September 16-19, 2003
Lisbon, Portugal

ETFA 2003

2003 IEEE Conference on Emerging Technologies and Factory Automation Proceedings

IEEE

ies

UNINOVA

Universidade Nova de Lisboa
Faculdade de Ciências e Tecnologia

ETFA 2003

2003 IEEE Conference on
Emerging Technologies and Factory Automation
Proceedings

VOLUME 1

September 16-19, 2003
Lisbon, Portugal

Co-sponsored by
IEEE Industrial Electronics Society
UNINOVA and Universidade Nova de Lisboa-FCT-DEE

IEEE ies UNINOVA Universidade Nova de Lisboa
Faculdade de Ciências e Tecnologia

IEEE Catalog Number 03TH8696
ISBN 0-7803-7937-3
Library of Congress Number 2003104524

Contents

Volume 1

A Planning Architecture for Topological Robot Navigation in Uncertain Domains

Elena López Luis Miguel Bergasa Rafael Barea Marisol Escudero
Dept. of Electronics
University of Alcalá
Campus Universitario, 28871, Alcalá de Henares (Madrid)
SPAIN

***Abstract* – This paper presents a new navigation architecture for autonomous mobile robots working in uncertain domains. Partially Observable Markov Decision Processes (POMDPs) are suitable mathematical models for solving localization, planning and learning problems in uncertain navigation systems based on a topological representation of the environment. This paper focuses on the planning module, consisting of a two-level layered architecture (a local policy and a global policy) that simplifies the problem of finding optimal policies in POMDPs. The proposed system naturally integrates several planning objectives, such as guiding to a goal room, reducing location uncertainty, and exploring. Some experimental results are shown, carried out with an assistant robot developed in the Electronics Department of the University of Alcalá.**

I. INTRODUCTION

This work has been developed within the SIRAPEM project (Spanish acronym for Autonomous Robotic System for Elderly Assistance), which objective is to design an assistant robot for elderly and/or disabled people. Due to the dramatic increase of the elderly population in the last years, the society needs to find new technologies and alternative ways of providing care to this sector of the population. Aware of this necessity, nowadays there are several research groups working in this area, and some important projects such us "Nursebot" [1] and "Morpha" [2].

Figure 1 shows the global architecture of the SIRAPEM system, based on a commercial platform (the PeopleBot robot of ActivMedia Robotics [3]) 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 camera, a tactile screen and wireless Ethernet link. The system architecture includes telepresence and telemedicine interfaces, and 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 guide the user).

This paper focuses on the navigation module, and mainly, in the planning system. In this kind of care applications, in which the robot must perform tasks in indoor environments (such as houses, nursing homes or hospitals) for long periods of time, it's very important to achieve a robust navigation system capable of treat real world uncertainties, and solve global localization failures without any user supervision. Nowadays there are several important works ([4],[5]) about robot navigation systems specifically dealing with sensors and actuators uncertainties. 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.

