Autonomous Robots 19, 67–87, 2005
© 2005 Springer Science + Business Media, Inc. Manufactured in The Netherlands.

A Navigation System for Assistant Robots Using Visually Augmented POMDPs

MARÍA ELENA LÓPEZ, LUIS MIGUEL BERGASA, RAFAEL BAREA
AND MARÍA SOLEDAD ESCUDERO
Electronics Department, University of Alcalá, Campus Universitario s/n, 28805 Alcalá de Henares, Madrid, Spain
elena@depeca.uah.es
bergasa@depeca.uah.es
barea@depeca.uah.es
marisol@depeca.uah.es

Abstract. Assistant robots have received special attention from the research community in the last years. One of the main applications of these robots is to perform care tasks in indoor environments such as houses, nursing homes or hospitals, and therefore they need to be able to navigate robustly for long periods of time. This paper focuses on the navigation system of SIRA, a robotic assistant for elderly and/or blind people based on a Partially Observable Markov Decision Process (POMDP) to global localize the robot and to direct its goal-oriented actions. The main novel feature of our approach is that it combines sonar and visual information in a natural way to produce state transitions and observations in the framework of Markov Decision Processes. Besides this multisensorial fusion, a two-level layered planning architecture that combines several planning objectives (such as guiding to a goal room and reducing locational uncertainty) improves the robustness of the navigation system, as it's shown in our experiments with SIRA navigating corridors.

Keywords: probabilistic navigation, Partially Observable Markov Decision Processes, multisensorial fusion, planning under uncertainty, assistant robots

1. Introduction

In the last years, the number of elderly in need of care is increasing dramatically. In the European Union, it is estimated that 10–15% of the total population is over 60 years old. The society needs to find new technologies and alternative ways of providing care to this sector of the population, where the need of personal assistance is larger than in any other age group. Aware of this necessity, nowadays there are several projects and research groups working in the development of assistant robots, such as "Nursebot project", with robots Flo (Roy et al., 2000) and Pearl (Montemerlo et al., 2002), and "I.L.S.A" (Haigh et al.,2002) or "Morpha" (Lay et al., 2001) projects.

In order to contribute to this research field, the Electronics Department of the University of Alcalá is working on the SIRAPEM project (Spanish acronym of Robotic System for Elderly Assistance). The goal of this project is the development of a robotic aid that serves primary functions of tele-presence, telemedicine, intelligent reminding, safeguarding, mobility assistance and social interaction.

Figure 1 shows a simplified diagram of the SIRAPEM global architecture, based on a commercial platform (the PeopleBot robot of ActivMedia Robotics (2003)) endowed with a differential drive system, encoders, bumpers, two sonar rings (high and low), loudspeakers, microphone and on-board PC. The robot has been also provided with a PTZ color camera, a tactile screen and wireless Ethernet link. The system architecture includes several human-machine interaction systems, such as voice (synthesis and recognition speech) and touch screen for simple command selection (for example, a destination room to which the robot must go to carry out a service task).

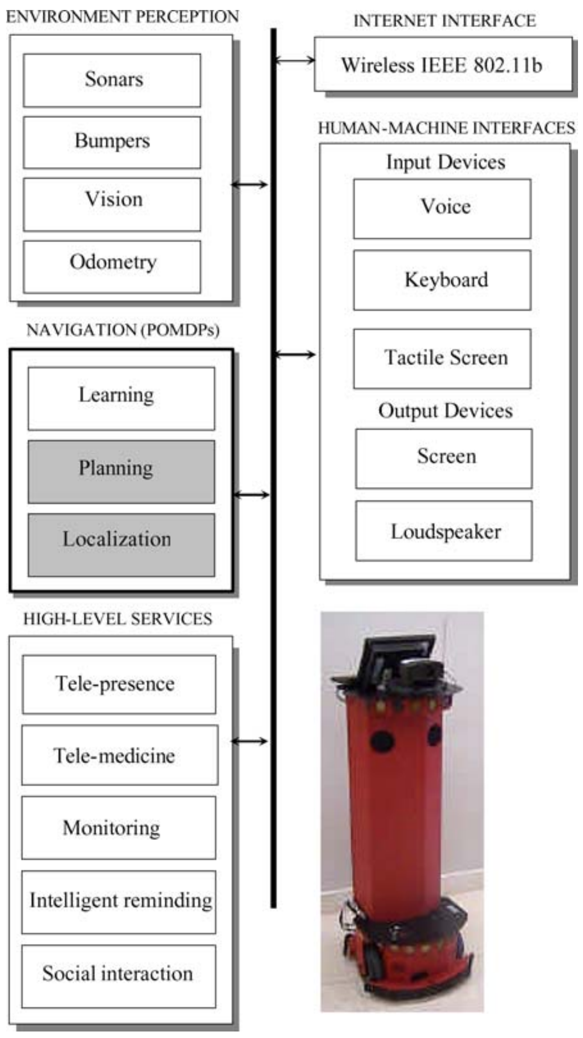


Figure 1. Global architecture of the SIRAPEM project.

This paper focuses on the navigation module, and mainly, in the localization and planning systems, whose final objective is to guide the robot to a goal room. In this kind of care applications, in which the robot must perform tasks in indoor environments for long periods of time, a decisive factor is to achieve a robust navigation system capable of treat real world uncertainties and solve global localization failures without any user supervision. Another desired feature for these assistant robotic systems is to simplify the installation process, in order to use it in different environments (houses, hospitals, etc.) without long or difficult configuration steps. So, they must use simple environment representations and natural landmarks that can be easily found in any indoor environment.

Many researchers have already pointed out and shown that probabilistic representation and reasoning is appropriate and very effective for navigating in noisy real world (Kaelbling et al., 1996; Burgard et al., 1996; Thrun, 2002). The research on this field has diverged to different approaches categorized with respect to the environment representation technique adopted: metric or topological. In both cases, Markov foundations (Fox, 1998) for robot localization have been applied with successful results. Even though topological approaches are less precise due to the coarse discretization of the environment, they allow to solve global navigation tasks in which it's not necessary to know the robot's pose in detail. Given robust low-level routines, it is only necessary to know that the robot is in some region to allow the navigation task.

In the framework of navigation with topological maps, probabilistic localization and decision-making adopt the form of Partially Observable Markov Decision Processes (POMDPs). These models provide solutions to localization, planning and learning in the robotics context, and have been used as probabilistic reasoning method in the three modules of the navigation system proposed in this work (see Fig. 1). One of the main contributions of this work is the addition of visual information to the Markov model, taking advantage of POMDPs as natural framework for sensor fusion. This visual information is used not only for observations, but also for state transition detection. As it will be shown in the results section, the addition of simple visual information reduces the positional uncertainty by increasing the observability of the environment, and makes the navigation system much more robust and reliable (without decreasing real-time execution ability). Another important contribution is the development of a two-level layered planning architecture that combines global and local policies to achieve several planning objectives: guidance to a room, reduction of locational uncertainty and exploration.

This paper is organized as follows. After placing this work within the context of previous similar ones, a brief overview of POMDPs foundations is presented as background in Section 2. Section 3 describes the Markov model used in this navigation application. Section 4 shows the global architecture of the navigation system. The localization module is described in Section 5 and the two layers of the planning system are shown in Section 6. Finally, we show some experimental results, whereas a final discussion and conclusion summarizes the paper.

1.1. Related Previous Work

Markov models, and particularly POMDPs, have already been widely used in robotics, and especially in robot navigation. The robots DERVISH (Nourbakhsh et al., 1995), developed in the Stanford University, and Xavier (Koenig and Simmons, 1998), in the Carnegie-Mellon University, were the first robots successfully using this kind of navigation strategies for localization and action planning. In the nursing applications field, in which robots interact with people and uncertainty is pervasive, robots such as Flo (Roy et al., 2000) or Pearl (Montemerlo et al., 2002) use POMDPs at all levels of decision making, and not only in low-level navigation routines. However, in all these successful navigation systems, only proximity sensors are used to perceive the environment. Due to the typical high perceptual aliasing of these sensors in office environments, using only proximity sensors makes the Markov model highly non-observable, and the initial global localization stage is rather slow.

On the other hand, there are quite a lot of recent works using appearance-based methods for robot navigation with visual information. Some of these works, such as Gechter et al.(2001) and Regini et al. (2002), incorporate POMDP models as a method for taking into account previous state of the robot to evaluate its new pose, avoiding the teleportation phenomena. However, these works are focused on visual algorithms, and very slightly integrate them into a complete robot navigation architecture. So, the above referenced systems don't combine any other sensorial system, and use the POMDP only for localizing the robot, and not for planning or exploring.

This work is a convergence point between these two research lines, proposing a complete navigation architecture that adds visual information to proximity sensors to improve previous navigation results, making more robust and faster the global localization task. Furthermore, a new Markov model is proposed that better adapts to environment topology, being completely integrated with a planning system that simultaneously contemplates several navigation objectives.

2. POMDPs Review

Although there is a wide literature about POMDPs theory (Papadimitriou and Tsitsiklis, 1987; Puterman, 1994; Kaelbling et al., 1998), in this section some terminology and main foundations are briefly introduced as theoretical background of the proposed work.

2.1. Markov Decision Processes

A Markov Decision Process (MDP) is a model for sequential decision making, formally defined as a tuple $\{\mathbf{S}, \mathbf{A}, \mathbf{T}, \mathbf{R}\}$, where,

- $\mathbf{S}$ is a finite set of states ($s \in \mathbf{S}$).
- $\mathbf{A}$ is a finite set of actions ($a \in \mathbf{A}$).
- $\mathbf{T} = \{p(s'|s, a) \,\forall\, (s, s' \in \mathbf{S}\; a \in \mathbf{A})\}$ is a state transition model which specifies a conditional probability distribution of posterior state s' given prior state s and action executed a.
- $\mathbf{R} = \{r(s, a) \,\forall\, (s \in \mathbf{S}\; a \in \mathbf{A})\}$ is the reward function, that determines the immediate utility (as a function of an objective) of executing action a at state s.

