A POMDP approach to cooperative localization in sparse environments

Francisco S. Melo* Isabel Ribeiro*

Abstract

In this paper we adopt the reinforcement learning formalism to address the problem of cooperative localization and navigation in sparse environments. We consider situations in which a group of robots navigates in a sparse environment described by a topological map. Each robot must move from an initial position to a goal position, known only to that single robot. In this paper we discuss how communication can be used advantageously for cooperative navigation in sparse environments. Specifically, we analyze the tradeoff between communication cost and efficient completion of the navigation task. We make use of a partially observable Markov decision process (POMDP) to model each robot in the environment. The use of the POMDP framework allows to explicitly consider the tradeoff between information-gathering actions and actions that move the robot towards the goal. By explicitly including communication in the POMDP model as an information-gathering action with an associated cost, we are able to optimally settle this tradeoff between the gain in information arising from the use of communication and the corresponding cost. We illustrate our results in a small test application.

1 Introduction

Consider a group of N robots moving in a *sparse environment*. This environment is described by a topological map with M nodes (see, for example, Fig. 1). We refer to the nodes in the map as *states*. Each robot must navigate from an initial state to a goal state, known only to that single robot. We admit that several states in the environment have *distinctive landmarks* that the robots can generally perceive through its sensors. However, until a robot is able to reach one such state and observe the corresponding landmark, it must generally navigate through several other states receiving no sensorial feedback from the environment (*e.g.*, using only dead-reckoning).

*The authors are with the Institute for Systems and Robotics, Instituto Superior Técnico, Lisboa, Portugal. E-mail: [fmelo,mir]@isr.ist.utl.pt. This work was partially supported by POS_C that includes FEDER funds. The first author acknowledges the PhD grant SFRH/BD/3074/2000.

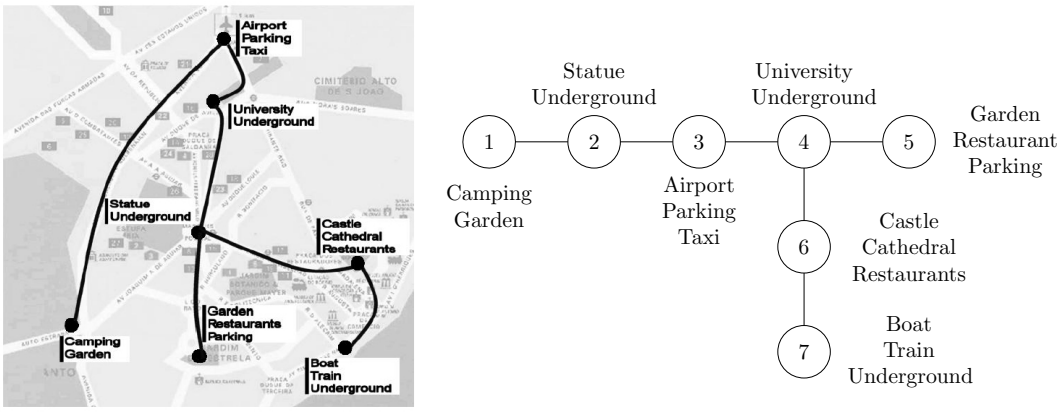

Figure 1: Example of a topological map with 7 states (or nodes).

Throughout the paper we admit that local navigation (such as obstacle avoidance and motion control) is handled by low-level controllers and focus only on the problems of global localization and navigation/planning. We model the navigation task as a sequence of decisions: at each decision instant, each robot must choose from a set of action primitives that control its movements. Examples of such primitives can be "Move north" or "Go to state i".

The robots are allowed to communicate with each other and *share* sensorial information. This received sensorial data can then be used by the robot to improve its localization in the environment. The use of communication to extend the sensorial capabilities of individual robots is particularly useful in the class of scenarios considered here, where the robots navigate for long periods using only dead-reckoning. Navigation in such sparse environments, where little sensorial information is available, may greatly benefit from the presence of multiple robots: the use of distributed sensing will greatly improve the localization capabilities of each robot in the group.

We emphasize that, even if each robot can reach its goal without ever considering the existence of other robots in the environment, communication can help a robot to do it more effectively. This defines the cooperation mechanism for this scenario: the robots *cooperate* by *sharing information*. In this paper we analyze those situations in which it may be advantageous for a robot to *communicate* so as to more effectively accomplish its mission.

If, as argued above, communication can be used to improve the sensorial capabilities of each robot, it is also true that the communication process generally takes time and consumes resources. Furthermore, it may happen occasionally that no useful information is received. Therefore, it is important to realize in which situations is communication advantageous and in which situations it should be avoided. For example, it may happen that the cost of communication is much higher than the benefit obtained from it. To settle this problem, we make use of a *partially observable Markov decision process* (POMDP) to model each robot in the environment. POMDPs have successfully been used for topological navigation [7, 13, 15, 19]

and are particularly amenable to the use of Markov localization methods [9]. Furthermore, POMDPs explicitly consider the tradeoff between choosing actions to *disambiguate the state of the robot* (information-gathering) and actions to *move the robot towards the goal*. By explicitly including *communication* in the POMDP model as an information-gathering action with an associated cost, we are able to optimally settle this tradeoff between the gain in information arising from the use of communication and the corresponding cost. This appealing feature of the POMDP framework has led some researchers to address *active sensing* using POMDP models [22].

The paper is organized as follows. In Section 2 we introduce the general POMDP framework. In Section 3 we apply this framework to model the particular problems addressed in the paper. We introduce a simple illustrative example that is used throughout the paper to illustrate the main ideas. In Section 4 we present the results obtained with our approach and outlining several important aspects of our results. Finally, Section 5 concludes the paper with a summary of the main conclusions and points out several possible directions for future work.

