Journal of Intelligent and Robotic Systems **40:** 233–265, 2004.
© 2004 *Kluwer Academic Publishers. Printed in the Netherlands.*

A Human–Robot Cooperative Learning System for Easy Installation of Assistant Robots in New Working Environments

MARÍA ELENA LÓPEZ, RAFAEL BAREA, LUIS MIGUEL BERGASA and MARÍA SOLEDAD ESCUDERO
Electronics Department, University of Alcalá, Spain

(Received: 8 August 2003; in final form: 11 March 2004)

Abstract. One of the applications of service robots with a greater social impact is the assistance to elderly or disabled people. In these applications, assistant robots must robustly navigate in structured indoor environments such as hospitals, nursing homes or houses, heading from room to room to carry out different nursing or service tasks. Among the main requirements of these robotic aids, one that will determine its future commercial feasibility, is the easy installation of the robot in new working domains without long, tedious or complex configuration steps. This paper describes the navigation system of the assistant robot called SIRA, developed in the Electronics Department of the University of Alcalá, focusing on the learning module, specially designed to make the installation of the robot easier and faster in new environments. To cope with robustness and reliability requirements, the navigation system uses probabilistic reasoning (POMDPs) to globally localize the robot and to direct its goal-oriented actions. The proposed learning module fast learns the Markov model of a new environment by means of an exploration stage that takes advantage of human–robot interfaces (basically speech) and user–robot cooperation to accelerate model acquisition. The proposed learning method, based on a modification of the EM algorithm, is able to robustly explore new environments with a low number of corridor traversals, as shown in some experiments carried out with SIRA.

Key words: probabilistic navigation, partially observable Markov decision processes, learning under uncertainty, expectation–maximization algorithm, assistant robots.

1. Introduction

The world population of people over the age of 65 that can no longer live independently is growing rapidly in the last decades. The great explosion that service robotics have undergone in these last years makes it possible to think in new, alternative ways of providing care to this sector of the population. A meaningful indicator of the growing interest in these new technologies is the number of projects and research groups currently working in the development of assistant robots, such as the “Nursebot” project, with robots Flo (Roy et al., 2000) and Pearl (Montemerlo et al., 2002), “I.L.S.A (Independent LifeStyle Assistant^{TM})” (Haigh et al., 2002) and “Morpha” (Lay et al., 2001) projects, and other assistance systems such as that developed by Hoppenot (2002) and Morere et al. (2002).

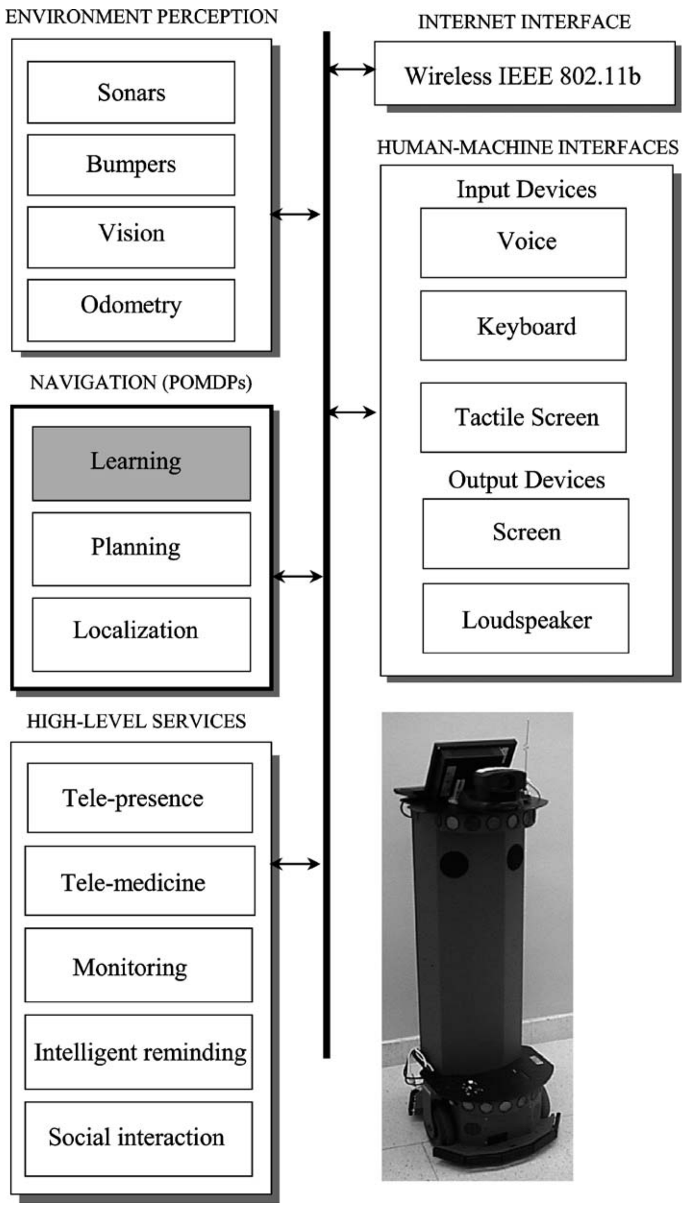


Figure 1. Global architecture of the SIRAPEM project.

The Electronics Department of the University of Alcalá is contributing to this research field with the SIRAPEM Project (Spanish acronym of *Robotic System for Elderly Assistance*), whose goal is the development of a robotic aid that serves primary functions of tele-presence, tele-medicine, 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).

In this kind of care applications, in which the robot must perform tasks in indoor environments for long periods of time, there are a number of requirements the navigation system must carry out:

- 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 is to *simplify the installation process*, in order to use it in new environments (hospitals, houses, etc.) without long or difficult configuration steps. So, it must use a simple environment representation and avoid environment modifications by using natural landmarks that can be easily found in any indoor domain. The incorporation of learning methods is crucial to reduce designer intervention in the installation step, and to ensure a good adaptation of navigation parameters to real environments.
- Localization, planning and learning algorithms must be executed in real robotic platforms, and so real-time execution and limited memory must be taken into account.

A suitable framework to cope with all these requirements is Partially Observable Markov Decision Processes (POMDPs). These models use probabilistic reasoning (Thrun, 2000) to deal with uncertainties, and a topological representation of the environment to reduce memory and time requirements of the algorithms. For the proposed global navigation system, in which the objective is the guidance to a goal room and some low-level behaviors perform local navigation, a topological discretization is appropriate to facilitate the planning and learning tasks.

POMDP 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 of SIRA (see Figure 1). The main contributions of the navigation architecture of SIRA, regarding other similar ones (that we'll be referenced in next subsection), are the following:

- Addition of visual information to the Markov model, not only as observations, but also for improving state transition detection. This visual information reduces typical perceptual aliasing of proximity sensors, accelerating the process of global localization when initial pose is unknown.
- Development of a new planning architecture that selects actions to combine several objectives, such as guidance to a goal room, localization to reduce uncertainty and environment exploration.

- Development of a new exploration and learning strategy that takes advantage of human–machine interaction to robustly and quickly learn new working environments.

The first two contributions have already been shown in previous publications (López et al., 2003a, 2003b). This paper focuses on the learning module, that adjusts the probabilities of the initial Markov model, firstly by a user-supervised active exploration of corridors, and later by passively observing the robot's interactions with its environment. The proposed algorithm is a modification of the EM algorithm (that we call "EM with Certainty Break Points", EM-CBP) that exploits specific POMDP structure and some "certainty points" introduced by user-supervision to reduce the amount of training data needed to learn good Markov models.

This paper is organized as follows. After placing this work within the context of previous similar ones, a brief overview of POMDP's foundations and EM algorithm is presented as background in Section 2. Section 3 describes the Markov model used in this navigation application. Section 4 describes the main features of the learning system, while Sections 5, 6 and 7 explain the initial POMDP compilation, training data collection, and EM-CBP algorithms respectively. Finally, we show some experimental results, whereas a final conclusion summarizes the paper.

1.1. RELATED PREVIOUS WORK

Markov models, and particularly POMDPs, have 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. Other successful robots guided with POMDPs are those proposed by Zanichelli (1999) or Asoh et al. (1996). 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. Although POMDP navigation has demonstrated to be very robust in these systems, most of them need a previous "hand-made" introduction of the Markov model of the environment.

There are, however, a number of works about topological and probabilistic map learning in robotics navigation. Learning a POMDP involves two main issues: (1) obtaining its topology (structure), and (2) adjusting the parameters (probabilities) of the model. The majority of the works deals with the last problem, using the well-known EM algorithm (known as the Baum–Welch algorithm in the field of Hidden Markov Models) to learn the parameters of a Markov model whose structure is known. Examples of these works are (Shatkay and Kaelbling, 1997; Thrun et al., 1998) or (Koenig and Simmons, 1996). However, because the computational complexity of the learning process increases exponentially as the number of states increases, these methods are still time consuming, and its working ability is limited to reduced environments.

Other works try to obtain the topology (structure) of the POMDP models from experimental data. Brants (1996) proposed the "model merging and splitting" techniques which estimate Markov models structures by successively merging or splitting the states so that it maximally preserves or improves the transition model likelihood. Another recent work in this field, applied to robotic navigation is (Yairi et al., 2003). However, these methods have only been theoretically formulated (they have not been experimentally validated), and the obtained models are not reliable for robot navigation yet.

This paper presents a POMDP model for robot navigation that can be easily obtained for new environments by means of human–robot cooperation, being an optimal solution for assistant robots endowed with human–machine interfaces. There are several previous systems that also include the human operator in the navigation loop. In some cases, it is done by means of a tele-operation of the movements of the robot (Haegele et al., 2001; Hoppenot, 2002; Thrun, Burgard and Fox, 1998). In other systems, the human–robot interaction is introduced in a higher level, in which the user acts as an expert teacher (Asoh et al., 1996) using several communication channels, such as voice or gestures. In this work, we propose a practical solution to learn a new environment in the POMDP framework, in which human–robot cooperation is performed in a very intuitive way that can be carried out by any non-expert user. The topological representation of the environment is intuitive enough to be easily defined by the designer (avoiding unreliability and unstability of the above mentioned methods). The uncertainties and observations that constitute the parameters of the Markov model are learned by the robot using a modification of the EM algorithm that exploits slight user supervision (using the voice interface) and topology constraints to highly reduce memory requirements and computational cost of the standard EM algorithm.

2. Theoretical Background

Although there is a wide literature about POMDPs theory (Papadimitriou and Tsitsiklis, 1987; Puterman, 1994; Kaelbling et al., 1996) and parameter learning methods (Rabiner and Juang, 1986; Russell et al., 1994), in this section some terminology and main foundations are briefly introduced as theoretical background of the proposed work. Firstly, we describe some foundations about Partially Observable Markov Decision Processes (POMDPs), and then we introduce the basis of EM algorithm.