A MDP assumes the Markov property, which establishes that actual state and action are the only information needed to predict next state:

$$p(s_{t+1} \mid s_0, a_0, s_1, a_1, \ldots, s_t, a_t) = p(s_{t+1} \mid s_t, a_t) \quad (1)$$

In a MDP, the actual state s is always known without uncertainty. So, planning in a MDP is the problem of action selection as a function of the actual state (Howard, 1960). A MDP solution is a policy $a = \pi(s)$, which maps states into actions and so determines which action must be executed at each state. An optimal policy $a = \pi^*(s)$ is that one that maximizes future rewards. Finding optimal policies for MDPs is a well known problem in the artificial intelligent field, to which several exact and approximate solutions (such as the value iteration algorithm) have been proposed (Howard, 1960; Papadimitriou and Tsitsiklis, 1987; Puterman, 1994).

2.2. Partially Observable Markov Decision Processes

A POMDP is used under domains where there is not certainty about the actual state of the system. Instead, the agent can do observations, and so the model includes the following elements:

- $\{\mathbf{S}, \mathbf{A}, \mathbf{T}, \mathbf{R}\}$, the same that in the MDP context.
- $\mathbf{O}$, a finite set of observations ($o \in \mathbf{O}$)

- $\vartheta = \{p(o \mid s) \, \forall \, o \in \mathbf{O}, s \in \mathbf{S}\}$ is an observation model which specifies a conditional probability distribution over observations given the actual state s.

Because in this case the agent has not direct access to the current state, it uses actions and observations to maintain a probability distribution over all possible states, known as the belief distribution, **Bel**(**S**). A POMDP is still a markovian process in terms of this probability distribution, which only depends on the prior belief, prior action and current observation.

In a POMDP, a policy $a = \pi(\mathbf{Bel})$ maps beliefs into actions. However, what in a MDP was a discrete state space problem, now is a high-dimensional continuous space. Although there are numerous studies about finding optimal policies in POMDPs (Cassandra, 1994; Littman, 1994; Kaelbling et al.,1998), the size of state spaces and real-time constraints make them infeasible to solve navigation problems in robotic contexts. This paper proposes an alternative approximate solution for planning in POMDP-based navigation contexts, dividing the problem into two layers and applying some heuristic strategies for action selection. This method provides successful results in this kind of robot navigation applications.

3. Markov Model for Global Navigation

A POMDP model for robot navigation is constructed from two sources of information: the topology of the environment and some experimental or learned information about action and sensor errors and uncertainties. Taking into account that the final objective of the SIRAPEM navigation system is to direct the robot from one room to another to perform guiding or service tasks, we discretize the environment into coarse-grained regions (nodes) of variable size in accordance with the topology of the environment, in order to make easier the planning task. As it's shown in Fig. 2 for a virtual environment, only one node is assigned to each room, while the corridor is discretized into thinner regions. The limits of these regions correspond to any change in lateral features of the corridor (such as a new door, opening or piece of wall). This is a suitable discretization method in this type of structured environments, since nodes are directly related to topological locations in which the planning module may need to change the commanded action.

3.1. The Elements: States, Actions and Observations

In the context of robot navigation, the states of the Markov model are the locations (or nodes) of a topological representation of the environment. Actions are local navigation behaviors that the robot can execute to move from one state to another, and observations are all kind of environment information the robot can extract from its sensors. In this case, the Markov model is partially observable because the robot may never know exactly which state it is in.

The states (**S**) of our Markov model are directly related to the nodes of the topological graph. A single state corresponds to each room node, while four states are assigned to each corridor node, one for each of the four orientations the robot can adopt during corridor navigation.

The actions (**A**) selected to produce transitions from one state to another correspond to local navigation behaviors of the robot. We assume imperfect actions, so the effect of an action can be different of the expected one (this will be modelled by the transition model **T**). These actions are:

(1) "Go out room" (a_O): to traverse doors using sonar an visual information in room states,
(2) "Enter room" (a_E): only defined in corridor states oriented to a door,
(3) "Turn right" (a_R): to turn 90° to the right,
(4) "Turn Left" (a_L): to turn 90° to the left,
(5) "Follow Corridor" (a_F): to continue through the corridor to the next state, and
(6) "No Operation" (a_{NO}): used as a directive in the goal state.

Finally, the observations (**O**) in our model come from the two sensorial systems of the robot: sonar and vision. Markov models provide a natural way to combine multisensorial information, as it will be shown in Section 3.2. In each state, the robot makes three kinds of observations:

(1) "Abstract Sonar Observation" (o_{ASO}). Each of the three nominal directions around the robot (left, front and right) is classified as "free" or "occupied" using sonar information, and an abstract observation is constructed from the combination of the percepts in each direction (thus, there are eight possible abstract sonar observations, as it's shown in Fig. 3(a)).

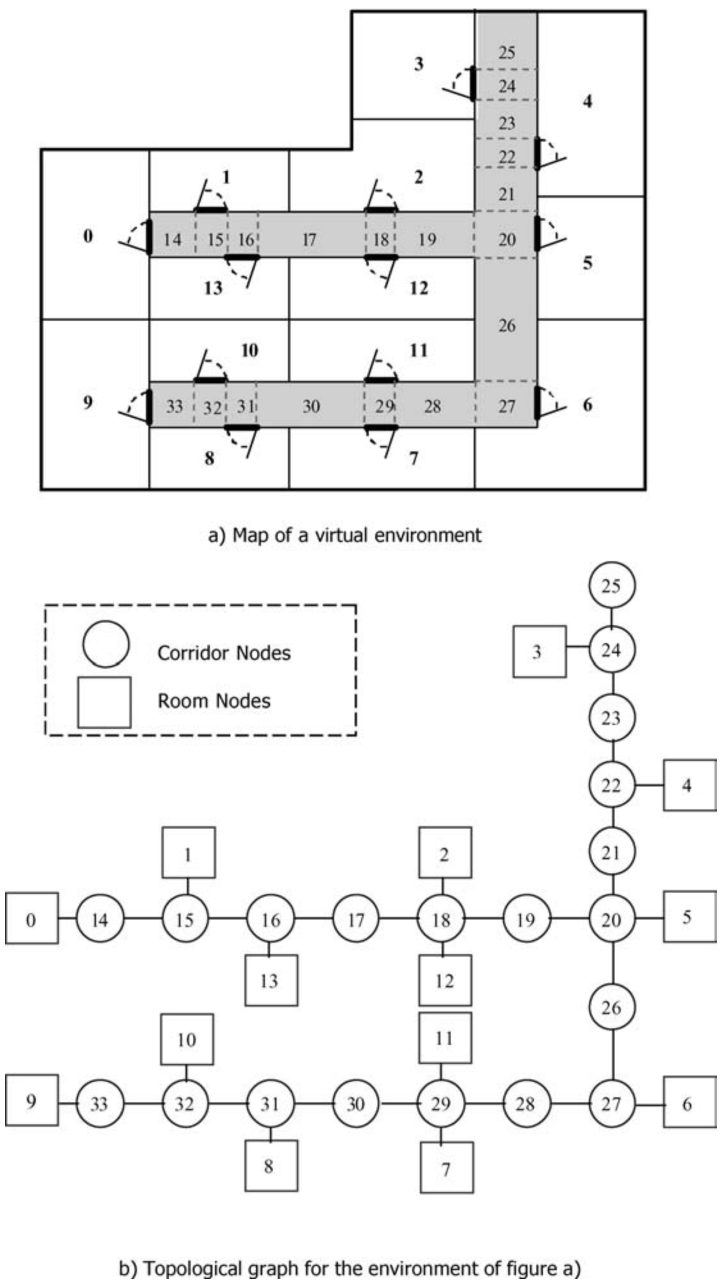
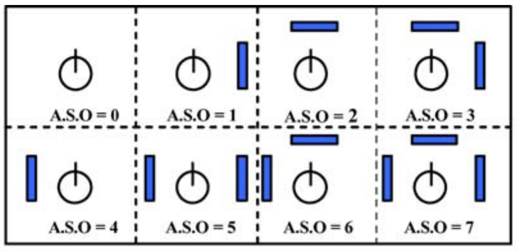

a) Map of a virtual environment
Corridor Nodes
Room Nodes
b) Topological graph for the environment of figure a)

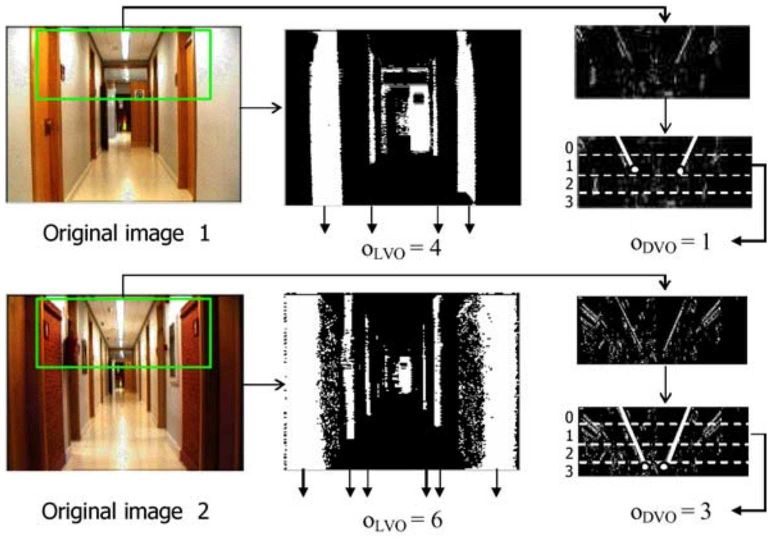
Figure 2. Topological graph for a virtual environment.

(2) “Landmark Visual Observation” (o_{LVO}). Doors are considered as natural visual landmarks, because they exist in all indoor environments and can be easily segmented from the image using color (previously trained) and some geometrical restrictions. This observation is the number of doors (in lateral walls of the corridor) extracted from the image (see Fig. 3(b)), and it reduces the perceptual aliasing of sonar by distinguishing states at the beginning from states at the end of a corridor. However, in long corridors, doors far away from the robot can’t be easily segmented from the image (this is the case of image 2 of Fig. 3(b)), and this is the reason because we introduce a third visual observation.

(3) “Depth Visual Observation” (o_{DVO}). As human-interaction robots have tall bodies with the camera



a) Abstract Sonar Observation (o_{ASO}) of the Markov model.



b) Examples of visual observations: o_{LVO} and o_{DVO}.

