# Learning state-variable relationships for improving POMCP performance

Maddalena Zuccotto
University of Verona
Verona, Italy
maddalena.zuccotto@univr.it

Alberto Castellini
University of Verona
Verona, Italy
alberto.castellini@univr.it

Alessandro Farinelli
University of Verona
Verona, Italy
alessandro.farinelli@univr.it

## ABSTRACT

**Accepted version of the manuscript. Please refer to https://dl.acm.org/doi/10.1145/3477314.3507049 for the published version.** We address the problem of learning state-variable relationships across different episodes in Partially Observable Markov Decision Processes (POMDPs) to improve planning performance. Specifically, we focus on Partially Observable Monte Carlo Planning (POMCP) and we represent the acquired knowledge with Markov Random Fields (MRFs). We propose three different methods to compute MRF parameters while the agent acts in the environment. Our techniques acquire information from agent action outcomes, and from the belief of the agent, which summarizes the knowledge acquired from observations. We also propose a stopping criterion to determine when the MRF is accurate enough and the learning process can be stopped. Results show that the proposed approach allows to effectively learn state-variable probabilistic constraints and to outperform standard POMCP with no computational overhead.

## CCS CONCEPTS

• **Computing methodologies** → **Planning under uncertainty**; **Partially-observable Markov decision processes**; **Sequential decision making**;

## KEYWORDS

Planning under uncertainty, POMCP, Planning and Learning, Markov Random Fields

**ACM Reference Format:**
Maddalena Zuccotto, Alberto Castellini, and Alessandro Farinelli. 2022. Learning state-variable relationships for improving POMCP performance. In *The 37th ACM/SIGAPP Symposium on Applied Computing (SAC '22), April 25–29, 2022, Virtual Event, .* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3477314.3507049

## 1 INTRODUCTION

Planning under uncertainty is an important problem in artificial intelligence with important applications in robotics and cyber-physical system control. In many real-world applications, agents act in partially unknown environments and they know only the dynamics of these environments. However, in several cases there are also other known properties of the environment which can be used to improve planning performance. In this work, we focus on applications in which the state is representable by a set of variables whose values are probabilistically related to each other. An example are the variables representing rock values in the well known rocksample domain [24]. Suppose to have an agent moving through a grid containing valuable and valueless rocks: by knowing in advance which rocks have the same value (i.e., knowing probabilistic relationships on rock values), the agent can collect valuable rocks faster. This improves planning performance by reducing the number of steps needed to obtain the reward.

POMDPs [11, 25] provide a sound and complete framework for planning under uncertainty. To tackle partial observability they consider all possible states of the (agent-environment) system and assign to each of them a probability value expressing its likelihood of being the true state. These probabilities, considered as a whole, constitute a probability distribution over states, called belief. A solution for a POMDP is a policy that maps beliefs into actions. Finding optimal policies is unfeasible in general [18], therefore approximated policies are typically used. Here, we consider a particular method for learning POMDP policies, called Partially Observable Monte Carlo Planning (POMCP) [23]. It is based on Monte-Carlo Tree Search (MCTS) [6], an efficient method to compute approximate Q-values [26] on large state spaces, and particle filter for representing the belief. Standard POMCP does not consider any kind of prior knowledge about state-variable relationships. An extended version of POMCP has recently been proposed [7] which considers state-variable constraints, expressed as Constraint Networks (CNs) or Markov Random Fields (MRFs). Introducing such knowledge improves planning performance with no overhead in terms of time complexity. In [7], however, state-variable relationships are assumed to be known in advance (e.g. by experts).

In this paper we aim to learn state-variable relationships as the agent acts in the environment. As in [7], we store this knowledge in a MRF representing probabilistic constraints between state-variable assignments. Furthermore, we answer a key question: "When can the MRF be trusted?". We propose, in particular, three different MRF learning methods to convert the knowledge acquired step-by-step by the agent into probabilistic constraints on state-variables, and a criterion based on confidence intervals of MRF potentials to decide when the learning stage can be stopped and the MRF can be used by POMCP to obtain performance improvement. The first MRF learning approach uses observations in the real world, while the other two consider information in the belief of the agent, and respectively, the maximum likelihood state and a weighted sum of the states (where weights are belief probabilities).