2.1. POMDPs OVERVIEW

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

- $\boldsymbol{S}$ is a finite set of states ($s \in \boldsymbol{S}$).
- $\boldsymbol{A}$ is a finite set of actions ($a \in \boldsymbol{A}$).
- $\boldsymbol{T} = \{p(s'|s, a)\ \forall(s, s' \in \boldsymbol{S}, a \in \boldsymbol{A})\}$ is a state transition model which specifies a conditional probability distribution of posterior state s' given prior state s and action executed a.
- $\boldsymbol{R} = \{r(s, a)\ \forall(s \in \boldsymbol{S}, a \in \boldsymbol{A})\}$ 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:

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

In a MDP, the actual state s is always known without uncertainty. However, Partially Observable Markov Decision Processes (POMDPs) are used under domains where there is not certainty about the actual state of the system. Instead, the agent can do observations and use them to compute a probabilistic distribution over all possible states. So, a POMDP adds:

- $\boldsymbol{O}$, a finite set of observations ($o \in \boldsymbol{O}$),
- $\vartheta = \{p(o|s)\ \forall o \in \boldsymbol{O}, s \in \boldsymbol{S}\}$ 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*, $\mathrm{Bel}(\boldsymbol{S})$. 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. This belief must be updated whenever a new action or perception is carried out. When an action a is executed, the new probabilities become:

$$\mathrm{Bel}_{\mathrm{posterior}}(s') = K \cdot \sum_{s \in S} p(s'|s, a) \cdot \mathrm{Bel}_{\mathrm{prior}}(s), \tag{2}$$

where K is a normalization factor to ensure that the probabilities all sum one. When a sensor report o is received, the probabilities become:

$$\mathrm{Bel}_{\mathrm{posterior}}(s) = K \cdot p(o|s) \cdot \mathrm{Bel}_{\mathrm{prior}}(s). \tag{3}$$

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 perceptions of the environment that 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.

2.2. EM ALGORITHM REVIEW

Learning Markov models of partially observable environments is a hard problem, because it involves inferring the hidden state at each step from observations, as well as estimating the transition and observation models, while these two procedures are mutually dependent.

The EM algorithm (in Hidden Markov Models context known as Baum–Welch algorithm) is an expectation–maximization algorithm for learning the parameters (entries of the transition and observation probabilistic models) of a POMDP from observations (Bilmes, 1997). The input for applying this method is an execution trace, containing the sequence of actions-observations executed and collected by the robot at each execution step $t = 1, \dots, T$ (T is the total number of steps of the execution trace):

$$\textbf{trace} = [o_1, a_1, o_2, a_2, \dots, o_t, a_t, \dots, o_{T-1}, a_{T-1}, o_T]. \tag{4}$$

The EM algorithm is a hill-climbing process that iteratively alternates two steps to converge to a POMDP that locally best fits the trace. In the E-Step (expectation step), probabilistic estimates for the robot states (locations) at the various time steps are estimated based on the currently available POMDP parameters (in the first iteration, they can be uniform matrixes). In the M-Step (maximization step), the maximum likelihood parameters are estimated based on the states computed in the E-step. Iterative application of both steps leads to a refinement of both, state estimation and POMDP parameters.

The E-Step

This is a forward–backward process that, for estimating state s_t at time t, uses previous to t data with a forward state propagation (the result is a probability distribution $\alpha_t(s_t)$), and posterior to t data with a backward propagation (resulting in probability distribution $\beta_t(s_t)$). The combination of these two distributions provides a better estimation of the process state s_t (by means of a final distribution $\gamma_t(s_t) = \alpha_t(s_t) \cdot \beta_t(s_t)$).

So, $\alpha(s)$ distributions are forwarded computed (from $t = 1$ to $t = T$) using the typical Markov localization updates (Equations (2) and (3)):

$$\begin{aligned} \alpha_t(s'_t) &= p(s'_t|o_1, a_1, \dots, o_t) \\ &= \eta \cdot p(o_t|s'_t) \cdot \sum_{\forall s_{t-1} \in S} p(s'_t|s_{t-1}, a_{t-1}) \cdot \alpha_{t-1}(s_{t-1}) \end{aligned} \tag{5}$$

while $\beta(s)$ distributions are backward computed (from $t = T$ to $t = 1$) using a similar based expression:

$$\begin{aligned} \beta_t(s_t) &= p(s'_t|a_t, \dots, o_T) \\ &= \eta \sum_{\forall s_t \in S} p(s'_{t+1}|s_t, a_t) \cdot p(o_{t+1}|s'_{t+1}) \cdot \beta_{t+1}(s'_{t+1}). \end{aligned} \tag{6}$$

Finally, $\gamma(s)$ distributions are computed as the product of $\alpha(s)$ and $\beta(s)$ distributions. Another useful distribution is a bidimensional one that represents at each time t the probability of transition from s_t to s'_{t+1}, known as *transition gamma distribution* $\gamma(s, s')$:

$$\gamma_t(s) = \alpha_t(s) \cdot \beta_t(s), \tag{7}$$

$$\gamma_t(s, s') = \alpha_t(s) \cdot p(s'|s, a_t) \cdot p(o_{t+1}|s') \cdot \beta_{t+1}(s'). \tag{8}$$

The M-Step

This step uses some frequency-counting re-estimation formulas to calculate the improved POMDP (initial state probabilities, and transition and observation probabilities). The overlined symbols represent the probabilities of the improved POMDP:

Initial state distribution re-estimation: $\bar{p}(s_1 = s) = \gamma_1(s)$. (9)

Transition probabilities re-estimation:

$$\bar{p}(s'|s, a) = \frac{\sum_{t=1,\dots,T-1|a_t=a} \gamma_t(s, s')}{\sum_{t=1,\dots,T-1|a_t=a} \gamma_t(s)} \quad \forall s, s' \in \boldsymbol{S},\ a \in \boldsymbol{A}. \tag{10}$$

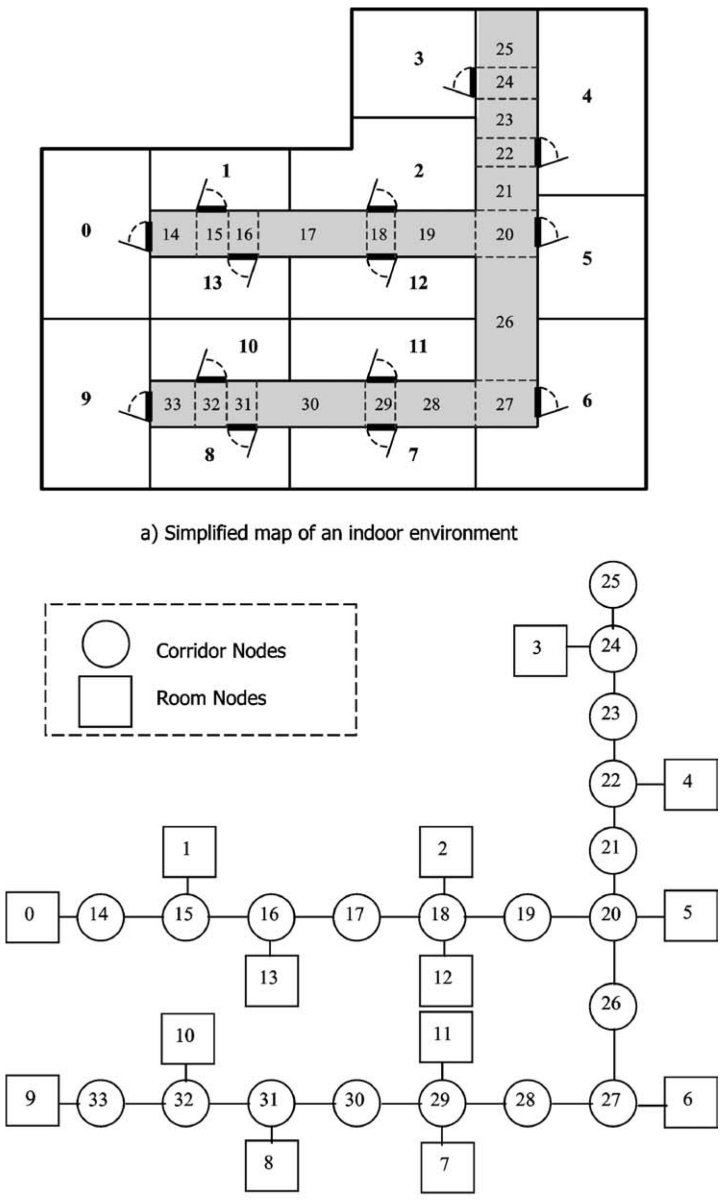
Observation probabilities re-estimation:

$$\bar{p}(o|s) = \frac{\sum_{t=1,\dots,T|o_t=o} \gamma_t(s)}{\sum_{t=1,\dots,T} \gamma_t(s)} \quad \forall s \in \boldsymbol{S},\ o \in \boldsymbol{O}. \tag{11}$$

The limitations of the standard EM algorithm are well known. One of them is that it converges to a local optimum, and so, the initial POMDP parameters have some influence on the final learned POMDP. But the main disadvantage of this algorithm is that it requires a large amount of training data. As the degrees of freedom (settable parameters) increase, so does the need for training data. There are some works proposing alternative approximations of the algorithm to lighten this problem, such as (Koenig and Simmons, 1996) or (Liu et al., 2001). We propose a new method that takes advantage of human–robot interfaces of assistant robots and the specific structure of the POMDP model to reduce the amount of data needed for convergence.

3. Markov Model for Global Navigation

The first step to define a POMDP model for robot global navigation is to establish a topology for mapping the environment. 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 Figure 2



a) Simplified map of an indoor environment

b) Topological graph for the environment of figure a)

Figure 2. Topological graph of an environment map.

for an environment example, 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).

3.1. THE ELEMENTS OF THE MARKOV MODEL: STATES, ACTIONS AND OBSERVATIONS

With this topology, states ($\boldsymbol{S}$) of the 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 following.

The actions ($\boldsymbol{A}$) 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 modeled by the transition model $\boldsymbol{T}$). These actions are:

(1) "*Go out room*" (a_{O}): to traverse door using sonar and 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.

The "Follow Corridor" action has a great influence in the final performance of the localization and planning systems. While following corridors, the robot must detect state transitions to update the belief about its position and select new actions to reach the goal room (López et al., 2003a). To detect these transitions, sonar (that reliably detects transitions with opened doors) and vision (that detects doorframes to estimate the distance to travel to new states) information are combined. So, the color of doors must be trained in each new environment, and its contrast with color of walls directly affects the uncertainty associated with the "Follow Corridor" action.