Figure 3. Observations of the proposed Markov model.

on the top, it's possible to detect the vanishing ceiling lines and classify its length into a set of discrete values (in this case, we use four quantification levels, as it's shown in Fig. 3(b)). This is a less sensitive to noise observation than using floor vanishing lines (mainly to occlusions due to people walking through the corridor) and it provides complementary information to o_{LVO}.

Figure 3(b) shows two scenes of the same corridor from different positions and their corresponding o_{LVO} and o_{DVO} observations. It's shown that these are obtained by means of very simple image processing techniques (color segmentation for o_{LVO} and edge detection for o_{DVO}), and have the advantage, regarding correlation techniques used in Gechter et al. (2001) or Regini et al. (2002), that they are less sensitive to slight pose deviations within the same node.

3.2. *Visual Information Utility and Improvements*

Visual observations increase the robustness of the localization system by reducing perceptual aliasing. On the other hand, visual information also improves state transition detection, as it's shown in the following subsections.

3.2.1. Sensor Fusion to Improve Observability. Using only sonar to perceive the environment makes the Markov model highly non-observable due to perceptual aliasing. Furthermore, the "Abstract Sonar Observation" is highly dependent on doors state (opened or closed). The addition of the visual observations proposed in this work augments the observability of states. For example, corridor states with an opened door on the left and a wall on the right produces the same abstract sonar observation ($o_{\mathrm{ASO}} = 1$) independently if they are at the beginning or at the end of the corridor. However, the number of doors seen from the current state (o_{LVO}) allows to distinguish between these states.

POMDPs provide a natural way for using multisensorial fusion in their observation models ($p(o \mid s)$ probabilities). In this case, **o** is a vector composed by the three observations proposed in the former subsection. Because these are independent observations, the observation model can be simplified in the following way:

$$\begin{aligned} p(o \mid s) &= p(o_{\mathrm{ASO}}, o_{\mathrm{LVO}}, o_{\mathrm{DVO}} \mid s) \\ &= p(o_{\mathrm{ASO}} \mid s) \cdot p(o_{\mathrm{LVO}} \mid s) \cdot p(o_{\mathrm{DVO}} \mid s) \end{aligned} \tag{2}$$

3.2.2. Visual Information to Improve State Transition Detection. To ensure that when the robot is in a corridor, it only adopts the four allowed directions without large errors, it's necessary that, during the execution of a "Follow Corridor" action, the robot becomes aligned with the corridor longitudinal axis. So, when the robot stands up to a new corridor, it aligns itself with a subtask that uses visual vanishing points. Besides, during corridor following, it uses sonar buffers to detect the walls and construct a local model of the corridor, using the method proposed by Konolige (1998). This method uses an occupancy map to extract the local model of the corridor (Konolige, 1997) and updates this model when robot moves using an anchoring process (Saffiotti, 1994). A fuzzy controller allows the robot to move along the obtained corridor (Saffiotti et al., 1993). Finally, an individual "Follow Corridor" action terminates when the robot reaches a new state of the corridor. Detecting these transitions only with sonar readings is very critical when doors are closed.

To solve this problem, we add visual information to detect door frames as natural landmarks of state transitions (using color segmentation and some geometrical restrictions). The advantage of this method is that the image processing step is fast and easy, being only necessary to process two lateral windows of the image as it's shown in Fig. 4. Whenever a vertical transition from wall to door color (or vice versa) is detected in a lateral window, the distance to travel as far as that new state is obtained from the following formula, using a pin-hole model of the camera (see Fig. 4):

$$d = \frac{l}{tg\,(\alpha)} = K \cdot l \tag{3}$$

where l is the distance of the robot to the wall in the same side as the detected door frame (obtained from sonar readings) and α is the visual angle of the door

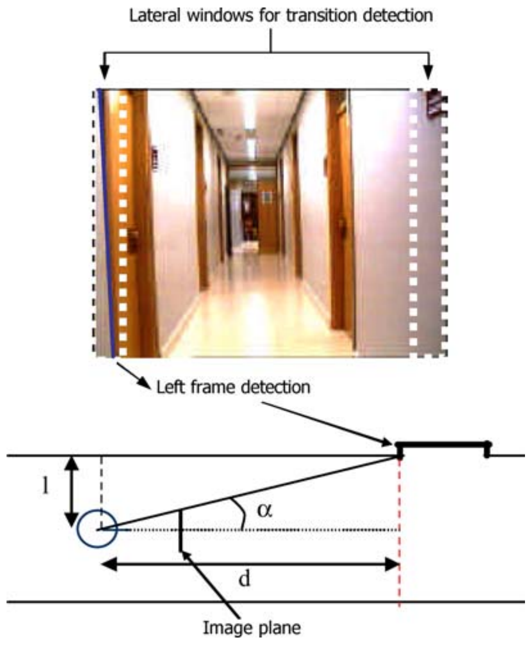


Figure 4. State transition detection by means of visual information.

frame. As the detected frame is always in the edge of the image, the visual angle α only depends on the focal distance of the camera that is constant for a fixed zoom (and known from camera specifications). After covering distance d (measured with relative odometry readings), the robot reaches the new state. This transition can be confirmed (fused) with sonar if the door is opened. Another advantage of this transition detection approach is that no assumptions are made about doors or corridor widths.

3.3. Action and Observation Uncertainties

To automatically calculate (compile) the transition and observation matrixes of the POMDP model, it's necessary to define some action an observation uncertainties. The POMDP compiler, described in next subsection, supports two ways of defining these uncertainties. The first one is to introduce some experimental "handmade" uncertainty rules (this method is also used in Koenig and Simmons (1998) and Zanichelli (1999)). The empirical rules used in our robot are shown in Table 1 for action and observation uncertainties. For example, if a "Follow" action (a_F) is commanded, the probability of making a state transition (F) is 70%, while there is a 5% probability of remaining in the same

Table 1. Rules for constructing the Markov model from empirical action and observation uncertainties.

Action Uncertainty (F = Follow, L = Left, R = Right, O = Out, E = Enter, N = No action)

Command	Efect of Command (% probabilities)			
a_F	N = 10	F = 70	FF = 10	FFF = 10
a_L	N = 5		L = 90	LL = 5
a_R	N = 5		R = 90	RR = 5
a_O	N = 10		O = 80	OF = 10
a_E	N = 10		E = 90	

Observation Uncertainty

ASO model	
Open door probability (for all doors)	50%
Prob. of detecting something being nothing	10%
Prob. of detecting nothing being something	5%
LVO model	
Assigned probability to real number of doors	70%
Maximum deviation	±2 doors
DVO model	
Assigned probability to real o_{DVO} observation	80%
Maximum deviation	±1

state (N = no action), a 5% probability of making two successive state transitions (FF), and a 5% probability of making three state transitions (FFF) (similar rules are used in the observation model). Experience with this method has shown it to produce reliable navigation. The second one consists of learning probabilities to more closely reflect the actual environment of the robot. Adopting this last method, we have developed a learning module (see Fig. 1) that adjusts observations and probabilities during an initial exploration stage, and maintains these parameters updated when the robot is performing another guiding or service tasks (López et al., 2004). This module, that also makes easier the installation of the system in a new environment, is out of the scope of this paper.

3.4. POMDP Compilation

As it's shown in Fig. 5, the POMDP model is automatically constructed from two sources of information:

- The topology of the environment, represented as a graph with nodes and connections. This graph fixes the states ($s \in \mathbf{S}$) of the model and establishes the ideal transitions among them by means of logical connectivity rules.
- The uncertainty model, that characterizes the errors or ambiguities of actions and observations, and together with the graph, makes possible to generate the transition and observation matrixes of the POMDP.

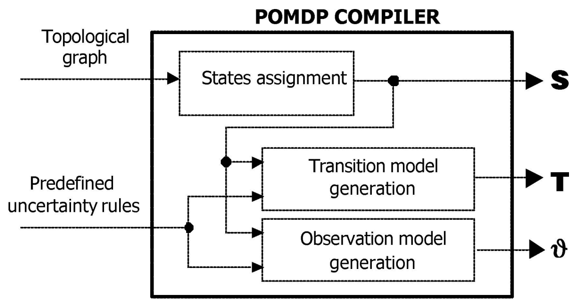
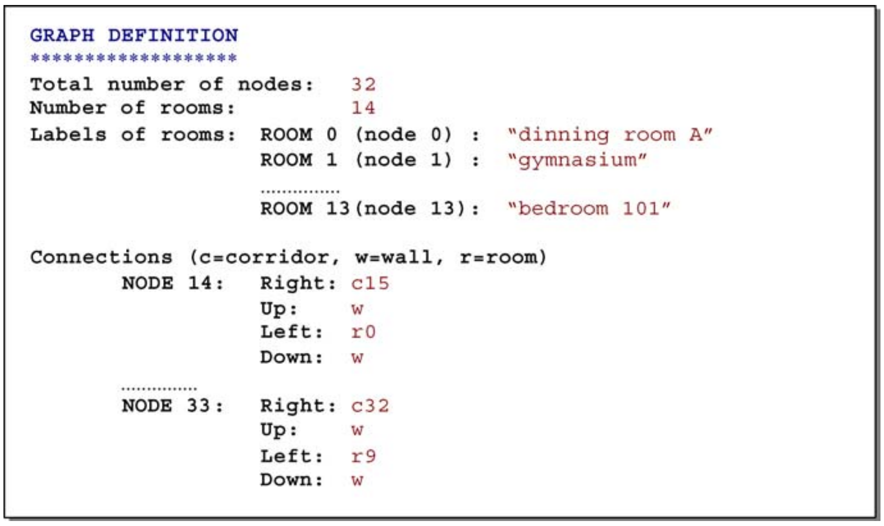

POMDP COMPILER
Topological graph
Predefined uncertainty rules
States assignment
Transition model generation
Observation model generation
S
T
ϑ

Figure 5. Structure of the POMDP compiler.

Several recent works (Brants, 1996; Yairi et al., 2003) try to learn the structure (topology) of POMDP models from experimental data. These methods need long training stages to produce reliable results, and have not been validated in real robotic applications yet. Taking into account that a reliable graph is crucial for the localization and planning systems to work properly, and the topological representation proposed in this work is very close to human environment perception, we propose a manual introduction of the graph. To do this, the SIRAPEM system incorporates an application to help the user to introduce the graph of the environment (this step is needed only once when the robot is installed in a new working domain, because the graph is a static representation of the environment). After numbering the nodes of the graph (the only condition to do this is to assign the lower numbers to room nodes, starting with 0), the connections in the four directions of each corridor node must be indicated. Figure 6 shows an example of the "Graph definition" application (for the environment of Fig. 2), which also allows to associate a label to each room. These labels will be identified by the voice recognition interface and used as user commands to indicate goal rooms.

