered wrong data (e.g. failed to see a mark). The planner never failed to find a plan, and typically did so in a fraction of a second, and was still below 5 s in the scenarios where observation points were generated online.

It should be noted that each one of the scenarios below could potentially be solved by a *ad hoc* procedure, provided that the type of perceptual anomaly encountered were known beforehand. One of the strengths of our approach is that one and the same generic domain model was used to deal with all these situations in a uniform way.

5.1. (a) Recovery using complementary sensors

In this first series of experiments we show how ambiguous cases due to partial matching of the description can be resolved by using a complementary sensor modality. The experiments are performed by using a number of cups containing different substances. The cups are scattered throughout a room containing other objects and the task of Pippi is to find a cup that is characterized by both a visual and an odor description (Figure 1 (a)).

When Pippi attempts to anchor an object with the description "the green cup that smells of ethanol", it finds several objects matching the visual description. In order to recover from this situation, Pippi determined that the odor part of the description ("smells of ethanol") was needed and used its e-nose to sample the different cups.

A number of configurations of the above scenario where tried with 2 to 5 cups containing different substances. Each cup contained one of: Ethanol, Hexanal, 3-Hexanol, Octanol, and Linalool. These substances are part of an ASTM atlas of odor descriptions whose characters which are best described as alcoholic, sour, woody, oily and fragrant respectively [16].

The odor part of the description as well as the positions of the cups and other objects where varied. Pippi was given the description of the target object and generated and executed a conditional plan examining all candidate cups until the correct one was found.

The occasional failures were due to either misclassification of odors and/or problems with the vision module.

5.2. (b) Reacquiring Displaced Objects

The second series of experiments concerned the resolution of ambiguous situations when an object is to be reacquired. The scenario involved two or more identically colored gas bottles in a room together with other objects such as boxes (Figure 1 (b)). One gas bottle named `gb-b` was distinguished by a white mark on one side. Pippi started from one position in the room, and tried to find `gb-b` by visually scanning the room. As the mark was initially turned toward Pippi, the correct gas bottle was easily found. While the robot performed other tasks the gas bottles where rearranged so that the position of `gb-b` was no longer accurate. The next time Pippi needed to reacquire `gb-b`, it could not determine which gas bottle was the correct one by using position information or by observing the, now occluded, mark.

Pippi then generated a plan to go to different positions in the room observing the gas bottles and looking for the mark. Once the mark was found, she approached that gas bottle. A number of variation of the scenario were tried, varying the number of gas bottles, their rotation, initial and final positions. For the result in Table 2 we count here only the reacquire operations since the initial find operations where all trivial to perform. Counting also the find operations would instead have yielded success rates in the range 90-95%.

When several gasbottles were involved the recovery was non-trivial because multiple perceptual faults often led to additional recursive recovery plans being executed.

5.3. (c) Anchoring with relations

Two series of experiments involving relations to other objects have also been run. The first scenario uses a description “the green garbage can near the red ball”, where “near” is defined as a fuzzy relation on the estimated positions of percepts. There are two green cans visible, but the red ball is hidden behind one of them. Here, Pippi generates a plan to tour the room looking for the red ball, and when it is detected, to check which can it is near. The second scenario uses the description “the green garbage can near the red ball with a mark”, and involves a single can and a red ball with the mark turned in different directions. In this scenario, Pippi generates a plan where it searches for a mark on the secondary object, i.e. the ball.

5.4. (d) Multiple recoveries

This set of experiments was meant to test recovery from multiple anchoring failures while executing a longer top-level plan. We gave Pippi the goal that three known garbage cans should be inspected and that Pippi should return home. From the initial information the planner generated a plan consisting of 15 steps, including 3 anchoring operations and which was executed successfully in only some of the setups. In some other setups perceptual ambiguities were encountered during the execution: a mark was not visible and/or a nearby ball was occluded. In different cases different top-level anchoring operations required additional actions resulting in the total execution of up to 27 actions. Pippi successfully handled most of these cases in ways similar to those described in the previous experiments and achieved the top-level goals. Most of the failures were caused by faults in the low-level perception, and a few by bad self localization. The cases where subsequent anchoring operations where not executed due to earlier failures where also counted as failures here. Otherwise we would have a success rate of 86%.

5.5. (e) Discovery of new objects

The final set of experiments involved replanning due to the discovery of objects not initially visible. The ability to replan is vital in order to achieve robustness; failing to observe all relevant objects initially is not an uncommon occurrence in a moderately cluttered environment. There were two scenarios: one where Pippi was to find a gas bottle with a mark on it, and one where Pippi was to search for a gas bottle near a red ball with a mark.