Finally, the observations ($\boldsymbol{O}$) in our model come from the two sensorial systems of the robot: sonar and vision. In each state, the robot makes three kind 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 Figure 3(a)).
(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 (as it was said, previously trained) and some geometrical restrictions. This observation is the extracted number of doors in lateral walls from the image (see Figure 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

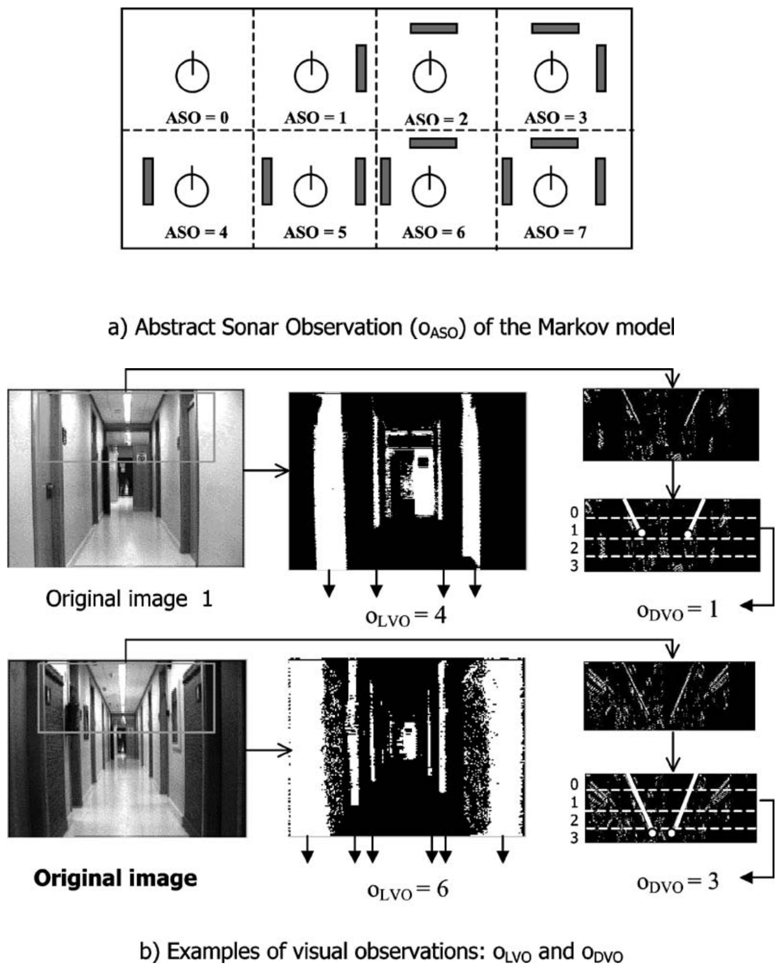


Figure 3. Observations of the proposed Markov model.

from the robot can't be easily segmented from the image (this is the case of image 2 of Figure 3(b)), and this is the reason why we introduce a third visual observation.

(3) "*Depth Visual Observation*" (o_{DVO}). As human–interaction robots have tall bodies with the camera 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 Figure 3(b)). This is a less sensitive to noise observation than using floor vanishing lines (for example, less sensitive to walking people influence), and 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.

As it was demonstrated in a previous publication (López et al., 2003a), the addition of the visual observations 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 distinguishing between these states.

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

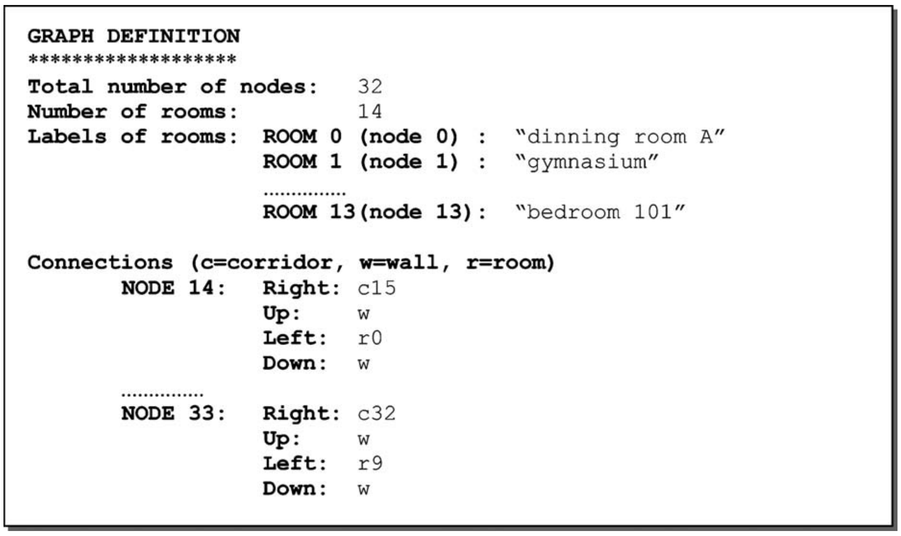
$$p(\mathbf{o}|s) = p(o_{\mathrm{ASO}}, o_{\mathrm{LVO}}, o_{\mathrm{DVO}}|s) = p(o_{\mathrm{ASO}}|s)p(o_{\mathrm{LVO}}|s)p(o_{\mathrm{DVO}}|s). \qquad (12)$$

3.2. ACTION AND OBSERVATION UNCERTAINTIES

Besides the topology of the environment, it's necessary to define some action and observation uncertainties to generate the final POMDP model (transition and observation matrixes). A first way of defining these uncertainties is by introducing some experimental "hand-made" rules (this method is used in (Koenig and Simmons, 1998) and (Zanichelli, 1999)). For example, if a "Follow" action (a_{F}) is commanded, the expected probability of making a state transition (F) is 70%, while there is a 10% probability of remaining in the same state (N = no action), a 10% probability of making two successive state transitions (FF), and a 10% probability of making three state transitions (FFF). Experience with this method has shown it to produce reliable navigation. However, a limitation of this method is that some uncertainties or parameters of the transition and observation models are not intuitive for being estimated by the user. Besides, results are better when probabilities are learned to more closely reflect the actual environment of the robot. So, our proposed learning module adjusts observation and transition probabilities with real data during an initial exploration stage, and maintains these parameters updated when the robot is performing another guiding or service tasks. This module, that also makes easier the installation of the system in new environments, is described in detail in the following sections.

