Journal name
000(00):1–13

DOI:doi number
http://mms.sagepub.com

Reliable Confirmation of an Object Identity by a Mobile Robot: A Mixed Appearance/Localization-Driven Motion Approach

Israel Becerra*, Luis M. Valentín-Coronado and Rafael Murrieta-Cid
Centro de Investigación en Matemáticas, CIMAT, Guanajuato, México
{israelb,luismvc,murrieta}@cimat.mx

Jean-Claude Latombe
Artificial Intelligence Laboratory, Computer Science Department, Stanford University, Stanford, CA 94305, USA
latombe@cs.stanford.edu

Abstract

This paper investigates the problem of confirming the identity of a candidate object (expected to be a target based on some crude visual clues) with a mobile robot equipped with visual sensing capabilities. We present a method whose main novelty is to mix localization of the robot relative to the candidate object and confirmation that it is the sought target. This twofold approach drastically reduces false positives. Identity confirmation with this twofold goal is modeled as a Partially-Observable Markov Decision Process where the states are the cells of the space decomposition. It is solved using Stochastic Dynamic Programming with imperfect state information. A robotic system using this method has been implemented and tests have been carried out both in simulation and with a real robot. The experiments empirically validate the method, evaluate its performance using various metrics, and demonstrate its ability to perform well in different settings.

Keywords
Object identification, motion strategies, appearance-guided motion, robot localization

1. INTRODUCTION

In this paper we investigate the problem of confirming the identity of an object using a mobile robot equipped with a camera, a range sensor (to avoid collisions), and a vision software module referred to as the *target detector* (or, more simply, the *detector*). More precisely, we will describe a method to perform the third step of the following scenario:

* Corresponding author; e-mail: israelb@cimat.mx.

1. The robot is instructed to find a certain object T, hereafter called the *target*, in its environment.
2. Using a space coverage method (e.g., the method described in [Espinoza et al. 2011]), the robot detects an object C, called the *candidate*, that looks like T based on some crude (but easy to detect) visual clues, such as color, texture, or general shape.
3. The robot moves on a horizontal floor around C to achieve suitable viewpoints allowing the target detector to eventually confirm (or infirm) that C is actually T.

The main novelty of our method is to mix localization of the robot relative to the candidate object C and confirmation that C is the target T. This twofold approach drastically reduces false positives. The robot is given a probabilistic observation model of the response of the target detector trained over a cell decomposition of the space surrounding T (Section 3). During the confirmation process, the robot never knows its exact position; instead, the robot's belief about its position is also modeled by a probability distribution over this cell decomposition (Section 4). By performing successive moves (Section 5) the robot acquires several images of C from different viewpoints and runs the target detector on each image. After each move it uses the observation model to update its position belief. The robot's goal is twofold (Section 4). One part of the goal is to compute a motion strategy that will eventually reach a position P where the target detector confirms with high confidence that C is actually T. But this goal alone would be prone to produce false positives, especially if other objects look very similar to T from some viewpoints. The other part of the goal is to compute the motion strategy so that the robot expects with high probability that, upon reaching P, the detector will confirm with high confidence that C is actually T. This twofold goal drastically reduces the risk that the detector incorrectly confirms that C is T. The identity confirmation process is modeled as a Partially-Observable Markov Decision Process (POMDP) [Candido and Hutchinson 2011; Kaelbling et al. 1998] where states are the cells of the space decomposition. Motion policies are computed using Stochastic Dynamic Programming with imperfect state information [Bertsekas 2000].

The ability of a mobile robot to reliably confirm the presence of a target object in an environment has a wide range of potential applications, especially as an end-module for search problems, as suggested in the above scenario. There has already been a substantial amount of research studying how a robot or a team of robots should search an environment to find a static object or a moving agent (adversarial or not) [Chung at al. 2011; Guibas et al. 1999; Tovar and LaValle 2008]. In this research the target is usually considered found when a robot has established a line of sight with it. However, this condition requires the robot's detector to reliably identify the target from any viewpoint at any distance. Some research attempts to fulfill this condition by creating powerful detectors [Felzenzwalb et al. 2008; Grauman and Darrel 2005; Lowe 2004]. Instead, we consider here that a detector will always be imperfect and that the ability of the robot to move around an object and take images from different viewpoints should be exploited to overcome the limitations of the detector. This combination is particularly suitable when the target differs from other objects by few small distinctive features visible only from limited viewpoints or when elements of the environment (e.g., poor illumination and textured background) may deceive the detector. For instance, to reliably identify your luggage on a conveyor belt at an airport, you look to it from different viewpoints in order to detect special features that differentiate it from similar luggage, for which you had no prior models, and you do not remain static. So, the problem considered here is a target confirmation problem, which differs significantly from the traditional object recognition problems, as only the observation model of the target is available. Confirming that an object is a given target and not another, possibly similar object, of unknown model, is a challenging problem.

A robotic system using our method has been implemented and tests have been carried out both in simulation and with a real robot. Our simulation experiments were intended to perform basic tests of our approach under well-controlled conditions. We created a textured virtual environment and the detector was applied to synthetic images generated from the successive robot's viewpoints. In this paper we report on four experiments in simulation (Section 7). Some of our goals were to verify that our system is able to discriminate between similar objects and is not confused by targets with similar appearances from different viewpoints, and to analyze its behavior when localization errors increase. We then present five experiments with a real robot, a Pioneer P3-DX (differential-drive robot) equipped with an USB camera

and a Sick-LMS200 range sensor, aimed at further assessing our approach (Section 8). In these experiments, we vary two sets of parameters. Parameters in one set are related to the planner, e.g.: planning horizon, number of cells in space decomposition. Parameters in the other set are related to the environment and the target detector: illumination of the environment, environment background, textures, sizes, and colors of the targets, size of the training set of images. Overall, the experiments empirically validate our method, evaluate its performance along various metrics (success rate, planning time, number of sensing operations, length of robot path), and demonstrate its ability to perform well in a broad variety of settings, even with an imperfect detector. They also reveal interesting insights and suggest topics for additional future research.

A preliminary version of a portion of this paper appeared in [Becerra et al. 2014]. The main differences are the following:

1. We no longer assume that the robot knows the position of the center of the candidate object. Instead, only the direction of the object center is now needed and it is estimated using the data provided by the range sensor. Moreover, a new experiment in simulation shows that our system tolerates large errors in the estimated direction.
2. The preliminary version did not include experiments with a real robot. Here, we present and discuss five such experiments. They test in particular the impact of variations in scene illumination and in the number of cells of the space decomposition.
3. A new experiment in simulation which considers the presence of obstacles around the candidate object. These obstacles create both motion and visibility constraints.

The main contributions of our work are threefold. (1) To reduce false positives our method mixes localization of the robot relative to the candidate object C and confirmation that C is the target T. (2) Our experiments show that if an observation model contains too much uncertainty then planning too far ahead may have a negative effect on the effectiveness of the confirmation process. (3) Experiments demonstrate that our method handles correctly candidate objects that are distinct from targets, but similar-looking, and that it tolerates poor illumination conditions, targets with no neat feature and obstacles that produces both motion and visibility constraints.

Section 2 discusses related work. Section 3 describes the target detector and the observation model of a target. Section 4 formulates the mixed identification/localization process as a POMDP. Section 5 defines the motion commands available to the robot and the corresponding transition model of the POMDP. Section 6 describes the use of Stochastic Dynamic Programming to compute motion policies. Section 7 presents and discusses four experiments carried out in simulation. Section 8 reports on five experiments executed with the real robot. Section 9 concludes the paper and suggests directions for future research. Appendix A provides details on how transition probabilities of the POMDP are computed. Appendix B describes the implemented robotic system and the software modules running on it.

2. RELATED WORK

Several variants of search and pursuit-evasion problems have attracted considerable interest in the robotics community [Chung at al. 2011; Guibas et al. 1999; Isler at al. 2005; Lee et al. 2002; Park et al. 2011; Sarmiento et al. 2003; Tovar and LaValle 2008]. A particularly interesting one is the problem of finding a target in an environment of given geometry using the fewest number of robot searchers. For instance, in [Guibas et al. 1999] the searchers, which are equipped with visibility sensors, are required to plan and perform coordinated trajectories that guarantee the eventual visual detection of a mobile target (also called the evader), even if it tries to escape the searchers. However, a common strong assumption in this type of work is to represent both the searchers and the target as points and to consider that the target is detected as soon as the line segment between a searcher and the target intersects no obstacle, i.e, a line of sight exists between them. As mentioned in the Introduction (Section 1), this kind of work would benefit from using a more realistic target detector.

The problem of searching static objects in an environment has also been treated as an active vision problem [Tsotsos and Shubina 2007; Ye and Tsotsos 1999] by selecting several successive viewpoints. In fact, active vision [Aloimonos et al. 1988; Bajcsy 1988]—the use of motion to acquire pertinent sensing information about an environment—is a recurrent theme in robotics. It arises in many applications, for instance, in robotic manipulation [Jang et al. 2008 ; Motai and Kosaka 2008], surveillance [Bakhtari et al. 2009; Schroeter et al. 2009], tracking [Barreto et el. 2010], object modeling [Pito 1999], localization and mapping [Fintrop and Jensfelt 2008; Kaess and Dellaert 2010; Zingaretti and Frontoni 2006], and object recognition (see below), just to mention some. A rich survey of active vision methods used in robotic systems is given in [Chen et al. 2011]. The active vision approach that we propose in this paper mixes localization and identification. Unlike in [Fintrop and Jensfelt 2008; Kaess and Dellaert 2010; Zingaretti and Frontoni 2006] localization is not the main goal. It is only used as a source of information to better plan the next viewpoints and in the end confirm the identity of an object with greater reliability.

The goal of object recognition is usually to detect and identify important objects and in some cases to estimate their poses. Some of the earliest works are presented in [Bajcsy 1988; Krotkov and Bajcsy 1993]. This line of research can be organized according to different criteria, e.g. whether the sensing device is static [Lowe 2004] or not [Browatzki et al. 2012], whether view planning optimizes only over the next viewpoint [Browatzki et al. 2012; Eidenberg and Scharinger 2010; Meger et al. 2010] or over a sequence of viewpoints [Atanasov et al. 2014], and which optimization function is used for view planning. Regarding optimization, a common approach is to use an entropy-related information gain criterion [Eidenberg and Scharinger 2010; Laporte and Arbel 2006; Meger et al. 2010; Pito 1999]. A different type of optimization criterion quantifies a tradeoff between the length of the sensor movement, the energy expenditure to move the sensor, and the expected cost of an incorrect classification [Atanasov et al. 2014]. Here we propose yet another type of optimization criterion aimed at guiding the robot toward a position where, with high probability, the target detector will confirm with high confidence that the candidate object is actually the target. Moreover, unlike some recognition methods (e.g. [Atanasov et al. 2014]) that require an accurate estimate of the sensor pose to plan the next best views, our approach models the position of the robot (and its sensors) relative to the candidate object as a probabilistic distribution over a cell decomposition of the space around the object.

Also relevant to our work, a method for learning viewpoint detection models for object categories is presented in [Meger et al. 2010]. The generated models are used in a sequential object category recognition task, where the robot must infer the likelihood that an object has a category label x given N detector's responses received so far. Each new viewpoint is chosen to minimize the entropy over the posterior belief of the category labels of the object. Our approach is also based on the idea that the choice of viewpoints is critical to identify an object. However, our formulation of the identification problem as a POMDP does not lead to concentrating the peak of a probability distribution over a collection of object categories on a particular category, as is done in [Eidenberg and Scharinger 2010; Meger et al. 2010]. Instead, it is focused on one single object, the target, and only considers the model of this object; so, it does not need models for many objects that are not potential targets.

As we mentioned in the Introduction the problem of sweeping an environment to find a target object has been addressed in [Espinoza et al. 2011]. The work presented in this paper is complementary to this previous work. We think that, to be performed quickly, the detection of a target during an environment sweep must only rely of crude, but easy-to-detect visual clues; hence it cannot be fully reliable. Therefore, once a candidate object has been detected, a different approach is needed to confirm (or infirm) that this object is actually the target. The rest of this paper presents such an approach.

3. TARGET DETECTOR AND OBSERVATION MODEL

We assume that during the confirmation process the candidate object C is static and not significantly hidden by obstacles, and that the robot moves on a planar horizontal surface around C. We define the position of the robot to be that of the

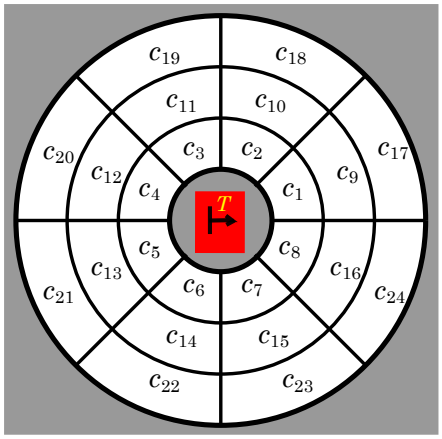


Fig. 1. Cell decomposition of the space around T

camera mounted on it. Either the robot can rotate in place (e.g., a differential-drive robot), or there is a revolute joint (around a vertical axis) between the robot and the camera, so that pointing the camera toward C does not introduce additional constraints on the path followed by the robot.

Before the robot can be used to confirm that a candidate object C is a certain target T, it must be equipped with a vision software module D_T, the target detector for T, capable of identifying T with high confidence, at least from some viewpoints. Given an arbitrary image taken by the robot's camera, D_T returns a *confidence score* $y \in \{o_1, o_2, .., o_n\}$, where $o_1 < o_2 < ... < o_n$, which measures how well a subset of the image matches the appearance of T. The response o_n indicates very high confidence, whereas o_1 means very poor confidence. We expect that D_T can occasionally make false positive mistakes, for instance because the image shows an object similar to T or because the detector is deceived, e.g., due to poor illumination and/or a textured background. In all the experiments reported in this paper, we have used detectors that return 6 levels of confidence, i.e., $n = 6$.

In our implemented system, the detection method used in D_T is the deformable part model algorithm proposed in [Felzenzwalb et al. 2010] (see Section 7.1). Our framework is independent from this specific method; other detection methods (or combinations of methods) could be used as well. However, the success rate of the overall system, i.e., its ability to correctly confirm or infirm the presence of the target, depends on both the framework (along with its inner parameters) and on the particular method used by the detector D_T.

The robot should be equipped with different detectors D_{T_i} to identify different targets T_i, $i = 1, 2, \cdots$. These detectors may all use the same method, each with different data, or different methods. In our current system, we assume that each target has a single equilibrium pose on a horizontal surface. Although many objects encountered in practice have one preferred pose, this is a limitation for practical use in the future. For a target with multiple possible poses, a possible extension of our method would be to train several detectors, one for each possible pose (see Section 9).

The *observation model* of T is a trained probabilistic model of the response of D_T as a function of the position of the robot relative to T. Since we assume that the target object is over a given two-dimensional plane, to train this model, we decompose the space around T into non-overlapping cells $c_1, ..., c_m$.

Figure 1 shows a particular decomposition made of $m = 24$ cells created by drawing concentric circles, the center of which is roughly the center of mass of the projection of T on the horizontal plane, and erecting regularly-spaced radial rays from this center. The decomposition is bounded by an inner circle slightly larger than the enclosing circle of the horizontal projection of T and an outer circle several times larger than the inner circle. Although other decompositions are possible, we will use this decomposition throughout the rest of this paper. This decomposition is well adapted to the identification

of relatively small objects. For large objects with elongated shapes, another type of decomposition might be preferable. Nevertheless, the proposed decomposition enjoys some interesting properties described in Appendix A.