Once the graph is introduced, the POMDP compiler automatically assigns a number (n_s) to each state of the graph as a function of the number of the node to which it belongs (n) and its orientation within the node (head = {0(right), 1(up), 2(left), 3(down)}) in the following way (*n*_rooms being the number of room



```
GRAPH DEFINITION
*******************
Total number of nodes:   32
Number of rooms:         14
Labels of rooms:  ROOM 0  (node 0)  :  "dinning room A"
                  ROOM 1  (node 1)  :  "gymnasium"
                  ...............
                  ROOM 13(node 13):  "bedroom 101"

Connections (c=corridor, w=wall, r=room)
        NODE 14:   Right: c15
                   Up:    w
                   Left:  r0
                   Down:  w
        ...............
        NODE 33:   Right: c32
                   Up:    w
                   Left:  r9
                   Down:  w
```

Figure 6. Example of graph definition for the environment of Fig. 2.

nodes):

$$\text{Room states}: n_s = n$$
$$\text{Corridor states}: n_s = n\_\text{rooms} + (n - n\_\text{rooms}) \times 4 + \text{head}$$

Finally, the compiler generates the initial transition and observation matrixes using the predefined uncertainty rules. Besides, during normal working of the navigation system (performing guiding tasks), the learning module uses current actions and observations to maintain the parameters updated in the face of possible changes (López et al., 2004).

4. Navigation System Architecture

The problem of acting in partially observable environments can be decomposed into two components: a state estimator, which takes as input the last belief state, the most recent action and the most recent observation, and returns an updated belief state, and a policy, which maps belief states into actions. In robotics context, the first component is robot localization and the last one is task planning.

Figure 7 shows the global navigation architecture of the SIRAPEM project, formulated as a POMDP model. At each process step, the planning module selects a new action as a command for the local navigation module, that implements the actions of the POMDP as local navigation behaviors. As a result, the robot modifies its state (location), and receives a new observation from its sensorial systems. The last action executed, besides the new observation perceived, are used by the *localization* module to update the belief distribution **Bel**(**S**).

After each state transition, and once updated the belief, the planning module chooses the next action to execute. Instead of using an optimal POMDP policy (that involves high computational times), this selection is simplified by dividing the planning module into two layers:

- A local policy, that assigns an optimal action to each individual state (as in the MDP case). This assignment depends on the planning context. Three possible contexts have been considered: (1) *guiding* (the objective is to reach a goal room selected by the user to perform a service or guiding task), (2) *localizing* (the objective is to reduce location uncertainty) and (3) *exploring* (the objective is to learn or adjust observations and uncertainties of the Markov model).
- A global policy, that using the current belief and the local policy, selects the best action by means of different heuristic strategies proposed by Kaelbling et al. (1996).

This proposed two-layered planning architecture is able to combine several contexts of the local policy to simultaneously integrate different planning objectives, as will be shown in subsequent sections.

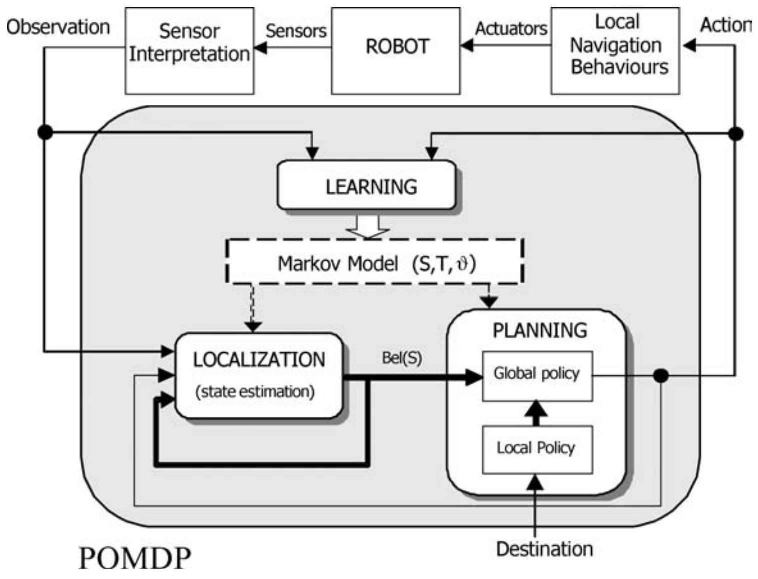


Figure 7. Global architecture of the navigation system.

Finally, the learning module (López et al., 2004) uses action and observation data to learn and adjust the observations and uncertainties of the Markov model.

5. Localization and Uncertainty Evaluation

The localization module updates the belief distribution after each state transition, using the well known Markov localization equations (Thrun, 2002). These equations are applied in two steps:

- Prediction step, that can be calculated just after a new action a_i is commanded:

$$Bel_{\text{posterior}}(s') = K \cdot \sum_{s \in S} p(s' \mid s, a) \cdot Bel_{\text{prior}}(s) \quad \forall s' \in \boldsymbol{S} \quad (4)$$

where K is a normalization factor to ensure that the probabilities all sum one.

- Estimation step, that must be calculated after action execution, once the new observation **o** (at new state) is perceived, using the Bayes rule:

$$Bel_{\text{posterior}}(s) = K \cdot p(o \mid s) \cdot Bel_{\text{prior}}(s) \quad \forall s \in \boldsymbol{S} \quad (5)$$

In the first execution step, the belief distribution can be initialized in one of the two following ways: (a) If initial state of the robot is known, that state is assigned probability 1 and the rest 0, (b) If initial state is unknown, a uniform distribution is calculated over all states.

Although the planning system chooses the action based on the entire belief distribution, in some cases it's necessary to evaluate the degree of uncertainty of that distribution (this is, the locational uncertainty). A typical measure of discrete distributions uncertainty is the entropy. The normalized entropy (ranging from 0 to 1) of the belief distribution is:

$$H(\boldsymbol{Bel}) = -\frac{\sum_{s \in \mathrm{S}} Bel(s) \cdot \log(Bel(s))}{\log(n_s)} \quad (6)$$

where n_s is the number of states of the Markov model. The lower the value, the more certain the distribution. This measure has been used in all previous robotic applications for characterizing locational uncertainty (Kaelbling, 1996; Zanichelli, 1999).

However, this measure is not appropriate for detecting situations in which there are a few maximums of similar value, being the rest of the elements zero, because it's detected as a low entropy distribution. In fact, even being only two maximums, that is a not good result for the localization module, because they can correspond to far locations in the environment. A more suitable choice should be to use a least square measure respect to ideal delta distribution, that better detects the convergence of the distribution to a unique maximum (and so, that the robot is globally localized). However, we propose another approximate measure that, providing similar results to least squares, is faster calculated by using only the two first maximum values of the distribution (it's also less sensitive when uncertainty is high, and more sensitive to secondary maximums during the tracking stage). This is the normalized divergence factor, calculated in the following way:

$$D(\boldsymbol{Bel}) = 1 - \frac{n_s\,(d_{\max} + p_{\max}) - 1}{2 \cdot n_s - 1} \qquad (7)$$

where $d_{\max}$ is the difference between first and second maximum values of the distribution, and $p_{\max}$ the absolute value of the first maximum. Again, a high value indicates that the distribution converges to a unique maximum. In the results section we'll show that this new measure provides much better results when planning in some kind of environments.

6. Planning under Uncertainty

A POMDP model is a MDP model with probabilistic observations. Finding optimal policies in the MDP case (that is a discrete space model) is easy and quickly for even very large models. However, in the POMDP case, finding optimal control strategies is computationally intractable for all but the simplest environments, because the beliefs space is continuous and high-dimensional.

There are several recent works that use a hierarchical representation of the environment, with different levels of resolution, to reduce the number of states that take part in the planning algorithms (Theocharous and Mahadevan, 2002; Pineau and Thrun, 2002). However, these methods need more complex perception algorithms to distinguish states at different levels of abstraction, and so they need more prior knowledge about the environment and more complex learning algorithms. On the other hand, there are also several recent approximate methods for solving POMDPs, such as those that use a compressed belief distribution to accelerate algorithms (Roy, 2003) or the point-based value iteration algorithm (Pineau et al., 2003) in which planning is performed only on a sampled set of reachable belief points.

The solution adopted in this work is to divide the planning problem into two steps: the first one finds an optimal local policy for the underlying MDP ($a^* = \pi^*(s)$, or to simplify notation, $a^*(s)$), and the second one uses a number of simple heuristic strategies to select a final action ($a^*(\mathbf{Bel})$) as a function of the local policy and the belief. This structure is shown in Fig. 8 and described in subsequent sections.

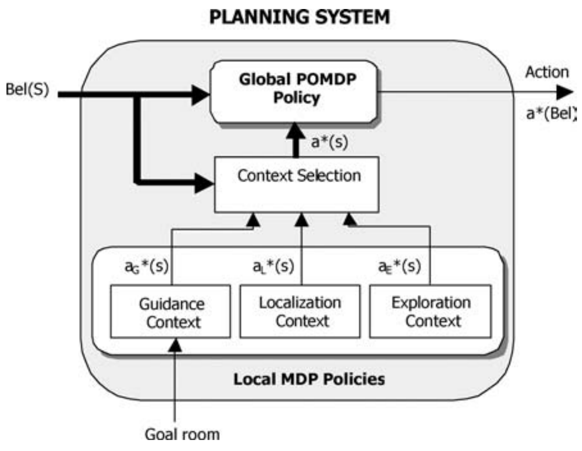


Figure 8. Planning system architecture, consisting of two layers: (1) Global POMDP Policy, and (2) Local MDP Policies.

6.1. *Contexts and Local Policies*

The objective of the local policy is to assign an optimal action ($a^*(s)$) to each individual state s. This assignment depends on the planning context. The use of several contexts allows the robot to simultaneously achieve several planning objectives. The localization and guidance contexts try to simulate the optimal policy of a POMDP, which seamlessly integrates the two concerns of acting in order to reduce uncertainty and to achieve a goal. The exploration context is to select actions for learning the parameters of the Markov model.