4. Objectives and Overview of the Learning System

As it was said in the last section, the POMDP model is constructed from two sources of information:



```
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 4. Example of graph definition for the environment of Figure 2.

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

As it was indicated in Section 1.1, several recent works 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 4 shows an example of the "Graph definition" application (for the environment of Figure 2), that 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 defined the graph, the objective of the learning module is to adjust the parameters of the POMDP model (entries of transition and observation matrixes).

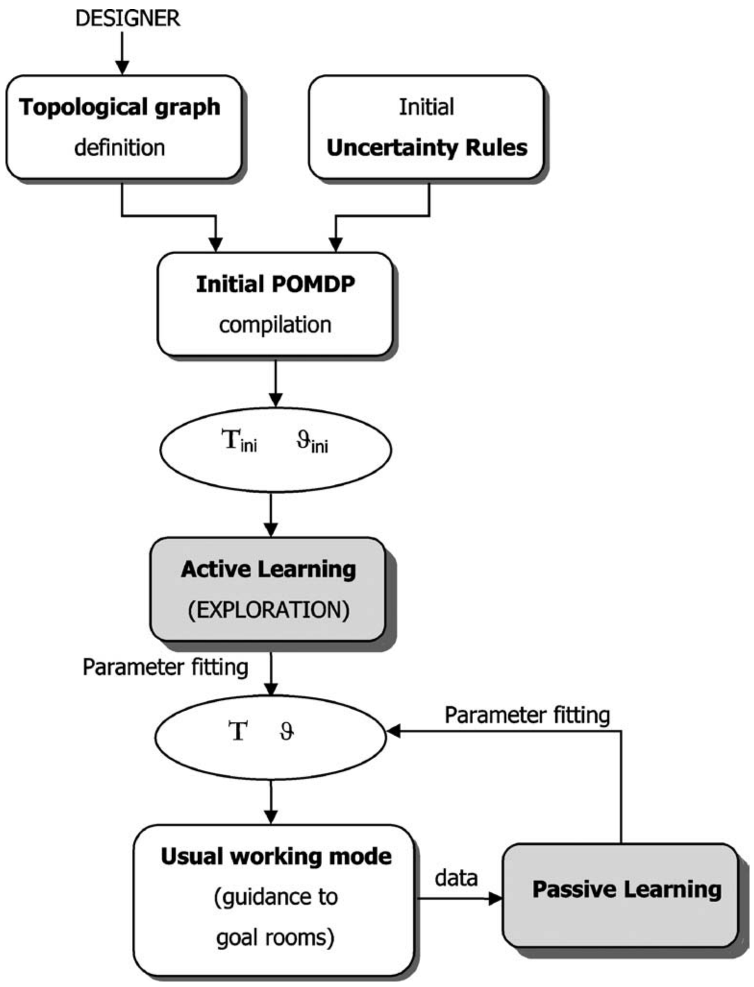

DESIGNER
Topological graph definition
Initial Uncertainty Rules
Initial POMDP compilation
T_{ini} ϑ_{ini}
Active Learning (EXPLORATION)
Parameter fitting
T ϑ
Parameter fitting
Usual working mode (guidance to goal rooms)
data
Passive Learning

Figure 5. Steps for the introduction and learning of the Markov model of a new environment.

Figure 5 shows the steps involved in the POMDP generation of a new working environment. The graph introduced by the designer, together with some predefined initial uncertainty rules are used to generate an initial POMDP. This initial POMDP, described in next section, provides enough information for corridor navigation during an exploration stage, whose objective is to collect data in an optimum manner to adjust the settable parameters (no the whole POMDP, as it will be also justified in next section) with minimum memory requirements and ensuring a reliable convergence of the model to fit real environment data (this is the *active learning* stage). Besides, during normal working of the navigation system (performing guiding tasks), the learning module carries on working (*passive learning* stage), collecting actions and observations to maintain the parameters updated in the face of possible changes.

5. Settable Parameters and Initial POMDP Compilation

A method used to reduce the amount of training data needed for convergence of the EM algorithm is to limit the number of model parameters to be learned. There are two reasons because some parameters can be excluded off the training process:

- Some parameters are only robot dependent, and don't change from one environment to another. Examples of this case are the errors in turn actions (that are nearly deterministic due to the accuracy of odometry sensors in short turns), or errors of sonars detecting "free" when "occupied" or vice versa.
- Other parameters directly depend on the graph and some general uncertainty rules, being possible to learn the general rules instead of its individual entries in the model matrixes. This means that the learning method constrains some probabilities to be identical, and updates a probability using all the information that applies to any probability in its class. For example, the probability of losing a transition while following a corridor can be supposed to be identical for all states in the corridor, being possible to learn the general probability instead of the particular ones.

Taking these properties into account, Table I shows the uncertainty rules used to generate the initial POMDP in the SIRAPEM system. Figure 6 shows the process of initial POMDP compilation. Firstly, the 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 nodes):

Room states: $n_s = n$.

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

Finally, the compiler generates the initial transition and observation matrixes using the predefined uncertainty rules. Settable parameters are shown over gray background in Figure 6, while the rest of them will be excluded of the training process. The choice of settable parameters is justified in the following way:

(a) *Transition probabilities*. Uncertainties for actions "Turn Left" (a_L), "Turn Right" (a_R), "Go out room" (a_O) and "Enter room" (a_E) depends on odometry and the developed algorithms, and can be considered environment independent. However, the "Follow corridor" (a_F) action highly depends on the ability of the vision system to segment doors color, that can change from one environment to another. As a pessimistic initialization rule, we use a 70% probability of making the ideal "follow" transition (F), and 10% probabilities for autotransition (N), and two (FF) or three (FFF) successive transitions, while the rest of possibilities are 0. However, these probabilities will be adjusted by the learning system to better fit real environment conditions. In this case, instead of learning each individual transition probability, the general rule (values for N, F, FF and FFF) will be trained (so, transitions that initially are 0 will be kept unchanged).

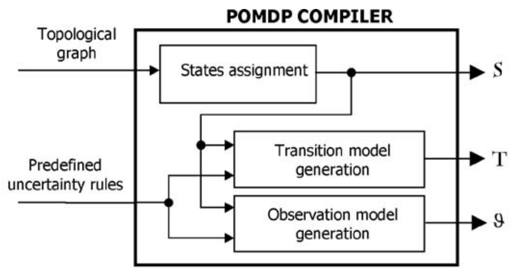
Table I. Predefined uncertainty rules for constructing the initial POMDP model

TRANSITION MODEL UNCERTAINTIES (F=Follow, L=Left, R=Right, O=Out, E=Enter,N=No action)				
Command	**Effect of Command (% probabilities)**			
$\mathbf{a_F}$	**N** = 10	**F** = 70	**FF** = 10	**FFF**= 10
$\mathbf{a_L}$	**N** = 5	**L** = 90	**LL** = 5	
$\mathbf{a_R}$	**N** = 5	**R** = 90	**RR** = 5	
$\mathbf{a_O}$	**N** = 5	**O** = 85	**OF** = 10	
$\mathbf{a_E}$	**N** = 10	**E** = 90		

OBSERVATION MODEL UNCERTAINTIES	
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	
Room states	Uniform distribution over o_{LVO}={0,1,2}
Corridor states perpendicular to corridor direction and oriented to a door	Uniform distribution over o_{LVO}={0,1,2}
Corridor states perpendicular to corridor direction and oriented to a wall	Uniform distribution over o_{LVO}={0,1}
Corridor states aligned with corridor direction	Uniform distribution over o_{LVO}
DVO Model	
Room states	Delta distribution in o_{DVO}=0
Corridor states perpendicular to corridor direction	Delta distribution in o_{DVO}=0
Corridor states aligned with corridor direction	Uniform distribution over o_{DVO}

The new learned values are used to recompile the rows of the transition matrix corresponding to corridor states aligned with corridor directions (the only ones in which the “Follow Corridor” action is defined).

(b) *Observation probabilities*. The Abstract Sonar Observation can be derived from the graph, the state of doors, and a model of the sonar sensor characterizing its probability of perceiving “occupied” when “free” or vice versa. The last one is no environment dependent, and the state of doors can change with high frequency. So, the initial model contemplates a 50% probability for states “closed” and “opened” of all doors. During the learning process, states containing doors will be updated to provide the system with some memory about past state of doors. Regarding the visual observations, it’s obvious that they are not intuitive for being predefined by the user or deduced from the graph. So, in corridor states aligned with corridor direction, the initial model for both visual observations consists of a uniform distribution, and the probabilities will be later learned from robot experience during corridor following in the exploration stage.


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

a_o=Go Out Room

s \ s'	0	1	2	3	4	5	6	7	8	...
0	5	0	0	0	0	0	0	0	0	
1	0	5	0	0	0	0	0	0	0	
2	0	0	5	0	0	0	0	0	0	
3	0	0	0	5	0	0	0	0	0	
4	0	0	0	0	5	0	0	0	0	...
5	0	0	0	0	0	5	0	0	0	
6	0	0	0	0	0	0	5	0	0	
7	0	0	0	0	0	0	0	5	0	
8	0	0	0	0	0	0	0	0	5	
...					...					

a_E=Enter Room

s \ s'	0	1	2	3	4	5	6	7	8	...
...					...					
16	90	0	0	0	0	0	0	0	0	
17	0	0	0	0	0	0	0	0	0	
18	0	0	0	0	0	0	0	0	0	
19	0	90	0	0	0	0	0	0	0	...
20	0	0	0	0	0	0	0	0	0	
21	0	0	0	0	0	0	0	0	0	
22	0	0	0	0	0	0	0	0	0	
23	0	0	0	0	0	0	0	0	0	
...					...					

a_F=Follow Corridor

s \ s'	...	22	23	24	25	26	27	28	29	...
...					...					
22		10	0	0	0	70	0	0	0	
23		0	0	0	0	0	0	0	0	
24		0	0	10	0	0	0	0	0	
25	...	0	0	0	0	0	0	0	0	...
26		0	0	0	0	10	0	0	0	
27		0	0	0	0	0	0	0	0	
28		0	0	70	0	0	0	10	0	
29		0	0	0	0	0	0	0	0	
...					...					

a_L=Turn Left

s \ s'	...	14	15	16	17	18	19	20	21	...
...					...					
14		5	90	5	0	0	0	0	0	
15		0	5	90	5	0	0	0	0	
16		5	0	5	90	0	0	0	0	
17	...	90	5	0	5	0	0	0	0	...
18		0	0	0	0	5	90	5	0	
19		0	0	0	0	0	5	90	5	
20		0	0	0	0	5	0	5	90	
21		0	0	0	0	90	5	0	5	
...					...					

a_R=Turn Right

s \ s'	...	14	15	16	17	18	19	20	21	...
...					...					
14		5	0	5	90	0	0	0	0	
15		90	5	0	5	0	0	0	0	
16		5	90	5	0	0	0	0	0	
17	...	0	5	90	5	0	0	0	0	...
18		0	0	0	0	5	0	5	90	
19		0	0	0	0	90	5	0	5	
20		0	0	0	0	5	90	5	0	
21		0	0	0	0	0	5	90	5	
...					...					

$p(o_{ASO} \mid s)$

s \ ASO	0	1	2	3	4	5	6	7
...				...				
14	0	4	0	0	4	83	0	9
15	2	0	42	4	2	0	46	4
16	0	2	0	2	2	43	2	49
17	2	2	42	46	0	0	4	4
18	2	42	0	4	2	46	0	4
19	40	4	44	4	4	0	4	0
20	2	2	0	0	42	46	4	4
21	4	0	78	9	0	0	9	0
...				...				

$p(o_{LVO} \mid s)$

s \ LVO	0	1	2	3	4	5	6
...				...			
14	16	16	16	16	16	16	160
15	50	50	0	0	0	0	0
16	16	16	16	16	16	16	160
17	33	33	33	0	0	0	0
18	16	16	16	16	16	16	160
19	330	33	33	0	0	0	0
20	16	16	16	16	16	16	160
21	50	50	0	0	0	0	0
...				...			

$p(o_{DVO} \mid s)$

s \ DVO	0	1	2	3
...				
14	25	25	25	25
15	100	0	0	0
16	25	25	25	25
17	100	0	0	0
18	25	25	25	25
19	100	0	0	0
20	25	25	25	25
21	100	0	0	0
...				

Figure 6. Initial POMDP compilation, and structure of the resulting transition and observation matrixes. Parameters over gray background will be adjusted by the learning system.

As a resume, the parameters to be adjusted by the learning system are:

- the general rules N, F, FF and FFF for the "Follow Corridor" action. Their initial values are shown in Table I;

- the probabilities for the Abstract Sonar Observation of corridor states in which there is a door in left, right or front directions (to endow the system with some memory about past door states, improving the localization system results). Initially, it's supposed a 50% probability for "opened" and "closed" states. In this case, the adjustment will use a low gain because the state of doors can change with high frequency;
- the probabilities for the Landmark Visual Observation and Deep Visual Observation of corridor states aligned with corridor direction, that are initialized as uniform distributions.

6. Training Data Collection

As it has been said, the main disadvantage of the EM algorithm is the large amount of data required for convergence, that is conditioned by the limited memory resources of the on-board robot computer. Models with a high number of states (such as models used in navigation systems) need large amounts of memory, not only to store training data, but also the *alfa*, *beta* and *gamma* distributions used by the algorithm. Besides, in order to the algorithm to converge properly, and taking into account that EM is in essence a frequency-counting method, the robot needs to traverse several times the whole environment to obtain the training data. Given the relative slow speed at which mobile robots can move, it's desirable that the learning method learns good POMDPs with as few corridor traversals as possible.

To reduce the memory requirements, we take advantage of the strong topological restrictions of our POMDP model in two ways:

- All the parameters to be learned (justified in the last section) can be obtained during corridor following by sequences of "Follow Corridor" actions. So, it's not necessary to alternate other actions in the execution traces, apart from turn actions needed to start the exploration of a new corridor (that in any case will be excluded off the execution trace).