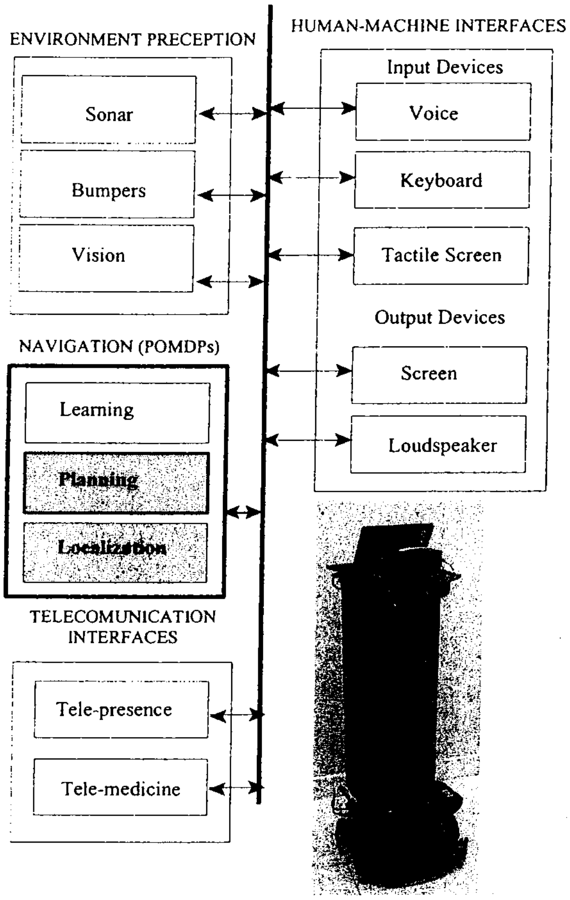

Fig 1. Global architecture of the SIRAPEM project.

So, they must use simple environment representations and natural landmarks that can be easily found in any indoor environment.

A suitable mathematical framework for robust navigation under uncertainty, based on a topological model of the environment, are 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 figure 1). The robots DERVISH [6], developed in the Stanford University, and Xavier [7], in the Carnegie- Mellon University, use these kind of navigation strategies for localization and action planning. The SIRAPEM navigation system provides several contributions in this research line. One of them, presented in previous works [8][9], is the incorporation of visual information (besides typical proximity sensors) to the Markov model in order to improve the robustness of the localization system. Another one, in which we focus this paper, is the development of a two-layered planning architecture that combines global an local policies to achieve several planning objectives: guidance to a room, reduction of location uncertainty, and exploration.

This paper is organized as follows. After a brief overview of POMDPs foundations (section 2), we describe the Markov model used in this navigation application (section 3). Section 4 shows the global architecture of the navigation system. The localization module is briefly revised in section 5. The two layers of the planning system, in which we focus the paper, are shown in section 6. Finally, we show some experimental results (section 7), whereas a final conclusion summarizes the paper (section 8).

II. POMDP MODELS REVIEW

In this section we introduce some terminology and foundations about POMDPs. Firstly, we describe Markov Decision Processes (MDPs) as the underlying model of a Partially Observable Markov Decision Process (POMDP), and then we introduce the concept of partial observability. In both cases, a brief review about conventional planning methods is presented.

A. *Markov Decision Processes*

A MDP is a model for sequential decision making, formally defined as a tuple $\{S,A,T,R\}$, where,

- S is a finite set of states ($s \in S$).
- A is a finite set of actions ($a \in A$).
- $T=\{p(s'|s,a) \ \ \forall (s,s' \in S \ \ a \in A)\}$ is a state transition model which specifies a conditional probability distribution of posterior state s' given prior state s and action executed a.
- $R=\{r(s,a) \ \ \forall (s \in S \ \ a \in A)\}$ 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*, that establishes that actual state and action are the only information needed to predict next state:

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

In a MDP, the actual state s is always known without uncertainty. Planning in a MDP is the problem of action selection as a function of the actual state. A MDP solution is a *policy* $a=\pi(s)$, that 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 algorithms (such as the *value iteration* algorithm) have been proposed [10][11].

B. *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:

- $\{S,A,T,R\}$, the same that in the MDP context.
- O, a finite set of observations ($o \in O$)
- $\vartheta=\{p(o|s) \ \ \forall \ o \in O, \ s \in S\}$ 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, that only depends on the prior belief, prior action, and current observation, as will be seen in a posterior section.

In a POMDP, a policy $a=\pi(Bel)$ maps beliefs into actions. So, 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 [12], 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 POMDPs, dividing the problem into two layers, and applying some heuristic strategies adopted from previous similar works [5]. This method, as will be shown in section 7, provides successful results in this kind of robot navigation applications.