Given D_T and the decomposition around T, the observation model of T is created in the form of a probability distribution $P(o_j|c_i)$, $i \in \{1,...,m\}$ and $j \in \{1,...,n\}$, where $P(o_j|c_i)$ is the probability that the response of D_T is o_j given that the input image was taken from within cell c_i. For each cell c_i, a set of images of T is taken at random positions of the robot in c_i (with the camera pointing toward T at each position). The normalized frequency of each response o_j of D_T for all these images gives $P(o_j|c_i)$.

4. CONFIRMATION PROCESS AND ROBOT LOCALIZATION

Consider now the situation reached by Step 2 of the scenario presented at the beginning of the Introduction (Section 1): based on simple visual clues the robot has detected an object C that may be the target T, but it is not sure yet that C is T. The confirmation process (Step 3 of the scenario) consists of computing and executing a series of motion commands u_t at times $t = 0, 1, 2, ...$, until it is confirmed (or infirmed) that C is T.

We assume around C the existence of the same cell decomposition $X = \{c_1, ..., c_m\}$ that was used around T to create the observation model of T (see Section 3). But the robot knows neither the orientation of this decomposition, nor the location of its center. In fact, if C is not T, any positioning of the cell decomposition would be arbitrary.

At each time t, the robot's belief about its position relative to the potential target is modeled at the cell resolution of X by a probability distribution $P(x_t)$ over the m cells of X, i.e., x_t ranges over $\{c_1, ..., c_m\}$. One way to interpret $P(x_t)$ is the following: the robot hypothesizes that C is the target T and treats $P(x_t)$ as the distribution of its possible positions (at the cell resolution) relative to T. In the case where C is not T, $P(x_t)$ is meaningless, but the confirmation process should fail anyway.

The visual sensors of the robot in our implemented system includes a range sensor (see Section 8.1) that is used at Step 2 of the scenario to bring the robot within a distance of C that is compatible with the inner and outer radii of the decomposition, for instance at a distance rougly equal to the outer radius minus the inner one. The robot knows nothing else about its initial position. So, at time -1 (by convention the time just before running the detector D_T for the first time at time 0), the robot's belief about its position is uniformly distributed over the m cells, i.e., $P(x_{-1} = c_i) = 1/m$ for all $i = 1, ..., m$.

At every time $t \geq 0$ during the confirmation process the robot points its camera toward C, takes an image of C, and runs the detector D_T on this image. The new probability distribution of the robot's position over X, $P(x_t|I_t)$, is then inferred as follows from the observation model of T and the history $I_t = \{y_t, u_{t-1}, I_{t-1}\}$ of applied motion commands and scores returned by the detector D_T [LaValle 2006; Thrun et al. 2005]:

$$P(x_t|I_t) = \frac{P(y_t|x_t) \sum_{x_{t-1}} P(x_t|x_{t-1}, u_{t-1}) P(x_{t-1}|I_{t-1})}{\sum_{x_t} P(y_t|x_t) \sum_{x_{t-1}} P(x_t|x_{t-1}, u_{t-1}) P(x_{t-1}|I_{t-1})} \quad (1)$$

where $t \geq 0$, x_t and x_{t-1} range over $\{c_1, ..., c_m\}$, and $P(x_{-1}|I_{-1})$ is the uniform distribution over the cells of X. The set of motion commands that the robot may perform at each step and the probability distribution $P(x_t|x_{t-1}, u_{t-1})$, called the *transition model*, will be presented in the next section (Section 5). Equation (1) combines two distinct steps: a prediction step that corresponds to applying the transition model $P(x_t|x_{t-1}, u_{t-1})$ to the last robot's position belief $P(x_{t-1}|I_{t-1})$, and an update step that corresponds to applying the observation model $P(y_t|x_t)$ after the prediction step.

To confirm that C is the target T the robot must reach a position at some time $t+1$ where the detector D_T returns a high confidence score $y_{t+1} \geq \hat{o}$, where $\hat{o}$ is a given threshold (in most of our experiments we set $\hat{o}$ to the highest score o_n, i.e., o_6). But this condition is not sufficient, because it could be achieved by chance (false positive), especially if C is not T but resembles T under some viewpoints. So, we impose another condition: at time t the robot must also expect with high probability that the detector D_T will return a high score at time $t+1$. This second condition can be written formally as

follows:

$$P(y_{t+1} \geq \hat{o}|I_t, u_t) > \lambda$$

where $P(y_{t+1} \geq \hat{o}|I_t, u_t)$ is computed from the observation and transition models (see Section 6) and λ is a given threshold defining the term "high probability". The achievement of this twofold condition results in an intricate mix of robot localization and target identification. When C is not T, the robot will estimate incorrectly its position relative to the candidate target. So, the mass of the probability distribution of the robot's position derived from the sequence of images up to time t will be unlikely to lie in cells where D_T is expected to return a high confidence score. In this way, the twofold termination condition drastically reduces the risk of a false positive, as will be empirically demonstrated by Experiment #2 in Section 7.3.

As formulated above, target confirmation is a Partially-Observable Markov Decision Process (POMDP) [Candido and Hutchinson 2011],[Kaelbling et al. 1998], where X is the set of states, $P(o_j|c_i)$ the observation model, and $P(x_t|x_{t-1}, u_{t-1})$ the transition model. This formulation differs from entropy minimization formulations used for related problems. Unlike in some object recognition systems, we do not try to concentrate the probability mass on a particular object (here, the target T) in a given collection of objects, as has been proposed in [Meger et al. 2010]. Instead, we focus on the sole appearance of the target and we avoid the need to generate observation models for many objects that are not potential targets. Entropy minimization is also commonly used in pure localization problems [Thrun et al. 2005]. Here, it would lead the robot to concentrate the mass of the probability distribution $P(c_i)$, $i = 1, \ldots, m$ in as few cells of X as possible, where the detector might not be guaranteed to return high confidence scores. Our formulation is more closely related to the POMDP formulation of object recognition presented in [Eidenberg and Scharinger 2010], but is nevertheless quite different. In the tradition of active perception, the work in [Eidenberg and Scharinger 2010] aims at recognizing all objects in a multi-object given scene, along with their respective 6D poses, by moving a visual sensor around the scene and taking successive images. The proposed method represents the collection of objects in the scene with probabilistic object hypotheses that are updated after each new sensing operation. Planning is used to refine probabilistic hypotheses more quickly, while dealing with occlusions. Unlike in our approach, there is no notion of target object and the sensor is not localized relative to any object to improve the reliability of its identification.

5. MOTION COMMANDS AND TRANSITION MODEL

To complete the formulation of the POMDP introduced above, we now present the state transition model, that is, the set of motion commands that the robot can perform and the probability distribution $P(x_t|x_{t-1}, u_{t-1})$ for each one of these commands.

At each step t of the confirmation process, we allow the robot to choose among 4 motion commands:

$$U = \{forward, backward, left, right\}$$

defined as follows and illustrated in Figure 2:

- The command *forward* makes the robot move in the direction of the estimated center of C in the image taken at t. The command *backward* makes the robot move in the opposite direction.

- The commands *left* and *right* make the robot move perpendicularly to the estimated direction of the center of C at time t, respectively in the clockwise and counter-clockwise directions.

In our implemented system the length of each motion is set to be slightly greater than the distance between two consecutive circles of the decomposition. This choice ensures that the robot crosses cell boundaries often and thus gets a variety of viewpoints over a few moves. However, for the commands *forward* and *backward* the robot uses its range sensor

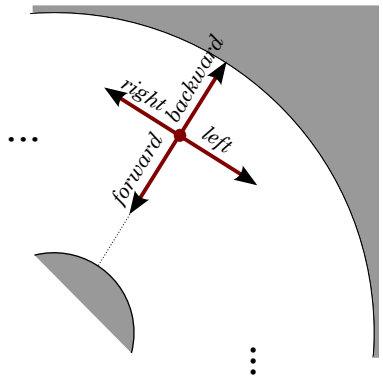


Fig. 2. Motion commands

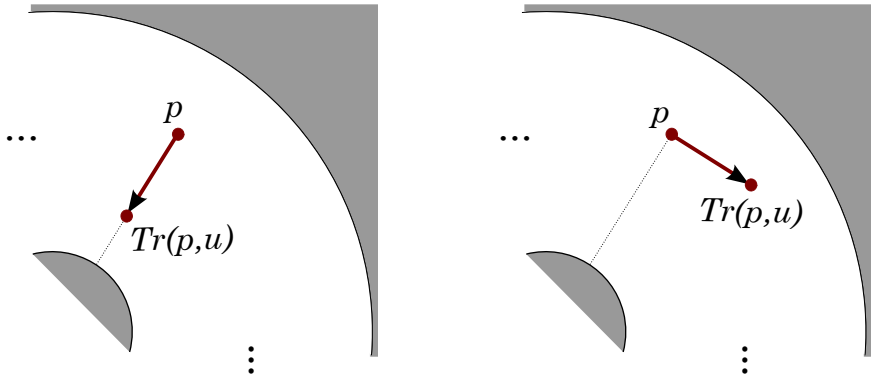
to remain within a distance of C compatible with the inner and outer radii of the decomposition X: if it gets too close to C or too far from it, then the length of the motion is reduced.

Recall from Section 4 that the robot estimates its position at the cell resolution in X. This means that when the robot believes that its position is within a cell c_i, it can be anywhere in c_i with a uniform distribution. The transition probability $P(x_t = c_j | x_{t-1} = c_i, u_{t-1})$ used in Equation (1) estimates the position of the robot after executing the command u_{t-1} (one of the 4 commands introduced above) from within cell $x_{t-1} = c_i$. To compute $P(x_t = c_j | x_{t-1} = c_i, u_{t-1})$ let us first define the transform of a cell $c \in X$ by a motion command $u \in \{forward, backward, left, right\}$. This transform is a region denoted by $Tr(c,u)$.

Consider an arbitrary point p in $c \in X$. We define the transform $Tr(p,u)$ of p by u as follows:

- If $u = forward$ (resp. *backward*), then $Tr(p,u)$ is the point obtained by translating p along a radial line of X toward (resp. away from) the center of X by the length of the motion. See Figure 3a.

- If $u = left$ (resp. *right*), then $Tr(p,u)$ is the point obtained by translating p perpendicularly to the direction of the center of C, respectively in the clockwise and counter-clockwise directions, by the length of the motion. See Figure 3b.



(a) $u = forward$ (b) $u = left$

Fig. 3. Transform $Tr(p,u)$ of p by u

We define $Tr(c,u)$ to be the set formed by the transforms of all the points p in c by u, hence: $Tr(c,u) = \{Tr(p,u) \,|\, p \in c\}$.

The probability $P(x_t = c_j | x_{t-1} = c_i, u_{t-1})$ is calculated as the ratio of the area of the region $Tr(x_{t-1} = c_i, u_{t-1})$ that is contained in cell $x_t = c_j$. To illustrate, Figure 4 shows the transform of c_9 by command *right*. This transform is the region $c'_9 = Tr(c_9, u_{t-1} = right)$. The region c'_9 is a region that overlaps several cells.

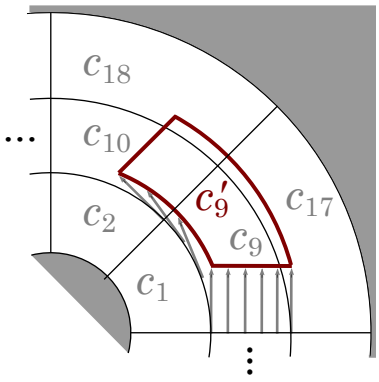


Fig. 4. $c'_9 = Tr(x_{t-1} = c_9, u_{t-1} = right)$

This calculation makes the implicit assumption that the robot moves are "ideally" performed toward and away from the center of X (for *forward* and *backward*) and around this center (for *left* and *right*). However, the robot never tries to localize the decomposition X relative to C, as it does not even know whether the object C is actually the target T. Nevertheless, we believe that, when C is T, the moves performed by the robot are close to these ideal moves. The experiments with a real robot (Section 8) validate this claim. Moreover, the Experiment #4 in simulation (Section 7.5), which analyzes the impact of increasing errors in the estimate of the direction of the center of X, demonstrates that our method tolerates large errors quite well, as long as the candidate object remains within the field of view of the camera. In the future, if needed, the uncertainty in the direction of the center of X could be taken into account in the computation of the probability $P(x_t = c_j | x_{t-1} = c_i, u_{t-1})$ by enlarging the region $Tr(x_{t-1} = c_i, u_{t-1})$ defined above.

The transition probabilities $P(x_t = c_j | x_{t-1} = c_i, u_{t-1})$ can be calculated analytically for each one of the four commands, as shown in Appendix A.

6. COMPUTATION OF A MOTION STRATEGY

In Section 4 we have formulated the process of confirming the identity of C as a POMDP in which the states are the cells of X. Here, we compute a motion strategy by using Stochastic Dynamic Programming (SDP) with imperfect state information as proposed in [Bertsekas 2000].

We specify two horizons: a *planning horizon* N and an *execution horizon* $N_e = N_p \times N$. The SDP-based planner computes a motion strategy up to horizon N. It is invoked up to N_p times and each generated plan is immediately executed. The process stops when the twofold confirmation condition presented in Section 4 is achieved—then the identity of C as T is confirmed—or when N_p plans have been executed without success—then C is considered to be an object other than T.

The SDP-based planner computes a motion strategy in the form of a policy $\pi(t, I_t)$ that gives the motion command to be applied at every time $t = 0, 1, 2, ..., N-1$ for every possible history I_t. For this computation, SDP is applied with Equations (2), (3), (4), (5), and (6), where $\tilde{g}(I_t, u_t)$ and $g_F(x_N)$ are gain functions [Bertsekas 2000]:

$$J_{N-1}(I_{N-1}) = \max_{u_{N-1} \in U_{N-1}} \left[\tilde{g}(I_{N-1}, u_{N-1}) + \underset{x_{N-1}}{E} \left\{ \underset{x_N}{E} \{ g_F(x_N) | x_{N-1}, u_{N-1} \} | I_{N-1}, u_{N-1} \right\} \right], \tag{2}$$

$$\pi(N-1, I_{N-1}) = \arg \max_{u_{N-1} \in U_{N-1}} \left[\tilde{g}(I_{N-1}, u_{N-1}) + \underset{x_{N-1}}{E} \left\{ \underset{x_N}{E} \{ g_F(x_N) | x_{N-1}, u_{N-1} \} | I_{N-1}, u_{N-1} \right\} \right], \tag{3}$$

and for $t < N-1$

$$J_t(I_t) = \max_{u_t \in U_t} \left[\tilde{g}(I_t, u_t) + \underset{y_{t+1}}{E} \{ J_{t+1}(I_t, y_{t+1}, u_t) | I_t, u_t \} \right], \tag{4}$$

$$\pi(t, I_t) = \arg \max_{u_t \in U_t} \left[\tilde{g}(I_t, u_t) + \underset{y_{t+1}}{E} \{ J_{t+1}(I_t, y_{t+1}, u_t) | I_t, u_t \} \right]. \tag{5}$$

Since we want the robot to reach a position where the condition $P(y_{t+1} \geq \hat{o}|I_t, u_t) > \lambda$ is satisfied, we set the gain function $\tilde{g}(I_t, u_t)$ to $P(y_{t+1} \geq \hat{o}|I_t, u_t)$ as given by Equation (6) below. This equation makes use of the observation model, the transition model, and the current robot's belief about its position:

$$P(y_{t+1} \geq \hat{o}|I_t, u_t) = \sum_{x_{t+1}} P(y_{t+1} \geq \hat{o}|x_{t+1}) \sum_{x_t} P(x_{t+1}|x_t, u_t) P(x_t|I_t). \quad (6)$$

In Experiments #1 and #2 reported later in this paper, the other gain function $g_F(x_N)$ used in Equations (2) and (3) will be set to the identically null function. In Section 7.4 (Experiment #3), we will introduce a non-zero function $g_F(x_N)$ used in all other experiments that reduces the risk of local maxima in the function J_t.