*SAC '22, April 25–29, 2022, Virtual Event,*

ACM ISBN 978-1-4503-8713-2/22/04…$15.00
https://doi.org/10.1145/3477314.3507049

Our empirical analysis shows that the learning approach based on the maximum likelihood state in the belief generates the most accurate MRFs. Its usage in tandem with the stopping criterion based on confidence intervals of MRF potentials yields a statistically significant performance improvement over the standard POMCP without any increase of time complexity.

In summary, the main contributions of this work to the state-of-the-art are listed in the following:

(1) We propose three approaches for learning state-variable relationships, represented by a MRF, in POMCP;
(2) We propose a criterion for stopping the learning process, based on confidence intervals of MRF potentials;
(3) We empirically evaluate the proposed approach on two robotics-inspired domains, showing that it yields a statistically significant improvement of the discounted return without any additional overhead in terms of time complexity.

## 2 RELATED WORK

Recent works highlight the benefits of introducing prior knowledge in POMCP. In [7], authors show that the introduction of such prior knowledge leads to a performance improvement. In particular, they make use of probabilistic constraints expressed as MRFs [17] and CNs [10]. The main limitation of these works regards the requirement to have a full specification of the prior knowledge in advance. This is not always feasible in practice, especially in complex applications such as robotic ones, due to the need of complete and exact knowledge about state-variable relationships. What differentiates our work from [7] is that we aim to overcome this limitation by learning the MRF while acting in the environment.

Some works in the literature propose approaches to learn arbitrary MRF structures [1, 4, 19, 21, 28]. These approaches have a higher complexity than the approach we propose because our approach is specialized on pairwise MRF and on the POMCP application. In [22] authors also focus on pairwise MRF but the methodology they propose learns continuous pairwise MRF. The MRFs that we use in our approach are instead discrete.

Another research field related to our work is that of Bayesian adaptive learning in POMDPs [20]. An elegant method for learning the transition and reward models in POMCP is presented in [12]. Authors extend the POMCP algorithm to the Bayes-Adaptive case, proposing a so-called Bayes-Adaptive Partially Observable Monte Carlo Planning (BA-POMCP) approach. It learns, however, the parameters of the transition model (i.e., the probability of a state to change to another state in a step of the dynamics), while our method learns probabilities of pairs of state-variables to have equal values in single states (i.e., we do not consider any information about how the state changes over time). We assume that the state can change only from an episode to another and each state has a probability to occur that depends on some (unknown) state-variable probabilistic relationships. We notice that this setting is very common in practice (see the example of the warehouse in the problem definition section, below) but it cannot be naturally encoded in the transition model. The information encoded in our MRF is instead very simple to understand and useful to initialize and update the belief. For the same reason, our approach differentiates also from Factored BA-POMCP [13], which learns a compact model of the dynamics by exploiting the underlying structure of a POMDP, allowing to better scale to large problems. Also this approach deals with knowledge about the transition from one state to another across the steps of an execution and it cannot learn the probability distribution of states considering probabilistic state-variable relationships, as our MRF does. We remark that we do not factorize the POMDP to learn compact model of dynamics.

There are also works that address the problem of adding constraints to planning for improving the performance or scaling to large environments. In [15] MCTS is used to generate policies for constrained POMDPs and in [3] the multi-agent structure of some specific problems is explored to decompose the value function. We instead constrain the state space on the basis of state-variable relationships to refine the belief during execution. Specifically, we learn a MRF able to express variable constraints.

## 3 BACKGROUND

We provide formal definitions of the main frameworks used in this work, namely, POMDP, POMCP and MRF.