2 Partially observable Markov decision processes

A POMDP is a tuple $(\mathcal{X}, \mathcal{A}, \mathcal{Z}, \mathsf{P}, \mathsf{O}, r, \gamma)$, where $\mathcal{X}$ is the finite set of possible states of the system, $\mathcal{A}$ is a finite set of control primitives and $\mathcal{Z}$ corresponds to a finite set of possible observations. At each time instant t, a decision-maker chooses an action A_t depending on the past history of events, causing the system to move from its current state X_t to state X_{t+1} according to the transition probabilities

$$\mathbb{P}\left[X_{t+1} = j \mid X_t = i, A_t = a\right] = \mathsf{P}_a(i, j),$$

where $i, j \in \mathcal{X}$ and $a \in \mathcal{A}$. Therefore, $\mathsf{P}_a(i, j)$ represents the probability of moving from state i to state j under action a and depends only on the state X_t of the system at time instant t and the corresponding action at that time instant, A_t. As soon as the transition occurs, the decision-maker receives an observation Z_{t+1} that depends on the new state of the system according to the observation probabilities

$$\mathbb{P}\left[Z_{t+1} = z \mid X_{t+1} = j, A_t = a\right] = \mathsf{O}_a(j, z),$$

where $j \in \mathcal{X}$, $a \in \mathcal{A}$ and $z \in \mathcal{Z}$. Therefore, $\mathsf{O}_a(j, z)$ represents the probability of observing z when the state of the system is j and action a was taken. Also, as soon as the transition occurs, the decision-maker is granted a numerical reward $r(i, a, j)$, verifying $|r(i, a, j)| \leq R_{\max}$. The purpose of the decision-maker is to choose the control sequence $\{A_t\}$ so as to maximize the functional

$$V(b_0, \{A_t\}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(X_t, A_t, X_{t+1}) \mid X_0 \sim b_0\right], \tag{1}$$

where $\gamma < 1$ is a positive discount factor, b_0 is the *initial belief* for the process and $X_0 \sim b_0$ denotes the fact that X_0 is distributed according to b_0. The initial belief b_0 is a probability vector describing the initial distribution of the state of the system.

In this paper, we are interested in using a POMDP model to describe a robot moving in a sparse environment described topologically. Using a POMDP model, the state-space $\mathcal{X}$ corresponds to the set of sites in the environment. The state of the system at time t, X_t, corresponds to the position of the robot in the environment at time t. The control primitives, or *actions*, correspond to the high-level navigation commands that allow the robot to move between sites in the environment and the observations correspond to the sensorial information. With a POMDP model, Markov localization [9, 10] can be implemented in a straightforward way: at each time instant t the robot maintains a *belief-vector* b_t, each component $b_t(i)$ describing the probability of being in a particular state $i \in \mathcal{X}$. This belief-vector is updated componentwise as

$$b_{t+1}(j) = \mathsf{B}_a(b,z)_j = \frac{\sum_{i\in\mathcal{X}} b_t(i)\mathsf{P}_a(i,j)\mathsf{O}_a(j,z)}{\sum_{i,k\in\mathcal{X}} b_t(i)\mathsf{P}_a(i,k)\mathsf{O}_a(k,z)},$$

where $A_t = a$ and $Z_{t+1} = z$ and $\mathsf{B}_a(b,z)_j$ represents the jth component of the vector $\mathsf{B}_a(b,z)$.

Given the POMDP $(\mathcal{X}, \mathcal{A}, \mathcal{Z}, \mathsf{P}, \mathsf{O}, r, \gamma)$, the *optimal value function* V^* is defined for each belief b as

$$V^*(b) = \max_{\{A_t\}} V(\{A_t\}, b) = \max_{\{A_t\}} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(X_t, A_t, X_{t+1}) \mid X_0 \sim b_0\right]$$

and verifies

$$V^*(b) = \max_{a\in\mathcal{A}} \sum_{i,j\in\mathcal{X}} b(i)\mathsf{P}_a(i,j)\left[r(i,a,j) + \gamma \sum_{z\in\mathcal{Z}} \mathsf{O}_a(j,z)V^*\big(\mathsf{B}_a(b,z)\big)\right]$$

which is a form of the Bellman optimality equation. We remark that $V^*(b)$ represents the expected total discounted reward received along an optimal trajectory starting from the initial state distribution b. The optimal decision rule can be defined by means of the mapping

$$\pi^*(b) = \arg\max_{a\in\mathcal{A}} \sum_{i,j\in\mathcal{X}} b(i)\mathsf{P}_a(i,j)\left[r(i,a,j) + \gamma \sum_{z\in\mathcal{Z}} \mathsf{O}_a(j,z)V^*\big(\mathsf{B}_a(b,z)\big)\right].$$

The optimal control process is then given by $A_t = \pi^*(b_t)$ and the mapping π^* is the *optimal policy* for the POMDP $(\mathcal{X}, \mathcal{A}, \mathcal{Z}, \mathsf{P}, \mathsf{O}, r, \gamma)$.

2.1 Incremental pruning

There are numerous works in the literature surveying POMDPs [1, 6]. The methods specifically developed to address decision problems in partially observable scenarios can grossly be divided into three main classes, namely:

- *Exact methods* that seek to determine the exact optimal policy or a corresponding value-function from which the optimal policy can be determined. Exact methods include the *witness algorithm* [11] and the *incremental pruning algrithm* [8];

- *Methods for particular problems*, designed to address specific problems, exhibiting particular properties [17, 18];

- *Approximate methods* that provide sub-optimal solutions. This is, by far, the widest and most diverse of all three classes and includes numerous algorithms supported by either formally justified approximations [2, 14, 20] or by more heuristic reasoning [13, 16].

For the purposes of our study, we adopt the incremental prunning (IP) algorithm [8], that we now describe.