Every plan generated by the SDP-based planner is immediately executed. If the confirmation condition is achieved—i.e., if the robot reaches at time t a position where the condition $P(y_{t+1} \geq \hat{o}|I_t, u_t) > \lambda$ is satisfied *and* if the detector actually returns a confidence score greater than $\hat{o}$ on the image taken at time $t+1$—then the confirmation process ends with success. Otherwise, a new plan is computed with the current robot's belief about its position, again with horizon N. As indicated above, this iterative process ends with failure if N_p plans have been executed without achieving the confirmation condition. Then the robot returns that C is not the target.

There are clearly tradeoffs in setting the values of N and $N_e = N_p \times N$. If N is too small, the successive plans may be too short-sighted and the computation may get trapped into a local maximum[1] in the function J_t. We will specifically discuss this issue in Section 7.4. Ideally, N should be set large enough so that the planner's computation reaches states where large confidence scores are likely to happen. But planning time increases quickly with N. Moreover, if uncertainty is too large (e.g., if the the quality of the observation model is poor), then the computed plans may not be better (as will be discussed in Section 8.3). On the other hand, if the ratio $N_p = N_e/N$ is too small then the robot may incorrectly fail to confirm that C is the target T, whereas if this ratio is too large and C is not T, the robot may waste too much time before returning that C is not T. One goal of our experiments will be to investigate these tradeoffs (see Section 7.3). To summarize, our experiments show that low-resolution cell decompositions, hence with relatively few cells, work best. Such decompositions combined with the gain function g_F do not require large planning horizons. Most of the best results are obtained with $N = 3$.

7. EXPERIMENTS WITH SIMULATION ROBOT

7.1. General setting

We created the textured virtual environment shown in Figure 5 in order to perform basic tests of our approach under well-controlled conditions. The object to be identified is placed at the center of the circular table in the middle of the room. The simulated robot can move around this table. Synthetic images from the viewpoint of a virtual camera mounted on the robot are generated at successive positions of the robot, and the detector D_T is fed with those images.

For each target T, the detector D_T uses the deformable part model algorithm presented in [Felzenzwalb et al. 2010]. This same algorithm will also be used in the experiments carried with a real robot (Section 8). Our choice was motivated by the fact that this algorithm has been tested on large databases of images taken under very diverse conditions, e.g., of scale and illumination. However, our framework does not rely on this particular algorithm; it could use other object recognition algorithms or combinations of algorithms.

In all the experiments reported in this section the planner uses the 24-cell decomposition shown in Figure 1. (We will vary the number of cells in Section 8.5.) For each target T, the detector D_T is trained on a set of images taken from a single cell c_g of the decomposition. This cell is selected so that distinctive features of the target are visible from it. Therefore, for each target, the detector returns high confidence scores when the robot is in c_g. The scores tend to decrease when the robot moves away from c_g, except when the target has a similar appearance under other viewpoints. The scores range over

[1] SDP delivers the global maximum for a given planning horizon, but such maximum might correspond to a local maximum for a larger planning horizon.

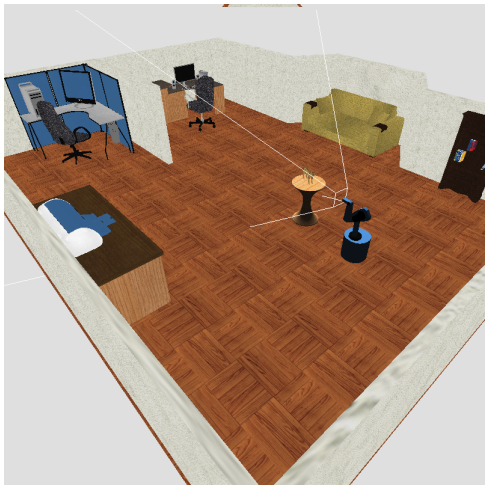


Fig. 5. Virtual environment.

6 values, i.e., $n = 6$. We also simulate a range sensor that prevents the robot from going too close or too far away from the candidate object C.

In all experiments the center of the cell decomposition X is the center of the circular table. In Experiments #1, #2, and #3, the robot knows exactly the position of the table's center and therefore performs the four motion commands with accurate knowledge of the direction of this center. In Experiment #4, we will consider the case where this knowledge is imperfect and we will analyze the impact of increasing errors in the estimate of the direction of the center. In Experiments #1 and #2 the gain function g_F used in Equations (2) and (3) is set to the identically null function.

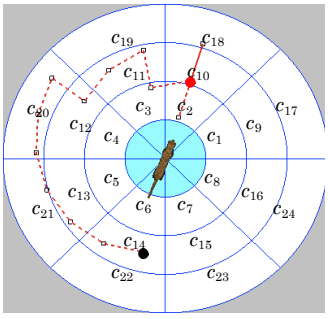
In all the statistics tables shown below, the average numbers of sensing locations, path lengths and planning times are computed only over the runs that achieve confirmation.

7.2. Experiment #1: Plastic Cat

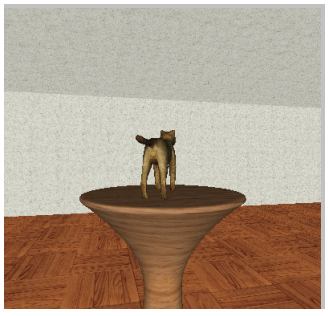
	o_1	o_2	o_3	o_4	o_5	o_6
c_1	0.236	0.324	0.220	0.140	0.068	0.012
c_2	0.000	0.000	0.000	0.132	0.644	0.224
c_3	0.196	0.376	0.224	0.160	0.044	0.000
c_4	0.916	0.084	0.000	0.000	0.000	0.000
c_5	0.312	0.560	0.128	0.000	0.000	0.000
c_6	0.000	0.876	0.124	0.000	0.000	0.000
c_7	0.144	0.688	0.168	0.000	0.000	0.000
c_8	1.000	0.000	0.000	0.000	0.000	0.000
c_9	0.152	0.412	0.156	0.180	0.092	0.008
c_{10}	0.000	0.000	0.000	0.000	0.428	0.572
c_{11}	0.112	0.360	0.268	0.096	0.144	0.020
c_{12}	0.896	0.104	0.000	0.000	0.000	0.000
c_{13}	0.276	0.616	0.108	0.000	0.000	0.000
c_{14}	0.000	0.420	0.580	0.000	0.000	0.000
c_{15}	0.200	0.540	0.260	0.000	0.000	0.000
c_{16}	1.000	0.000	0.000	0.000	0.000	0.000
c_{17}	0.596	0.204	0.124	0.068	0.008	0.000
c_{18}	0.000	0.040	0.320	0.372	0.260	0.008
c_{19}	0.440	0.244	0.232	0.064	0.020	0.000
c_{20}	1.000	0.010	0.290	0.390	0.290	0.010
c_{21}	0.804	0.196	0.000	0.000	0.000	0.000
c_{22}	0.068	0.880	0.052	0.000	0.000	0.000
c_{23}	0.740	0.260	0.000	0.000	0.000	0.000
c_{24}	1.000	0.000	0.000	0.000	0.000	0.000

Table 1: Observation model $P(y = o_j | x = c_i)$ of the plastic cat in Experiment #1

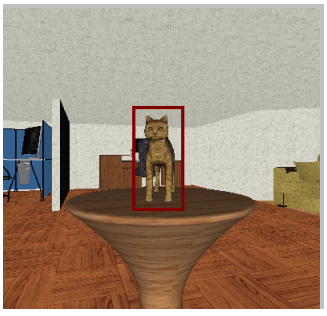
(a) Plastic cat as target



(b) Path generated with $N = 3$ and $\lambda = 0.55$



(c) Initial view of candidate object C



(d) View of object C when it is confirmed to be the target

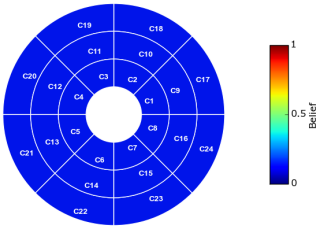
Fig. 6. Illustrations for Experiment #1

The purpose of this experiment is to verify the ability of our method to reliably identify an object and to compare its performance with other motion strategies. The target is the plastic cat shown in Figure 6a. The placement of the cat on the table and the cell decomposition used for training the detector are shown in Figure 6b. We chose cell c_{10} as c_g and we trained the detector on 100 images obtained from viewpoints randomly picked in c_{10}. After training the detector, we sampled 200 images over each of the 24 cells to obtain the detector's observation model shown in Table 1. As expected, with high probability, the detector returns high scores, i.e. o_5 or o_6, in c_{10}. Conversely, the probability that the detector returns the lowest scores o_1 and o_2 is greater in cells distant from c_{10}.

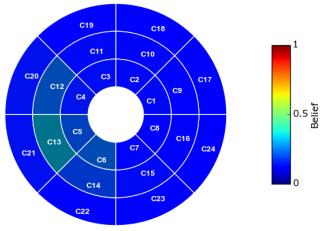
To test our method, we placed the robot in cell c_{14}, which is located far away from c_{10}. Using the SDP-based planner of Section 6, we generated motion strategies with 4 planning horizons $N = 1, 2, 3$ and 4. The execution horizon was set to $N_e = 100$ and the threshold $\hat{o}$ to o_6 (the highest confidence score of the detector). Runs were performed with 3 values of the probability threshold λ: 0.45, 0.50, and 0.55.

Note that the combination of Equation (6) and the observation model of Table 1 determines the upper-bound $\max_{c_i}[P(y \geq \hat{o}|c_i)]$ for λ, beyond which it is impossible to confirm that the candidate object X is the target T. So, the value selected for λ should be less than this upper-bound. Here the bound is 0.572. It depends on the shape and resolution of the decomposition, as well as on the method used by the detector. Methods that yield higher bounds are preferable.

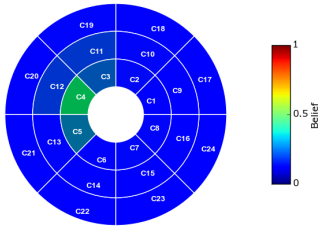
The path followed by the robot in a run with $N = 3$ and $\lambda = 0.55$ is plotted in Figure 6b. The initial position of the robot is shown with a black circle, its final configuration as a red (gray) circle, and the intermediate sensing locations as small squares. Figure 7 shows the probability distribution $P(x_t|I_t)$ at the beginning (a), one fourth (b), midway (c), three fourths (d), and end (e) of the confirmation process. In Figure 7a the distribution is uniform over the 24 cells. The initial appearance of the cat (then the candidate object C) is shown in Figure 6c. The detector returns a low score, so that the robot needs to move to better viewpoints. Figure 6d shows C's final appearance when the robot confirms that C is the cat.



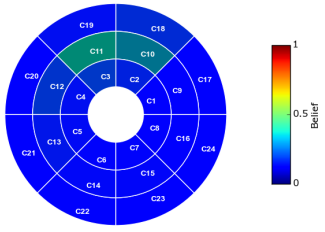
(a) $P(x_t|I_t)$ at the beginning of the confirmation process



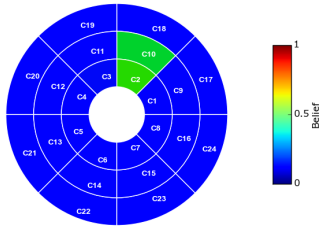
(b) $P(x_t|I_t)$ at one fourth of the confirmation process



(c) $P(x_t|I_t)$ at the middle of the confirmation process



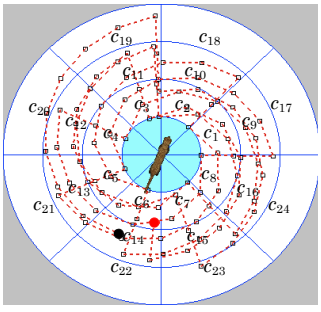
(d) $P(x_t|I_t)$ at three fourths of the confirmation process



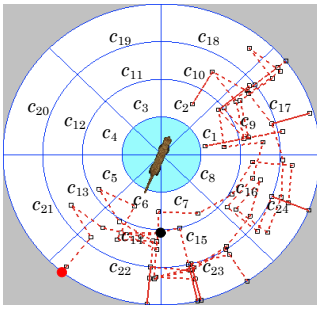
(e) $P(x_t|I_t)$ at the confirmation moment

Fig. 7. Evolution of probability distribution $P(x_t|I_t)$ during a trial with the plastic cat as target ($N = 3$)

In this run, the confirmation process required 15 moves (some sensing locations at the end of the path are repeated as the robot moves back and forth to refine its localization and eventually confirm the object's identity).



(a) Fixed strategy path



(b) Random strategy path

Fig. 8. Two sample paths using fixed and random strategies with $\lambda = 0.55$

For comparison, we also ran separately two different motion strategies. One is a "fixed strategy" that consists for the robot to move around the candidate object in clockwise direction. The other is a "random strategy" in which the robot performs one of the 4 motion commands picked uniformly at random at each step. For both strategies, the maximum number of steps was set to $N_e = 100$. Figures 8a and 8b plots the paths of the robot in a run of the fixed strategy and a run of the random strategy, respectively. In each of these two runs the robot was unable to confirm the identity of the object (using the same confirmation test as described in Section 6).

Table 2 shows some statistics (average number of sensing locations, average path length, and average total planning time) for Experiment #1, as well as the percentage of successful confirmation. Each line of the table was obtained over 200 runs with different initial positions of the robot within cell c_{14} (more runs did not produce significantly different results).

Planning horizon	λ	# of sensing locations	Path length	Planning time (ms)	% of confirmation
1	0.45	33.698	32.580	9.854	93
	0.50	42.994	41.837	12.693	88
	0.55	42.493	41.385	12.513	88
2	0.45	12.915	11.816	37.428	100
	0.50	13.150	12.068	38.004	100
	0.55	14.385	13.305	42.443	100
3	0.45	12.837	11.471	405.655	100
	0.50	13.120	11.715	415.278	100
	0.55	13.875	12.385	440.959	100
4	0.45	12.230	11.028	35485.734	100
	0.50	12.587	11.402	37040.285	100
	0.55	13.655	12.431	40236.170	100
Random	0.45	55.750	43.344	-	22
	0.50	62.962	46.266	-	26.5
	0.55	61.354	47.201	-	15.5

Table 2: Statistics for Experiment #1 (cat)

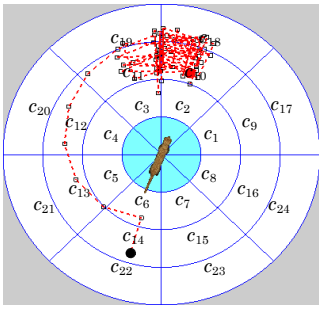


Fig. 9. Path generated with $N = 1$ and $\lambda = 0.55$

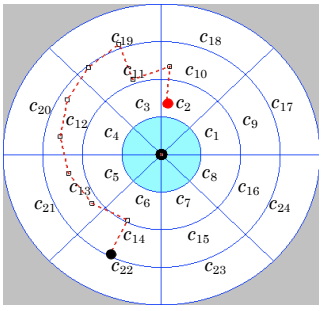
Statistics for the fixed strategy are not included in the table because all 200 runs failed to confirm the target's presence. With planning horizons N set to 2, 3, and 4, the presence of the target is confirmed by our method in all runs for all values of λ. When N is set to 1, the confirmation rate drops to about 90%, but it is still much higher than the rate obtained with the random strategy, which is less than 30%. However, the average number of sensing locations is much larger than for $N = 2$, 3, or 4. In fact, when $N = 1$, it is harder for the robot to localize itself with a precision sufficient to achieve the condition $P(y_{t+1} \geq \hat{o}|I_t, u_t) > \lambda$. This leads the robot to perform a long series of moves around cell c_g, during which it progressively refines its localization. Figure 9 shows an example of such a path. For all values of λ, the smallest number of sensing locations and the shortest path lengths are obtained with our method with $N = 4$. The number of sensing locations and the path lengths are the largest for the random strategy. Note that neither our method, nor the random and fixed strategies know in advance the locations of the viewpoints from which the detector returns a high score. In fact, the random and fixed strategies reach such viewpoints, but they fail to achieve the twofold confirmation condition designed to avoid false positives.