- States corresponding to different corridors (and different directions within the same corridor) can be broken up from the global POMDP to obtain reduced sub-POMDPs. So, a different execution trace will be obtained for each corridor and each direction, and only the sub-POMDP corresponding to the involved states will be used to calculate de EM algorithm, reducing in this way the memory required to store training data, and *alfa*, *beta* and *gamma* matrixes.

As it was shown in Figure 5, there are two learning modes, that also differ in the way in which data is collected: the active learning mode during an initial exploration stage, and the passive learning mode during normal working of the navigation system. They are explained in the following subsections:

(a) Supervised Active Learning. Corridors Exploration

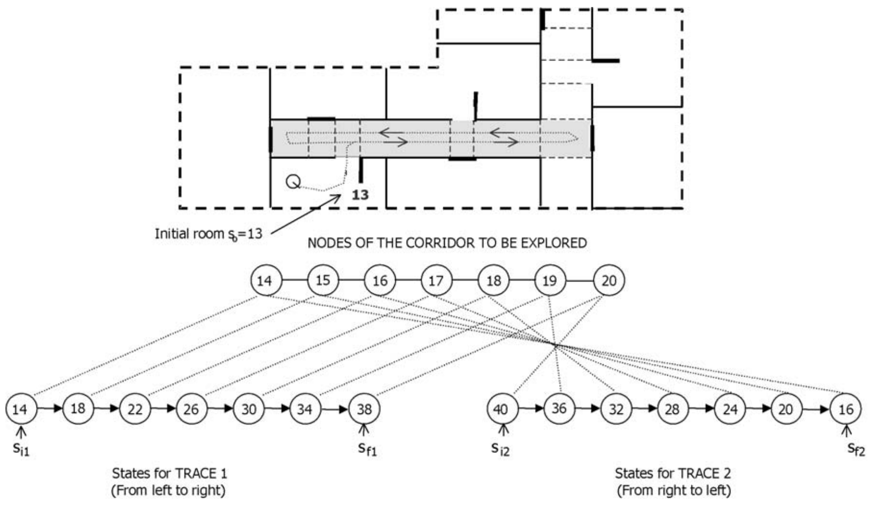
The objective of this exploration stage is to obtain training data in an optimized way to facilitate the initial adjustment of POMDP parameters, reducing the amount of data of execution traces, and the number of corridor traversals needed for convergence. The distinctive features of this exploration process are:

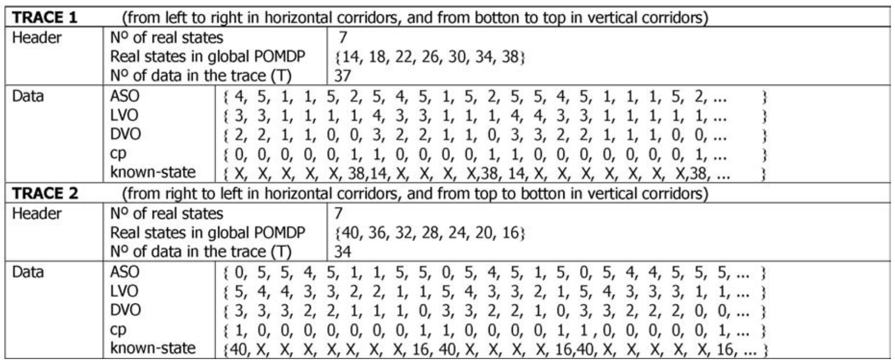
- The objective of the robot is to explore (*active learning*), and so, it independently moves up and down each corridor, collecting a different execution trace for each direction. Each corridor is traversed the number of times needed for the proper convergence of the EM algorithm (in the results section it will be demonstrated that the number of needed traversals ranges from 3 to 5).
- We introduce some *user supervision* in this stage, to ensure and accelerate convergence with a low number of corridor traversals. This supervision can be carried out by a non-expert user, because it consists in answering some questions the robot formulates during corridor exploration, using the speech system of the robot. To start the exploration, the robot must be placed in any room of the corridor to be explored, whose label must be indicated with a talk as the following:

Robot:	*I'm going to start the exploration. ¿Which is the initial room?*
Supervisor:	*dinning room A*
Robot:	*Am I in dinning room A?*
Supervisor:	*yes*

  With this information, the robot initializes its belief **Bel**(S) as a delta distribution centered in the known initial state. As the initial room is known, states corresponding to the corridor to be explored can be extracted from the graph, and broken up from the general POMDP as it's shown in Figure 7. After executing an "Out Room" action, the robot starts going up and down the corridor, collecting the sequences of observations for each direction in two independent traces (trace 1 and trace 2 of Figure 7). Taking advantage of the speech system, some "certainty points" (CPs) are introduced in the traces, corresponding to initial and final states of each corridor direction. To obtain these CPs, the robot asks the user *"Is this the end state of the corridor?"* when the belief of that final state is higher than a threshold (we use a value of 0.4). If the answer is "*yes*", a CP is introduced in the trace (flag cp = 1 in Figure 7), the robot executes two successive turns to change direction, and introduces a new CP corresponding to the initial state of the opposite direction. If the answer is "no", the robot continues executing "Follow Corridor" actions. This process is repeated until traversing the corridor a predefined number of times.

Figure 7 shows an example of exploration of the upper horizontal corridor of the environment of Figure 2, with the robot initially in room 13. As it's shown, an independent trace is stored for each corridor direction, containing a header with





TRACE 1	(from left to right in horizontal corridors, and from botton to top in vertical corridors)	
Header	Nº of real states	7
	Real states in global POMDP	{14, 18, 22, 26, 30, 34, 38}
	Nº of data in the trace (T)	37
Data	ASO	{ 4, 5, 1, 1, 5, 2, 5, 4, 5, 1, 5, 2, 5, 5, 4, 5, 1, 1, 1, 5, 2, ... }
	LVO	{ 3, 3, 1, 1, 1, 1, 4, 3, 3, 1, 1, 1, 4, 4, 3, 3, 1, 1, 1, 1, 1, ... }
	DVO	{ 2, 2, 1, 1, 0, 0, 3, 2, 2, 1, 1, 0, 3, 3, 2, 2, 1, 1, 1, 0, 0, ... }
	cp	{ 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, ... }
	known-state	{ X, X, X, X, X, 38,14, X, X, X, X,38, 14, X, X, X, X, X, X, X,38, ... }
TRACE 2	(from right to left in horizontal corridors, and from top to botton in vertical corridors)	
Header	Nº of real states	7
	Real states in global POMDP	{40, 36, 32, 28, 24, 20, 16}
	Nº of data in the trace (T)	34
Data	ASO	{ 0, 5, 5, 4, 5, 1, 1, 5, 5, 0, 5, 4, 5, 1, 5, 0, 5, 4, 4, 5, 5, 5, ... }
	LVO	{ 5, 4, 4, 3, 3, 2, 2, 1, 1, 5, 4, 3, 3, 2, 1, 5, 4, 3, 3, 3, 1, 1, ... }
	DVO	{ 3, 3, 3, 2, 2, 1, 1, 1, 0, 3, 3, 2, 2, 1, 0, 3, 3, 2, 2, 2, 0, 0, ... }
	cp	{ 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1 , 0, 0, 0, 0, 0, 1, ... }
	known-state	{40, X, X, X, X, X, X, X, 16, 40, X, X, X, X, 16,40, X, X, X, X, X, 16, ... }

Figure 7. Example of exploration of one of the corridors of the environment of Figure 2 (involved nodes, states of the two execution traces, and stored data).

the number of real states contained in the corridor, its numeration in the global POMDP, and the total number execution steps of the trace. The trace stores, for each execution step, the reading values of ASO, LVO and DVO, the "cp" flag indicating CPs, and their corresponding "known states". These traces are the inputs for the EM-CBP algorithm shown in the next section.

This is no the first work that takes advantage of human intervention during the exploration stage to improve a learned probabilistic map. Another successful system is that proposed by (Thrun, Burgard and Fox, 1998), in which a human expert tele-operates the robot during exploration, pushing a button whenever it reaches a *significant place*. This makes easier to acquire a landmark-based metric map of a new environment, using the EM algorithm to concurrent localization and mapping. The topological representation proposed in this work gives several advantages regarding Thrun's work. The first one is that the graph and predefined uncertainty rules provide and initial POMDP model with enough information to allow the robot to autonomously navigate during the exploration stage, avoiding the user tele-operation. Besides, the proposed system is not landmark-based, but transition-based. These transitions are inherent to the local navigation schemes, and avoid the necessity of introducing a certainty point in all states (whereas in Thrun's work a button should be pressed with high probability in all significant places). In second place, the speech interface used in this work is much more intuitive that pushing a button. So, the human doesn't act as a teacher, but as a non-expert assistant of the robot during exploration. Finally, in Thrun's work *significant places* are indistinguishable, and so they are not exploited by the learning module. In this work, *certainty points* are turned into known places that are used to improve the learning algorithm, as it's shown in Section 7.

(b) Unsupervised Passive Learning

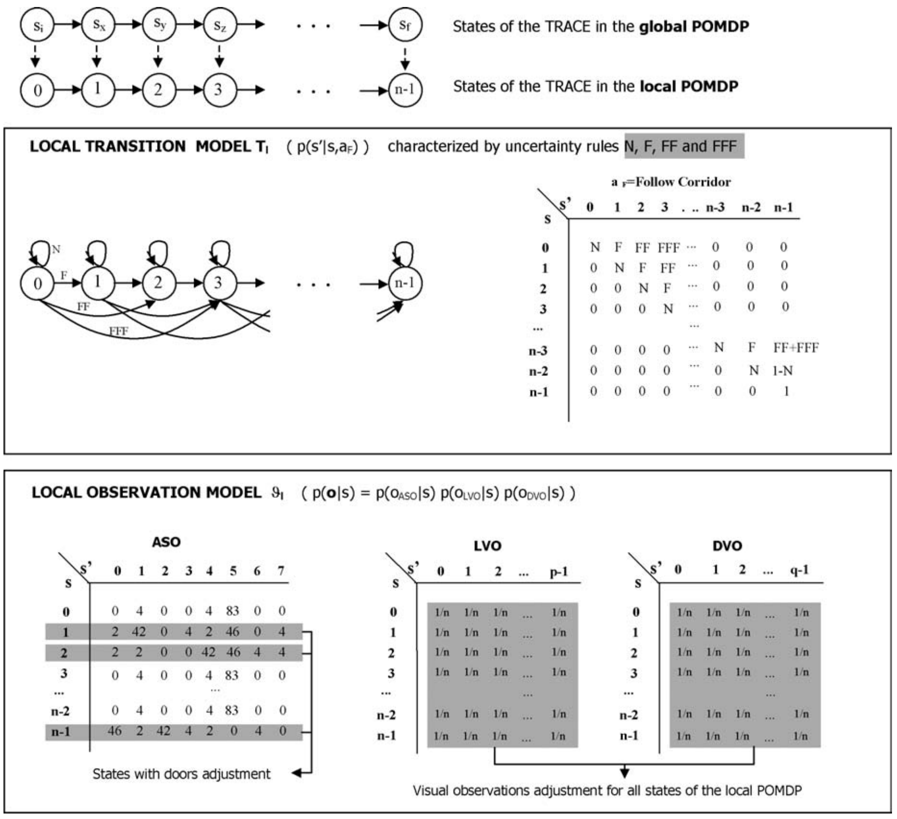
The objective of the passive learning is to keep POMDP parameters updated during the normal working of the navigation system. These parameters can change, mainly the state of doors (that affects the Abstract Sonar Observation), or the lighting conditions (that can modify the visual observations or the uncertainties of "Follow Corridor" actions). Because during the normal working of the system (passive learning), actions are not selected to optimize execution traces (but to guide the robot to goal rooms), the standard EM algorithm must be applied. Execution traces are obtained by storing sequences of actions and observations during the navigation from one room to another. Because they usually correspond to only one traversal of the route, sensitivity of the learning algorithm must be lower in this passive stage, as it's explained in the next section.

7. EM-CBP Algorithm

The EM with Certainty Break Points (EM-CBP) algorithm proposed in this section can be applied only in the active exploration stage, with the optimized execution traces. In this learning mode, an execution trace corresponds to one of the directions of a corridor, and involves only "Follow Corridor" actions.

The first step to apply the EM-CBP to a trace is to extract the local POMDP corresponding to the corridor direction from the global POMDP, as it's shown in Figure 8. To do this, states are renumbering from 0 to $n - 1$ (n being the number of real states of the local POMDP). The local transition model T_l contains only the matrix corresponding to the "Follow Corridor" action (probabilities $p(s'|s, a_F)$, whose size for the local POMDP is $(n - 1) \times (n - 1)$, and can be constructed from the current values of N, F, FF and FFF uncertainty rules (see Figure 8). The local observation model ϑ_l also contains only the involved states, extracted from the global POMDP, as it's shown in Figure 8.

The main distinguishing feature of the EM with Certainty Break Points algorithm is that it inserts delta distributions in *alfa* and *beta* (and so, *gamma*) distrib-


States of the TRACE in the **global POMDP**
States of the TRACE in the **local POMDP**
LOCAL TRANSITION MODEL T_l (p(s'|s,a_F)) characterized by uncertainty rules N, F, FF and FFF
N
F
FF
FFF
LOCAL OBSERVATION MODEL ϑ_l (p(**o**|s) = p(o_{ASO}|s) p(o_{LVO}|s) p(o_{DVO}|s))
States with doors adjustment
Visual observations adjustment for all states of the local POMDP

a_F=Follow Corridor

s \ s'	0	1	2	3	. ..	n-3	n-2	n-1
0	N	F	FF	FFF	...	0	0	0
1	0	N	F	FF	...	0	0	0
2	0	0	N	F	...	0	0	0
3	0	0	0	N	...	0	0	0
...					...			
n-3	0	0	0	0	...	N	F	FF+FFF
n-2	0	0	0	0	...	0	N	1-N
n-1	0	0	0	0	...	0	0	1

ASO

s \ s'	0	1	2	3	4	5	6	7
0	0	4	0	0	4	83	0	0
1	2	42	0	4	2	46	0	4
2	2	2	0	0	42	46	4	4
3	0	4	0	0	4	83	0	0
...					...			
n-2	0	4	0	0	4	83	0	0
n-1	46	2	42	4	2	0	4	0

LVO

s \ s'	0	1	2	...	p-1
0	1/n	1/n	1/n	...	1/n
1	1/n	1/n	1/n	...	1/n
2	1/n	1/n	1/n	...	1/n
3	1/n	1/n	1/n	...	1/n
...				...	
n-2	1/n	1/n	1/n	...	1/n
n-1	1/n	1/n	1/n	...	1/n

DVO

s \ s'	0	1	2	...	q-1
0	1/n	1/n	1/n	...	1/n
1	1/n	1/n	1/n	...	1/n
2	1/n	1/n	1/n	...	1/n
3	1/n	1/n	1/n	...	1/n
...				...	
n-2	1/n	1/n	1/n	...	1/n
n-1	1/n	1/n	1/n	...	1/n

Figure 8. Extraction of the local POMDP corresponding to one direction of the corridor to be explored.

utions corresponding to time steps with certainty points. This makes the algorithm to converge in a more reliable and fast way with shorter execution traces (and so, less corridor traversals) than the standard EM algorithm, as will be demonstrate in the results section.

Figure 9 shows the pseudocode of the EM-CBP algorithm. The expectation and maximization steps are iterated until convergence of the estimated parameters. The stopping criteria is that all the settable parameters remain stable between iterations (with probability changes lower than 0.05 in our experiments).

The update equations shown in Figure 9 (items 2.4 and 2.5) differ from Equations (10) and (11) in that they use Baye's rule (Dirichlet distributions) instead of frequencies. This is because, although both methods produce asymptotically the same results for long execution traces, frequency-based estimates are not very reliable if the sample size is small. So, we use the factor K $(K > 0)$ to indicate the confidence in the initial probabilities (the higher the value, the higher the confidence, and so, the lower the variations in the parameters). Note that the original