Several important observations are now in order. We start by conveying the idea of *finite horizon*. In (1), we claimed that the purpose of the decision-maker was to choose a sequence of actions $\{A_t\}$ so as to maximize the total expected discounted reward, received along an *infinite trajectory* of the process. This is known as an *infinite-horizon* problem: the decision-maker must choose its actions taking into account the fact that there are still infinitely many decisions remaining. When the number of decisions is known and finite, the problem is referred as a *finite-horizon* problem.[1] In particular, an *h-horizon problem* corresponds to a situation where the decision-maker has h decisions to make.

The fundamental difference between finite and infinite-horizon problems is that, in finite-horizon problems, the optimal policy is *non-stationary*, *i.e.*, it explicitly depends on time. To see why this is so, consider the simple situation depicted in Figure 2.

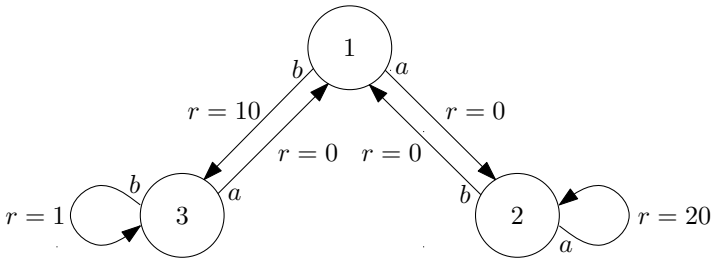

Figure 2: A simple example to illustrate the non-stationarity of finite-horizon policies.

The optimal policy for this system, in the 1-horizon and 2-horizon situations, is summarized in Table 1.a). Let us consider the 1-horizon situation first. In this situation, the decision-maker has a single decision to make, trying to maximize the received reward. This means that, if the decision-maker is in state 1 it should choose action b, so as to collect a reward

[1] Infinite-horizon problems also model situations in which the horizon is finite but unknown. In that case, the discount factor γ can be interpreted as the probability of the process continuing after one decision is taken.

Table 1: a) Optimal policy for the 1-horizon and 2-horizon problems, corresponding to the system in Figure 2. b) Value of each action in the 2-horizon situation, taking into account the value of the next state (we consider $\gamma = 0.95$).

a) Optimal policy;

State	$h = 1$	$h = 2$
1	b	a
2	a	a
3	b	a

b) Value of state-action pairs.

State/Action	a	b
1	**19**	10.95
2	**39**	9.5
3	1.95	**9.5**

of 10. In state 2 the obvious choice is action a, leading to a reward of 20 and in state 3 the optimal choice is action b, leading to a reward of 1.

However, if we consider the 2-horizon situation, the situation is very different. In this situation, considering that there are still *two* decisions to take, the decision-maker should take into account the *value* of the subsequent state. This value is the reward of the 1-horizon policy and in Table 1.b) we summarize the value of each action in each state for the 2-horizon policy. The difference between the two columns should be obvious and clearly illustrates the point that the optimal policy in the finite-horizon problem is time-dependent.

This, in turn, means that the optimal value-function in the finite-horizon case, is also time-dependent, and now verifies the following recursion

$$V_{h+1}^*(b) = \max_{a \in \mathcal{A}} \sum_{i,j \in \mathcal{X}} b(i) \mathsf{P}_a(i,j) \left[r(i,a,j) + \gamma \sum_{z \in \mathcal{Z}} \mathsf{O}_a(j,z) V_h^*\big(\mathsf{B}_a(b,z)\big) \right], \tag{2}$$

where V_h^* is the value-function for the h-horizon problem.

The second important observation is that, for any finite-horizon problem, the optimal value function is *piecewise-linear and convex* (PWLC) [6]. A piecewise linear and convex function can be compactly represented by resorting to a finite set of vectors $\{\alpha_1, \ldots, \alpha_n\}$, that we henceforth refer as α-vectors [6]. Any such function V at a belief point b takes the general form

$$V(b) = \max_{\alpha_k} \sum_i \alpha_k(i) b(i) = \max_{\alpha_k} \alpha_k^\top b.$$

We denote the minimum set of vectors necessary to represent a PWLC function V by $\Gamma(V)$ and refer to it as its *parsimonious representation*. To illustrate the importance of the concept of parsimonious representation, consider the PWLC function depicted in Figure 3. In this case we have $\Gamma(V) = \{\alpha_1, \alpha_2, \alpha_3\}$ and α_4 is not used in this representation. This is due to the fact that there is a single point b^* at which $V(b^*) = \alpha_4^\top b^*$. Each vector α_k in $\Gamma(V)$ has a

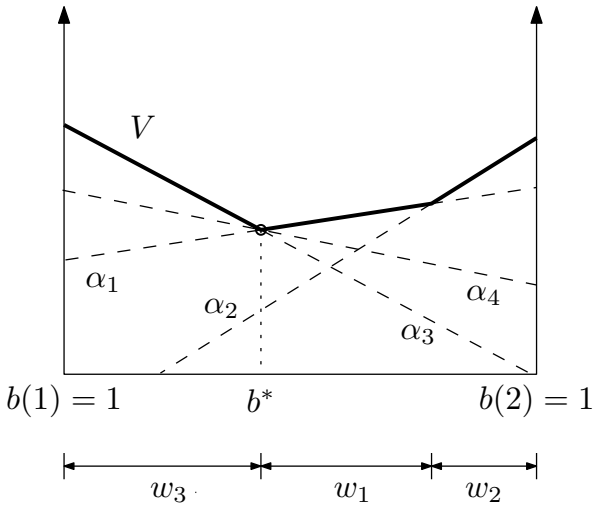

Figure 3: Piecewise linear and convex function V, witness regiones w_k, and unused α-vector α_4.

witness region, *i.e.*, a region w_k in which, for all $b \in w_k$,

$$V(b) = \alpha_k^\top b.$$

Finally, we introduce some additional notation that greatly simplifies the description of the incremental prunning algorithm. From (2), we have

$$\begin{aligned} W^*_{h+1}(b,a,z) &= \sum_{i,j\in\mathcal{X}} b(i)\mathsf{P}_a(i,j)\left[\frac{r(i,a,j)}{|\mathcal{Z}|} + \gamma \mathsf{O}_a(j,z)V^*_h\big(\mathsf{B}_a(b,z)\big)\right]; \\ Q^*_{h+1}(b,a) &= \sum_{z\in\mathcal{Z}} W^*_{h+1}(b,a,z); \\ V^*_{h+1}(b) &= \max_{a\in\mathcal{A}} Q^*_{h+1}(b,a). \end{aligned}$$

It is clear, by replacing the definitions of W^*_{h+1} and Q^*_{h+1}, that V^*_{h+1} as defined above is equivalent as V^*_{h+1} as defined in (2). For fixed a and z, each of the functions $W^*_{h+1}(\cdot,a,z)$, $Q^*_{h+1}(\cdot,a)$ and V^*_{h+1} is PWLC and has, therefore, a parsimonious representation.