In this experiment our method produces smaller numbers of sensing locations and shorter path lengths when the planning horizon N increases. However, planning time also increases quickly with N (and also with λ). Here, planning takes approximately 90 times longer when $N = 4$ than when $N = 3$, for approximately the same average numbers of sensing locations and lengths of paths. In general, a greater value of N will be required to provide the planner with the same look-ahead when the resolution of the decomposition X increases. So, the best tradeoff value of N will also increase with the resolution of the decomposition X. This experiment suggests that for relatively small objects our 24-cell decomposition and $N = 3$ offer a good tradeoff in terms of planning time, confirmation rate, numbers of sensing locations, and path lengths.

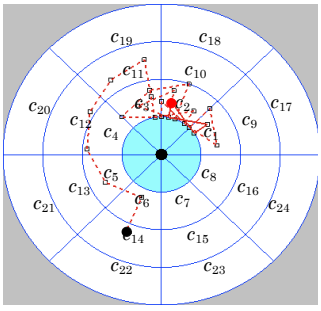
7.3. Experiment #2: Two similar bottles

(a) True bottle (target)

(b) False bottle (not the target)



(c) Path generated with $N = 3$ and $\lambda = 0.8$ in the presence of the true bottle



(d) Path generated with $N = 3$ and $\lambda = 0.8$ in the presence of the false bottle

(e) Final view for the true bottle

(f) Final view for the false bottle

Fig. 10. Illustrations for Experiment #2

The purpose of this experiment is to verify that our robot is able to discriminate between similar objects. We have two bottles, the "true" bottle shown in Figure 10a and the "false" bottle shown in Figure 10b. The true bottle has a circular label that the false bottle does not have. Apart from this label, the two bottles are identical. The target is the true bottle.

Like in Experiment #1, cell c_{10} was selected as c_g. We placed the true bottle at the center of the table so that its two labels are visible from c_{10}. We trained the detector on 100 images obtained from viewpoints randomly picked in c_{10}. We then constructed the observation model in the same way as in Experiment #1. The result is shown in Table 3. Here the probability of getting the highest score o_6 is greater in cell c_2 than in cell c_{10}. This is not very surprising since c_2 offers the same set of viewpoints over the bottle as c_{10}, but at a closer distance.

	o_1	o_2	o_3	o_4	o_5	o_6
c_1	0.000	0.000	0.010	0.477	0.427	0.086
c_2	0.000	0.000	0.000	0.000	0.095	0.905
c_3	0.000	0.000	0.010	0.558	0.392	0.040
c_4	0.000	0.000	0.839	0.161	0.000	0.000
c_5	0.000	0.000	1.000	0.000	0.000	0.000
c_6	0.000	0.000	1.000	0.000	0.000	0.000
c_7	0.000	0.000	0.995	0.005	0.000	0.000
c_8	0.000	0.000	0.834	0.166	0.000	0.000
c_9	0.000	0.000	0.036	0.472	0.437	0.055
c_{10}	0.000	0.000	0.000	0.000	0.317	0.683
c_{11}	0.000	0.000	0.065	0.518	0.392	0.025
c_{12}	0.000	0.010	0.900	0.090	0.000	0.000
c_{13}	0.000	0.000	1.000	0.000	0.000	0.000
c_{14}	0.000	0.055	0.945	0.000	0.000	0.000
c_{15}	0.000	0.000	1.000	0.000	0.000	0.000
c_{16}	0.000	0.126	0.819	0.055	0.000	0.000
c_{17}	0.000	0.276	0.422	0.251	0.051	0.000
c_{18}	0.000	0.000	0.186	0.523	0.291	0.000
c_{19}	0.015	0.171	0.477	0.292	0.045	0.000
c_{20}	0.070	0.598	0.332	0.000	0.000	0.000
c_{21}	0.216	0.533	0.251	0.000	0.000	0.000
c_{22}	0.256	0.693	0.051	0.000	0.000	0.000
c_{23}	0.241	0.573	0.186	0.000	0.000	0.000
c_{24}	0.312	0.523	0.165	0.000	0.000	0.000

Table 3: Bottle observation model $P(y = o_j|x = c_i)$

Scene object	λ	# of sensing locations	Path length	Planning time (ms)	% of confirmation
True Bottle	0.80	10.820	9.346	367.723	100
	0.85	10.825	9.122	361.993	100
	0.90	12.030	9.244	415.965	99.5
False Bottle	0.80	21.333	18.002	721.861	1.5
	0.85	17	14.561	621.074	0.5
	0.90	-	-	-	0

Table 4: Statistics for Experiment #2 (similar bottles)

We tested the ability of our method to confirm (or infirm) the identity of each of the two bottles by placing them at the center of the table with their labels oriented as during the training of the detector. For each bottle we performed runs with planning horizon N set to 3 and execution horizon N_e set to 30. We set the probability threshold λ to 0.80, 0.85, and 0.90. (Note that here the observation model allows us to select greater values of λ than in Experiment #1 with the plastic cat.) In each case, we collected data over 200 runs. Table 4 shows the results. The robot was able to differentiate quite well between the two bottles. When the true bottle (i.e., the target) was present, it confirmed its identity in 100% of the runs when λ was set to 0.80 and 0.85, and in 99.5% of the runs (199 runs out of 200) when λ was set to 0.90 (a threshold very close to the upper limit, 0.905, beyond which confirmation would be impossible). On the other hand, when the false bottle was present, the robot incorrectly confirmed the presence of the true bottle in 1.5% of the runs (3 runs out of 200) when $\lambda = 0.80$, 0.5% of the runs (1 out 200) when $\lambda = 0.85$, and 0% of the runs when $\lambda = 0.90$. Figures 10c and 10d show two paths generated with $\lambda = 0.80$, respectively with the true bottle and with the false bottle. In Figure 10c the confirmation process ends in cell c_2. In Figure 10d the planner also guided the robot toward cell c_2, but the detector failed to return high enough scores and the confirmation process ended with no confirmation after 30 steps.

7.4. Experiment #3: A bottle with two similar faces

Here our goal was to observe the behavior of our method when the target has similar appearances from two different viewpoints. The target is a bottle that looks as in Figure 10a on one side and as in Figure 10b on the opposite side. The detector is trained in cell c_{10} after placing the bottle so that the face with the circular label is visible. The observation

model is shown in Table 5. Like in Experiment #2, the highest probabilities of score o_6 are in cells c_2 and c_{10}. But now, as expected, high scores (mainly o_5) are also likely in cells c_6 and c_{14}, which are opposite to c_2 and c_{10}. Figure 11 plots the probability of getting score o_6 in each of the cells of the decomposition; note the local maximum at cell c_6.

	o_1	o_2	o_3	o_4	o_5	o_6
c_1	0.000	0.000	0.021	0.437	0.472	0.070
c_2	0.000	0.000	0.000	0.000	0.105	0.895
c_3	0.000	0.000	0.015	0.533	0.397	0.055
c_4	0.000	0.000	0.568	0.432	0.000	0.000
c_5	0.000	0.000	0.000	0.377	0.623	0.000
c_6	0.000	0.000	0.000	0.000	0.980	0.020
c_7	0.000	0.000	0.000	0.417	0.583	0.000
c_8	0.000	0.000	0.608	0.392	0.000	0.000
c_9	0.000	0.000	0.080	0.503	0.367	0.050
c_{10}	0.000	0.000	0.000	0.000	0.362	0.638
c_{11}	0.000	0.000	0.050	0.548	0.362	0.040
c_{12}	0.000	0.000	0.829	0.171	0.000	0.000
c_{13}	0.000	0.000	0.050	0.467	0.483	0.000
c_{14}	0.000	0.000	0.000	0.025	0.975	0.000
c_{15}	0.000	0.000	0.005	0.543	0.452	0.000
c_{16}	0.000	0.100	0.779	0.121	0.000	0.000
c_{17}	0.005	0.271	0.513	0.166	0.045	0.000
c_{18}	0.000	0.000	0.196	0.437	0.367	0.000
c_{19}	0.020	0.191	0.447	0.292	0.050	0.000
c_{20}	0.055	0.558	0.387	0.000	0.000	0.000
c_{21}	0.000	0.201	0.523	0.276	0.000	0.000
c_{22}	0.000	0.025	0.447	0.473	0.055	0.000
c_{23}	0.000	0.136	0.548	0.306	0.010	0.000
c_{24}	0.146	0.573	0.281	0.000	0.000	0.000

Table 5: Observation model $P(y = o_j|x = c_i)$ of the bottle with two similar faces (Experiment #3)

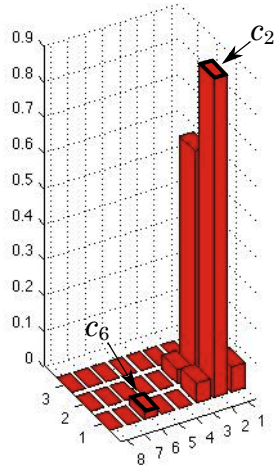
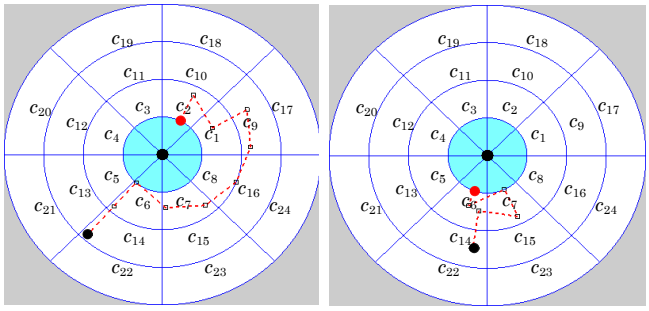


Fig. 11. $P(y = o_6|x = c_i)$

For both $N = 2$ and $N = 3$ we performed 200 runs with the execution horizon N_e set to 30 and λ set to 0.85. With $N = 3$ the presence of the bottle was confirmed in all 200 runs. Figure 12a shows a path followed by the robot in one of these runs. In contrast, when $N = 2$ the planner confirmed the presence of the bottle in only 1% of the runs. In the other runs it got trapped into a local maximum of the function J_t defined in Equations (2) and (4). This maximum is located in cell c_6, which is also a local maximum for score o_6. Figure 12b shows a path of the robot in one of the unsuccessful runs.

When $N = 2$ the computation performed by the planner is too short-sighted to consider cell c_2 where the detector returns score o_6 with probability greater than 0.85. So, instead, it is lured toward cell c_6, which is closer to the starting cell c_{14} and where score o_6 has a local maximum. In the path of Figure 12b the robot first moves from c_{14} to c_6 and then



(a) Path generated with $N = 3$ (b) Path generated with $N = 2$

Fig. 12. Sample paths for bottle with similar appearances from different viewpoints

performs moves between c_6 and c_7 until the execution horizon N_e is attained. Greater planning horizons (here, $N = 3$) are less prone to get trapped into spurious local maxima of J_t.

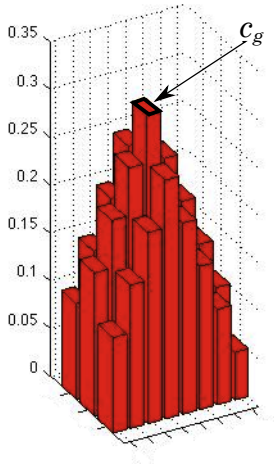


Fig. 13. Function g_F defined with a maximum in cell c_g

To address this local-maxima issue when small planning horizons are used, we added another gain function g_F (in addition to $\tilde{g}$) in Equations (2) and (3). We define g_F as a piecewise constant function (see Figure 13), with constant value over each cell, largest value in cell c_g (here, c_{10}), and decreasing values as we move away from c_g. Hence, g_F gives an incentive for the robot to get closer to c_g. With the addition of g_F, all 200 runs with $N = 2$, as well as all those with $N = 3$, confirmed the presence of the bottle. Figure 14 shows one robot path with $N = 2$.

In all the experiments reported in the rest of this paper (both in simulation and with a real robot) Equations (2) and (3) include the additional gain function c_g defined above.

7.5. Experiment #4: Imperfect knowledge of the table's center

To perform the four motion commands, the robot must estimate the direction of the center of the cell decomposition X. In our setup, this center coincides with the center of the circular table. In Experiments #1, #2, and #3, the robot knew exactly the position of the table's center and therefore performed the four motion commands with accurate knowledge of the direction of X's center. We now consider the case where this knowledge is imperfect and we analyze the impact of increasing errors in the estimate of the center's direction (from now on referred to as orientation errors).

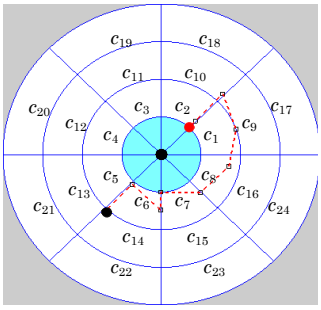
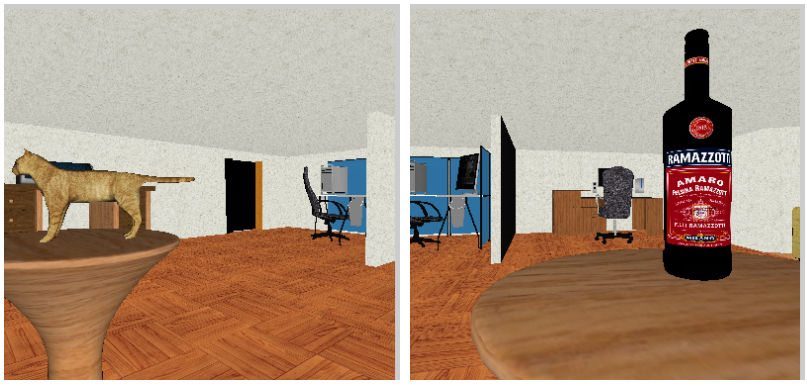


Fig. 14. Robot path with $N = 2$ after introducing function g_F



(a) Cat as target (b) Bottle as target

Fig. 15. Robot views of the cat and bottle with orientation errors

We perform runs with two successive targets: first the cat of Experiment #1, next the true bottle of Experiment #2. We test the impact of orientation errors with different planning horizons by setting N to 2 when the cat is the target and to 3 when the bottle is the target. To create an orientation error we shift the actual direction of the table's center by an angle picked according to a mean-zero normal distribution truncated between -0.5 and $+0.5$ radians. The maximal error of 0.5 radians corresponds to a very bad estimate of the center's direction, but still insures that the candidate object C remains in the camera's field of view. Figure 15 shows two images from the robot's viewpoint with large orientation errors; note the table's center no longer lies on the vertical centerlines of the images. We performed series of runs with standard deviation σ successively set to 0, 0.25, and 0.5.