In this subsection we show the three contexts separately. Later, they will be automatically selected or combined by the Context Selection and Global policy modules (Fig. 8).

6.1.1. Guidance Context. This local policy is calculated whenever a new goal room is selected by the user. Its main objective is to assign to each individual state s, an optimal action ($a_G{}^*$(s)) to guide the robot to the goal.

One of the most well known algorithms for finding optimal policies in MDPs is Value Iteration (Bellman, 1957). This algorithm assigns an optimal action to each state when the reward function $r(s, a)$ is available. In this application, the information about the utility of actions for reaching the destination room is contained in the graph. So, a simple path searching algorithm

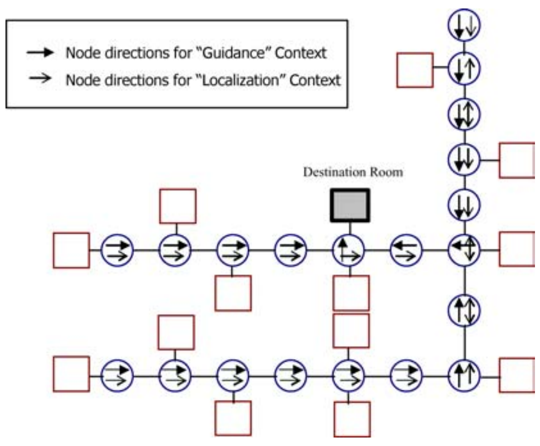

Node directions for "Guidance" Context
Node directions for "Localization" Context
Destination Room

Figure 9. Node directions for "Guidance" (to room 2) and "Localization" contexts for environment of Fig. 2.

can effectively solve the underlying MDP, without any intermediate reward function.

So, a modification of the A^* search algorithm (Winston, 1984) is used to assign a preferred heading to each node of the topological graph, based on minimizing the expected total number of nodes to traverse (shorter distance criterion cannot be used because the graph has not metric information). The modification of the algorithm consists of inverting the search direction, because in this application there is not an initial node (only a destination node). Figure 9 shows the resulting node directions for goal room 2 on the graph of environment of Fig. 2.

Later, an optimal action is assigned to the four states of each node in the following way: a "follow" (a_F) action is assigned to the state whose orientation is the same as the preferred heading of the node, while the remaining states are assigned actions that will turn the robot towards that heading (a_L or a_R). Finally, a "no operation" action (a_{NO}) is assigned to the goal room state.

Besides optimal actions, when a new goal room is selected, $Q(s, a)$ values are assigned to each (s, a) pair. In the MDPs theory, Q-values (Lovejoi, 1991) characterize the utility of executing each action at each state, and will be used by one of the global heuristic policies shown in next section. To simplify Q values calculation, the following criterion has been used: $Q(s, a) = 1$ if action a is optimal at state s, $Q(s, a) = -1$ (negative utility) if actions a is not defined at state s, and $Q(s, a) = -0.5$ for the remaining cases (actions that disaligns the robot from the preferred heading).

6.1.2. Localization Context. This policy is used to guide the robot to Sensorial Relevant States (SRSs) that reduce positional uncertainty, even if that requires moving it away from the goal temporarily. This planning objective was not considered in previous similar robots, such as DERVISH (Nourbakhsh et al., 1995) or Xavier (Koenig and Simmons, 1998), or was implemented by means of fixed sequences of movements (Cassandra, 1994) that don't contemplate environment relevant places to reduce uncertainty.

In an indoor environment, it's usual to find different zones that produce not only the same observations, but also the same sequence of observations as the robot traverses them by executing the same actions (for example, symmetric corridors). SRSs are states that break a sequence of observations that can be found in another zone of the graph.

Because a state can be reached from different paths and so, with different histories of observations, SRSs are not characteristic states of the graph, but they depend on the starting state of the robot. This means that each starting state has its own SRS. To simplify the calculation of SRSs, and taking into account that the more informative states are those aligned with corridors, it has been supposed that in the localization context the robot is going to execute sequences of "follow corridor" actions. So, the moving direction along the corridor to reach a SRS as soon as possible must be calculated for each state of each corridor. To do this, the Composed Observations (COs) of these states are calculated from the graph and the current observation model ϑ in the following way:

$$CO(s) = 100 \cdot o_{\text{DVO}}(s) + 10 \cdot o_{\text{LVO}}(s) + o_{\text{ASO}}(s)$$

with

$$\begin{aligned} o_{\text{DVO}}(s) &= \arg\max_{o_{\text{DVO}}} (p(o_{\text{DVO}}|s)) \\ o_{\text{LVO}}(s) &= \arg\max_{o_{\text{LVO}}} (p(o_{\text{LVO}}|s)) \\ o_{\text{ASO}}(s) &= \arg\max_{o_{\text{ASO}}} (p(o_{\text{ASO}}|s)) \end{aligned} \tag{8}$$

Later, the nearest SRS for each node is calculated by studying the sequence of COs obtained while moving in both corridor directions. Then, a preferred heading (among them that align the robot with any connected corridor) is assigned to each node. This heading points at the corridor direction that, by a sequence of "Follow Corridor" actions, directs the robot to the nearest SRS (Fig. 9 shows the node directions obtained for environment of Fig. 2). And finally, an optimal action is assigned to the four states of each corridor node to align the robot with this preferred heading (as it was described in the guidance context section). The optimal action assigned to room states is always "Go out room" (a_o).

So, this policy ($a^*_L(s)$) is only environment dependent and is automatically calculated from the connections of the graph and the ideal observations of each state.

6.1.3. Exploration Context. The objective of this local policy is to select actions during the exploration stage, in order to learn transition and observation probabilities. As in this stage the Markov model is unknown (the belief can't be calculated), there is not distinction between local and global policies, whose common function is to select actions in a reactive way to explore the environment. This context is strongly connected with the learning module (López et al., 2004) and they are out of the scope of this paper.

6.2. *Global Heuristic Policies*

The global policy combines the probabilities of each state to be the current state (belief distribution $Bel(\mathbf{S})$) with the best action assigned to each state (local policy $a^*(s)$) to select the final action to execute, $a^*(\mathbf{Bel})$. Once selected the local policy context (for example guidance context, $a^*(s) = a_G{}^*(s)$), some heuristic strategies proposed by Kaelbling et al. (1996) can be used to do this final selection.

The simpler one is the Most Likely State (MLS) global policy that finds the state with the highest probability and directly executes its local policy:

$$a^*_{\text{MLS}}(\boldsymbol{Bel}) = a^*\left(\arg\max_s (Bel(s))\right) \tag{9}$$

The Voting global policy first computes the probability mass of each action ($V(a)$) (probability of action a being optimal) according to the belief distribution, and then selects the action that is most likely to be optimal (the one with highest probability mass):

$$\begin{aligned} V(a) &= \sum_{s\,|_{a^*(s)=a}} Bel(s) \quad \forall a \in \boldsymbol{A} \\ a^*_{\text{vot}}(Bel) &= \arg\max_a (V(a)) \end{aligned} \tag{10}$$

This method is less sensitive to locational uncertainty, because it takes into account all states, not only the most probable one.

Finally, the Q_{MDP} global policy is a more refined version of the voting policy, in which the votes of each state are apportioned among all actions according to their Q-values:

$$\begin{aligned} V(a) &= \sum_{s\in S} Bel(s)\cdot Q^a(s) \quad \forall a \in A \\ a^*_{Q_{\text{MDP}}}(Bel) &= \arg\max_a (V(a)) \end{aligned} \tag{11}$$

This is in contrast to the "winner take all" behavior of the voting method, taking into account negative effect of actions.

Although there is some variability between these methods, for the most part all of them do well when initial state of the robot is known, and only the tracking problem is present. If initial state is unknown, the performance of the methods highly depends on particular configuration of starting states. However, MLS or Q_{MDP} global policies may cycle through the same set of actions without progressing to the goal when only guidance context is used. Properly combination of guidance and localization context highly improves the performance of these methods during global localization stage.

6.3. *Automatic Context Selection or Combination*

Apart from the exploration context, this section considers the automatic context selection (see Fig. 8) as a function of the locational uncertainty. When uncertainty is high, localization context is useful to gather information, while with low uncertainty, guidance context is the appropriate one. In some cases, however, there is benign high uncertainty in the belief state; that is, there is confusion among states that requires the same action. In these cases, it's not necessary to commute to localization context. So, an appropriate measure of uncertainty is the normalized divergence factor of the probability mass distribution, $D(V(a))$, (see Eq. (7)).

The thresholding-method for context selection uses a threshold ϕ for the divergence factor D. Only when divergence is over that threshold (high uncertainty), localization context is used as local policy:

$$a^*(s) = \begin{cases} a^*_G(s) & \text{if } D < \phi \\ a^*_L(s) & si\ D \geq \phi \end{cases} \tag{12}$$

However, the weighting-method combines both contexts using divergence as weighting factor. To do this, probability mass distributions for guidance and localization contexts ($V_G(a)$ and $V_L(a)$) are computed separately, and the weighted combined to obtain the final probability mass $V(a)$. As in the voting method, the action selected is the one with highest probability mass:

$$\begin{aligned} V(a) &= (1 - D) \cdot V_G(a) + D \cdot V_L(s) \\ a^*(Bel) &= \arg\max_a (V(a)) \end{aligned} \tag{13}$$

7. Simulation Results

To validate the proposed navigation architecture and compare the different planning methods and contexts, some experimental results are shown. Because the advantages of the localization system and some planning strategies can only be demonstrated in hard environments, we include two kinds of experiments. In this section, we show some results obtained with a simulator of the robot, in order to test the planning methods in hard virtual environments. The simulation platform used in these experiments (shown in Fig. 10) is based on Saphira commercial software (Konolige and Myers, 1998) provided by ActivMedia robotics, that includes a very realistic robot simulator that very closely reproduces real robot movements and ultrasound noisy measures on a user defined map. A visual 3D simulator using OpenGL software has been added to incorporate visual observations. So, simulation results can be reliably extrapolated to extract realistic conclusions about the system. Besides, in next section we show some experiments carried out with the real robot of the SIRAPEM project in one of the corridors of the Electronics Department (an "easier" environment), in order to validate the navigation system on a real robotic platform.