It is now possible to describe the IP algorithm. Each iteration of IP starts by computing, for each $a \in \mathcal{A}$ and each $z \in \mathcal{Z}$, the parsimonious representation $\Gamma\big(W^*_{h+1}(\cdot,a,z)\big)$ from $\Gamma(V^*_h)$. It then proceeds by computing $\Gamma\big(Q^*_{h+1}(\cdot,a)\big)$ from all $\Gamma\big(W^*_{h+1}(\cdot,a,z)\big)$, for each $a \in \mathcal{A}$. Finally, it computes $\Gamma\big(V^*_{h+1}\big)$ from all $\Gamma\big(Q^*_{h+1}(\cdot,a)\big)$. The details on how the computation of each set can be implemented efficiently are out of the scope of this paper, and we refer to [5, 8] for details.

3 The POMDP model

In the previous section, we have described the POMDP framework and the IP algorithm, a finite-horizon exact solution method. In this section we describe how this framework can be used to address the class of navigation problems at hand.

We start by remarking that any infinite-horizon problem with discount factor γ can be approximated by a h-horizon problem with discount factor $\gamma_u = \sqrt{\gamma}$, with

$$h = \log_{\gamma_u}\left(\frac{\varepsilon(1-\gamma_u)}{R_{\max}}\right),$$

where ε is the maximum error in the approximation (in terms of value-function). This means that we can use the IP algorithm to approximate the optimal infinite-horizon value-function as closely as desired, and that will be our course of action in this paper.

We are interested in addressing the situation in which a group of robots moves in a sparse environment, each robot trying to reach its goal location. We use a POMDP model to analyze how the robots can benefit from efficiently using communication, by explicitly including *communication* in the POMDP model as an information-gathering action with an associated cost. The optimal POMDP policy optimally settle this tradeoff between the gain in information arising from the use of communication and the corresponding cost.

Two important observations are in order. First of all, communication between two robots is modeled as a *directed exchange of sensorial information*. This means that, when communicating, a robot will explicitly request the sensorial information from a *specific robot*. In other words, we assume communication to be *peer-to-peer* and not broadcasted.

Secondly, even though we consider the existence of multiple communicating robots in the environment, *we do not address the interaction between these robots*. In particular, the actions of one robot do not affect the behavior of any other robot nor its ability to reach the goal. Therefore, each robot can be modeled independently of the other robots and there is no need to consider multi-agent decision models, such as Dec-POMDPs or stochastic games. Problems where the interaction of multiple robots (or, more generally, multiple decision-makers) must be explicitly considered are more suitably addressed from a game theoretic perspective and are thouroughly treated in the literature [4, 12], although the computational complexity involved in dealing with such models is prohibitive [3].

$\diamond$

From the discussion above, it should be clear that we can focus our analysis on the behavior of a *single robot*, considering the other robots as part of the environment. To that, we resort to a simplified model that encompasses all the fundamental features of the class of navigation problems considered in the paper. This model is represented in Figure 4.

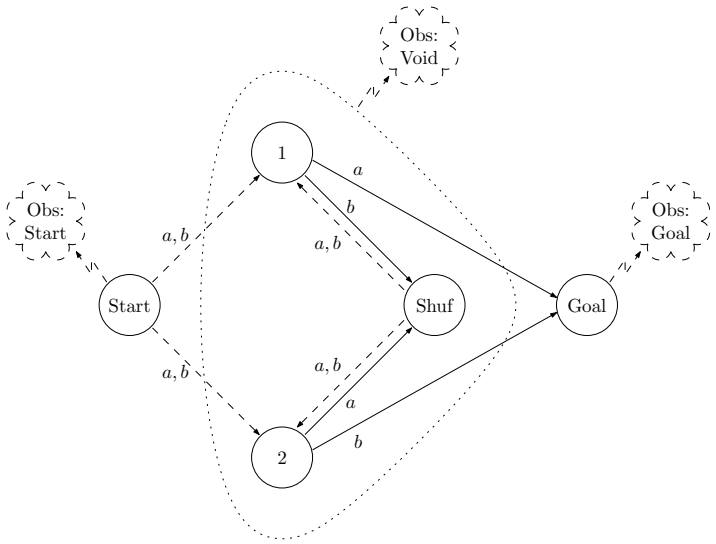

Figure 4: A general sparse environment. The dashed lines departing from a single node correspond to random, equiprobable transitions. For example, when at the starting state, the robot will move to states 1 or 2 with a 1/2 probability, independently of the chosen action.

In this simplified model, the robot departs from the state marked as "Start", by choosing any of the two available actions a or b. It then moves to either state 1 or state 2 with equal probability. All three states inside the dotted line are *indistinguishable*. The robot will then move to the state marked as "Goal" by choosing a in state 1 or b in state 2 and to the state marked as "Shuf", otherwise.

Upon reaching the Goal state, the robot receives a reward of $+20$ and upon reaching the Shuf state, it receives a reward of -5. At the Shuf state, independently of the robot's action, its position is, once again, randomly reset to either state 1 or 2, with equal probability.