Scene object	σ	# of sensing locations	Path length	Planning time (ms)	% of confirmation
Cat	0.00	14.385	13.305	42.443	100
	0.25	14.550	13.441	46.178	100
	0.50	14.320	13.201	45.284	100
Bottle	0.00	10.825	9.122	361.993	100
	0.25	11.470	9.418	394.074	100
	0.50	11.335	9.442	389.589	100

Table 6: Statistics for Experiment #4 (error in robot alignment towards C)

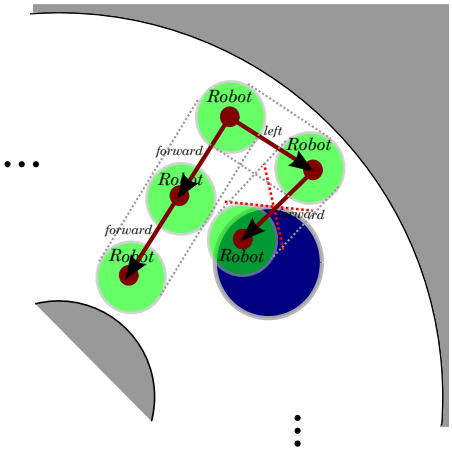
The quantitative results are summarized in Table 6. The most important result is that the two targets were confirmed in all runs for every value of σ. The other statistics in each row of the table are averages over the 200 runs. For both targets the small differences in the average numbers of steps and path lengths are not significant, which we confirmed by performing additional runs (not reported here). However, the average planning times tend to increase slightly when $\sigma \neq 0$.

This is likely due to the fact that the robot moves then diverge more from the ideal ones. Overall these results demonstrate that our approach tolerates large orientation errors quite well, as long as the candidate object remains within the field of view of the camera.

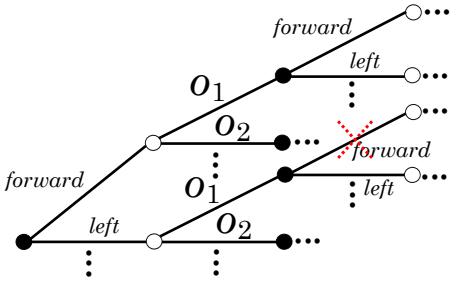
7.6. Dealing with Obstacles Generating Motion and Visibility Constraints

In this subsection we present an extension of our method to deal with obstacles that create both motion constraints and visibility occlusions. Since a crude visual method has detected the candidate object, this object is visible at the beginning of the confirmation process.

We use the range sensor to detect obstacles and measure their position relative to the robot. The position of the robot relative to C is still estimated probabilistically as before. When control commands are being planned, it is possible to predict if the robot will collide with obstacles (see Figure 16a), independently of its position relative to C. The branches of the search tree that cause collision are pruned (Figure 16b). Note that collision prediction can be done at the precision level provided by the range sensor, so that commands may allow the robot to traverse cells that are partially occupied by obstacles.



(a) The robot collides if a *forward* command is applied after a *left* command



(b) SDP Tree is pruned based on collision checking

Fig. 16. Integrating collision checking into the confirmation process

A policy generated with Stochastic Dynamic Programming (SDP) with Imperfect State Information might be described as a tree that relates state beliefs, controls and observations. Our main idea to deal with motion constrains is to prune branches of this tree corresponding to controls yielding to a collision (see Figure 16b). In the planning and execution stages the robot just takes into account collision-free controls, therefore, this allow us to still use the transition model that we have been using so far without the necessity to recompute it integrating the obstacles; indeed, the location of the obstacles in the cellular decomposition with respect to the target, in both the planning and execution stages, is unknown.

During the execution of a sequence of commands we can predict if obstacles occlude C by combining the knowledge of the position of the robot relative to the obstacles with the estimated direction of C. In the absence of obstacles, our method uses a Bayes filter to update the probabilistic model of the position of the robot integrating controls in a prediction step and observations in an update step. Now, if C is occluded by obstacles, only the prediction step is used to compute the probabilistic model. If the robot reaches a position where C is occluded, a new plan is generated from the probabilistic model computed with the prediction step only, temporarily ignoring the observations when updating the position distribution, just applying the transition model until C is no longer occluded. It is important that the planner may send the robot to positions where the target is not visible. Otherwise, the range of valid paths could decrease too much, even to the point where the workspace is disconnected (see Figure 17).

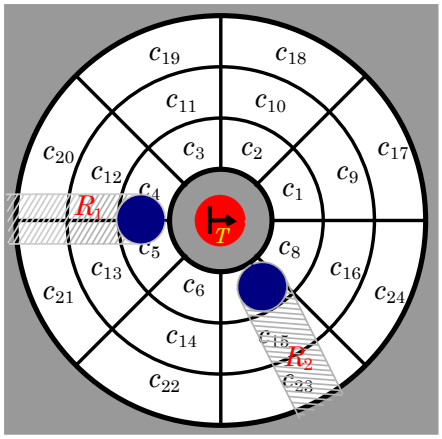


Fig. 17. If we do not allow the robot to reach regions R_1 nor R_2 (where the view of C is obstructed) the workspace would be disconnected

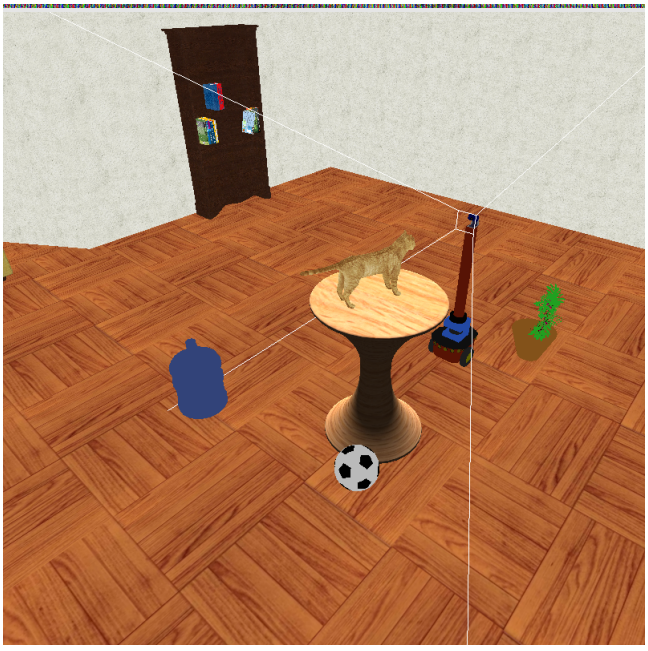


Fig. 18. Motion obstacles around the target (plastic cat) in the virtual environment

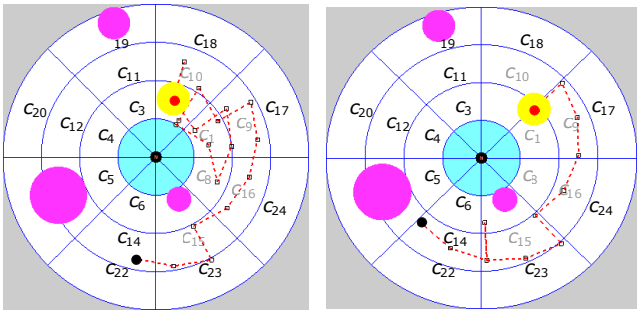
In a first set of runs we tested our method with obstacles that only produce motion constraints, pruning branches of the search tree yielding to collision. The target is the plastic cat of Experiment #1, with three obstacles placed around it as shown in Figure 18. Table 7 gives the average measurements over 100 runs with planning horizon $N = 3$ and execution horizon $N_e = 100$. Comparing these results to those in Table 2, we notice a significant increase in the average number of sensing locations, path length, and planning time. Such an increase was expected, as the set of feasible paths that the robot can now choose from has now decreased. Paths that led the robot to maximize its expected gain in the absence of obstacles are now no longer feasible. Nonetheless a 99% confirmation rate is still achieved.

In our second set of runs the robot is given the task to confirm the presence of the true bottle of Experiment #2. In that experiment (without obstacles) the robot had a tendency to move around the bottle clock-wisely as shown in Figure 10c. Now three obstacles (shown in pink) that produce motion obstructions are included with one of them blocking the clockwise path of the robot. A sample path followed by the robot in the presence of these three obstacles is shown in Figure 19a (the yellow disc is the final position of the robot). Figure 19b shows another path, where the robot starts moving

Planning horizon	# of sensing locations	Path length	Planning time (ms)	% of confirmation
3	22.24	20.84	840.13	99

Table 7: Statistics from 100 runs with motion obstacles and the plastic cat as target

to the right, then it moves forward (into a cell that is partially occupied by an obstacle), but it then moves backwards in order to avoid collision and keeps moving around the bottle until detection is achieved. Table 8 gives average measurements over 100 runs. Again, as expected, there is a neat increase in the number of sensing locations, path length, and planning time, relative to the results shown in Table 4 of Experiment #3. But the confirmation rate is still very good.



(a) Detection in 19 steps (b) Detection in 12 steps

Fig. 19. Sample paths for the true bottle while avoiding motion obstacles

Planning horizon	# of sensing locations	Path length	Planning time (ms)	% of confirmation
3	23.72	20.80	999.45	99

Table 8: Statistics from 100 runs with motion obstacles and the true bottle as target

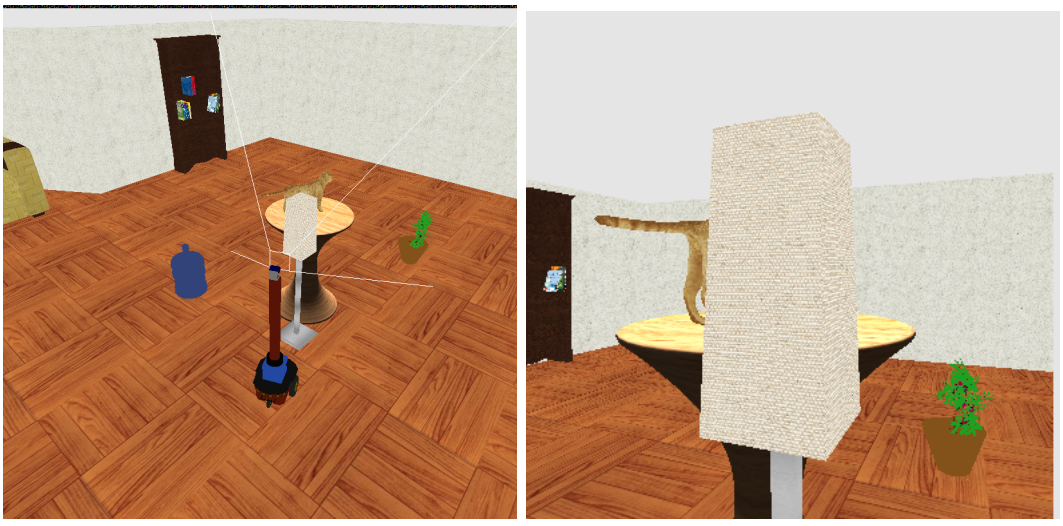
In our third set of runs we repeated the bottle set experiment, but now one of the three obstacles (the one in the inner ring of the cell decomposition) also produces visibility constraints (see Figure 20a and Figure 20b). In this case, whenever it is detected that C is occluded, only the prediction step is used to update the probabilistic model of the robot position relative to C. A sample path is shown in Figure 21, with robot positions where occlusion occurs marked by black ”x”. The size of the occluding obstacle relative to the length of a motion step allows the robot to escape from the zone where occlusion takes place in two steps. Table 9 shows statistics over 100 trials. The results are quite similar to those obtained when only motion obstacles were present. However, the confirmation rate is marginally smaller.

8. EXPERIMENTS WITH REAL ROBOT

8.1. General setting

In this section we present five experiments carried out with a real robot. The goal of these experiments is to further assess our approach. All our experiments took place in the Robotics Laboratory at CIMAT. Figure 22 shows the scene of the experiments.

Our robot is a Pioneer P3-DX (a differential-drive robot) equipped with an USB camera as our visual sensor and a Sick-LMS200 range sensor. The robotic system and its software architecture are presented in more detail in Appendix B.



(a) Environment with an obstacle that produces both a motion obstruction and a visibility occlusion

(b) Visibility occlusion from the robot's perspective

Fig. 20. Experiment with both motion obstructions and visibility occlusions

Planning horizon	# of sensing locations	Path length	Planning time (ms)	% of confirmation
3	23.70	20.77	918.72	97

Table 9: Statistics over 100 runs, with both motion and visibility constraints and the true bottle as target

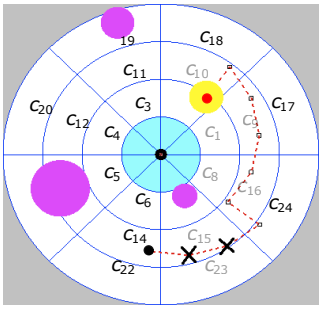


Fig. 21. Sample path in the presence of an obstacle creating occlusion constraints

The object to be identified is placed at the center of a rectangular support table (see center of Figure 22). The camera is used to take images that are fed into the target detector. The range sensor is used to keep the robot at an adequate distance from the support table to avoid collision with it and to maintain the candidate object within the field of view of the camera. The range sensor is also used to estimate the direction of the object C.

In the presented experiments we will vary two sets of parameters. Parameters in one set are related to the planner: planning horizon N and number of cells in the decomposition. Parameters in the other set are related to the environment and/or the target detector: illumination, scene background, textures, sizes, and colors of the targets, size of the sets of images used to train the detector and generate the observation models, number of cells used to train the detector.

In Experiments #5, #6 and #7, the planner uses the 24-cell decomposition shown in Figure 1. In Experiments #8 and #9 it uses a reduced version of this decomposition with only 16 cells, obtained by removing the outer ring of cells. For each target T, the detector D_T uses the same deformable part model algorithm as in Section 7. In Experiments #5 through #8

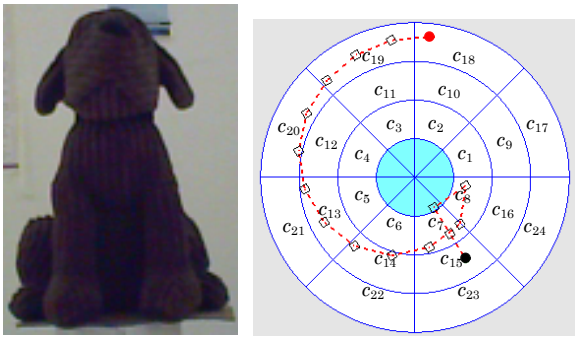
Fig. 22. Laboratory environment for Experiments #5 through #9

the detector is trained on a set of images taken from a single cell c_g of the decomposition. In Experiment #9 it is trained on a set of images taken from two adjacent cells c_{g1} and c_{g2}. In all cases, the confidence scores range over 6 values, i.e., $n = 6$ and the threshold $\hat{o}$ was set to o_6. The parameter λ was set to 0.5. The robot was initially located in cell c_{14} or c_{15}, and c_g, c_{g1} and c_{g2} were chosen far away from these cells. The execution horizon was set to $N_e = 40$.

In all experiments, the estimated direction of the center of the candidate object C is computed as follows. The laser scan of the range sensor intercepts the support of C. The two depth discontinuities corresponding to the two sides of this support are detected and the middle direction between them is the estimated direction.

As with the experiments in simulation, in all the statistics tables shown below, the average numbers of sensing locations, path lengths and planning times are computed only over the runs that achieve confirmation.

8.2. Experiment #5: A dark dog plush



(a) Dog target (b) Path generated with $N = 4$

(c) Initial view (d) View on detection

Fig. 23. Illustrations for Experiment #5

The main purpose of this experiment is to test the ability of our system to confirm the presence of a target in a real-world scene. The target is the dog plush shown in Figure 23a. It has uniform texture and dark color. Only 40 images were used to train D_T (instead of 200 in Experiments #1 through #4). We also took 40 images from each of the 24 cells of the decomposition to obtain D_T's observation model (instead of 200 in the previous experiments).