### 3.1 POMDP and POMCP

A POMDP [11] is a tuple $(S, A, O, T, \Omega, R, \gamma)$ where $S$ is a finite set of *states*, $A$ is a finite set of *actions*, $\Omega$ is a finite set of *observations*, $T : S \times A \to \Pi(S)$ is the *transition model* where $\Pi(S)$ is the space of probability distribution over states, $O : S \times A \to \Pi(\Omega)$ is the *observation model*, $R : S \times A \to \mathbb{R}$ is the *reward function* and $\gamma \in [0, 1)$ is the *discount factor*. The agent's goal is to maximize the *expected discounted return* $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)]$ acting optimally, namely, choosing in each state $s_t$, at time $t$, the action $a_t$ with the highest expected reward. In POMDPs, however, the agent is not able to directly observe the current state $s_t$ but it maintains a probability distribution over states $S$, called *belief*, which updates at each time-step. We represent by symbol $b(s)$ the probability of being in state $s$ according to belief $b$. The belief summarizes agent's previous experiences, i.e. the sequence of actions and observations that the agent took from an initial belief $b_0$ to the belief $b$. The solution of a POMDP is an optimal *policy*, namely, a function that optimally maps belief states into actions. A policy is optimal if it maximizes the expected discounted return. The discount factor $\gamma$ reduces the weight of long-term rewards guaranteeing convergence.

POMCP [23] is a Monte-Carlo based algorithm for planning in partially observable environments. A particle filter representing the belief is initialized with $k$ particles, each encoding a state $s$. At each step, MCTS is used to find the best action. The Upper Confidence bounds applied to Trees (UCT) strategy [14] is used to balance exploration and exploitation in the simulation phase. After the selected action $a$ is performed in the real environment, a real observation $o$ is collected and particles in the belief are updated by keeping only particles that explain the observation. Particle reinvigoration is used if the particle filter gets empty. In [7] MRFs are used to introduce in POMCP prior knowledge about state-variable relationships. The methodology improves planning performance while keeping the time complexity unchanged.

### 3.2 Markov Random Fields

A MRF is an undirected graph where nodes represent variables and edges represent probabilistic relationships between variable values [5, 17]. A *potential function* is a non-negative function representing the relative "compatibility" of different variable assignments. According to the Hammersley-Clifford theorem [27], the joint probability represented by the MRF can be computed as the product of potential functions over the maximal cliques of the graph, namely $p(\boldsymbol{x}|\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \prod_{c \in C} \psi_c(\boldsymbol{x}_c|\boldsymbol{\theta}_c)$, where $\boldsymbol{x}$ is a variable configuration (e.g., $\boldsymbol{x} = (1, 0 \ldots, 0)$), $\boldsymbol{\theta}$ is a parametrization of the MRF, $C$ is the set of maximal cliques, $\psi_c(\boldsymbol{x}_c|\boldsymbol{\theta}_c)$ is the potential function, and $Z(\boldsymbol{\theta})$ is the partition function, i.e., a normalization factor that can be computed as $Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \prod_{c \in C} \psi_c(\boldsymbol{x}_c|\boldsymbol{\theta}_c)$. Potentials can be represented by a Boltzmann distribution, i.e., exponentials, thus $\psi_c(\boldsymbol{x}_c|\boldsymbol{\theta}_c) = exp(-F(\boldsymbol{x}_c|\boldsymbol{\theta}_c))$ where $F(\boldsymbol{x}_c)$ is the energy function. Restricting the parametrization of the MRF to the edge rather than to the maximal clique of the graph, we obtain *pairwise MRF* in which the product of potentials can be computed by summing the energies of all pairwise relationships. We call $E$ the set of pairwise relationships $(i, j)$ in the MRF, where $i, j \in 1, \ldots, n$, and $n$ is the number of state-variables. For instance, given a pair of state-variables $(X_i, X_j)|(i, j) \in E$ a potential could be $\psi_{X_i,X_j}(0, 0) = 0.45$, which indicates a compatibility of 0.45 to have value 0 in both state-variables $X_i$ and $X_j$, or $\psi_{X_i,X_j}(0, 1) = 0.05$ which indicates a compatibility of 0.05 to have value 0 in state-variable $X_i$ and 1 in state-variable $X_j$. In the following, when we refer to a MRF we mean a set of potentials representing compatibilities of different variable assignments $\psi_{X_i,X_j}(l, h),\ (i, j) \in E,\ l, h \in \{1, \ldots, k\}$ where $k$ is the number of possible values of each variable.