```
EM-CBP algorithm
(this code must be applied to each execution trace)

1. Extraction of the local POMDP($T_l$ and $\vartheta_l$)

2. Repeat until convergence of parameters of (2.4) and (2.5)
   2.1.  Calculate alfa values  (forward propagation)
         For t=0 to t=T-1
            If cp(t)=1   //Certainty Point
                $\alpha$(t,known_state(t))=1, and the rest columns of $\alpha$(t) are 0
            else
                $\alpha(t,s') = p(o(t) \mid s') \cdot \sum_{\forall s \in S_l} p(s' \mid s, a_F) \cdot \alpha(t-1, s) \quad \forall s' \in S_l$
                Normalize $\alpha$(t)
   2.2.  Calculate beta values  (backward propagation)
         For t=T-1 to t=0
            If cp(t)=1   //Certainty Point
                $\beta$(t,known_state(t))=1 and the rest columns of $\beta$(t) are 0
            else
                $\beta(t,s) = \sum_{s' \in S_l} p(s' \mid s, a_F) \cdot p(o(t+1) \mid s') \cdot \beta(t+1, s') \quad \forall s \in S_l$
                Normalize $\beta$(t)
   2.3.  Calculate gamma values
         For t=0 to t=T-1
            $\gamma(t,s)=\alpha(t,s)\cdot\beta(t,s) \quad \forall s\in S_l$
            Normalize $\gamma$(t)
   2.4.  Update observation parameters (apply independently to each observation $o_i$)
            $p(o_i \mid s) = \left(K \cdot p(o_i \mid s) + \sum_{t=0 \ldots T-1 \mid o_l(t)=o_i} \gamma(t,s)\right) \Big/ \left(K + \sum_{t=0 \ldots T-1} \gamma(t,s)\right) \quad \forall s \in S_l \quad \text{and} \quad \forall o_i$

   2.5.  Update uncertainty rules of "Follow Corridor" action
            $n = \left(K \cdot n + n^{\circ}\,\text{times}\left(\max(\gamma(t)) - \max(\gamma(t-1)) = 0\right)\right) / (K + T)$
            $f = \left(K \cdot f + n^{\circ}\,\text{times}\left(\max(\gamma(t)) - \max(\gamma(t-1)) = 1\right)\right) / (K + T)$
            $ff = \left(K \cdot ff + n^{\circ}\,\text{times}\left(\max(\gamma(t)) - \max(\gamma(t-1)) = 2\right)\right) / (K + T)$
            $fff = \left(K \cdot fff + n^{\circ}\,\text{times}\left(\max(\gamma(t)) - \max(\gamma(t-1)) = 3\right)\right) / (K + T)$