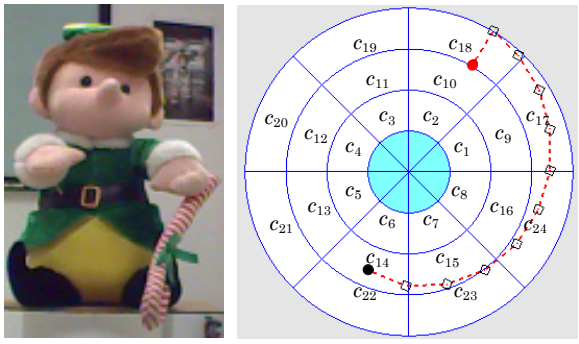
Planning horizon	# of sensing locations	Path length	Planning time (ms)	% of confirmation
4	21.0	8.677	214092.8	80

Table 10: Statistics for Experiment #5 (dog)

We experimented with planning horizons $N = 2, 3$, and 4. The best results were obtained with $N = 4$. Statistics with this horizon collected over 10 runs with random initial positions of the robot in cell c_{15} are shown in Table 10: average number of sensing locations, average path length, average total planning time and confirmation rate in %. Despite the fact that this target has few distinctive features and that the experiment was conducted under relatively poor illumination, the system managed to do so in 80% of the runs. Figure 23b plots the path followed by the robot in one of the runs. The initial position of the robot is shown with a black circle, its final position as a red (gray) circle, and the intermediate sensing locations as small squares. Figures 23c and 6d show, respectively, the initial image of the dog (then the candidate object) and its last image when it is conformed to be the target. In Extension 1 of this paper, a run of this experiment is shown at 2x speed.

The imperfect results obtained in this experiment led us to perform experiments with larger training sets of images and under diverse illumination conditions, as reported below.

8.3. Experiment #6: A multicolor elf plush



(a) Elf toy as target (b) Path generated with $N = 2$

(c) Initial view of candidate object C (d) View of object C when it is confirmed to be the target

Fig. 24. Illustrations for Experiment #6

This two main purposes of this experiment are to better analyze the impacts of the choice of the planning horizon N and of the number of images used to train D_T.

The target T is the multi-color elf plush shown in Figure 24a. Initially the detector was trained (like in Experiment #5) on 40 images taken from within cell $c_g = c_{10}$ and the observation model was built using 40 images from each of the

24 cells of the decomposition X. Using the SDP-based planner of Section 6, we generated motion strategies with three planning horizons $N = 2, 3$ and 4.

Planning horizon	# of sensing locations	Path length	Planning time (ms)	% of confirmation
2	14.00	6.685	78.52	100
3	15.00	6.713	1641.39	100
4	14.50	6.851	154880.6	80

Table 11: Statistics for Experiment #6 (elf plush)

Figure 24b plots the path followed by the robot in a run with $N = 2$. Figures 24c and 24d show, respectively, the initial image of the elf and its last image when it is confirmed to be the target.

Table 11 shows statistics collected for each value of N. Each line of the table was obtained over 10 runs with different initial positions of the robot within cell c_{14}. The numbers of sensing locations and the path lengths are approximately the same for all three values of N over the runs confirming the presence of the target. However, the main surprise comes from the fact that runs with $N = 4$ had a confirmation rate of only 80%, while the runs with $N = 2$ and $N = 3$ achieved a 100% rate. Although statistics over 10 runs may not be fully reliable, this result is counter-intuitive, as a planner with greater look-ahead computes policies taking more possible future states and observations into account. However, if models contains too much uncertainty, considering motion/sensing steps too far in the future may reduce information relevance and lead to poor decisions. Here, the detector D_T trained over only 40 images may result in a non-informative observation.

Planning horizon	# of sensing locations	Path length	Planning time (ms)	% of confirmation
4	12.00	5.732	119232.2	100

Table 12: Statistics for Experiment #6 (elf plush) with enhanced training of the detector

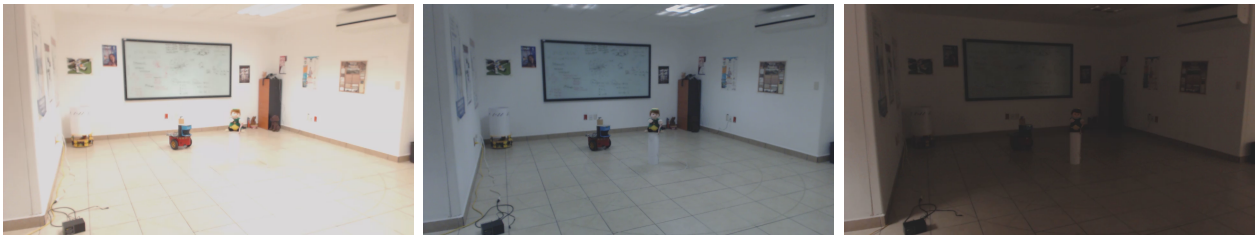
To test this hypothesis, we re-trained the detector D_T on a larger set of 75 images from cell c_{10}. Table 12 shows the new statistics for $N = 4$. The confirmation rate not only reaches 100%, but the average number of sensing locations and the average path length are also better.

The number of steps before confirmation is roughly the same as those reported above for Experiments #1, #2, and #4 and observed for other experiments. In Experiment #5 a slightly larger number of steps were needed to confirm the target, despite the combination of lack of distinctive features of the dog plush, poor illumination of the scene, and small size of the training set of images. This suggests that the resolution of the cell decomposition and the step size, which have been the same for these experiments, are key determinants of the number of steps. Hence, it may be possible to set a much smaller limit than the current one on the total number of steps before the planner decides that a candidate object C is not the target. This would avoid wasting too much time when the candidate object C is not the target, without impacting the reliability of the confirmation process.

8.4. Experiment #7: Changes in illumination

This experiment has been directly inspired by Experiment #5, which was carried out with reduced illumination. Its purpose is to analyze the impact of variations in illumination conditions between the training and the execution phases.

The target is again the elf plush used in Experiment #6 and shown in Figure 24a. We used the same trained detector and observation model as in Experiment #6. They were both obtained with the illumination conditions of Figure 25a (1650



(a) 1650 lux (b) 495 lux (c) 16 lux

Fig. 25. The same scene and target (elf plush) with three different illuminations (Experiment #7)

lux)[2]. Here, we perform two sets of additional runs with the reduced illumination conditions shown in Figure 25b (495 lux) and Figure 25c (16 lux).

Illumination conditions	# of sensing locations	Path length	Planning time (ms)	% of confirmation
1650 lux	15.00	6.713	1641.39	100
495 lux	18.44	8.125	2024.31	90
16 lux	14.33	7.029	1626.59	30

Table 13: Statistics for Experiment #7 (variations in illumination)

Table 13 shows the obtained statistics collected for each illumination over 10 runs with random initial positions of the robot in cell c_{14}, with planning horizon $N = 3$. The first row, which corresponds to runs performed with the same illumination as in the training phase, is the same as in Table 11. When illumination was reduced to 495 lux, the robot was still able to confirm the presence of the target in 9 runs out of 10. As expected, when illumination is reduced further the confirmation rate drops substantially. Nevertheless, with an illumination of only 16 lux the system achieves confirmation in 3 runs out of 10.

It is interesting to remark that when confirmation is achieved with 16-lux illumination, the average number of sensing locations and path length are similar in magnitude to those obtained in the 1650-lux and 495-lux scenarios. This suggests that with 16-lux illumination the subset of $c_g = c_{10}$ where the confidence score o_6 is obtainable is greatly reduced and may require finer motion steps of the robot to reach it. This hypothesis is confirmed in Figure 26. There, we show 40 robot positions sampled uniformly within cell c_{10}, and we attach two numbers to each position: the left number is the detection score o_i under 1650-lux and the right number is the score under 16-lux. Under 16-lux, the confidence score o_6 is obtainable only in a small small subset of c_{10}. With the current step size, if the robot does not reach this subset soon, it keeps moving back and forth around cell c_{10} without ever stopping into it.

Detection methods may exist that are more robust to variations in illumination conditions than the deformable part model algorithm used in our detector, or better suited in poorly illuminated scenes. If it is known in advance that the illumination conditions during execution runs are likely to be poor or can differ widely from those during training, such a method should then be used in our system.

8.5. Experiment #8: Woodcraft of a jaguar head

The main goal of this experiment is to test the performance of the system with fewer cells in the decomposition X when the target is smaller in size.

[2] The lux is a unit of luminous flux per unit area. In photometry, it measures the intensity of light that hits a surface.

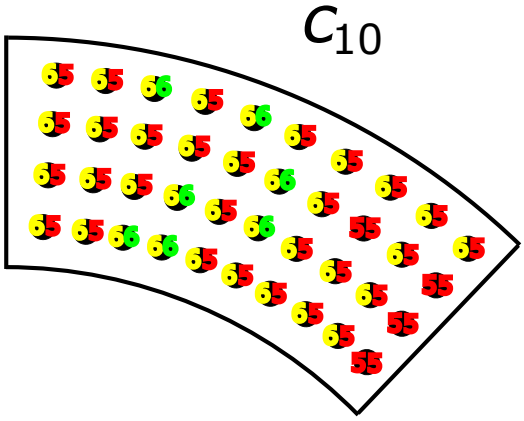


Fig. 26. At each position the left number is the score o_i under 1650-lux and the right number the score under 16-lux

	o_1	o_2	o_3	o_4	o_5	o_6
c_1	0.000	0.000	0.275	0.225	0.400	0.100
c_2	0.000	0.000	0.000	0.000	0.000	1.000
c_3	0.000	0.000	0.200	0.275	0.400	0.125
c_4	0.300	0.675	0.025	0.000	0.000	0.000
c_5	0.000	0.450	0.550	0.000	0.000	0.000
c_6	0.275	0.675	0.050	0.000	0.000	0.000
c_7	0.025	0.275	0.025	0.000	0.000	0.000
c_8	0.575	0.400	0.025	0.000	0.000	0.000
c_9	0.100	0.150	0.275	0.300	0.175	0.000
c_{10}	0.000	0.000	0.000	0.250	0.550	0.200
c_{11}	0.100	0.175	0.200	0.325	0.200	0.000
c_{12}	0.850	0.150	0.000	0.000	0.000	0.000
c_{13}	0.600	0.375	0.025	0.000	0.000	0.000
c_{14}	0.900	0.100	0.000	0.000	0.000	0.000
c_{15}	0.675	0.250	0.075	0.000	0.000	0.000
c_{16}	0.825	0.175	0.000	0.000	0.000	0.000

Table 14: Observation model $P(y = o_j|x = c_i)$ in Experiment #8 (jaguar's head)

The target is the colorful woodcraft of a jaguar head shown in Figure 27a. It is smaller in overall size than the dog and elf used in previous experiments. For this reason we reduced the number of cells in the decomposition to 16 by removing the outer ring (8 cells) from the 24-cell decomposition shown in Figure 1.

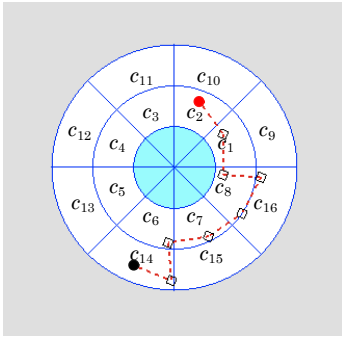
The detector was trained on 40 sample images using c_2 (rather than c_{10}) as c_g. The observation model was generated using 40 images from each of the 16 cells. The result is shown in Table 14.

Planning horizon	# of sensing locations	Path length	Planning time (ms)	% of confirmation
2	9.00	4.311	26.585	100

Table 15: Statistics for Experiment #8 (jaguar)

The smaller number of cells allows for a smaller planning horizon N, which we set to 2. Table 15 presents statistics collected over 10 runs with different initial positions of the robot within cell c_{14}. The confirmation rate is 100%. The average number of steps and path length are slightly lower than in previous experiments. This is likely due to the smaller number of cells in the decomposition and the choice of c_2 as c_g. Such choices were induced by the target's smaller size. Figure 27b plots the path followed by the robot in one of the runs. Figures 27c and 27d show, respectively, the initial image of the jaguar's head and its final image taken when it is confirmed to be the target.

(a) Woodcraft of a jaguar head as the target



(b) Path generated with $N = 2$

(c) Initial view of the jaguar

(d) View of the jaguar when it is confirmed to be the target

Fig. 27. Illustrations for Experiment #8

This result led us to perform the same experiment with the larger elf plush used in Experiment #6, i.e., with the same 16-cell decomposition as above and c_2 as c_g. Table 16 gives the statistics collected over 10 runs with $N = 3$. Comparison with the second row of Table 11 indicate a slightly lower confirmation rate (90% instead of 100%) and 20% smaller number of steps and path length. In the run that did not end up in target confirmation, the robot reached c_2 but kept moving towards the elf plush to positions where the top part of the target was out of camera field of view. At those positions the confidence scores returned by the detector were no greater than o_5. Because of its smaller size, this never happened with the jaguar head.

This comparison indicates that the size of the decomposition should be adapted to the size of the target. With targets larger than those used our experiments, one might have to use decompositions with 2 or 3 rings with larger inner and outer circles. In some cases, e.g., large elongated objects, non-circular decompositions may be more suitable. In Extension 2 of this paper, a run of this experiment is shown at 1x speed.

Planning horizon	# of sensing locations	Path length	Planning time (ms)	% of confirmation
3	11.55	5.423	699.26	90

Table 16: Statistics for Experiment #8 (elf plush with 2-ring 16-cell decomposition)

8.6. Experiment #9: Detector trained over two adjacent cells

The purpose of this experiment is to test our system when the detector D_T is trained over the union of two adjacent cells c_{g1} and c_{g2}, instead of a single cell. Training the detector over multiple cells can provide the robot with a broader range of viewpoints from which it can confirm the presence of the target.

	o_1	o_2	o_3	o_4	o_5	o_6
c_1	0.000	0.000	0.000	0.000	0.625	0.375
c_2	0.000	0.000	0.000	0.000	0.000	1.000
c_3	0.000	0.000	0.000	0.000	0.000	1.000
c_4	0.000	0.000	0.025	0.040	0.500	0.075
c_5	0.025	0.550	0.400	0.025	0.000	0.000
c_6	0.575	0.375	0.050	0.000	0.000	0.000
c_7	0.000	0.075	0.750	0.175	0.000	0.000
c_8	0.000	0.000	0.000	0.550	0.450	0.000
c_9	0.000	0.000	0.050	0.200	0.750	0.000
c_{10}	0.000	0.000	0.025	0.250	0.550	0.175
c_{11}	0.000	0.000	0.000	0.075	0.400	0.525
c_{12}	0.025	0.025	0.175	0.375	0.400	0.000
c_{13}	0.725	0.175	0.100	0.000	0.000	0.000
c_{14}	0.950	0.050	0.000	0.000	0.000	0.000
c_{15}	0.525	0.300	0.175	0.000	0.000	0.000
c_{16}	0.050	0.100	0.150	0.500	0.200	0.000

Table 17: Observation model $P(y = o_j|x = c_i)$ in Experiment #9 (jaguar's head)

The target is again the jaguar's head shown in Figure 27a, with the same 16-cell decomposition as in Experiment #8. The detector D_T was trained on 80 images, 40 from within $c_{g1} = c_2$ and 40 from within $c_{g2} = c_3$. The observation model shown in Table 17 was built using 40 images from each of the 16 cells. The detector has probability 1 to return the maximal score o_6 in both cells c_2 and c_3. It also has probability 0.525 to return score o_6 in cell c_{11}. Therefore, since λ is set to 0.5 (see Section 8.1) the region where the robot may achieve confirmation is more than twice larger than in Experiment #8.

Planning horizon	# of sensing locations	Path length	Planning time (ms)	% of confirmation
2	7.5	3.335	14.92	100

Table 18: Statistics for Experiment #9 (jaguar multiple cells training)

Table 18 shows statistics collected over 10 runs with different initial positions of the robot within cell c_{14} and planning horizon $N = 2$. The confirmation rate is 100%. Since the robot can confirm the presence of the target from a much larger endgame region and part of this region is closer to cell c_{14} than cell c_2, it moves less than in Experiment #8. Comparison of Tables 15 and 18 indicates that the average number of sensing locations decreased by 16% whereas the average path length decreased by 22%.