## 4 METHODOLOGY

We formalize the problem, introduce three methods for learning the MRF and present the stopping criterion. Then we define measures to evaluate the learnt MRF and the policy.

### 4.1 Problem Definition

In this work we aim to learn MRFs representing probabilistic equality relationships between state-variables in POMDPs. These MRFs are then used to improve planning performance of POMCP. The advantage of using state-variable relationships inside POMCP was proved in [7, 9] where authors show that its use speeds up the convergence of the belief to the true state. Here, instead, we focus on learning the MRF while the agent acts in the environment. As an example of probabilistic equality relationship between state-variables, consider two variables $x_1, x_2$ assuming values in $\{0, 1\}$. A possible relationship is "$x_1$ is equal to $x_2$ with probability 0.8". In a warehouse, this could represent, for instance, the probability of two aisles to have similar traffic levels. Interestingly, this relationship does not concern the dynamics of the state since it reflects a relative "compatibility" between state-variable values in a single state, hence it cannot be encoded in the transition model. Instead, it is a property of state distribution which can be well represented by a MRF that uses state-variables as nodes and probabilistic relationships between pairs of state-variable values as edges. In other words, our method learns the probabilities of pairs of state-variables to have equal values in single states (i.e., we do not consider any information about how the state changes over time). We assume that the state can change only from an episode to another and each state has a probability to occur that depends on some (unknown) state-variable probabilistic relationships.

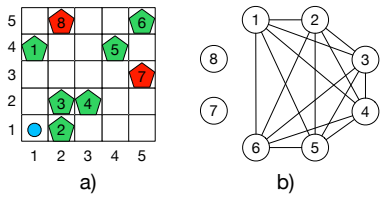


**Figure 1: Running example on rocksample. *a)* Board with valuable (green) and valueless (red) rocks. *b)* MRF topology.**

A second problem we tackle is that of identifying the right moment to stop learning the MRF and to start using it. If the learning process is stopped too early, the parameters of the MRF could be over specialized on the relationships of few episodes, yielding a decrease of planning performance instead of an improvement. Instead, if the learning process is stopped too late than the last episodes could be non informative for improving the MRF. In this case it would be more convenient to use the learnt MRF in those episodes in order to start benefiting from the improvement that it provides.

### 4.2 Running example

We introduce here *rocksample* [24] a benchmark domain which we use as a running example in the following to explain the proposed methodology. The same domain is also used in the result section, together with a robotics-inspired domain, to evaluate the performance of the proposed approach. We perform our tests on rocksample(5,8), consisting of a grid with 5 rows and 5 columns in which 8 rocks are placed in fixed positions (see pentagons in Figure 1.a). Each rock can be valuable or valueless and the rock value configuration changes at each episode. The agent (see the circle in Figure 1.a) knows the rock locations but it cannot observe rock values (which is the hidden part of the state). At each step, the agent performs one action among *moving* (up, down, left, right), *sensing* a rock (i.e., checking its value) or *sampling* a rock (i.e., collecting its value) and its goal is to maximize the discounted reward. When a rock is checked (by the sensing action), however, its true value is observed with a probability inversely proportional to the distance between the agent and the rock. The reward obtained by moving and sensing is 0, while sampling a rock gives a reward of 10 if the rock is valuable, -10 if it is valueless. The MRF used in our running example, shown in Figure 1.b, has a clique in rocks 1, 2, 3, 4, 5, 6 and edge probabilities equal to 1.0, for simplicity, hence admissible configurations of rock values must have all these rocks with the same value to satisfy the constraint. The values of rocks 7 and 8 can instead be randomly assigned individually since there is no constraint on their values in the MRF.