There are some things that make one world more difficult to navigate than another. One of them is its degree of perceptual aliasing, which substantially affects the agent's ability for localization and planning. The localization and two-layered planning architecture proposed in this work improves the robustness of the system in typical "aliased" environments, by properly combining two planning contexts: guidance and localization. As an example to demonstrate this, we use the virtual aliased environment shown in Fig. 2, in which there are two identical corridors. Firstly, we show some results concerning only the localization system. After that, we also include the planning module in some guidance experiments to compare the different planning strategies.

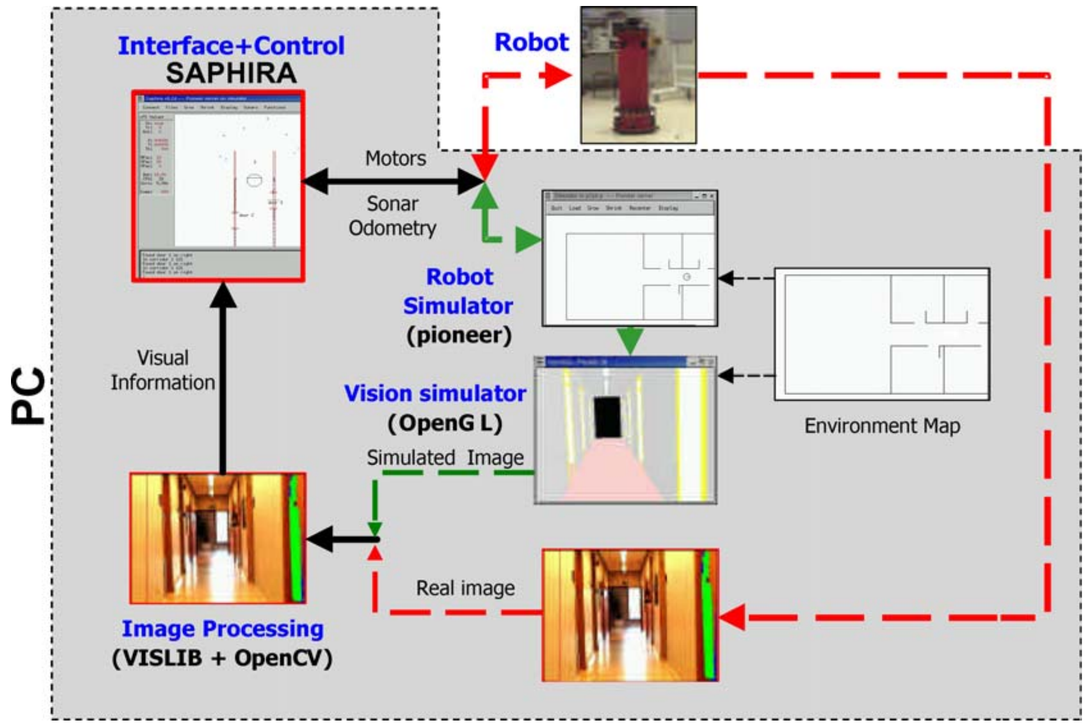


Figure 10. Diagram of test platforms: real robot or simulator.

7.1. *Localization Results*

Two are the main contributions of this work to Markov localization in POMDP navigation systems. The first one is the addition of visual information to accelerate the global localization stage from unknown initial position, and the second one is the usage of a novel measure to better characterize locational uncertainty. To demonstrate them, we executed the trajectory shown in Fig. 11(a), in which the "execution steps" of the POMDP process are numbered from 0 to 11. The robot was initially at node 14 (with unknown initial position), and a number of "Follow corridor" actions were executed to reach the end of the corridor. Then it executes a "Turn Left" action and continues through the new corridor until reaching room 3 door.

In the first experiments, all doors were opened, ensuring a good transition detection. This is the best assumption for only sonar operation. Two simulations were executed in this case: the first one using only sonar information for transition detection and observation, and the second one adding visual information. As the initial belief is uniform, and there is an identical corridor to that in which the robot is, the belief must converge to two maximum hypotheses, one for each corridor. Only when the robot reaches node 20 (that is an SRS) it's possible to eliminate this locational uncertainty, appearing a unique maximum in the distribution and starting the "tracking stage". Figure 11(b) shows the real state assigned probability evolution during execution steps for the two experiments. Until step 5 there are no information to distinguish corridors, but it can be seen that with visual information the robot is better and sooner localized within the corridor. Figure 11(c) shows entropy and divergence of both experiments. Both measures detect a lower uncertainty with visual information, but it can be seen that divergence better characterizes the convergence to a unique maximum, and so, the end of the global localization stage. So, with divergence it's easier to establish a threshold to distinguish "global localization" and "tracking" stages.

Figures 11(d) and (e) show the results of two new simulations in which doors 13, 2 and 4 were closed. Figure 11(d) shows how using only sonar information some transitions are lost (the robots skips positions 3, 9 and 10 of Fig. 11(a). This makes much worse the localization results. However, adding visual information no transitions are lost, and results are very similar to that of Fig. 11(b).

After performing some statistics, it can be concluded that uncertainty effect of "Follow Corridor" action is reduced from 60% to 15% with closed doors. The reduction of convergence time ("global localization stage") is very dependent on the environment, but it can be assured and proved that it's always quite shorter when adding visual observations.

So, visual information makes the localization more robust, reducing perceptual aliasing of states in the same corridor, and more independent of doors state. Besides, the proposed divergence uncertainty measure better characterizes the positional uncertainty that the typical entropy used in previous works.

7.2. *Planning Results*

The two layered planning architecture proposed in this work improves the robustness of the system in "aliased" environments, by properly combining the two planning contexts: guidance and localization. To demonstrate this, we show the results after executing some simulations in the same fictitious environment of Fig. 11(a). In all the experiments the robot was initially at room state 0, and the commanded goal room state was 2. However, the only initial knowledge of the robot about its position is that it's a room state (initial belief is a uniform distribution over room states). So, after the "go out room" action execution, and thanks to the visual observations, the robot quickly localizes itself within the corridor, but due to the environment aliasing, it doesn't know in which corridor it is. So, it should use the localization context to reach nodes 20 or 27 of Fig. 2, that are sensorial relevant nodes to reduce uncertainty.

Table 2 shows some statistical results (average number of actions to reach the goal, final values of entropy and divergence and skill percentage on reaching the

Table 2. Comparison of the planning strategies in the virtual environment of Fig. 11(a)).

	Only Guidance Context			
	No of Actions	Final H	Final D	Final State 2
MLS	6	0.351	0.754	54.3%
Voting	17	0.151	0.098	63.8%
Q_{MDP}	15	0.13	0.095	62.3%
	Guidance and Localization Contexts (always with voting global method)			
$H(V(a))$ threshold	14	0.13	0.05	83.5%
$D(V(a))$ threshold	13	0.12	0.04	100%
Weighted $D(V(a))$	13	0.12	0.04	100%

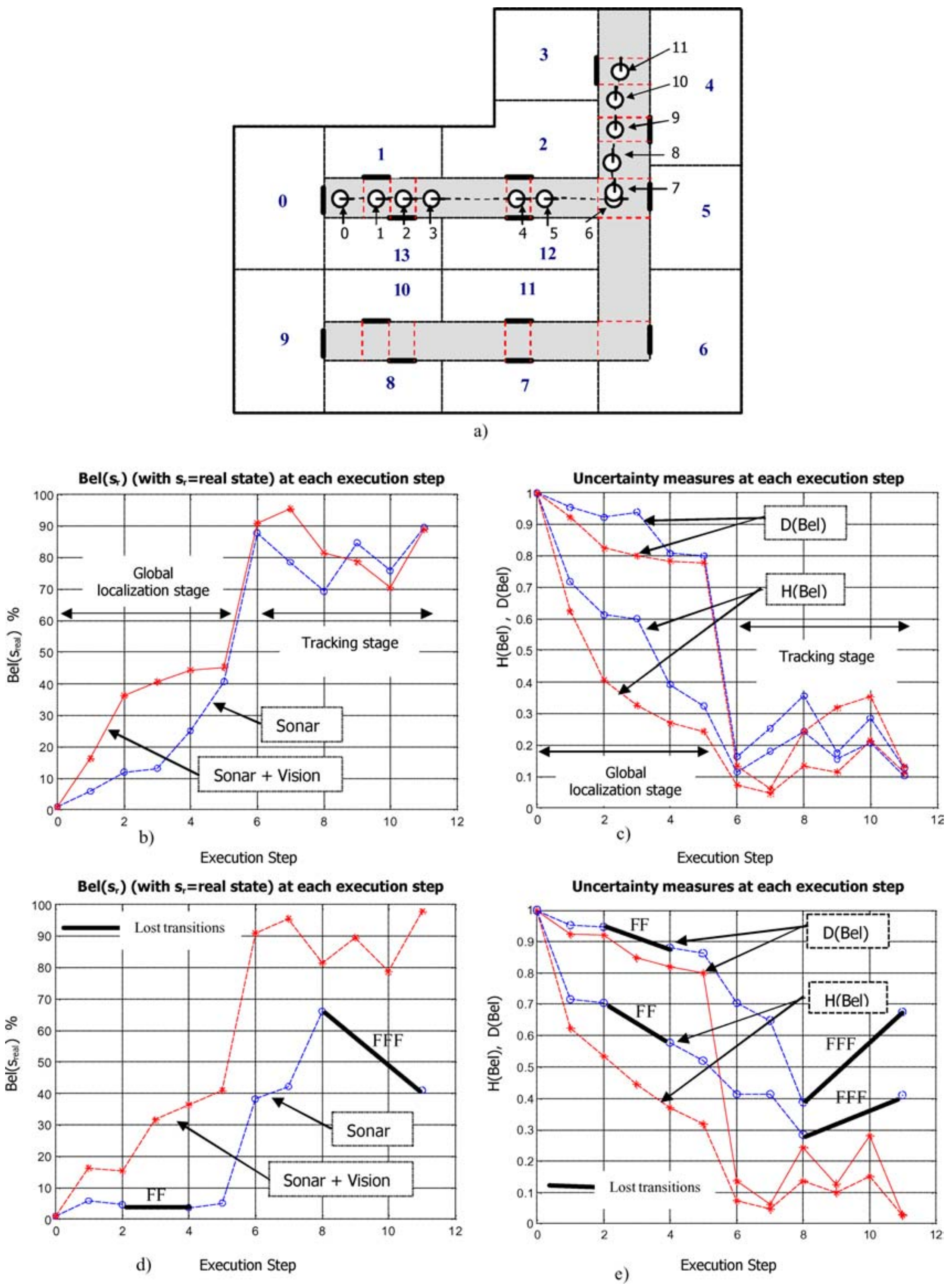
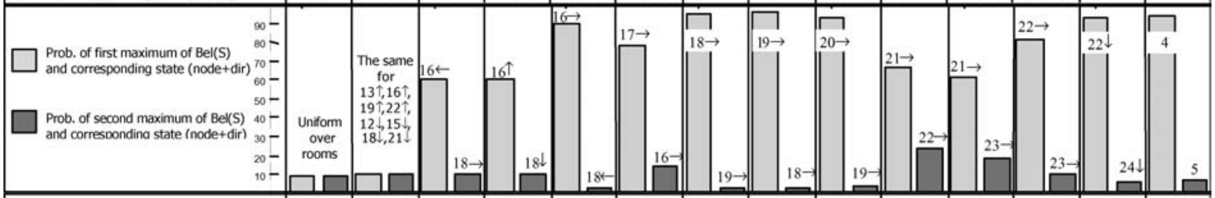