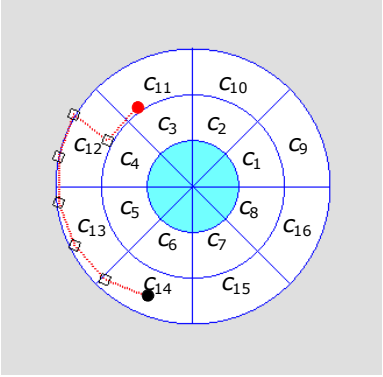


Fig. 28. Path generated with $N = 2$ with D_T trained in $c_{g1} = c_2$ and $c_{g2} = c_3$

Training the detector over a larger number of cells may speed up confirmation in some cases, but may not always be suitable. If the diversity of viewpoints increases, the detector may require more images from each cell to be able to

return high scores from each one. Moreover, if several objects have the same appearance from some viewpoints (as in Experiment #2 with two similar bottles), then incorrect confirmation of the target's presence will become more likely. Our ability to train the detector in cells where distinctive features of the target are visible would be reduced or even lost.

The pertinence of using a static detector trained from multiple viewpoints or a mobile robot using multiple detectors, or a mobile robot using a single detector depends on the appearance of the target. If the target has only one small feature that differentiate it from similar objects, then a single detector works well and the mobility of the sensor is essential. If the target has several distinctive features visible from across several viewpoints, then multiple detectors may be preferable. In addition, if at least one distinctive feature is visible from any viewpoint (a rare case in practice), then a static detector would work. In general, the mobility of the sensor makes it possible to reduce the number of viewpoints to train the detector.

Remark: As a final note regarding all the experiments selecting the ideal value of the threshold λ for a given target is not trivial as it might vary significantly depending on the target searched. The observation model of the target gives an upper bound on the probability of getting a given detection score greater than or equal to $\hat{o}$, for any selected value of $\hat{o}$, hence an upper-bound on the value of λ. In all our experiments we set $\hat{o}$ to the highest possible score of the target detector, but this might not always be the best choice, especially if the highest score is rarely achieved.

9. CONCLUSION AND FUTURE WORK

We have presented and tested a new approach allowing a mobile robot equipped with visual sensors to confirm that a candidate object C is a given target T using an imperfect target detector. Our main contribution is a POMDP formulation of the problem that mixes localization of the robot relative to C and confirmation that C is the target T. To confirm the target's presence while avoiding false positives, the robot must achieve a twofold condition: (1) it must reach a position where the target detector has high probability to return a high confidence score and (2) the detector must actually return a high confidence score on the image taken at this position.

Motion strategies are computed as policies with given horizons using Stochastic Dynamic Programming (SDP) with Imperfect State Information. Computing exact solutions in SDP with Imperfect State Information does not scale well with the planning horizon but our experiments show that space decompositions with relatively few cells suffice. Such decompositions do not require very long horizons. Moreover, we have introduced a gain function g_F that prevents the planner from being trapped into local maxima when planning horizons are set too short. Our method is a target confirmation method, not a general-purpose object recognition method. Hence, unlike most object recognition systems, it only takes the observation model of the target as input.

We have implemented our method and tested it both in simulation and with a real robot. In this paper, we have presented results obtained in nine different experiments (four in simulation and five with a real robot). Some of these experiments include several significant variants aimed at further understanding some results or validating certain hypotheses. Overall, these experiments demonstrate that our system is efficient and reliable under a broad range of situations. In particular, they show that it tolerates well similar-looking targets and candidate objects, variations in scene illumination, non-uniform scene backgrounds, targets with various colors, shapes, and textures, relatively small training sets of images, and large errors in the estimated direction of the candidate object's center. They also indicate that the sizes (24 and 16 cells) of our cell decompositions are appropriate for the small targets we have considered, that planning horizons 2 and 3 work well with these decompositions and the additional gain function g_F, and that the size of the decomposition and the length of a motion step are key determinants of the number of moves needed to achieve confirmation. A larger planning horizon ($N = 4$) significantly increases planning times, without affecting much the numbers of steps and the success rates (when the additional gain function g_F is used). Moreover, there is evidence that an increase of the planning horizon may require larger training sets of images.

Our goals for future research include the following:

- **Multiple detectors:** Training one detector over k cells, as we did in Experiment #9 (for $k = 2$ adjacent cells), is different from training k detectors, each over one distinct cell. Using multiple such detectors could have significant advantages, especially when the target has distinctive features visible from very different viewpoints. Then the presence of the target could be confirmed from a broader range of robot positions, hence more quickly. Moreover, false positives would be less likely than with a single detector trained over multiple cells (an issue discussed at the end of Section 8.6). However, efficiently managing multiple detectors and their responses could raise some non-trivial issues. The true benefits of using multiple detectors would also have to be demonstrated experimentally.
- **Bigger targets:** The type of practical application we have in mind for our method is the retrieval and localization of relatively small objects in typical indoor environments (home or office). So, In all our experiments, the target objects have been relatively small and, except for the plastic cat, compact. For such targets the experiments show that the circular 24- and 16-cell decompositions are appropriate. For larger targets, especially elongated and/or irregularly shaped ones, non-circular decompositions that conform better to the target shapes, e.g., elliptical or rectangular decompositions, might be preferable. Selecting the size of the cells (hence, their number), as well as appropriate motion commands and transition models could then be challenging. If cells are too small, long planning horizons could be needed, but planning times could then be prohibitive; if cells are too large, the observation models might not be informative enough to reliably determine the target's presence or absence.
- **Targets with arbitrary orientation in 3D space:** In our current work we assume that a target has a single pose on a horizontal surface. In the future, it could be useful to remove this assumption and let the target have any orientation in 3D space. This would be useful if the target has several equilibrium position on a plane or if it may lie on a non-flat-non-horizontal surface. One possible approach could be to train several detectors for different orientations and then to estimate the one that provides the best match with the candidate object.
- **Visual occlusions:** In Section 7.6 we have described an extension of our system to deal with motion and occluding obstacles. However, this extension is still preliminary. Another approach (that could be combined with our current method) would be to train detectors on small subsets of the target containing distinctive features, since such subsets are less likely to be occluded by obstacles than the entire target.
- **Camera with more degrees of freedom:** Many targets could be more reliably identified if the camera was moving in 3D space rather than a horizontal plane. This can be achieved by attaching the camera at the end-effector of a manipulator arm mounted on the mobile robot. However, as in the item above, this will also require new cell decompositions, motion commands, and transition models. Problems similar to those mentioned above will need to be addressed. The robot would also have to decide how to perform some motions, either using the degrees of freedom of its mobile base, or those of its manipulator arm, or both. Cameras with more degrees of freedom could also help dealing with targets with arbitrary orientations in 3D space.

In addition to these research goals, we are also interested in a more practical goal: building a robot that can routinely search for user-specified targets in an indoor environment.

References

[Aloimonos et al. 1988] J. Aloimonos, I. Weiss and A. Bandyopadhyay, "Active vision", *Int. J. Computer Vision*, vol. 1, no. 4, pp. 333-356, 1988.

[Atanasov et al. 2014] N. Atanasov, B. Sankaran, J. Le Ny, G. Pappas and K. Daniilidis, "Nonmyopic view planning for active object classification and pose estimation", *IEEE Trans. Robotics*, vol. 30, no. 5, pp. 1078-1090, 2014.

[Bajcsy 1988] R. Bajcsy, "Active perception", *Proc. IEEE*, vol. 76, no. 8, pp. 966-1005, 1988.

[Bakhtari et al. 2009] A. Bakhtari, M. Mackay and B. Benhabib, "Active-vision for the autonomous surveillance of dynamic, multi-object environments", *Journal of Intelligent and Robotic Systems*, vol. 54, no. 4, pp. 567-593, 2009.

[Barreto et el. 2010] J.P. Barreto, L. Perdigoto, R. Caseiro and H. Araujo, "Active stereo tracking of N $\leq$ 3 targets using line scan cameras", *IEEE Trans. Robotics*, vol. 26, no. 3, pp. 442-457, 2010.

[Becerra et al. 2014] I. Becerra, L. Valentin, R. Murrieta-Cid and J.-C. Latombe, "Appearance-based Motion Strategies for Object Detection", *Proc. of IEEE Int. Conf. on Robotics and Automation*, 2014, Hong Kong, China, pp. 6455-6461.

[Bertsekas 2000] D. Bertsekas, *Dynamic Programming and Optimal Control*, Athena Scientific, 2000.

[Browatzki et al. 2012] B. Browatzki, V. Tikhanoff, G. Metta, H. Bulthoff and C. Wallraven, "Active object recognition on a humanoid robot", *IEEE Int. Conf. on Robotics and Automation*, 2012, St. Paul, Minnesota, USA, pp. 2021-2028.

[Candido and Hutchinson 2011] S. Candido and S. Hutchinson, "Minimum uncertainty robot navigation using information-guided POMDP planning", *Proc. of IEEE Int. Conf. on Robotics and Automation*, 2011, Shanghai, China, pp. 6102-6108.

[Chen et al. 2011] S. Chen, Y. Li and N.M. kwok, "Active vision in robotic systems: A survey of recent developments", *Int. Journal of Robotics Research*, vol. 30 no. 11, pp. 1343-1377, 2011.

[Chung at al. 2011] T. Chung, G. Hollinger and V. Isler, "Search and pursuit-evasion in mobile robotics", *Autonomous Robots*, vol. 31, no. 3, pp. 299-316, 2011.

[Eidenberg and Scharinger 2010] R. Eidenberger and J. Scharinger, "Active perception and scene modeling by planning with probabilistic 6D object poses", *IEEE Int. Conf. on Intelligent Robots and Systems*, 2010, Taipei, Taiwan, pp. 1036-1043.

[Espinoza et al. 2011] J. Espinoza, A. Sarmiento, R. Murrieta-Cid and S. Hutchinson,"A motion planning strategy for finding an object with a mobile manipulator in 3-D environments", *Advanced Robotics*, vol. 25, no. 13-14, pp. 1627-1650, 2011.

[Felzenzwalb et al. 2008] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model", *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, Anchorage, Alaska, USA, pp. 1-8.

[Felzenzwalb et al. 2010] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627-1645, 2010.

[Fintrop and Jensfelt 2008] S. Frintrop and P. Jensfelt, "Attentional landmarks and active gaze control for visual SLAM", *IEEE Trans. Robotics*, vol. 24 no. 5,, pp. 1054-1065, 2008.

[Guibas et al. 1999] L. Guibas, J. Latombe, S. Lavalle, D. Lin and R. Motwani, "Visibility-based pursuit-evasion in a polygonal environment", *Int. Journal of Computational Geometry and Applications*, vol.9, no. 5, pp. 471-494, 1999.

[Grauman and Darrel 2005] K. Grauman and T. Darrell, "The pyramid match kernel: Discriminative classification with sets of image features", *Proc. of IEEE Int. Conference on Computer Vision (ICCV)*, 2005, Beijing, China, pp. 1458-1465.

[Isler at al. 2005] V. Isler, S. Kannan, and S. Khanna, "Randomized pursuit-evasion in a polygonal environment", *IEEE Trans. Robotics*, vol. 21, no. 5, pp. 875-884, 2005.

[Jang et al. 2008] H.Y. Jang, H. Moradi, P. Le Minh, S. Lee and J. Han, "Visibility-based spatial reasoning for object manipulation in cluttered environments", *Computer-Aided Design*,, vol. 40, no. 4, pp. 422-438, 2008.

[Kaelbling et al. 1998] L.P. Kaelbling, M.L. Littman and A.R. Casandra, "Planning and acting in partially observable stochastic domains", *Artificial Intelligence*, vol. 101, no. 1-2, pp. 99-134, 1998.

[Kaess and Dellaert 2010] M. Kaess and F. Dellaert, "Probabilistic structure matching for visual SLAM with a multi-camera rig", *Computer Vision and Image Understanding*, vol. 114, no. 2, pp. 286-296, 2010.

[Krotkov and Bajcsy 1993] E. Krotkov and R. Bajcsy, "Active vision for reliable ranging: Cooperating focus, stereo, and vergence", *Int. J. Comput. Vis.*, vol. 11, no. 2, pp. 187-203, 1993.

[Laporte and Arbel 2006] C. Laporte and T. Arbel, "Efficient discriminant viewpoint selection for active Bayesian recognition", *Int. Journal of Computer Vision*, vol. 68, no. 3, pp. 1405-1573, 2006.

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

[Lee et al. 2002] J.-H. Lee, S.-M. Park and K.-Y. Chwa, "Simple algorithms for searching a polygon with flashlights", *Information Processing Letters*, vol. 81, no. 5, pp. 265-270, 2002.

[Lowe 2004] D.G. Lowe, "Distinctive image features from scale-invariant keypoints", *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, 2004.

[Meger et al. 2010] D. Meger, A. Gupta and J. Little, "Viewpoint detection models for sequential embodied object category recognition", *IEEE Int. Conf. on Robotics and Automation*, pp. 5055-5061, Anchorage, AK, 2010.

[Motai and Kosaka 2008] Y. Motai and A. Kosaka, "Hand-eye calibration applied to viewpoint selection for robotic vision", *Transactions on Industrial Electronics*, vol. 55, no. 10, pp. 3731-3741, 2008.

[Park et al. 2011] S.-M. Park, J.-H. Lee and K.-Y. Chwa, "Searching a room by two guards", *Int. J. of Computational Geometry & Applications*, vol. 12, no. 4, pp. 339-352, 2011.

[Pito 1999] R. Pito, "A solution to the next best view problem for automated surface acquisition", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 10, pp. 1016-1030, 1999.

[Sarmiento et al. 2003] A. Sarmiento, R. Murrieta-Cid, and S. Hutchinson, "An efficient strategy for rapidly finding an object in a polygonal world", *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2003, Las Vegas, Nevada, USA, pp. 1153-1158.

[Schroeter et al. 2009] C. Schroeter, M. Hoechemer, S. Mueller and H.M. Gross, "Autonomous robot cameraman - observation pose optimization for a mobile service robot in indoor living space", *Proc. of IEEE Int. Conf. on Robotics and Automation*, 2009, Kobe, Japan, pp. 424-429.

[Tovar and LaValle 2008] B. Tovar and S.M. LaValle, "Visibility-based Pursuit Evasion with Bounded Speed", *International Journal of Robotics Research*, vol. 27. no. 11-12, pp. 1350-1360, 2008.

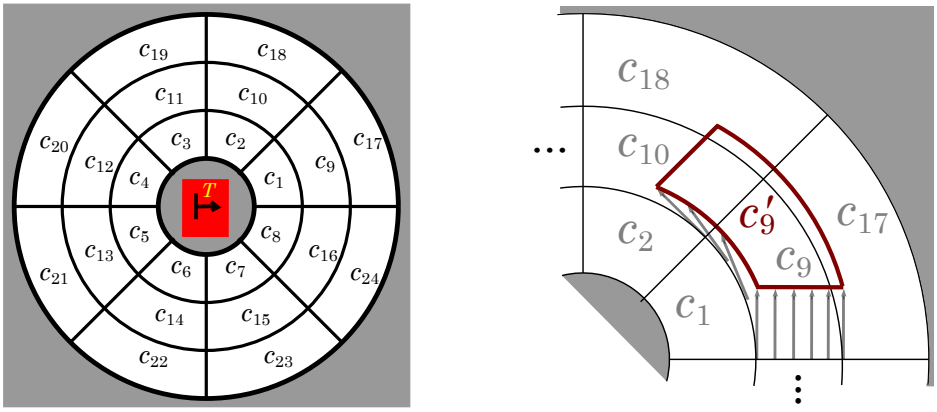
[Thrun et al. 2005] S. Thrun, W. Burgard and D. Fox, *Probabilistic Robotics*, MIT Press; 2005.