The *state* is characterized by the agent position on the grid, the rocks configuration (hidden), and a flag indicating rocks already sampled. The set of *actions* is composed by the four moving actions, the sample action and a sensing action for each rock. *Observations* have three possible values, namely: 1 (2) for valuable (valueless) rock

observation returned by sensing actions, 3 for *null* observations returned by moving actions. The discount factor used is $\gamma = 0.95$. Since we aim at maximizing the information learnt about state-variable relationships, we prevent the agent from exiting the grid.

## 4.3 MRF learning techniques

We present three different methods to learn the MRF during the execution of standard POMCP. The first, named *Sample-based MRF Learning (SL)* is based on knowledge gathered by the agent when it gets the true value of the state-variables, i.e., using sampling actions in rocksample. In this method we assume that the agent can somehow (e.g., from the reward or a specific action) get the true value of (hidden) state-variables; for instance, in rocksample we assume the agent can see the true value of the rock only when it collects it. We use this information only in the MRF learning algorithm, not to update the belief. The second and third methods, called *Maximum-likelihood Belief-based MRF Learning (MBL)* and *Weighted-likelihood Belief-based MRF Learning (WBL)* are instead based on the information present in the belief at the end of each learning episode. The two methods differ from each other in the way they use the information in the belief to compute the MRF: MBL uses only the state having maximum likelihood, while WBL uses all states in the belief, weighted by their likelihood. In all our tests we assume to learn the MRF in $NE$ episodes, where each episode $e$ is composed of a fixed number of steps. The methodology can simply adapt to the case of episodes with different number of steps. We initialize the MRF with uninformative priors and then update it at the end of each episode. Details about the three learning methods and common data structures are reported in the following.

*4.3.1 Data structures used in the MRF-learning algorithms.* Learning the MRF means learning the potentials of pairwise MRF representing state-variable relationships. Given two variables $X_i$ and $X_j$ connected by a pairwise relationship in the MRF and having $k$ possible values each, we learn the potential $\psi_{X_i,X_j}(l,h)$ for each pair $(l,h)$ with $l \in \{1,\dots,k\}$ and $h \in \{1,\dots,k\}$, where variable equality occurs when $l = h$. We keep track of intermediate quantities using three data structures:

- A *vector of state-variable values* $\mathcal{V}_e(i)$, $i = 1,\dots,n$, for each episode $e$, where $n$ is the number of state-variables and the value in cell $i$, namely, $\mathcal{V}_e(i) \in \{1,\dots,k\}$ is the value of state-variable $X_i$ observed or extracted from the final belief in episode $e$. This vector is initialized to $\mathcal{V}_e(i) = 0, \forall i \in \{1,\dots n\}$, and then each value $\mathcal{V}_e(i)$ is updated when the value of variable $X_i$ in episode $e$ becomes available.
- A four-dimensional array of *counts of state-variables equalities and inequalities*, for each episode $e$, $\mathcal{M}_e(i,j,l,h)$, where $(i,j) \in E$ and $l,h, \in \{1,\dots,k\}$. The value $\mathcal{M}_e(i,j,l,h)$ is the number of times variable $X_i$ had value $l$ and variable $X_j$ had value $h$ in the previous $e$ episodes, where $e \in \mathbb{N}$. Array $\mathcal{M}_e$ is updated at the end of each episode $e$ using the values in $\mathcal{V}_e(i)$, and the MRF potentials $\psi_{X_i,X_j}(l,h)$ are directly computed using values in $\mathcal{M}_e(i,j,l,h)$ (see Equation 4), hence the MRF can be updated using values in $\mathcal{M}_e$.
- A matrix of *probabilities of state-variables equalities*, $\mathcal{P}_e(i,j)$, where $(i,j) \in E$. The value $\mathcal{P}_e(i,j)$ is the probability that state-variables $X_i$ and $X_j$ had equal values until episode $e$ (see Equation 5).