Figure 11. Localization examples. (a) Real position of the robot at each execution step, (b) Bel(s_{real}) with all doors opened, with only sonar (—) and with sonar + vision (- - -), (c) uncertainty measures with all doors opened, (d) the same as (b), but with doors 13, 2, 4 closed, (e) the same as (c) but with doors 13, 2, 4 closed.

Table 3. Real guidance example using the SIRAPEM prototype of Fig. 1 navigating in a corridor of the Electronics Department (map and graph shown in Fig. 12). The robot was initially in room 2 with unknown initial room state, and room 4 was commanded as goal state. Guidance and localization contexts are combined using thresholding method with divergence of probability mass as uncertainty measure. The table shows, for each execution step: the real robot state (indicated by means of node number and direction); the first and second most likely states and divergence of the belief $D(Bel)$; the most voted action in guidance context and the divergence of its probability mass distribution $D(V)$ (when the last one is higher than 0.5, the most voted action of localization context is used); the action command selected at each process step and the real effect (transition) of actions from step to step.

Process execution step	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
Real robot state (node + dir)	2	16↑	16←	16↑	16→	17→	18→	19→	20→	21→	21→	22→	22↓	4
														
Divergence of Bel(S) (D(Bel))	0.961	0.940	0.453	0 .453	0.113	0.290	0.067	0.055	0.082	0.453	0.473	0.231	0.100	0.097
Most voted action in GUIDANCE context (and votes in %)	O 91 %	L 51 %	R 80 %	R 80 %	F 95 %	F 98 %	F 100 %	F 100 %	F 100 %	F 74 %	F 62 %	R 97 %	E 93 %	N 94 %
Divergence of V(a) (D(V))	0.148	**0.801**	0.327	0.327	0.081	0.032	0	0	0	0.414	**0.621**	0.049	0.114	0.098
Context (LOCALIZ. if D(V)>0.5)	GUIDE	LOCAL	GUIDE	GUIDE	GUIDE	GUIDE	GUIDE	GUIDE	GUIDE	GUIDE	LOCAL	GUIDE	GUIDE	GUIDE
Most voted action in LOCALIZ. context if activated (votes in %)		L 62 %									F 67 %			
Action command selected	O	L	R	R	F	F	F	F	F	F	F	R	E	N
Real effect (transition) of action	O	L	R	R	F	F	F	F	F	N	F	R	E	

90
80
70
60
50
40
30
20
10
Prob. of first maximum of Bel(S) and corresponding state (node+dir)
Prob. of second maximum of Bel(S) and corresponding state (node+dir)
Uniform over rooms
The same for 13↑,16↑, 19↑,22↑, 12↓,15↓, 18↓,21↓
16←
18→
16↑
18↓
16→
18←
17→
16→
18→
19→
19→
18→
20→
19→
21→
22→
21→
23→
22→
23→
22↓
24↓
4
5

correct room) after repeating each experiment a number of times. Methods combining guidance and localization contexts are clearly better, because they direct the robot to node 20 before acting to reach the destination, eliminating location uncertainty, whereas using only guidance context has a unpredictable final state between rooms 2 and 11. On the other hand, using the divergence factor proposed in this work, instead of entropy, improves the probability of reaching the correct final state, because it better detects the convergence to a unique maximum (global localization).

8. Real Robot Results

Finally, Table 3 shows a real guidance example using the SIRAPEM prototype of Fig. 1 navigating in a corridor of the Electronics Department (map and graph shown in Fig. 12). The robot was initially in room 2 with unknown initial room state, and room 4 was commanded as goal state. In this example, guidance and localization contexts are combined using thresholding method with divergence of probability mass as uncertainty measure. Table 3 shows, for each execution step, the real robot state (indicated by means of node number and direction), the first and second most likely states and divergence of the belief *D(Bel)*. It also shows the most voted action in guidance context and the divergence of its probability mass distribution *D(V)*. When the last one is higher than 0.5, the most voted action of localization context is used. Finally, it shows the action command selected at each process step and the real effect (transition) of actions from step to step.

It can be seen that after going out the room, localization context is activated and the robot turns left, in the opposite direction of the destination, but to the best direction to reduce uncertainty. After this movement, uncertainty is reduced, and starts the movement to room 4. The trajectory shown with dotted line in Fig. 12 was obtained from odometry readings, and shows the real movement of the robot. As a global conclusion, divergence factor and context combination reduces the number of steps the robot is "lost", and so the goal reaching time.

9. Discussion and Future Work

The proposed navigation system, based on a topological representation of the world, allows the robot to

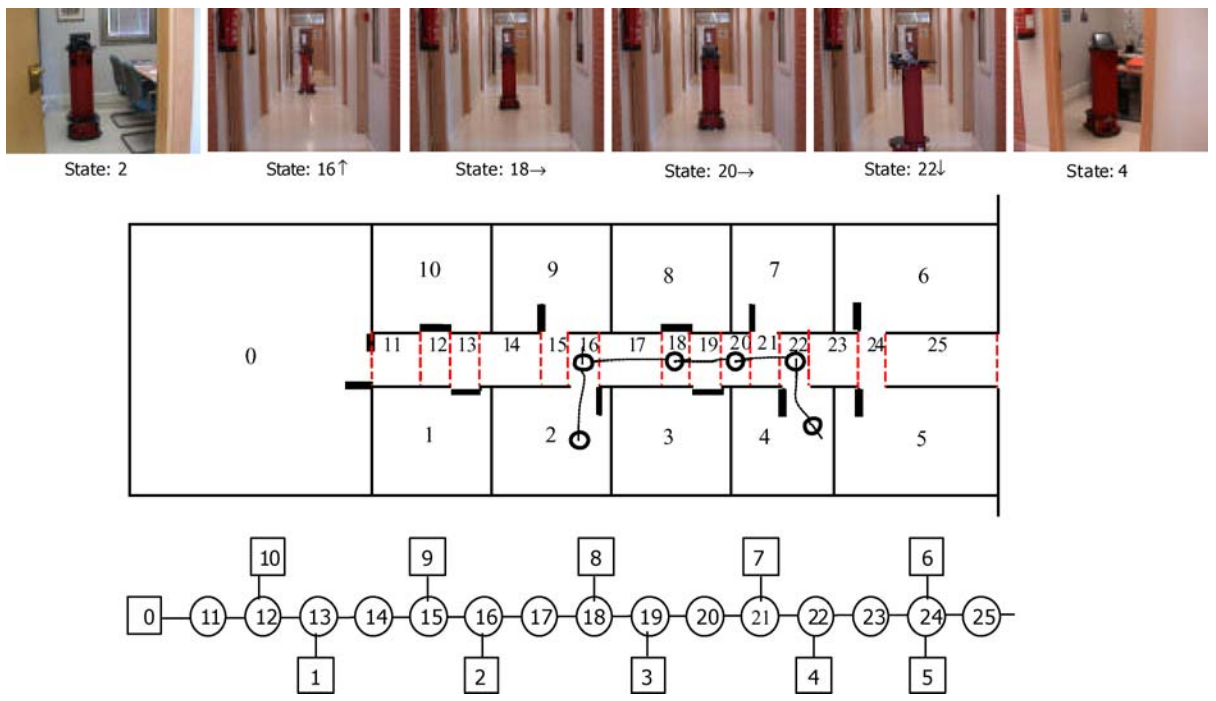


Figure 12. Topological graph model for a corridor of the Electronics Department, and executed trajectory from room 2 to room 4 (process evolution shown in Table 3).

robustly navigate in corridor and structured environments. This is a very practical issue in assistance applications, in which robots must perform guidance missions from room to room in environments typically structured in corridors and rooms, such as hospitals or nursing homes. Although the topological map consists of very simple and reduced information about the environment, a set of robust local navigation behaviors (the actions of the model) allow the robot to locally move in corridors, reacting to sensor information and avoiding collisions, without any previous metric information.

This topological POMDP-based method presents several advantages regarding other efficient probabilistic navigation techniques based on metric representations of the environment (Gutmann et al., 1998), such as particle filtering (also known as Monte Carlo filtering) (Fox et al., 1999). In corridor environments, that can be easily modelled with a topological description, the hard introduction (or learning) of metric maps is avoided. Besides, the level of locational resolution needed for navigation in these environments doesn't justify using a high number of states (particles) corresponding to metric positions of the robot along the corridors. Some previous experiments with particle filters in the same corridor environment of the Electronics Department of the University of Alcalá (López et al., 2002) show that the number of states can be reduced from thousands of particles to 71 topological states without worsening the navigation capabilities of the system. Besides, planning actions is much easier using a topological discretization of the environment such as that proposed in this work.

Another important subject in robot navigation is robustness in dynamic environments. It is demonstrated that topological representations are more robust to dynamic changes of the environment (people, obstacles, doors state, etc.) because they are not modelled in the map. In this case, in which local navigation is also based on an extracted local model of the corridor, the system is quite robust to people traversing the corridor. People are another source of uncertainty in actions and observations, which is successfully treated by the probabilistic transition and observation models. Regarding doors state, the learning module (López et al., 2004) adapts the probabilities to its real state, making the system more robust to this dynamic aspect of the environment.