III. MARKOV MODEL FOR ROBOT NAVIGATION

The POMDP model used for robot navigation is constructed from two sources of information: the topology of the environment, and some experimental information about action and sensor errors and uncertainties. Taking into account that the final objective of the navigation system will be to direct the robot from one room to another, we discretize the environment into coarse-grained "regions" of variable size in accordance with the topology of the environment, in order to make the planning task easier. As it's shown in figure 2 for a corridor of the Electronics Department, 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).

A. *The elements: States, Actions and Observations*

States (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.

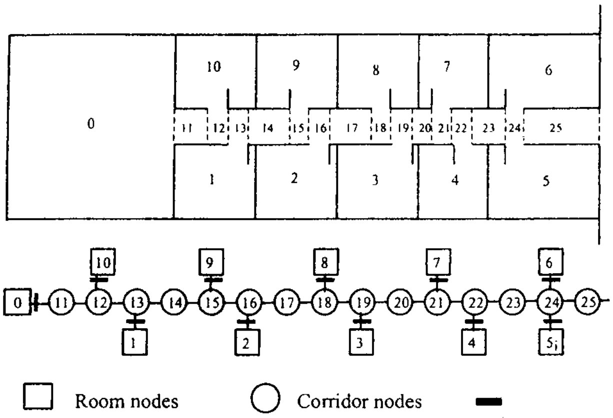

Fig.2. Topological Graph Model for a corridor of the Electronics Department

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:

- *"Go out room"(a_O)*: to traverse door using sonar and visual information in room states,
- *"Enter room" (a_E)*: only defined in corridor states oriented to a door,
- *"Turn right" (a_R)*: to turn 90° to the right,
- *"Turn Left" (a_L)*: to turn 90° to the left
- *"Follow Corridor"(a_F)*: to continue through the corridor to the next state.
- *"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 [9]. In each state, the robot makes two kind of "observations". The first one is an "*Abstract Sonar Observation*" (o_{ASO}): it can perceive, in each of three nominal directions (left, front and right), whether it's "free" or "occupied" and construct an abstract observation from the combination of the percepts in each direction (thus, there are 8 possible abstract sonar observations). The second one is a "*Landmark Visual Observation* " (o_{LVO}) consisting of the number of doors captured by the camera from actual state.

As it was demonstrated in previous published works about SIRAPEM project [8][9], the incorporation of visual information improves the observability of the process, providing much better results in the localization module.

B. The uncertainties: Transition and Observation Models

As it was said, besides the topology of environment, it's necessary to define some action and observation uncertainties to calculate the final POMDP model. The *rules* to define these errors in our robot and their *initial values* are shown in table 1 for action and observation uncertainties. For example, if a "Follow" action (F) is commanded, the probability of making a state transition (F) is 85%, while there is a 5% probability of remaining in the same state

ACTION UNCERTAINTIES (F=Follow, L=Left, R=Right, O=Out, E=Enter,N=No action)			
Command	**Efect of Command (% probabilities)**		
F	N = 5	F = 85	F-F = 10
L	N = 5	L = 90	L-L = 5
R	N = 5	R = 90	R-R = 5
O	N = 5	O = 85	O-F = 10
E	N = 10	E = 90	
OBSERVATION UNCERTAINTIES			
Sonar Model (%probabilities)			
Open door probability (for all doors)			50
Prob. of detecting something being nothing			10
Prob. of detecting nothing being something			5
Vision Model			
Assigned probability to real number of doors			70
Maximum deviation			±2 doors

Table 1. Uncertainty rules for constructing the Markov model

(N=no action), and a 10% probability of making two successive state transitions (F-F). These uncertainty rules provide initial parameters for the entries of transition (T) and observation (ϑ) matrixes, that are later on-line adapted by the learning module to fit real experience data.

IV. NAVIGATION SYSTEM ARCHITECTURE

Figure 3 shows the global navigation architecture of the SIRAPEM project, formulated as a POMDP model.

At each process step, the navigation system (specifically 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, is 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 (this involves high computational times [12]), this selection is simplified by dividing the planning module in 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), (2) *localizing* (the objective is to reduce location uncertainty) and (3) *exploring* (the objective is to learn or adjust topology 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 [5].