[Tsotsos and Shubina 2007] J.K. Tsotsos and K. Shubina, "Attention and visual search: Active robotic vision systems that search", *Proc. of Int. Conf. on Computer Vision Systems*, 2007, Washington, DC, USA, pp. 539.

[Ye and Tsotsos 1999] Y. Ye and J.K. Tsotsos, "Sensor planning for 3D object search", *Computer Vision and Image Understanding*, vol. 73, no. 2, pp. 145-168, 1999.

[Zingaretti and Frontoni 2006] P. Zingaretti and E. Frontoni, "Appearance based robotics", *IEEE Robotics and Automation Magazine*, vol. 13, no. 1, pp. 59-68, 2006.

Appendix A COMPUTATION OF TRANSITION PROBABILITIES



(a) Cell decomposition of the space around T

(b) $c'_9 = Tr(x_{t-1} = c_9, u_{t-1} = right)$

Fig. 29.

We now present the analytical calculation of the transition probabilities $P(x_t = c_j | x_{t-1} = c_i, u_{t-1})$ for the motion commands *forward*, *backward*, *left*, and *right* available to our robot. Recall from Section 5 that the length (hereafter denoted by S) of each motion step is set to a value slightly greater than the distance between two concentric circles of the

decomposition X. However, the motion step may be truncated by the range sensor if the robot gets too close or too far away from the candidate object.

Let us start with the case where $u_{t-1} = forward$. Assume that the command *forward* is applied to cell $x_{t-1} = c_i$, where c_i lies in the middle or outer layer of the decomposition X (hence, $i \notin \{1,...,8\}$). The neighboring cell of c_i in the forward radial direction of the decomposition X is cell c_{i-8}. Based on the definition of the transition probabilities given in Section 5, we have $P(x_t = c_{i-8}|x_{t-1} = c_i, forward) = 1$ since the transform $Tr(c_i, forward)$ is fully contained in c_{i-8}. Of course, we then have $P(x_t = c_j|x_{t-1} = c_i, forward) = 0, \forall c_j \neq c_{i-8}$. If the robot is in c_i, where $i \in \{1,...,8\}$, then the motion command *forward* will leave the robot in the same cell c_i thanks to the range sensor that prevents the robot from getting too close to the candidate object; so, in that case, $P(x_t = c_i|x_{t-1} = c_i, forward) = 1$.

The command $u_{t-1} = backward$ can be treated in a similar way. If $i \notin \{17,...,24\}$ then the neighboring cell of c_i in the backward radial direction of the decomposition X is c_{i+8}. We have $P(x_t = c_{i+8}|x_{t-1} = c_i, backward) = 1$. If $i \in \{17,...,24\}$ then $P(x_t = c_i|x_{t-1} = c_i, backward) = 1$.

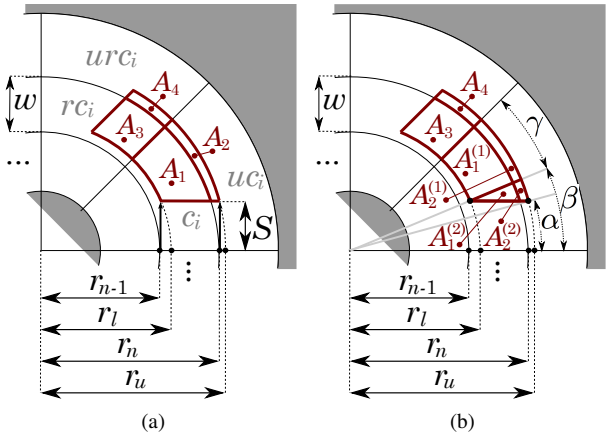


Fig. 30. Notations for calculating $P(x_t = c_j\,|\,x_{t-1} = c_i, u_{t-1} = right)$ for any cell c_i in the inner or middle layer of the decomposition X

The cases where $u_{t-1} = right$ and $left$ are more complicated. Let us consider the case where $u_{t-1} = right$. (The case where $u_{t-1} = left$ can be treated in a similar way.) Consider the most general case where cell c_i lies in the inner or middle layer of the decomposition X, i.e., $i \in \{1,...,16\}$. Then the transform $Tr(x_{t-1} = c_i, u_{t-1} = right)$ of c_i by *right*, shown in red in Figure 30a, overlaps four cells denoted c_i, uc_i, rc_i, and urc_i. (*uc* stands for "upper neighboring cell", *rc* for "right neighboring cell", and *urc* for "upper right neighboring cell".) The four overlap regions are designated by A_1, A_2, A_3, and A_4. We further decompose A_1 and A_2 into simpler regions $A_1^{(1)}$, $A_1^{(2)}$, $A_2^{(1)}$, and $A_2^{(2)}$, as shown in Figure 30b. Finally, let:

- r_{n-1} and r_n denote the radii of the inner and outer boundaries of c_i,
- r_l and r_u denote the radii of the inner and outer boundary of $Tr(x_{t-1} = c_i, u_{t-1} = right)$.

The four non-zero transition probabilities can now be expressed as follows as ratios of region areas:

$$P(x_t = c_i\,|\,x_{t-1} = c_i, u_{t-1} = right) = \frac{A_1}{A_1+A_2+A_3+A_4}, \tag{7}$$

$$P(x_t = uc_i\,|\,x_{t-1} = c_i, u_{t-1} = right) = \frac{A_2}{A_1+A_2+A_3+A_4}, \tag{8}$$

$$P(x_t = rc_i\,|\,x_{t-1} = c_i, u_{t-1} = right) = \frac{A_3}{A_1+A_2+A_3+A_4}, \tag{9}$$

$$P(x_t = urc_i\,|\,x_{t-1} = c_i, u_{t-1} = right) = \frac{A_4}{A_1+A_2+A_3+A_4}, \tag{10}$$

where the region areas are derived using trigonometric arguments:

$$A_1 = A_1^{(1)} + A_1^{(2)}, \qquad A_1^{(1)} = \frac{\gamma}{2}(r_n^2 - r_l^2),$$
$$A_1^{(2)} = \frac{[\sqrt{r_n^2 - S^2} - r_{n-1}][r_n \sin(\beta) - S]}{2} + \frac{r_n^2[\beta - \alpha - \sin(\beta - \alpha)]}{2},$$
$$A_2 = A_2^{(1)} + A_2^{(2)}, \qquad A_2^{(1)} = \frac{\gamma}{2}(r_u^2 - r_n^2),$$
$$A_2^{(2)} = \frac{w[r_u \sin(\beta) - S]}{2} + \frac{r_u^2[\beta - \alpha - \sin(\beta - \alpha)]}{2} - A_1^{(2)},$$
$$A_3 = \frac{\beta}{2}(r_n^2 - r_l^2) - A_1^{(2)}, \qquad A_4 = \frac{\beta}{2}(r_u^2 - r_n^2) - A_2^{(2)},$$
$$r_l = \sqrt{r_{n-1}^2 + S^2}, \qquad r_u = \sqrt{r_n^2 + S^2}.$$

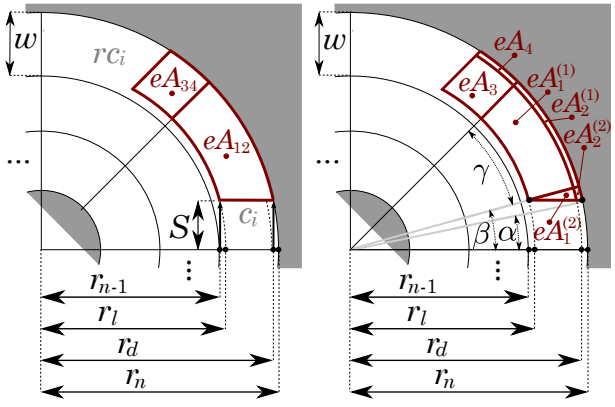


Fig. 31. Notations for calculating $P(x_t = c_j \,|\, x_{t-1} = c_i, u_{t-1} = right)$ for any cell c_i in the outer layer of the decomposition X

If cell c_i lies in the outer layer of the decomposition, i.e., if $i \in \{17, ..., 24\}$, then the range sensor prevents the robot from getting too far away from the candidate object by truncating some motion steps. In that case, there are no upper cells uc_i and urc_i (see Figure 31a). So, the transform $Tr(x_{t-1} = c_i, u_{t-1} = right)$ is now completely contained in c_i and rc_i. The two overlap regions are denoted by eA_{12} and eA_{34}. In Figure 31b) we introduce the radius r_d, which decomposes c_i into two subsets $s1(c_i)$ and $s2(c_i)$: in $s1(c_i)$ motion steps of full length S can be applied, whereas in $s2(c_i)$ motion steps are truncated by the range sensor. We compute the transition probabilities for $s1(c_i)$ in the same way as for a full cell contained in the inner or middle layer of the decomposition, just taking into account that here the transform $Tr(s1(c_i), right)$ consists of only two regions, eA_{12} and eA_{34}, instead of four. To compute the transition probabilities for $s2(c_i)$, we decompose this region further into two subsets: eA_c, which contains all the points that stay in c_i, and eA_{lc}, which contains all the points that moves to rc_i. Finally, the two non-zero transition probabilities for the entire cell c_i are calculated as weighted sums of the fractions of $s1(c_i)$ and $s2(c_i)$ that stay in c_i or move to rc_i. We get:

$$P(x_t = c_i \,|\, x_{t-1} = c_i, u_{t-1} = right) =$$
$$\left[\frac{eA_{12}}{eA_{12} + eA_{34}}\right]\left[\frac{r_d^2 - r_{n-1}^2}{r_n^2 - r_{n-1}^2}\right] + \left[\frac{eA_c}{eA_c + eA_{lc}}\right]\left[\frac{r_n^2 - r_d^2}{r_n^2 - r_{n-1}^2}\right], \tag{11}$$
$$P(x_t = rc_i \,|\, x_{t-1} = c_i, u_{t-1} = right) =$$
$$\left[\frac{eA_{34}}{eA_{12} + eA_{34}}\right]\left[\frac{r_d^2 - r_{n-1}^2}{r_n^2 - r_{n-1}^2}\right] + \left[\frac{eA_{lc}}{eA_c + eA_{lc}}\right]\left[\frac{r_n^2 - r_d^2}{r_n^2 - r_{n-1}^2}\right], \tag{12}$$

where the region areas are computed as follows:

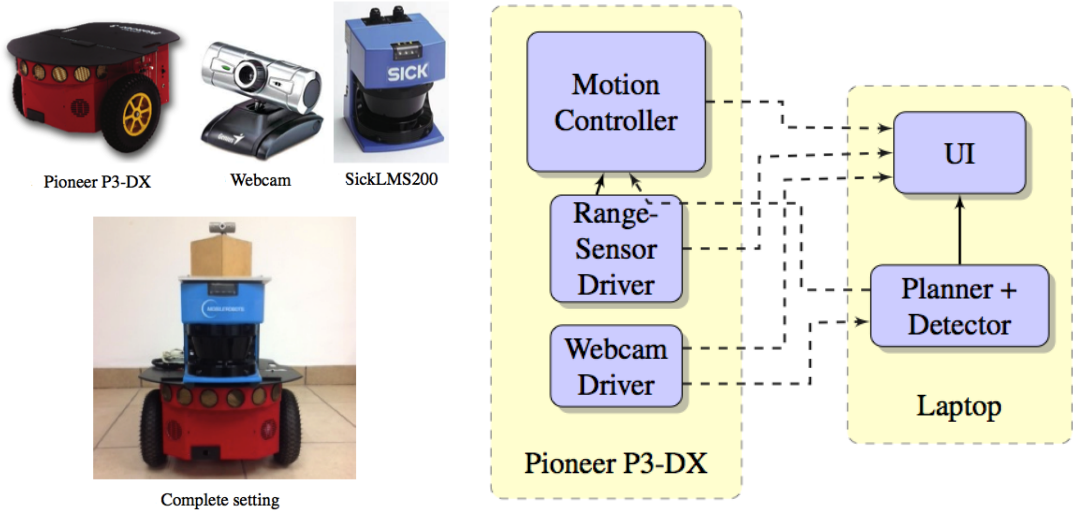
$$eA_{12} = eA_1^{(1)} + eA_2^{(1)} + eA_1^{(2)} + eA_2^{(2)},$$
$$eA_1^{(1)} = \frac{\gamma}{2}(r_d^2 - r_l^2), \qquad eA_2^{(1)} = \frac{\gamma}{2}(r_n^2 - r_d^2),$$
$$eA_1^{(2)} = \frac{[\sqrt{r_d^2 - S^2} - r_{n-1}][r_d \sin(\beta) - S]}{2} + \frac{r_d^2[\beta - \alpha - \sin(\beta - \alpha)]}{2},$$
$$eA_2^{(2)} = \frac{w[r_n \sin(\beta) - S]}{2} + \frac{r_n^2[\beta - \alpha - \sin(\beta - \alpha)]}{2} - eA_1^{(2)},$$
$$eA_{34} = eA_3 + eA_4,$$
$$eA_3 = \frac{\beta}{2}(r_d^2 - r_l^2) - eA_1^{(2)}, \qquad eA_4 = \frac{\beta}{2}(r_n^2 - r_d^2) - eA_2^{(2)},$$
$$eA_{lc} = \int_{r_d \cos\alpha}^{r_n} \sqrt{r_n x - x^2}\, \delta x - \left(\frac{r_d^2 \alpha}{2} - \frac{r_d^2 \sin 2\alpha}{4}\right),$$
$$eA_c = \frac{(\gamma + \beta)}{2}(r_n^2 - r_d^2) - A_{lc}, \qquad r_d = r_n \cos\alpha.$$

The transition probabilities $P(x_t = c_j \,|\, x_{t-1} = c_i, u_{t-1} = right)$ can be obtained in a similar way.

Appendix B HARDWARE AND SOFTWARE SETUP

In this appendix we briefly present the hardware components and the software architecture of the robotic system that we have implemented to perform the experiments described in Section 8.

The mobile robot is a Pioneer P3-DX with differential-drive wheels. It is equipped with an on-board 400 MHz computer. A usb webcam and a laser range sensor are mounted on the robot as depicted in Figure 32a. The webcam has a resolution of 640x480 pixels; it is used to feed the detector D_T with pictures of the candidate object C. The range sensor is a Sick-LMS200 with 10m range, 180dg scanning angle, and 1dg angular resolution; it is used to estimate the direction of the center of the candidate object C and to prevent the robot from going too close or too far away from C.



(a) Hardware (b) Software architecture

Fig. 32. Complete robotic system

Most of the system software is implemented using the ROS framework and programmed in C++. Due to the restricted computing power available on the Pioneer P3-DX, only some modules run on it. The other modules run on a laptop with an Intel Core-i7 2.20 GHz processor and 16 GB of RAM. The ROS framework is used to implement the communication protocols between the P3-DX and the laptop.

The main software modules running on the P3-DX are the motion controller, the webcam driver, and the range-sensor driver. The main modules running on the laptop are the motion planner, the target detector D_T, and the user interface (UI). The UI provides diverse information to the user, in particular the successive pictures taken by the webcam, the range-sensor measurements, and estimates of the current robot's position in the decomposition X. The motion planner applies the detector D_T to the images sent from the robot, computes motion policies using SDP, and sends motion commands back to the robot. Figure 32b summarizes this software architecture.

Appendix C Index to Multimedia Extensions

The multimedia extensions to this article are at:
http://www.ijrr.org.

Extension	Type	Description
1	Video	A run of the Experiment #5 at 2x speed
2	Video	A run of the Experiment #8 at 1x speed

Table 19: Table of Multimedia Extensions