In order to improve the navigation capabilities of the proposed system, we are working on several future work lines. The first one is to enlarge the action and observation sets to navigate in more complex or generic environments. For example, to traverse large halls or unstructured areas, a "wall-following" or "trajectory-following" action would be useful. Besides, we are also working on the incorporation of new observations from new sensors, such as a compass (to discriminate the four orientations of the graph) and a wireless signal strength sensor. Enlarging the model doesn't affect the proposed global navigation algorithms. Regarding the learning system, future work is focused on automatically learning the POMDP structure from real data, making even easier the installation process.

10. Conclusion

This paper shows a new navigation architecture for acting in uncertain domains, based on a POMDP model incorporating simple visual information. This new sensor provides better information to state transition and observation models, making possible a faster global localization when the initial position of the robot is unknown and a more robust navigation.

This paper also shows a new planning architecture for acting in uncertain domains. Instead of using POMDP exact solutions, we propose an alternative two-level layered architecture that simplifies the selection of the final action, combining several planning objectives. As local policies we propose a guidance context, whose objective is to reach the goal, and a localization context to reduce location uncertainty when necessary. As global policies, we have adopted some heuristic strategies proposed in previous works. We have demonstrated the validity of this architecture in highly aliased environments, in which the combination of the two contexts improves the robustness of the planning system, and in a real environment using the robot prototype of the SIRAPEM project. We also introduce a new uncertainty measure that better detects the convergence to a unique maximum that the typical entropy.

Acknowledgments

The authors wish to acknowledge the contribution of the Ministerio de Ciencia y Tecnología (MCyT) for SIRAPEM project financing (DPI2002-02193).

References

ActivMedia Robotics. 2003. PeopleBot Operations Manual.

Bellman, R. 1957. *Dynamic Programming*. Princeton University Press.

Brants, T. 1996. "Estimating Markov models structures". In *Proc. Fourth Conference on Spoken Language Processing*.

Burgard, W., Fox, D., Henning, D., and Schmidt, T. 1996. Estimating the absolute position of a mobile robot using position probability grids. In *Proc. of Thirteen National Conference on Artificial Intelligence.*

Cassandra, A. 1994. Optimal policies for partially observable Markov decision processes. Technical Report CS-94-14, Brown University, Department of Computer Science, Providence, RI.

Fox, D. 1998. Active Markov Localization for Mobile Robots. *Robotics and Autonomous System* (special issue Euromicro'97).

Fox, D., Burgard W., Dellaert, F. and Thrun, S. 1999. Monte Carlo Localization: Efficient position estimation for mobile robots. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI'99).*

Gechter, F., Thomas V., and Charpillet, F. 2001. Robot localization by stochastic vision based device. *The 5^{th} World Multi-Conference on Systemics, Cybernetics and Informatics, SCI 2001.*

Goldberg, D., and Mataric, M.J. 1999. Augmented Markov models. USC Institute for Robotics and Intelligent Systems Technical Report IRIS-99-367.

Gutmann, J.S., Burgard, W., Fox, D., and Konolige, K. 1998. An experimental comparison of localization methods. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'98).*

Haigh, K.Z., Phelps, J., and Geib, C.W. 2002. An open agent architecture for assisting elder independence. In the*First International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pp. 578–586.

Howard, R.A. 1960. Dynamic Programming and Markov Processes. The MIT Press: Cambridge, Massachusetts.

Kaelbling, L.P., Cassandra, A.R., and Kurien, J.A. 1996. Acting under uncertainty: Discrete bayesian models for mobile robot navigation. In *Proc. of the IEEE/RSJ International Conference on Itelligent Robots and Systems.*

Kaelbling, L.P., Littman, M.L., and Cassandra, A.R. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101: pp. 99–134.

Koenig, S. and Simmons, R. 1998. Xavier: A robot navigation architecture based on partially observable Markov decision process models. *Artificial Intelligence and Mobile Robots*, pp. 91–122.

Konolige, K. 1997. Improved occupancy grids for map building. *Autonomous Robots* 4(4):351–367.

Konolige, K., and Myers, K. 1998. The Saphira architecture for autonomous mobile robots. In *AI and Mobile Robots*, Kortenkamp et al. (Eds.) MIT Press, pp. 211–242.

Lay, K., Prassler, E., Dillmann, R., Grunwald, G., Hägele, M., Lawitzky, G., Stopp, A., and von Seelen, W. 2001. MORPHA: Communication and interaction with intelligent, anthropomorphic robot assistants. In *Tagungsband Statustage Leitprojekte Mensch-Technik-Interaktion in der Wissensgesellschaft*.

Littman, L. 1994. The witness algorithm: Solving partially observable Markov decision processes. Technical Report CS-94-04, Brown University, Department of Computer Science, Providence, RI.

López, E., Barea, R., Bergasa, L.M., and Escudero, M.S. 2002. Aplicación del método de Markov a la localización de un robot móvil en entornos interiores. In *Proceedings of the "Seminario Anual de Automática, Electrónica Industrial e Instrumentación" (SAAEI'02)*, vol. 1, pp. 339-342.

López, E., Barea, R., Bergasa, L.M., and Escudero, M.S. 2004. A human-robot cooperative learning system for easy installation of assistant robots in new working environments. *Journal of Intelligent and Robotic Systems*, 40:233–265.

Lovejoy, W.S. 1991. A survey of algorithmic methods for partially observed Markov decision processes. *Annals of Operations Research*, 28(1):47–65.

Montemerlo, M., Pineau, J., Roy, N., Thrun, S., and Verma, V. 2002. Experiences with a mobile robotic guide for the elderly. In *Proc. of the AAAI National Conference on Artificial Intelligence*. Edmonton, Canada.

Nourbakhsh, I., Powers, R., and Birchfield, S. 1995. DERVISH: An office navigating robot. *Artificial Intelligence Magazine*, 16(2).

Papadimitriou, C., Tsitsiklis, J. 1987. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3): 441–450.

Pineau, J., Gordon, G., and Thrun, S. 2003. Point-based value iteration: An anytime algorithm for POMDPs. *International Joint Conference on Artificial Intelligence (IJCAI)*. Acapulco, Mexico, pp. 1025–1032.

Pineau, J. and Thrun, S. 2002. An integrated approach to hierarchy and abstraction for POMDPs, Technical Report CMU-RI-TR-02-21, Carnegie Mellon University, Pittsburgh, PA.

Puterman, M.L. 1994. *Markov Decision Processes—Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc.: New York, NY.

Regini, L., Tascini, G., and Zingaretti, P. 2002. Appearance-based Robot Navigation. Artificial Intelligence and Intelligent Agents AIIA 2002.

Roy, N. 2003. Finding aproximate POMDP solutions through belief compression. PhD Thesis, CMU-RI-TR-03-25. Robotics Insititute, Carnegie Mellon University, 2003.

Roy, N., Baltus, G., Fox, D., Gemperle, F., Goetz, J., Hirsch, T., Magaritis, D., Montemerlo, M., Pineau, J., Schulte, J., and Thrun, S. 2000. Towards personal service robots for the elderly. In *Proc. of the Workshop on Interactive Robotics and Entertainment (WIRE)*, Pittsburgh, PA.

Saffiotti, A. 1994. Pick-up what? In *Current Trends in AI Planning*, X. Bäckström and E. Sandewall (Eds.), IOS Press: Amsterdam, Netherlands.

Saffiotti, A., Ruspini, E.H., and Konolige, K. 1993. Integrating reactivity and goal-directedness in a fuzzy controller. In *Proceedings of the 2nd Fuzzy IEEE Conference.*

Theocharous, G. and Mahadevan, S. 2002. Approximate planning with hierarchical partially observable markov decision processs for robot navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA).*

Thrun, S. 2002. Probabilistic algorithms in robotics. Technical Report CMU-CS-00-126. Carnegie Mellon University.

Winston, P.H. 1984. *Artificial Intelligence*. Addison-Wesley.

Yairi, T., Togami, M., and Hori, K. 2003. Learning topological structures of pomdp-based state transition models by state splitting method. In *Proceedings of the 21th IASTED International Conference on Applied Informatics*, pp. 7–12.

Zanichelli, F. 1999. Topological maps and robust localization for autonomous navigation. *IJCAI Workshop on Adaptive Spatial Representations of Dynamic Environments*.

María Elena López received her Ph.D. degree from University of Alcalá (Spain) in 2004. She also obtained the Graduete Level in Telecommunications and Engineering degree (MSc) in Electronics from University of Alcalá in 1994 and 1999 respectively. She has been a Lecturer in the Electronics Department at the University of Alcalá since 1995. Her research focus on intelligent control and artificial vision for robotics applications. She has numerous publications in international journals and conference proceedings about these research lines.

Luis Miguel Bergasa received the M.S. degree in Telecommunication Engineering in 1995 from the Technical University of Madrid, Spain and the Ph.D. degree in Telecommunications Engineering in 1999 from the University of Alcalá, Spain. He is currently an Associate Professor at the Department of Electronics of the University of Alcalá. His research interests include real-time computer vision and its applications, particularly in the field of the robotics and the assistance systems for elderly, disable people and drivers. He is the author of more than 50 publications in international journals, book chapters and conference proceedings.

Rafael Barea received a Ph.D. degree in Telecommunications from University of Alcalá in 2001, a M. S. degree in Telecommunications from the Polytechnic University of Madrid in 1997 and a B. S. degree in Telecommunications Engineering with first Class Honours from University of Alcalá in 1994. He has been a Lecturer in the Electronics Department at the University of Alcalá since 1994. His research interest include bioengineering, medical instrumentation, personal robotic aids, computer vision, system control and neural networks. He is the author of numerous refereed publications in international journals, book chapters and conference proceedings.

María Soledad Escudero obtained her electronics engineering degree (graduate level) from the University of Alcalá (Spain) in 1994 and her telecommunications engineering degree (M.Sc.) from the University of Valencia (Spain) in 1997. She has been a lecturer at the Electronics Department of the University of Alcalá since 1996. She has worked on several projects in relation to computer vision, robotics and intelligent control.