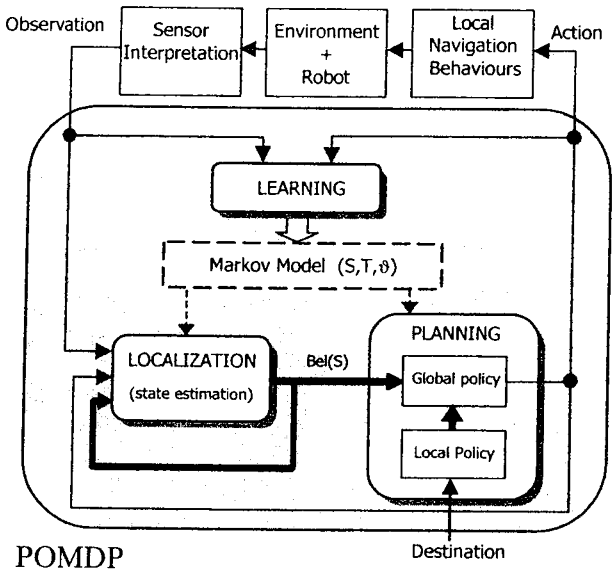

Fig. 3. Global architecture of the navigation system.

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.

Finally, the *learning* module (that is out of the scope of this paper) uses action and observation data to learn and adjust the topology and uncertainties of the Markov model.

V. LOCALIZATION SYSTEM

Although it's not the objective of this paper, a brief review about state estimation for robot localization is presented in this section, due to the strong connection between the localization and planning modules.

A. Markov localization by state estimation

The localization module updates the belief distribution after each state transition, using the well-known Markov localization equations [4]. These equations are applied in two steps:

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

$$Bel_{posterior}(s') = K \cdot \sum_{s \in S} p(s'|s,a) \cdot Bel_{prior}(s) \qquad (2)$$

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:

$$Bel_{posterior}(s) = K \cdot p(o|s) \cdot Bel_{prior}(s) \qquad (3)$$

In the first 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 1.0 and the rest 0.0. (b) If initial state is unknown, a uniform distribution is calculated over all states.

B. Uncertainty evaluation

Although the planning system chooses the action based on the entire belief distribution, in some cases it will be necessary to evaluate the degree of uncertainty of that distribution (this is, the localization uncertainty).

A typical measure of discrete distributions uncertainty is the *entropy* [13]. The *normalized entropy* (varying between 0 and 1) of the belief distribution is:

$$H(Bel) = -\frac{\sum_{s \in S} Bel(s) \cdot \log(Bel(s))}{\log(n_s)} \qquad (4)$$

where n_s is the number states of the Markov model. The lower the value, the more certain the distribution.

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. So, we propose another measure that better detects the convergence of the distribution to a unique maximum (and so, that the robot is globally localized). This is the *normalized divergence factor*, calculated in the following way:

$$D(Bel) = 1 - \frac{n_s(d_{max} + p_{max}) - 1}{2 \cdot n_s - 1} \qquad (5)$$

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.

VI. PLANNING SYSTEM

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 quick 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.

The solution adopted in this work is to divide the problem in 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 ($a^*(Bel)$) as a function of the local policy and the belief. This structure is shown in figure 4, and described in subsequent sections.

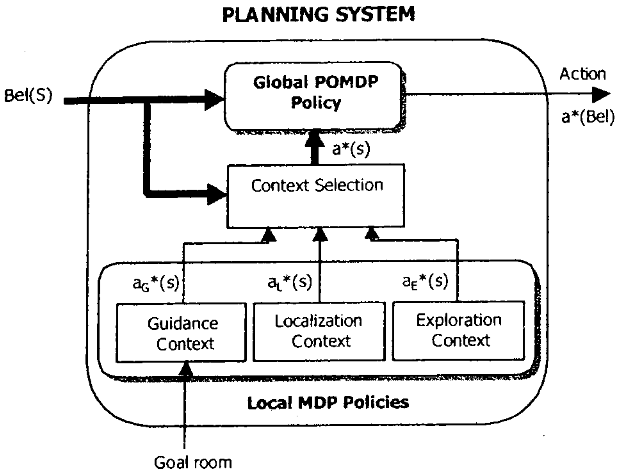

Fig. 4. Planning system architecture, consisting of two layers: (1) Global POMDP Policy, and (2) Local MDP Policies