Finally, in this model, the robot has 3 available observations: "Start", in the Start state, "Void", in the states inside the dotted line, and "Goal", in the goal state.

In terms of navigation in sparse environments, the set of three undistinguishable states describe those situations in which the robot gets lost due to long periods of dead-reckoning navigation. Upon reaching the Goal state, the robot is back to a location with distinguishing features and can use this information to localize once again. In this simplified model, the robot merely ignores the existence of other robots and just chooses its actions so as to maximize its total reward (reaching the Goal state as quiclly as possible).

This scenario can be described as a $(\mathcal{X}, \mathcal{A}, \mathcal{Z}, \mathsf{P}, \mathsf{O}, r, \gamma)$, where:

- $\mathcal{X} = \{\text{Start}, 1, 2, \text{Shuff}, \text{Goal}\}$;
- $\mathcal{A} = \{a, b\}$;
- $\mathcal{Z} = \{\text{Start}, \text{Void}, \text{Goal}\}$;

Table 2: Comparison between the performance of the cooperative robot and the non-cooperative.

Test	Total disc. reward		
Non-cooperative	58.118	±	22.185
Cooperative	81.498	±	9.867

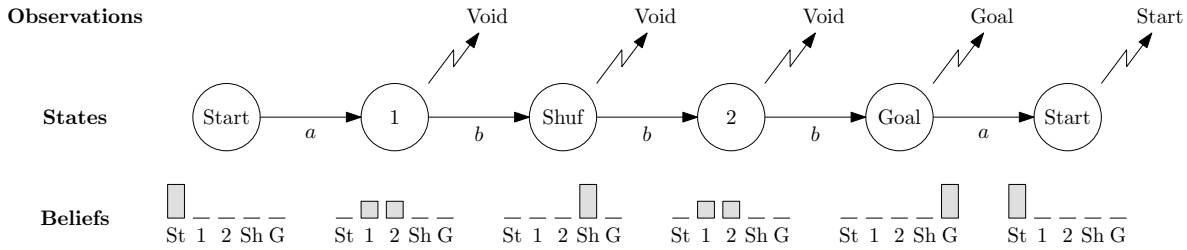

Figure 5: Sample trajectory when following the optimal policy in the non-cooperative robot navigation scenario.

- The transitions marked with solid lines in Figure 4 occur with probability 1, when the corresponding actions are chosen; The transitions marked with dashed lines reset the position of the robot to either state 1 or 2 with equal probability, independently of the action chosen by the robot;
- All observations occur with probability 1 in the corresponding states, illustrated in Figure 4;
- The reward function r is as described above;
- We consider $\gamma = 0.95$.

We ran IP to compute the optimal policy for this POMDP, to evaluate the performance of a robot navigating in a sparse environment discarding the existence of other robots in the environment. We then tested the performance of the policy, by running 2000 independent Monte Carlo trials, each consisting of a 10-time step trajectory, and computed the average total discounted reward. The results are reported in Table 2.

To better understand the behavior of the robot in this scenario, Figure 5 summarizes a sample trajectory under the optimal policy. The central diagram corresponds to the sequence of actual states visited by the robot and the bottom diagrams to the sequence of corresponding beliefs. Each belief is a 5-dimensional vector indicating the probability of being in each of the states. The observations after each transition are depicted on the top.

The robot starts on the "Start" state and since this state is distinguishable, the belief indicates that the robot is aware of its current position. However, after one transition (under

action a) the robot no longer knows its current state: it could be either state 1 or 2 with equal probability. The robot chooses action b and, since the underlying state was state 1, moves to state "Shuf".

At this point, the robot knows it is not in the "Goal" state, since otherwise it would have received the "Goal" observation. The only alternative is the "Shuf" state, as seen in the corresponding belief. At this state, independently of the robot's action, it will get lost between states 1 and 2. Once again, the robot chooses action b that, this time, successfully leads to the "Goal" state.

We emphasize that, when lost, the robot can only "bet" in one of the two possible actions a and b, hoping that it will lead to the desired outcome. In the sample trajectory described, the robot always chose action b, which lead to a successful outcome 50% of the time. One particularly "lucky" run can actually bring quite a large reward, while one particularly "unlucky" run could bring an alarmingly large penalty. This justifies the large standard deviation observed in the results portrayed in Table 2.

Another important aspect of the behavior just described is that it does match that expected from a robot navigating in a sparse environment: whenever it gets lost, it "keeps moving" in a direction where a recognizable state is expectable.

We now describe how the model in Figure 4 is modified to include the existence of another robot in the environment. As before, the robot departs from the state marked as "Start", by choosing any of the two available actions a or b. It then moves to either state 1 or state 2 with equal probability. The robot will then move to the state marked as "Goal" by choosing a in state 1 or b in state 2 and to the state marked as "Shuf", otherwise. Unlike the situation analyzed before, however, the robot has now two further actions available, dubbed as "Comm 1" and "Comm 2". None of the latter actions affects the position of the robot in the environment. Instead, each action "Comm i", $i = 1, 2$, sends a message "from" state i. This message is sent to a second robot (Robot B) who, upon receiving it, will send the first robot (Robot A) its sensorial information. Notice that such communication actions only suceed in the corresponding states. This means that, if the robot sends a message "Comm i" when at state $j \neq i$, there is a high probability of not receiving any reply.

Finally, as before, Robot A receives a reward of $+20$ upon reaching the goal and of -5 upon reaching the Shuf state. It receives a reward of -1 for every communication action.