In the first scenario, two gas bottles were used. One of them was initially visible, but the second one was positioned in such a way that it was initially occluded by the first one. The mark was either put on the first or the second gas bottle, and the position of the mark on the bottle was also varied. A few times, there was no mark on either gas bottle. Pippi handled this scenario in the following manner. First, she looked for gas bottles and found one but the mark was not visible (partial match). She then generated a recovery plan to look for the mark on that gas bottle. When arriving at the next position, Pippi discovered that there was a second gas bottle. This was yet another partial match. Hence, Pippi reformulated the current belief state to take this new gas bottle into account, and next generated a new recovery plan involving looking for marks on both gas bottles (on the remaining places). That plan was executed successfully in 8 of the 11 runs. The failures were due to low-level perceptual problems (in particular the mark was not detected).

The second scenario involved a red ball in addition to the two gas bottles. The task is to look for a gas bottle near a red ball. The scenario has been varied according to the positions of the bottles and the ball (at least one of them being initally occluded), and whether the ball or the gas bottle is supposed to have a mark. Several different configurations have been successfully solved by the robot.

6. Conclusions

To a mobile robot that is sensing, reasoning and acting in a complex environment, the ability to anchor symbols to perceptual data is essential. In this paper we have extended the anchoring framework to deal with ambiguous cases where there is more than one percept that matches the symbolic description of an object. We have proposed an approach for actively solving such cases. We first classify the situation based on the number of complete and partial matches, and whether the symbolic description is definite or indefinite. Based on this classification, we either declare failure or try to recover from the situation. Recovery is achieved by automatically formulating the present situation as a planning problem and then letting a planner find a suitable recovery plan, which typically consists of movement and perceptual actions. We also allow for replanning in case additional matching objects are perceived.

Our approach has been implemented and tested in a variety of experiments involving a mobile robot with sensing capabilities such as vision and olfaction. The algorithms and domain knowledge used in the recovery planning do not depend on the specific failure type and the larger examples could be handled automatically using the same mechanisms as in the single-failure cases.

We have also addressed the use of relations in the description of objects. Relations allow for richer descriptions and more complex ambiguity cases.

Uncertainty has both been adressed in possibilistic and probabilistic terms, so our approach is flexible regarding how to address uncertainty. Probability theory is concerned with precise quantifications of uncertainty, and is useful when one has reasonably precise probability estimates and a good idea of how they depend on each other (ideally, they should be independent). Possibility theory, on the other hand, is suitable when estimates are imprecise (one can only give upper and lower limits) and one cannot make any strong assumptions about dependencies [18].

Acknowledgements

This work has been supported partly by the Swedish Research Council (Vetenskapsrådet), partly by the Swedish KK foundation, and partly by ETRI (Electronics and Telecommunications Research Institute, Korea) through the project "Embedded Component Technology and Standardization for URC(2004-2008)".

References

[1] T. Arbel and F.P. Ferrie. Entropy-based gaze planning. *Image and Vision Computing*, 19(11):779–786, 2001.

[2] A. Atrash and S. Koenig. Probabilistic planning for behavior-based robots. In *Proceedings of the International FLAIRS conference*, pages 531–535, 2001.

[3] C. Barrouil, C. Castel, P. Fabiani, R. Mampey, P. Secchi, and C. Tessier. Perception strategy for a surveillance system. In *Proc. of the 13th European Conference on Artificial Intelligence*, pages 627–631, 1998.

[4] M. Beetz, T. Arbuckle, A.B. Cremers, and M. Mann. Transparent, flexible, and resource-adaptive image processing for autonomous service robots. In *Proc. of the 13th European Conference on Artificial Intelligence*, pages 158–170, 1998.

[5] M. Beetz and D. McDermott. Expressing transformations of structured reactive plans. In *Proc. of the European Conf. on Planning*, pages 64–76. Springer, 1997.

[6] H. Borotschnig, L. Paletta, M. Prantl, and A. Pinz. Appearance-based active object recognition. *Image and Vision Computing*, 18(9):715–727, 2000.

[7] A. Bouguerra, L. Karlsson, and A. Saffiotti. Situation assessment for sensor-based recovery planning. In *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI-06)*, pages 673–677, 2006.

[8] A. Bouguerra, L. Karlsson, and A. Saffiotti. Semantic-knowledge based monitoring for mobile robots. In *Proc. of 2007 IEEE International Conference on Robotics and Automation*, 2007.

[9] M. Broxvall, S. Coradeschi, L. Karlsson, and A. Saffiotti. Recovery planning for ambiguous cases in perceptual anchoring. In *Proc. of the 20th National Conference on Artificial Intelligence*, Pittsburgh, PA, 2005. AAAI Press.