A. 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, that seamlessly integrates the two concerns of acting in order to reduce uncertainty and acting for achieving a goal. The "exploration" context is to select actions in order to learn the topology and 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 (figure 4).

1. Guidance Context

This local policy is calculated whenever a new goal room is selected by the user. It's main objective is to assign to each individual state s, an optimal action (a_G*(s)) to guide the robot to the goal.

Firstly, a modification of the A* search algorithm [14] 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).

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 o 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 $Q^a(s)$ values are assigned to each (s,a) pair. In the MDPs theory, Q-values [11] 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: $Q^a(s)=1$ if action a is optimal at state s, $Q^q(s)=-1$ (negative utility) if actions a is not defined at state s, and $Q^a(s)=-0.5$ for the remaining cases (actions that disaligns the robot from preferred heading).

2. Localization Context

This policy is used to guide the robot to *sensorial relevant places* that reduce positional uncertainty, even if that requires to move it away from the goal temporarily. This planning objective was not considered in previous similar robots (such as DERVISH [6] or Xavier [7]), or was implemented by means of fixed sequences of movements [5] 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). Sensorial relevant states are those that *break* a sequence of observations that can be found in another zone of the graph.

So, this policy (a^*_L(s)) is only environment dependent, and is calculated from the connections of the graph and the ideal observations of each state. The optimal action assigned to room states is "Go out room" (a_O). To calculate optimal actions to corridor states, firstly 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 sensorial relevant state. Later, an optimal action is assigned to the four states of each corridor node to align the robot with the preferred heading, (as it was described in the guidance context section).

3. Exploration Context

The objective of this local policy is to select actions during the exploration stage, in order to construct the topological graph and learn transition and observation probabilities. As in this stage states are 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, and they are out of the scope of this paper.

B. Global Heuristic Policies

The global policy combines the probabilities of each state to be the current state (belief distribution Bel(S)) with the best action assigned to each state (local policy a*(s)) to select the final action to execute, a*(Bel). Once selected the local policy context (for example guidance context, a*(s)=a_G*(s)), some heuristic strategies proposed by [4] can be used to do this final selection (see figure 5).