This scenario can be described as a $(\mathcal{X}, \mathcal{A}, \mathcal{Z}, \mathsf{P}, \mathsf{O}, r, \gamma)$, where:

- $\mathcal{X} = \{\text{Start}, 1, 2, \text{Shuff}, \text{Goal}\}$;
- $\mathcal{A} = \{a, b, \text{Comm } 1, \text{Comm } 2\}$;
- $\mathcal{Z} = \{\text{Start}, \text{Void}, \text{Goal}, 1, 2\}$;
- Transitions under actions a and b occur as in the previous situation; communication actions leave the state unchanged;

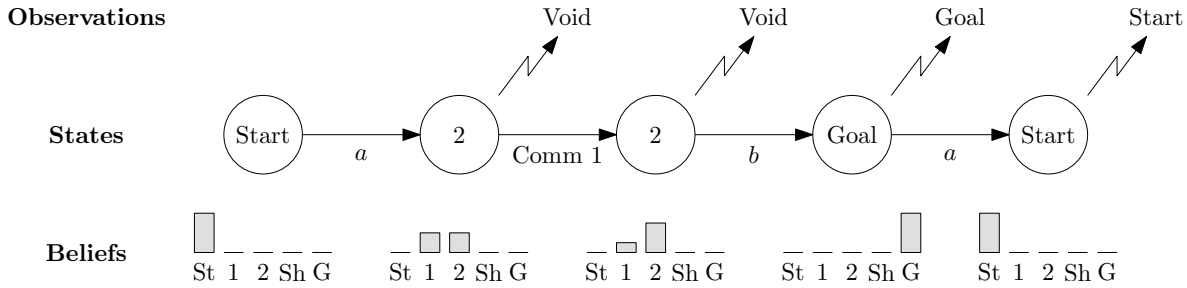

Figure 6: Sample trajectory when following the optimal policy in the cooperative robot navigation scenario.

- Observations "Start", "Void" and "Goal" occur with probability 1 in the corresponding states, when actions a or b are taken; observation i occurs in state i with 0.8 probability, when action "Comm i" is taken, $i = 1, 2$. With a 0.2 probability, the "Void" observation occurs; observation i occurs in state i with a 0.4 probability, when action "Comm j" is taken, with $i, j = 1, 2$ and $i \neq j$;
- The reward function r is as described above;
- We consider $\gamma = 0.95$.

Two important observations are in order. First of all, notice that there is a cost involved in communication, even if less significant than getting into the "Shuff" state. Secondly, the communication may not succeed. This can happen either because Robot A chose the "wrong" communication action, or simply because Robot B can give Robot A no information on its position (*e.g.*, Robot B cannot "see" Robot A).

We ran IP and computed the optimal policy for this POMDP, comparing the performance of the robot with the one observed from the non-cooperative robot. As before, we tested the performance of the optimal policy by running 2000 independent Monte Carlo trials, each consisting of a 10-time step trajectory, and computed the average total discounted reward. The results are reported in Table 2.

Notice, first of all, the tremendous difference in performance between the two robots. The robot relying on communication exhibited an average increase in performance of about 30%, even receiving the negative rewards arising from communication. Furthermore, the optimal policy in the presence of communication leads to a much more reliable performance, since the observed standard deviation is much smaller.

To better understand the behavior of the robot in this scenario, we once again summarize in Figure 6 a sample trajectory under the optimal policy. Notice that, in the particular trajectory depicted in Figure 6, communication was unsuccessful, but even the failing of communication contributes for the localization of the robot.

Finally, to assess the explicit tradeoff between the cost of communication and the "value

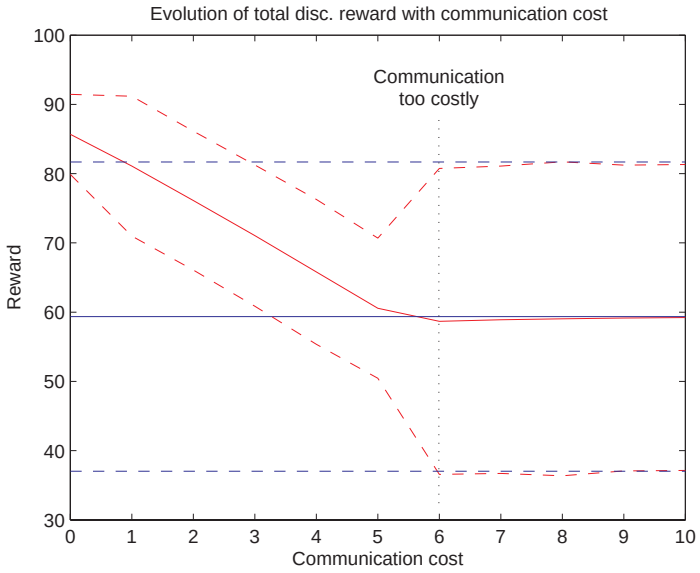

Figure 7: Tradeoff between the cost of communication, r_{comm}, and the value of information, in terms of the performance of the robot. The blue line corresponds to the non-cooperative robot and the red line to the cooperative robot. The line in solid corresponds to the mean performance from 2000 independent Monte-Carlo trials. The dotted lines correspond to the standard deviation.

of information", we have conducted similar tests, varying the communication cost, r_{comm}, between 0 and -10. The corresponding results are summarized in Figure 7.

Notice that, as the cost of communication increases, the performance of the cooperative (communicating) robot approaches that of the non-cooperative robot. The two performances reach a similar level when the cost of communication is similar to that of getting lost (*i.e.*, $r_{\text{comm}} = -5$). An important aspect to emphasize is that, when $r_{\text{comm}} = -5$, the performance of the optimal policy in the cooperative case is much more *reliable* than that of the non-cooperative case. This is easily seen from the standard deviation observed. Therefore, even at a high cost, the robot does rely on communication to navigate and this actually translates in an actual improvement in terms of performance. Finally, for $r_{\text{comm}} \geq 6$, communication becomes too costly, as indicated in Figure 7. This means that the optimal policy in both the cooperative and non-cooperative case are similar, as seen from the performance observed in Figure 7.

In the continuation, we describe a larger navigation scenario in which communication is successfully used to accomplish a navigation task, in the presence of little sensorial information.