3. Return the adjusted parameters to the corresponding rows of the global POMDP
```

Figure 9. Pseudo-code of the EM-CBP algorithm.

re-estimation formulas (10) and (11) are a special case with $K = 0$. Similarly, leaving the transition probabilities unchanged is a special case with $K \to \infty$.

In practice, we use different values of K for the different settable parameters. For example, as visual observations are uniformly initialized, we use $K = 0$ (or low values) to allow convergence with a low number of iterations. However, the adjustment of Abstract Sonar Observations corresponding to states with doors must be less sensitive (we use $K = 100$), because the state of doors can easily change, and all the probabilities must be contemplated with relative high probability. During passive learning we also use a high value of K ($K = 500$), because in this case the execution traces contain only one traversal of the route, and some confidence about previous values must be admitted.

The final step of the EM-CBP algorithm is to return the adjusted parameters from the local POMDP to the global one. This is carried out by simple replacing the involved rows of the global POMDP with their corresponding rows of the learned local POMDP.

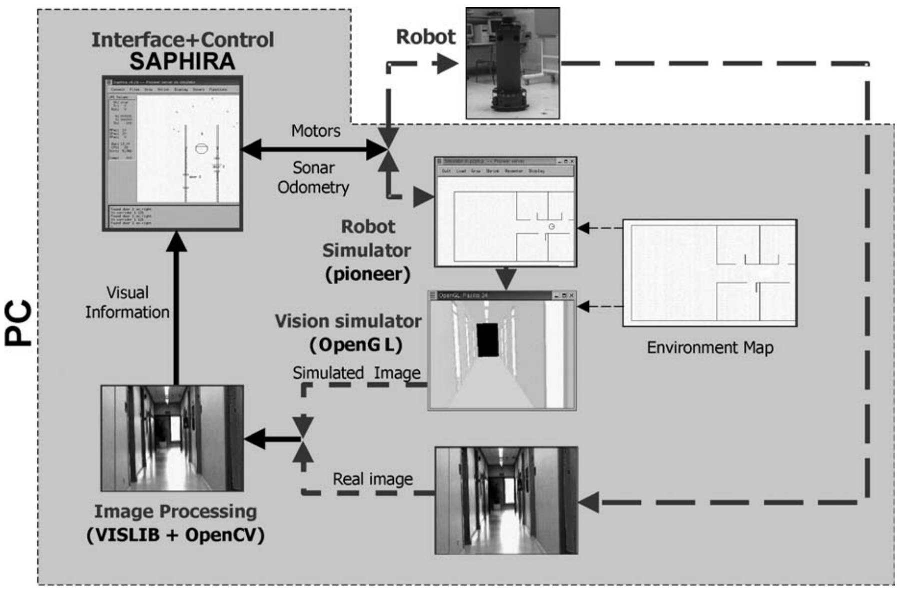


Figure 10. Diagram of test platforms: real robot and simulator.

8. Experimental Results

To validate the proposed learning system, demonstrate its advantage respect to standard EM based implementations, and test the effect of the different involved parameters, some experimental results are shown. Because some statistics must be extracted to test some features of the learning system, and it's also necessary to validate the methods in real robotic platforms and environments, two kind of experiments are shown. Firstly, we show some results obtained with a simulator of the robot, in order to extract some statistics without making long tests with the real robotic platform. The simulation platform used in this experiments (shown in Figure 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. Besides, to test the algorithms in extreme situations, we have incorporated to the simulator some methods to increase the non-ideal effect of actions, and noise in observations (indeed, these are higher that in real environment tests). So, simulation results can be reliably extrapolated to extract realistic conclusions about the system. Finally, we'll also show some experiments carried out with the real robot of the SIRAPEM project in one of the corridors of the Electronics Department, in order to validate the learning system on a real robotic platform.

The objective of the first simulation experiment is to learn the Markov model of the sub-POMDP corresponding to the upper horizontal corridor of the environment

Table II. Ideal local model to be learned for upper horizontal corridor of Figure 2

Local states	0 →	1 →	2 →	3 →	4 →	5 →	6
DVO:	4	4	3	3	2	1	1
ASO:	5	5,1	5,4	5	5,1,4,0	5	2,0
Follow Corridor rules: N=10%, F=80%, FF=5%, FFF=5%							

of Figure 2, going from left to right (so, using only the trace 1 of the corridor). Although the global graph yields a POMDP with 94 states, the local POMDP corresponding to states for one direction of that corridor has 7 states (renumbered from 0 to 6), and so, the sizes of the local matrixes are: 7×7 for the transition matrix $p(s'|s, a_F)$, 7×4 for the Deep Visual Observation matrix $p(o_{\mathrm{DVO}}|s)$, and 7×8 for the Abstract Sonar Observation matrix $p(o_{\mathrm{ASO}}|s)$. The Landmark Visual Observation has been excluded off the simulation experiments to avoid overloading the results, providing similar results to the Deep Visual Observation. In all cases, the initial POMDP was obtained using the predefined uncertainty rules of Table I. The simulator establishes that the "ideal" model (the learned model should converge to it) is that shown in Table II. It shows the "ideal" D.V.O. and A.S.O. for each local state (A.S.O. depends on doors states), and the simulated non-ideal effect of "Follow Corridor" action, determined by uncertainty rules N = 10%, F = 80%, FF = 5% and FFF = 5%.

In the first experiment, we use the proposed EM-CBP algorithm to simultaneously learn the "follow corridor" transition rules, D.V.O. observations, and A.S.O. observations (all doors were closed in this experiment, being the worst case, because the A.S.O. doesn't provide information for localization during corridor following). The corridor was traversed 5 times to obtain the execution trace, that contains a CP at each initial and final state of the corridor, obtained by user supervision. Figure 11 shows the learned model, that properly fits the ideal parameters of Table II. Because K is large for A.S.O. probabilities adjustment, the learned model still contemplates the probability of doors being opened. The graph on the right of Figure 11 shows a comparison between the real states that the robot traversed to obtain the execution trace, and the estimated states (maximum *gamma* values) using the learned model, showing that the model properly fits the execution trace.

Figure 12 shows the same results using the standard EM algorithm, without certainty points. All the conditions are identical to the last experiment, but the execution trace was obtained by traversing the corridor 5 times with different and unknown initial and final positions. It's shown that the learned model is much worse, and its ability to describe the execution trace is much lower.

Table III shows some statistical results (each experiment was repeated ten times) about the effect of the number of corridor traversals contained in the execution

s \ DVO	0	1	2	3
0	0.00	0.00	0.00	**1.00**
1	0.00	0.00	0.01	**0.99**
2	0.00	0.17	**0.83**	0.00
3	0.01	0.00	**0.89**	0.10
4	0.00	**1.00**	0.00	0.00
5	**1.00**	0.00	0.00	0.00
6	**1.00**	0.00	0.00	0.00

a_F rules:

N= 21.97%
F= 75.83%
FF= 1.2%
FFF= 1%

s \ ASO	0	1	2	3	4	5	6	7
0	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
1	0.00	0.37	0.00	0.00	0.00	**0.63**	0.00	0.00
2	0.00	0.00	0.00	0.00	0.40	**0.60**	0.00	0.00
3	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
4	0.19	0.19	0.00	0.00	0.19	**0.43**	0.00	0.00
5	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
6	0.38	0.00	**0.62**	0.00	0.00	0.00	0.00	0.00

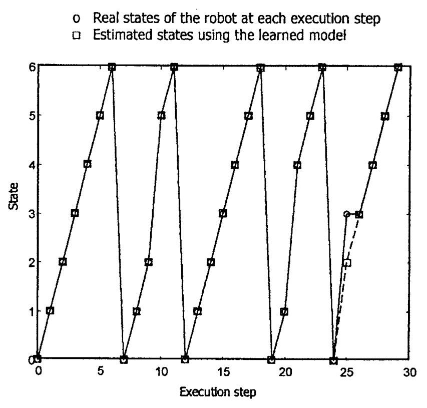


Figure 11. Learned model for upper corridor of Figure 2 using the EM-CBP algorithm.

s \ DVO	0	1	2	3
0	0.00	0.00	0.05	**0.95**
1	0.00	0.00	0.06	**0.93**
2	0.00	0.00	**0.28**	0.72
3	0.00	0.00	**0.49**	0.51
4	0.00	**0.47**	0.33	0.20
5	**0.45**	0.00	0.54	0.01
6	**1.00**	0.00	0.00	0.00

a_F rules:

N= 3.7%
F= 85.3%
FF= 7.2%
FFF= 3.8%

s \ ASO	0	1	2	3	4	5	6	7
0	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
1	0.00	0.45	0.00	0.00	0.00	**0.55**	0.00	0.00
2	0.00	0.00	0.00	0.00	0.41	**0.59**	0.00	0.00
3	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
4	0.19	0.19	0.00	0.00	0.19	**0.43**	0.00	0.00
5	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
6	0.38	0.00	**0.62**	0.00	0.00	0.00	0.00	0.00

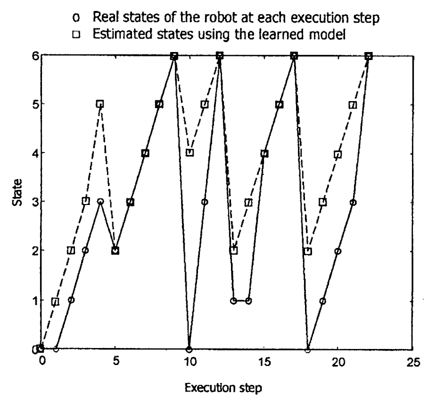


Figure 12. Learned model for upper corridor of Figure 2 using the standard EM algorithm.

Table III. Statistical results about the effect of corridor traversals and state of doors

	EM-CBP		**Standard EM**	
Nº of corridor traversals	**All doors closed**	**All doors opened**	**All doors closed**	**All doors opened**
1	37.6 %	6.0 %	57.0 %	10.2 %
2	19.5 %	0.6 %	33.8 %	5.2 %
3	13.7 %	0.5 %	32.2 %	0.8 %
5	12.9 %	0.0 %	23.9 %	0.1 %
10	6.7 %	0.0 %	12.6 %	0.0 %

trace, and the state of doors, using the EM-CBP and the standard EM algorithms. Although there are several measures to determine how well the learning method converges, in this table we show the percentage of faults in estimating the states of the execution trace. Opened doors clearly improve the learned model, because

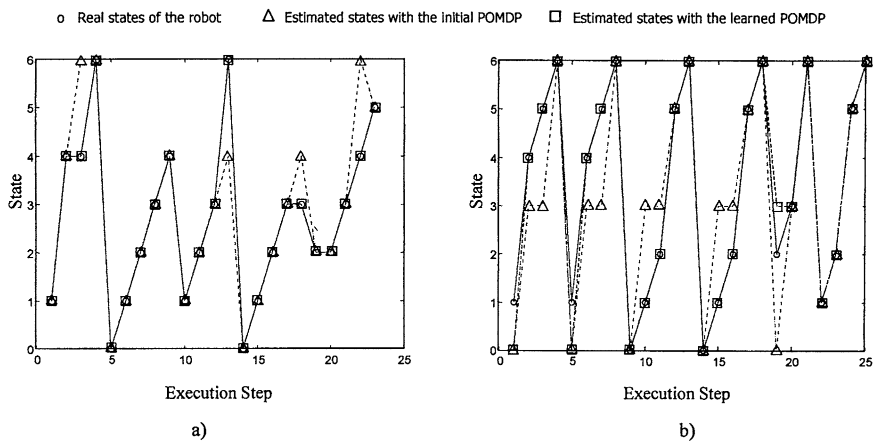
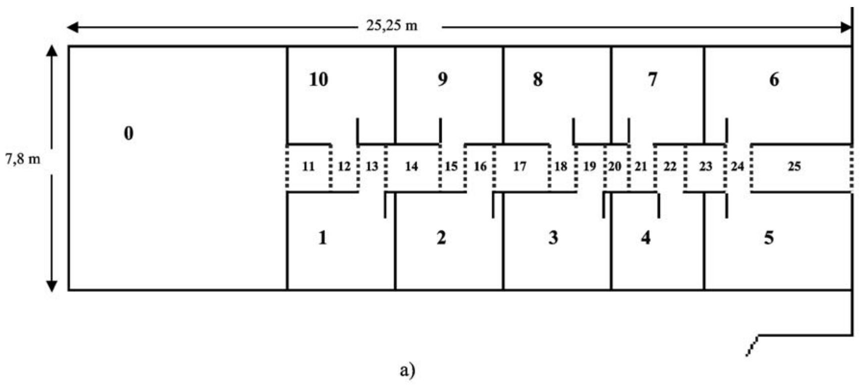


Figure 13. Experiments with the real robot, comparing position estimation with the initial "hand-made" POMDP and the learned POMDP: (a) with doors 1 and 2 opened; (b) with all doors closed.

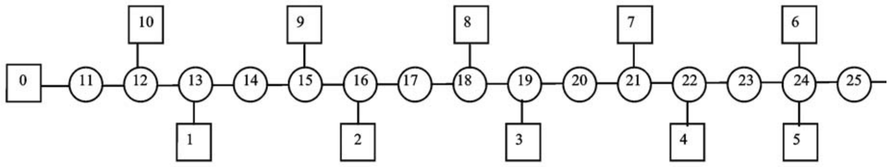
they provide very useful information to estimate states in the expectation step of the algorithm (so, it's a good choice to open all doors during the active exploration stage). As it's shown, using the EM-CBP method with all doors opened provides very good models even with only one corridor traversal. With closed doors, the EM-CBP needs between 3 and 5 traversals to obtain good models, while standard EM needs around 10 to produce similar results. In our experiments, we tested that the number of iterations for convergence of the algorithm is independent of all these conditions (number of corridor traversals, state of doors, etc.), ranging from 7 to 12.

A final simulated experiment is shown, whose objective is to demonstrate the advantages of using a learned model instead of a "hand-made" one. Firstly, we used the EM-CBP algorithm with a 5 traversals execution trace to obtain a learned model of the corridor (in this exploration stage, only doors of rooms 1 and 2 where opened). After that, we placed the robot in several random positions within the corridor, and tested the localization system using the initial "hand-made" POMDP, and the learned one. Figure 13(a) shows the results of 5 corridor traversals (they are linked together in only one trace) in which doors 1 and 2 remained opened. It can be seen how the learned model correctly estimates all states, while the initial POMDP has several fails. To obtain the results of Figure 13(b), all doors were closed. In this case, the initial POMDP has poor information (for example, about visual observations) to estimate states, while the learned POMDP still performs very well (only one state is not correctly estimated).

Although the environment of the last experiments can seem relatively simple, more complicated environments provide similar results because the proposed method always extracts local POMDPs of corridors. Average length (in number of states) of corridors in typical office environments ranges from 5 to 20, and for these