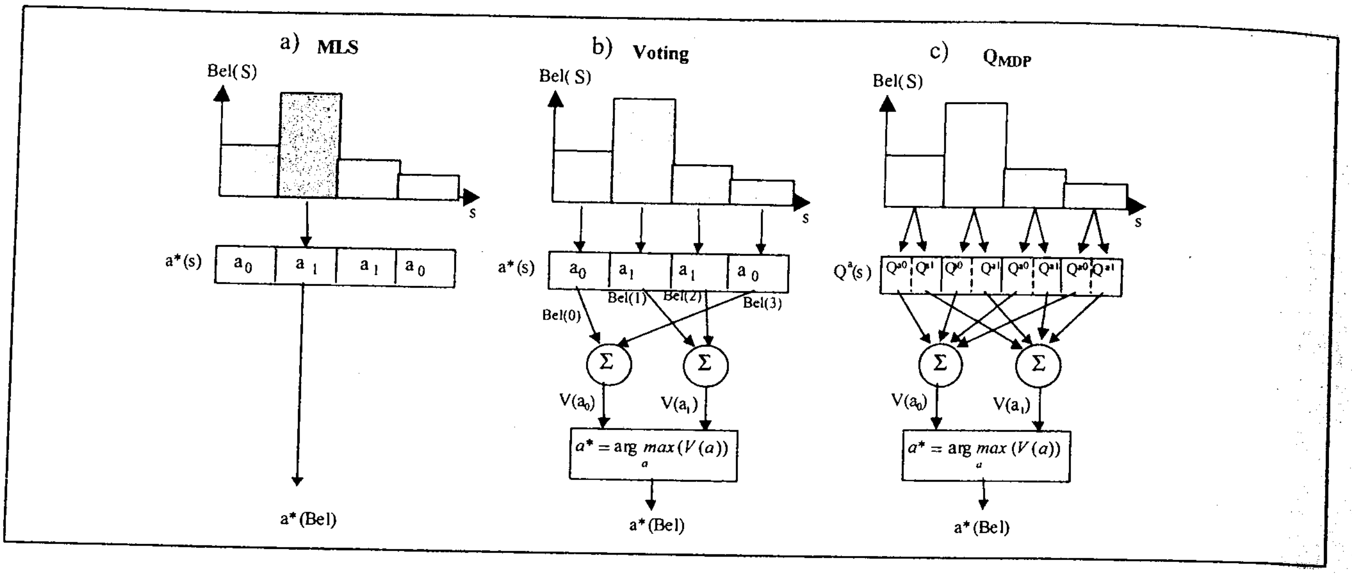

Fig. 5. Graphical interpretation of global heuristic policies: MLS, Voting and Q_{MDP}

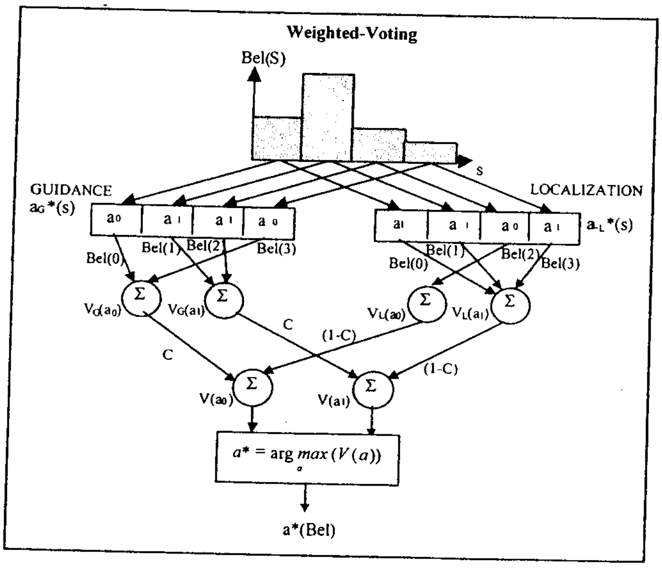

Fig. 6. The weighted-voting context combination method

The simplest one is the *Most Likely State* (MLS) global policy that finds the state with the highest probability and executes the local policy of that state (figure 5.a):

$$a^{\bullet}_{MLS}(Bel) = a^*\left(\arg\max_s(Bel(s))\right) \qquad (6)$$

The *Voting* global policy first computes the *probability mass* of each action (V*(a)*) (probability of action *a* is optimal) according to the belief distribution, and then selects the action that is most likely to be optimal (the one with highest probability mass) (figure 5.b):

$$\begin{aligned} V(a) &= \sum_{s \,|\, a^*(s)=a} Bel(s) \quad \forall a \in A \\ a^{\bullet}_{vot}(Bel) &= \arg\max_a(V(a)) \end{aligned} \qquad (7)$$

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 (figure 5.c):

$$\begin{aligned} V(a) &= \sum_{s \in S} Bel(s) \cdot Q^a(s) \quad \forall a \in A \\ a^{\bullet}_{Q_{MDP}}(Bel) &= \arg\max_a(V(a)) \end{aligned} \qquad (8)$$

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

C. *Automatic Context Selection or Combination*

Apart from the exploration context (out of the scope of this paper), this section considers the automatic context selection (see figure 4) 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. 5).