4 A simple illustrative result

Consider the environment described in Figure 8. A robot starts in any state between 1 and 16 with equal probability and must reach the state marked with 17 and a double line. The

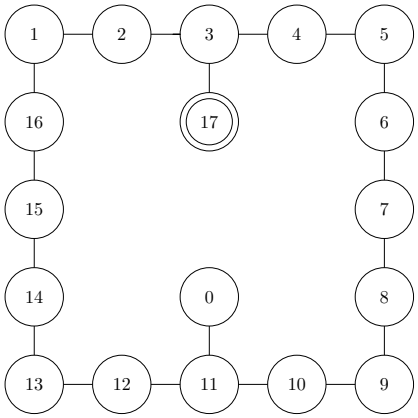

Figure 8: A two-robot navigation environment.

state marked as 0 is an undesirable state.

The robot has 4 navigation actions available, namely N, S, E and W. If no adjacent state exists in the corresponding direction, the robot will remain in the same state with probability 1. If *one* adjacent state exists in the corresponding direction, the robot will move to this state with a 0.75 probability and, with a 0.25 probability, remain in the same cell. Finally, if *two* consecutive adjacent states exist in the corresponding direction, the robot will remain in the same state with a 0.25 probability, it will move to the immediately adjacent state with a probability of 0.5 and to the third state with a 0.25 probability. The robot receives a reward of $+20$ upon reaching the goal state, -5 upon reaching the 0 state and 0 otherwise.

We admit that the robot is able only to detect the goal state, *i.e.*, state 17. This means that the robot essentially navigates using dead-reckoning and only the goal state provides sensorial feedback that the robot can use to localize. The discount factor was 0.95.

In the multi-robot situation, we admit the robot (henceforth referred to as Robot A) to have one extra action, dubbed as "Comm". This action allows Robot A to communicate with a second robot (Robot B) wandering in the environment, as long as this robot is within reach. If the Robot B is within reach, it will send robot A a message indicating the position of Robot A *as perceived by Robot* B. Robot B will accurately perceive the position of Robot A with a 0.5 probability and, with a 0.25 probability each, it will perceive the position of Robot A to be one of its adjacent states. This is summarized in Figure 9.

Once again, for every communication action, Robot A receives a reward of -1 and its state remains unchanged. Notice that, unlike the scenario described in the previous section, where communication succeeded very often, in this scenario communication will *fail* very often, since Robot B will often be out of reach.

We run IP for both problems, computing the optimal policy with and without communication. We then tested both optimal policies in the environment for a trial period of 10

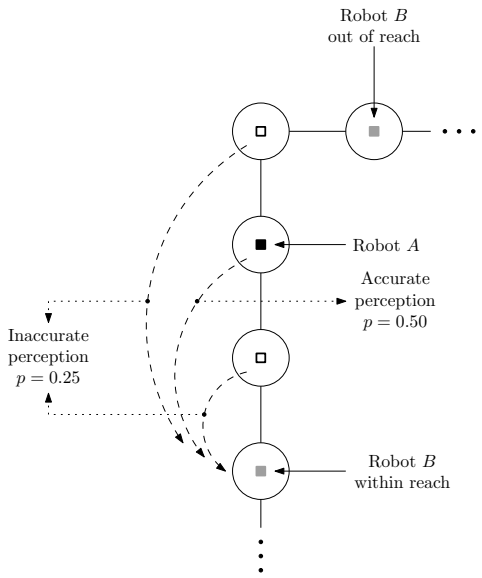

Figure 9: Illustration of the interaction between the two robots. The black rectangle corresponds to a possible position of Robot A. The bottom-most gray rectangle corresponds to the furthest position for Robot B to be within range, except if a corner is in between Robots A and B, as indicated in the top-most gray rectangle. Also, when Robot A is within reach, Robot B will have an accurate perception of the right position of Robot A only with a probability $p = 0.5$.

time-steps.[2] We ran 2000 independent Monte-Carlo runs and determined the average total discounted reward obtained using each of the two policies. The results are summarized in Table 3.

Table 3: Comparison between the performance of the cooperative robot and the non-cooperative in the large navigation scenario of Figure 8.

Test	**Total disc. reward**
Non-cooperative	0
Cooperative	73.756

Several important observations are in order. First of all, it is worth mentioning that, in the non-cooperative case, the robot *starts lost* and is never able to recover from this lost situation, since there is no sensorial feedback available in any state but the goalstate. One remarkable thing that is worth mentioning is that, in spite of being completely lost, the robot is still able to *avoid the undesirable state*, even if this is clearly easier than reaching the goal.

[2]Notice that 10 time-steps is the time it takes the robot to move from the furthest state to the goal state, if *every* transition succeeds.

Secondly, the cooperative robot is able to reach the goal, in average, *more than 3 times in the* 10 *time-steps period*. This is clear from the average obtained reward: since only reaching the goal does the robot receive a reward of +20, a total reward superior to +60 indicates that the goal must have been reached more than 3 times. This is an extraordinary difference from the performance of the non-cooperative robot.

Finally, it is important to mention that the optimal policy for the non-cooperative robot is more "cautious" than its cooperative counterpart. This will sometimes cause the cooperative robot to go into the undesirable state, causing it to receive a negative reward.

5 Conclusions and future work

In this paper we addressed the problem of cooperative localization and navigation in sparse environments. We considered situations in which a group of robots navigates in a sparse environment described by a topological map. We modeled each robot as an independent decision-maker and assumed each individual task to require no explicit interaction with the other robots to be completed.

We showed that, even if it is possible for each robot to reach its goal state without ever considering the existence of other robots in the environment, communication can greatly improve the performance of each robot. The cooperation mechanism in this class of cooperative problems takes the form of *information sharing*.

We made use of a POMDP to model each robot in the environment and were able to explicitly consider the tradeoff between choosing actions to *disambiguate the state of the robot* (information-gathering) and actions to *move the robot towards the goal*. By explicitly including *communication* in the POMDP model as an information-gathering action with an associated cost, we were able to optimally settle this tradeoff between the gain in information arising from the use of communication and the corresponding cost. We concluded by applying our approach to a small test problem.