a)



b)

c)

Figure 14. (a) Map of one of the corridors of the Electronics Department of the University of Alcalá; (b) corresponding topological graph; (c) view of the robot exploring the corridor.

values the number of traversals needed for convergence remains small (from 3 to 5 traversals).

To validate the learning system in larger corridors and real conditions, we show the results obtained with SIRA learning the model of one of the corridors of the Electronics Department of the University of Alcalá. Figure 14 shows the corridor map and its corresponding graph (with 71 states). The local POMDP of each

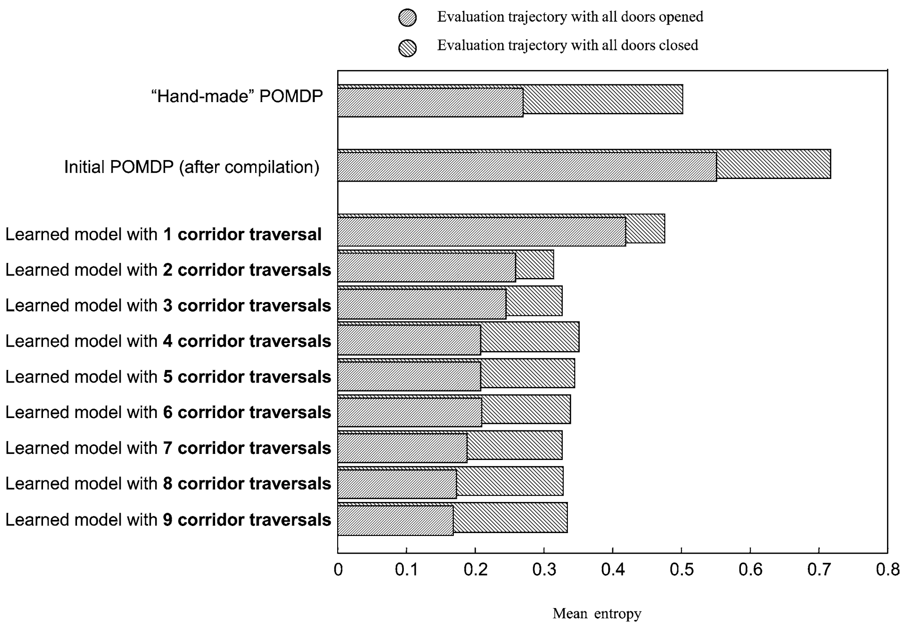


Figure 15. Comparison of the mean entropy of the belief distribution using different models of the corridor to localize the robot in two evaluation trajectories.

corridor direction contains 15 states. Figure 14(c)) shows a view of the robot navigating along the corridor during exploration. In these experiments, some people were continuously walking along the corridor to test the robustness of the learning system in real and dynamic environment conditions. To accelerate convergence, all doors were kept opened during the active exploration stage. We evaluated several POMDP models, obtained in different ways:

- The initial model generated by the POMDP compiler, in which visual observations of corridor aligned states are initialized with uniform distributions.
- A "hand-made" model, in which visual observations were manually obtained (placing the robot in the different states and reading the observations).
- Several learned POMDP models (using the EM-CBP algorithm), with different number of corridor traversals (from one to nine) during exploration.

Two "evaluation trajectories" were executed using these different models to localize the robot. In the first one, the robot crossed the corridor with unknown initial position and all doors opened, and in the second one, all doors were closed. The localization system was able to global localize the robot in less than 5 execution steps in both cases with all models. However, the uncertainty of the belief distribution was higher with worse models. Figure 15 shows the mean entropy of the belief distribution for all the evaluation trajectories. The "initial POMDP model" is the worst, because it doesn't incorporate information about visual observations. The learned model with one corridor traversal is not better that the "hand-made" one, but from two traversals, the obtained entopy and easy installation justifies the usage of the learning module. It can also be deduced that a good number of corridor traversals ranges from 2 to 4 in this case, because later adjustments of the model can be carried out during “active exploration”. Because all doors were opened during exploration, as the number of corridor traversals increases, so does the evidence about opened doors in the model and so, the uncertainty in the “evaluation trajectory” with opened doors decreases, while in that with closed doors increases. So, the model adapts this kind of environment changes.

The time required for exploring one corridor with three traversals was about 5 minutes (with a medium speed of 0.3 m/s). The computation time of the EM-CBP algorithm, using the onboard PC of the robot (a 850 MHz Pentium III) was 62 ms. These times are much lower that the ones obtained in Thrun’s work (Thrun, Burgard and Fox, 1998), in which for learning a metric map of an environment of 60×60 meters (18 times larger than our corridor), an exploration time of 15 minutes and computation time of 41 minutes were necessary.

9. Discussion and Future Work

The proposed navigation system, based on a topological representation of the world, allows the robot to 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. Navigation inside rooms is also performed by means of local navigation behaviours based on people searching and following, that are out of the scope of this paper.

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 modeled 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, that is successfully treated by the probabilistic transition and observation models. Regarding doors state, the proposed learning module 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 strengh 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 presents the learning module of a POMDP based navigation system for assistant robots. The objective is to cope with practical issues of this kind of applications, developing a learning system that makes easier the installation and configuration of the robot in new working environments. To do this, we take advantage of the specific topology of the Markov model, and the features of the settable parameters, that make possible to adjust the parameters of the transition and observation matrixes by independently traversing each corridor of the environment. So, a local POMDP of each corridor can be extracted, reducing the size of matrixes that take part in the EM algorithm. Besides, some "certainty points" are introduced by minimal user supervision (using the speech interface), also reducing the number of corridor traversals needed for convergence of the algorithm.

The system has been validated on a robotic platform navigating in real environments, and the parameters are correctly adjusted after a few (depending on the corridor length) corridor traversals, always quite less than when using standard EM implementations. Although some assumptions are made to derive the EM-CBP algorithm, all of them are satisfied in the working environments of assistant robots, making this learning method a good practical choice in this kind of navigation applications.

Acknowledgements

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.*

Asoh, H., Motomura, Y., Hara, I., Akaho, S., Hayamizu, S. and Matsui, T.: 1996, Combining probabilistic map and dialog for robust life-long office navigation, in: *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'96)*, pp. 807–812.

Bilmes, J. A.: 1997, Gentle tutorial on the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models, Technical Report ICSI-TR-97-021, University of Berkeley.

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

Gechter, F., Thomas, V. and Charpillet, F.: 2001, Robot localization by stochastic vision based device, in: *The 5th World Multi-Conferene on Systemics, Cybernetics and Informatics (SCI 2001)*.

Haegele, M., Neugebauer, J. and Schraft, R. D: 2001, From robots to robot assistants, in: *Proc. of the 32nd ISR (International Symposium on Robotics)*, pp. 19–21.

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.

Hoppenot, E. Colle: 2002, Mobile robot command by man–machine co-operation – Application to disabled and elderly people assistance, *Journal of Intelligent and Robotic Systems* **34**(3), 235–252.

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**, 99–134.

Koenig, S. and Simmons, R.: 1996, Unsupervised learning of probabilistic models for robot navigation, in: *Proceedings of the International Conference on Robotics and Automation*, pp. 2301–2308.

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

Konolige, K. and Myers, K.: 1998, The Saphira architecture for autonomous mobile robots, *Artificial Intelligence and Mobile Robots*, 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*.

Liu, Y., Emery, R., Chakrabarti, D., Burgard, W. and Thrun, S.: 2001, Using EM to learn 3D models with mobile robots, in: *Proc. of the International Conference on Machine Learning (ICML)*.

López, E., Bergasa, L. M., Barea, R. and Escudero, M. S.: 2003a, Topological robot navigation using multisensorial event-based POMDPs, in: *Proc. of the 11th IEEE International Conference on Advanced Robotics (ICAR 2003)*, pp. 216–221.

López, E., Bergasa, L. M., Barea, B. and Escudero, M. S.: 2003b, A planning architecture for topological robot navigation in uncertain domains, in: *Proc. of the 9th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2003)*, Vol. 1, pp. 597–604.

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.

Morere, Y., Pruski, A. and Ennaji, M.: 2002, A multi-agent control structure for an intelligent wheelchair, *AMSE Modelling, Measurement, Control Journal*.

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

Papadimitriou, C. and Tsitsiklis, J.: 1987, The complexity of Markov decision processes, *Math. Oper. Res.* **12**(3), 441–450.

Puterman, M. L.: 1994, *Markov Decision Processes – Discrete Stochastic Dynamic Programming*, Wiley, New York.

Rabiner, L. R. and Juang, B. H.: 1986, An introduction to hidden Markov models, *IEEE ASSP Magazine* **4**.

Regini, L., Tascini, G. and Zingaretti, P.: 2002, Appearance-based robot navigation, in: *Artificial Intelligence and Intelligent Agents (AIIA 2002)*.

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.

Russell, S., Binder, J. and Koller, D.: 1994, Adaptive probabilistic networks, Technical Report CSD-94-824, July 1994, University of California, Berkeley.

Shatkay, H. and Kaelbling, L.: 1997, Learning topologicla maps with weak local odometric information, in: *Proc. International Joint Conference on Artificial Intelligence*, pp. 920–927.

Thrun, S.: 2000, Probabilistic algorithms in robotics, in: *AI Magazine* **21**(4), 93–109.

Thrun, S., Gutmann, J., Fox, D., Burgard, W. and Kuipers, B.: 1998, Integrating topological and metric maps for mobile robot navigation: A statistical approach, in: *Proc. National Conference on Artificial Intelligence*, pp. 989–995.

Thrun, S., Burgard, W. and Fox, D.: 1998, A probabilistic approach to concurrent mapping and localization for mobile robots, *Machine Learning and Autonomous Robots (joint issue)* **31**(5), 1–25.

Yairi, T., Togami, M. and Hori, K.: 2003, Learning topological structures of POMDP-based state transition models by state splitting method, in: *Proc. of the 21th IASTED International Conference on Applied Informatics*.

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