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

$$a^*(s) = \begin{cases} a^{\bullet}_G(s) & \text{if} \quad D > \phi \\ a^{\bullet}_L(s) & \text{si} \quad D \le \phi \end{cases} \qquad (9)$$

However, the *weighting-method* combines both contexts using convergence 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 then 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 (see figure 6):

$$V(a) = (1 - D) \cdot V_G(a) + D \cdot V_L(s)$$
$$a^*(Bel) = \arg\max(V(a)) \qquad (10)$$

VII. EXPERIMENTAL RESULTS

In order to validate the navigation system and compare the different planning methods and contexts, some experimental results are shown. Because the advantages of some planning strategies can only be demonstrated in hard environments, we include two kind of experiments. Firstly, we show some results obtained with a simulator of the robot, in order to test the planning methods in a hard fictitious environment. After that, 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.

A. Simulation results in an aliased environment

There are some things that make one world more difficult to navigate that another. One of them is its degree of perceptual aliasing, that substantially affects the agent's ability for localization and planning. 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 used the fictitious aliased environment shown in figure 7, in which there are two identical corridors.

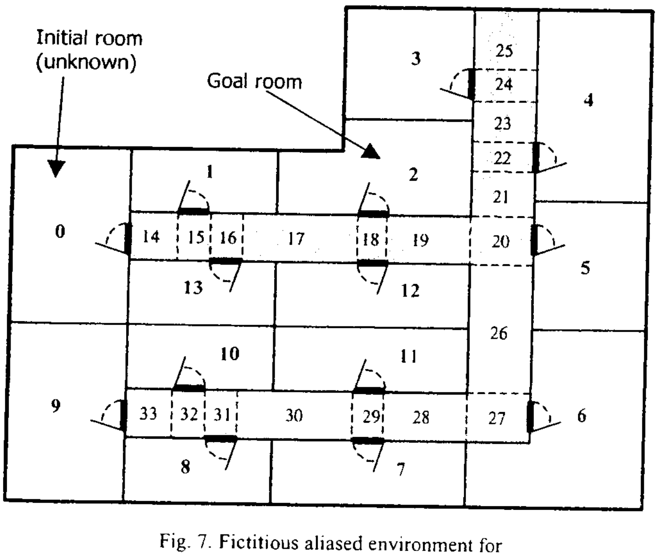

Fig. 7. Fictitious aliased environment for simulation experiments

ONLY GUIDANCE CONTEXT				
	Nº 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)				
	Nº Actions	Final H	Final D	Final State 2
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%

Table 2. Comparison of the planning strategies in the fictitious environment

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), 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 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).

B. Real robot example

Finally, table 3 shows a real guidance example using the SIRAPEM prototype of figure 1 navigating in a corridor of the Electronics Department (graph shown in figure 2). 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 commanded action, real action and final 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 first and second most voted actions in the guidance contexts, and their divergence. When divergence is higher than 0.5, the planner commutes to localization context.