As seen in this paper, research into more general active sensing problems can greatly benefit from models and methods used in reinforcement learning and dynamic programming. It is possible to address active sensing from a decision-theoretic perspective by casting any active sensor as an *autonomous decision-maker*. Models such as POMDPs and the powerful methods available from reinforcement learning and dynamic programming can, therefore, advantageously be applied to address the challenging problem of active sensing [21, 22]. A decision-theoretic approach to active sensing circumvents the need to provide exact or complete domain knowledge *a priori* and naturally leads to adaptive but robust decision systems. On the other hand, many decision-theoretic models such as POMDPs naturally encompass partial observability and uncertain environment dynamics, which makes these models adequate to address problems of perceptual aliasing.

This naturally leads to several interesting avenues for future research. First of all, decision-theoretic models as the one described in this paper can be further explored to analyze situations in which communication is used to combine the sensing information from different sources. In this paper we optimized the resort to communication for the purpose of navigation in sparse environments. However, the same underlying idea can also be explored for the purpose of sensor fusion and so as to optimize the total information obtained from the sensorial data (this information can be quantified for example in terms of the entropy regarding the distribution of the underlying state of the process).

Secondly, multi-agent variations of the model used here can also be used to address active sensor networks, by casting sensor networks as *communicating, cooperative multi-agent systems*. A possible approach is to consider independent decision-making and identify the conditions under which an optimal decision-rule can be implemented in a distributed way when multiple decision-makers exist. In this process, the trade-off between the loss in performance due to the independent implementation of this decision-rule and the cost of communication should become apparent. Therefore, it should be possible to design an algorithm in which this trade-off is explicitly addressed or, at least, can be explicitly settled by an external user.

References

[1] Douglas A. Aberdeen. A (revised) survey of approximate methods for solving partially observable Markov decision processes. Technical report, National ICT Australia, Canberra, Australia, 2003.

[2] Peter L. Bartlett and Jonathan Baxter. Estimation and approximation bounds for gradient-based reinforcement learning. In *Proceedings of the 13th Annual Conference on Computional Learning Theory (COLT'00)*, pages 133–141, San Francisco, CA, 2000. Morgan Kaufmann Publishers.

[3] Daniel S. Bernstein, Shlomo Zilberstein, and Neil Immerman. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27 (4):819–840, 2002.

[4] Michael Bowling and Manuela Veloso. An analysis of stochastic game theory for multiagent reinforcement learning. Technical Report CMU-CS-00-165, School of Computer Science, Carnegie Mellon University, 2000.

[5] Anthony R. Cassandra. *Exact and approximate algorithms for partially observable Markov decision processes*. PhD thesis, Brown University, May 1998.

[6] Anthony R. Cassandra. Optimal policies for partially observable Markov decision pro-

cesses. Technical Report CS-94-14, Department of Computer Sciences, Brown University, August 1994.

[7] Anthony R. Cassandra, Leslie Kaelbling, and James A. Kurien. Acting under uncertainty: Discrete Bayesian models for mobile-robot navigation. *Mathematics of Operations Research*, 12(3):441–450, 1987.

[8] Anthony R. Cassandra, Michael L. Littman, and Nevin L. Zhang. Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes. In *Proceedings of the 13th Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 54–61, Providence, Rhode Island, 1997. Morgan Kaufmann Publishers.

[9] Dieter Fox. *Markov localization: A probabilistic framework for mobile robot localization and navigation*. PhD thesis, University of Bonn, Germany, 1998.

[10] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11:391–427, 1999.

[11] Michael L. Littman. The Witness algorithm: Solving partially observable Markov decision processes. Technical Report CS-94-40, Department of Computer Sciences, Brown University, December 1994.

[12] Michael L. Littman. Value-function reinforcement learning in Markov games. *Journal of Cognitive Systems Research*, 2(1):55–66, 2001.

[13] Francisco S. Melo and M. Isabel Ribeiro. Transition entropy in partially observable Markov decision processes. In T. Balch T. Arai, R. Pfeifer and H. Yokoi, editors, *Proceedings of the 9th International Conference on Intelligent Autonomous Systems (IAS-9)*, pages 282–289. IOS Press, 2005.

[14] Nicolas Meuleau, Leonid Peshkin, Kee-Eung Kim, and Leslie P. Kaelbling. Learning finite-state controllers for partially observable environments. In *Proceedings of the 15th International Conference on Uncertainty in Artificial Intelligence (UAI'99)*, pages 427–436, 1999.

[15] Nicholas Roy and Sebastian Thrun. Coastal navigation with mobile robot. In *Proceedings of 1999 Conference on Neural Information Processing Systems (NIPS'99)*, pages 1043–1049, 1999.

[16] Nicholas Roy, Geoffrey Gordon, and Sebastian Thrun. Finding approximate POMDP solutions through belief compression. *Journal of Artificial Intelligence Research*, 23:1–40, 2005.

[17] Paat Rusmevichientong and Benjamin Van Roy. A tractable POMDP for a class of sequencing problems. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence (UAI'01)*, pages 480–487. Morgan Kaufmann, 2001.

[18] Jamieson Schulte and Sebastian Thrun. A heuristic search algorithm for acting optimally in Markov decision processes with deterministic hidden state. Unpublished Manuscript, 2001.

[19] Reid Simmons and Sven Koenig. Probabilistic robot navigation in partially observable environments. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 1080–1087, 1995.

[20] Matthijs T. J. Spaan and Nikos Vlassis. Perseus: Randomized Point-based Value Iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24:195–220, 2005.

[21] Steven D. Whitehead and Dana H. Ballard. Learning to perceive and act by trial and error. *Machine Learning*, 7(1):45–83, 1991.

[22] Steven D. Whitehead and Long-Ji Lin. Reinforcement learning of non-Markov decision processes. *Artificial Intelligence*, 73(1-2):271–306, 1995.