[10] M. Broxvall, L. Karlsson, and A. Saffiotti. Steps toward detecting and recovering from perceptual failures. In *Proc. of the 8th Conf. on Intelligent Autonomous Systems*, pages 793–800, 2004.

[11] S. Coradeschi and A. Saffiotti. Anchoring symbols to sensor data: preliminary report. In *Proc. of the 17th National Conf. on Artificial Intelligence*, pages 129–135, Menlo Park, CA, 2000.

[12] S. Coradeschi and A. Saffiotti. Perceptual anchoring of symbols for action. In *Proc. of the 17th International Joint Conf. on Artificial Intelligence*, pages 407–412, 2001.

[13] S. Coradeschi and A. Saffiotti. An introduction to the anchoring problem. *Robotics and Autonomous Systems*, 43(2-3):85–96, 2003. Special issue on perceptual anchoring.

[14] Cyranose Sciences Inc. The cyranose 320 electronic nose, 2000. User's Manual Revision D.

[15] J. Denzler and C.M. Brown. Information theoretic sensor data selection for active object recognition and state estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):145–157, 2002.

[16] A. Dravnieks. *Atlas of Odor Character profiles (ASTM Data Series Publication DS 61).* American Society for Testing, USA, 2000.

[17] D. Dubois and H. Prade. *Possibility theory — an approach to computerized processing of uncertainty.* Plenum Press, 1988.

[18] D. Dubois and H. Prade. When upper probabilities are possibility measures. *Fuzzy sets and systems*, 49(1):65–74, 1992.

[19] R.E. Fikes, P. Hart, and N.J. Nilsson. Learning and executing generalized robot plans. *Artificial Intelligence*, 3(4):251–288, 1972.

[20] J. Gancet and S. Lacroix. PG2P: A perception-guided path planning approach for long range autonomous navigation in unkown natural environments. In *Proc. of 2003 IEEE/RSJ International Conference of Intelligent Robots and Systems (IROS 2003)*, Las Vegas, NV, 2003.

[21] G. De Giacomo, L. Iocchi, D. Nardi, and R. Rosati. Planning with sensing for a mobile robot. In *Proc. of the 4th European Conf. on Planning*, pages 158–170, 1997.

[22] K.Z. Haigh and M.M. Veloso. Interleaving planning and robot execution for asynchronous user requests. *Autonomous Robots*, 5(1):79–95, 1998.

[23] L.P. Kaebling, M.L. Littman, and A.R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998.

[24] L. Karlsson. Conditional progressive planning under uncertainty. In *Proc. of the 17th International Joint Conf. on Artificial Intelligence*, pages 431–438, 2001.

[25] L. Karlsson and T. Schiavinotto. Progressive planning for mobile robots: a progress report. In M. Beetz, J. Hertzberg, M. Ghallab, and M. Pollack, editors, *Advances in Plan-Based Control of Robotic Agents*, pages 106–122. Springer, 2002.

[26] S. Kovacic, A. Leonardis, and F. Pernus. Planning sequences of views for 3-D object recognition and pose determination. *Pattern Recognition*, 31:1407–1417, 1998.

[27] A. Lazanas and J.C. Latombe. Motion planning with uncertainty: A landmark approach. *Artificial Intelligence*, 76(1-2):285–317, 1995.

[28] A. Loutfi, M. Broxvall, S. Coradeschi, and L. Karlsson. Object recognition: A new application for smelling robots. *Robotics & Autonomous Systems*, 52:272–289, 2005.

[29] A. Loutfi, S. Coradeschi, L. Karlsson, and M. Broxvall. Putting olfaction into action: Using an electronic nose on a multi-sensing mobile robot. In *Proc. of the IEEE Int. Conf. on Intelligent Robots and Systems*, 2004.

[30] S.M. Majercik and M.L. Littman. Contingent planning under uncertainty via stochastic satisfiability. *Artificial Intelligence*, 147:119–162, 2003.

[31] N. Onder and M.E. Pollack. Conditional, probabilistic planning: A unifying algorithm and effective search control mechanisms. In *Proc. of the 16th National Conf. on Artificial Intelligence*, pages 577–584, 1999.

[32] B. Pell, D.E. Bernard, S.A. Chien, E. Gat, N. Muscettola, P.P. Nayak, M.D. Wagner, and B.C. Williams. An autonomous spacecraft agent prototype. *Autonomous Robots*, 5(1):1–27, 1998.

[33] A. Saffiotti, K. Konolige, and E.H. Ruspini. A multivalued-logic approach to integrating planning and control. *Artificial Intelligence*, 76(1–2):481–526, 1995.

[34] L. Seabra-Lopes. Failure recovery planning in assembly based on acquired experience: learning by analogy. In *Proc. IEEE Intl. Symp. on Assembly and Task Planning*, Porto, PT, 1999.