Step	Commanded Action	Real action and final state	Most likely state (node and dir)	2nd Most likely state (node and dir)	Guidance Context			Local. Cont.	Selected context	Next best action selected	D(Bel)
					Most voted action	2nd most voted action	D(V(a))	Most voted action (only if D>0.5)			
0	Known that initial state is one of the rooms (so, states from 0 to 10 has p=9.0909%, and the rest p=0%)				O(90.9%)	N(9.09%)	0.148	no needed	GUIDE	O	0.9613
1	O	O (16 ↑)	13, 16,19,22 ↑ 12,15,18,21 ↓	24 ↑, 24 ↓	L(51%)	R(49%)	0.801	L(62%)	LOCAL	L	0.94
2	L	L (16←)	16← (60%)	18→ (10%)	R(80%)	F(20%)	0.327	no needed	GUIDE	R	0.453
3	R	R (16 ↑)	16 ↑ (60%)	18↓ (10%)	R(80%)	L(20%)	0.327	no needed	GUIDE	R	0.453
4	R	R (16→)	16→ (90%)	18← (2.5%)	F(95%)	R(5%)	0.081	no needed	GUIDE	F	0.113
5	F	F (17→)	17→ (78.7%)	16→ (15%)	F(98%)	R(2%)	0.032	no needed	GUIDE	F	0.290
6	F	F (18→)	18→ (94.8%)	19→ (3%)	F(100%)	F(100%)	0	no needed	GUIDE	F	0.067
7	F	F (19→)	19→ (96%)	18→ (3%)	F(100%)	F(100%)	0	no needed	GUIDE	F	0.055
8	F	F (20→)	20→(93.5%)	19→(3.3%)	F(100%)	F(100%)	0	no needed	GUIDE	F	0.082
9	F	F (21→)	21→(67%)	22→(24%)	F(74.5%)	R(25%)	0.414	no needed	GUIDE	F	0.453
10	F	N (21→)	21→(62%)	23→ (18%)	F(62%)	R(38%)	0.621	F(67%)	LOCAL	F	0.473
11	F	F (22→)	22→(82%)	23→(10%)	R(97%)	F(3%)	0.049	no needed	GUIDE	R	0.231
12	R	R (22 ↓)	22 ↓ (93%)	24 ↓ (6%)	E(93%)	R(7%)	0.114	no needed	GUIDE	E	0.100
13	E	E (4)	4 (93.6%)	5 (6.5%)	N(94%)	O (6%)	0.098	no need	GUIDE	N	0.097

Table 3. Experimental results navigating towards room 4 with unknown intitial room state (real initial room 2).

As it's shown in this example, using a localization context when there is uncertainty about which action to take quickly reduces location uncertainty, avoiding the selection of a wrong action.

VIII. CONCLUSIONS

This paper 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.

IX. ACKNOWLEDGEMENTS

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

X. REFERENCES

[1] Nursebot Project. http://www-2.cs.cmu.edu/~nursebot

[2] Morpha Project. http://www.morpha.de

[3] ActivMedia Robotics. http://robots.activmedia.com

[4] S. Thrun, "Probabilistic Algorithms in Robotics". *Technical Report CMU-CS-00-126*. 2002.

[5] A.R. Cassandra, L. P. Kaelbling and J.A. Kurien, "Acting under uncertainty: discrete bayesian models for mobile robot navigation". *Proc of IEEE/RSJ Int. Conference on Intelligent Robots and Systems*. 1996.

[6] I. Nourbakhsh, R. Powers and S. Birchfield. "DERVISH: an office navigating robot". *Artificial Intelligence Magazine*, 16(2). 1995.

[7] S. Koenig and R. Simmons. "Xavier: a robot navigation architecture based on partially observable Markov decision process models". *Artificial Intelligence and Mobile Robots*, pp. 91-122. 1998.

[8] M.E. López, R. Barea, L.M. Bergasa y M.S. Escudero. "Visually augmented POMDP for indoor robot navigation". *Proc. of the 21th IASTED Int. Multi-Conference on Applied Informatics*, pp. 183-187. 2003.

[9] M.E. López, L.M. Bergasa, R. Barea, M.S. Escudero. "Topological robot navigation using multisensorial event-based POMDPs". *Proc. of the 11th International Conference on Advanced Robotics (ICAR 03)*, pp. 216-221. 2003.

[10] D.P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, Belmont, Massachusetts, vol. 1,2. 1995.

[11] M.L. Puterman. *Markov Decision Processes-Discrete Stochastic Dynamic Programming*. John Wiley & sons, NY. 1994.

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

[13] J.L. Cuenca, M.C. Reyes. "La entropia: medida de la desigualdad". I Congreso de Ciencia Regional de Andalucía, pp. 853-861. 1997.

[14] P.J. McKerrow. *Introduction to Robotics*. Addison Wesley. 1991.