In summary, our pipeline at each episode $e$ first computes $\mathcal{V}_e$, then $\mathcal{M}_e$, afterwards $\psi^e$, and finally $\mathcal{P}_e$. What differentiates the three MRF-learning strategies here proposed is how they populate $\mathcal{V}_e$ and how they update $\mathcal{M}_e$ after each episode. In the next sections we present the three proposed learning algorithms and the related strategies for populating $\mathcal{V}_e$ and updating $\mathcal{M}_e$.

*4.3.2 Learning method 1: Sample-based MRF Learning (SL).* In this approach $\mathcal{V}_e(i)$ is the value observed for state-variable $X_i$ when the variable is sampled in episode $e$. We call this $\mathcal{V}_e^{SL}$. If in rocksample the i-th rock has been sampled and it was valuable, then $\mathcal{V}_e^{SL}(i) = 1$, if it was valueless then $\mathcal{V}_e^{SL}(i) = 2$, and if it was not sampled then $\mathcal{V}_e^{SL}(i) = 0$. $\mathcal{M}^{SL}$ is initialized to $\mathcal{M}^{SL}(i,j,l,h) = 0$ for each $(i,j) \in E$, and $l,h \in \{1,\dots k\}$. After each episode $\mathcal{M}_e^{SL}$ is updated using vector $\mathcal{V}_e^{SL}$ as:

$$\mathcal{M}_{e+1}^{SL}(i,j,l,h) = \\ = \begin{cases} \mathcal{M}_e^{SL}(i,j,l,h) + 1, & \text{if } \mathcal{V}_e^{SL}(i) = l \wedge \mathcal{V}_e^{SL}(j) = h \\ \mathcal{M}_e^{SL}(i,j,l,h), & \text{otherwise} \end{cases} \tag{1}$$

*4.3.3 Learning method 2: Maximum-Likelihood Belief-based MRF Learning (MBL).* In this approach $\mathcal{V}_e(i)$ is populated with the value of state-variable $X_i$ in the state having maximum likelihood in the belief of the agent at the end of episode $e$. We call this $\mathcal{V}_e^{MBL}$. $\mathcal{M}^{MBL}$ is initialized to $\mathcal{M}^{MBL}(i,j,l,h) = 0 \ \forall (i,j) \in E, \forall l,h \in \{1,\dots k\}$. At the end of each episode the array $\mathcal{M}_e^{MBL}$ is updated using vector $\mathcal{V}_e^{MBL}$ as:

$$\mathcal{M}_{e+1}^{MBL}(i,j,l,h) = \\ = \begin{cases} \mathcal{M}_e^{MBL}(i,j,l,h) + 1, & \text{if } \mathcal{V}_e^{MBL}(i) = l \wedge \mathcal{V}_e^{MBL}(j) = h \\ \mathcal{M}_e^{MBL}(i,j,l,h), & \text{otherwise} \end{cases} \tag{2}$$

*4.3.4 Learning method 3: Weighted-likelihood Belief-based MRF Learning (WBL).* In this approach we consider all the states in the belief at the end of episode $e$. Each state $s$ is considered with a weight depending on its probability $b(s)$ in the belief. Therefore, we compute $|S|$ vectors, $\mathcal{V}_{e,s}^{WBL}, s = 1,\dots,|S|$, and we update $\mathcal{M}_e^{WBL}$ for each $s \in S$ and for each $(i,j) \in E$ as:

$$\mathcal{M}_{e+1}^{WBL}(i,j,l,h) = \\ = \begin{cases} \mathcal{M}_e^{WBL}(i,j,l,h) + \mathcal{X}_e(i,j,l,h), & \text{if } \mathcal{V}_e^{WBL}(i) = l \wedge \mathcal{V}_e^{WBL}(j) = h \\ \mathcal{M}_e^{WBL}(i,j,l,h), & \text{otherwise} \end{cases} \tag{3}$$

where $\mathcal{X}_e(i,j,l,h) = \sum_{s \in S} b(s) \cdot \mathbb{1}_{(X_i=l, X_j=h)}$, and $\mathbb{1}$ is the Kronecker delta.

*4.3.5 Computation of potentials $\psi$ from $\mathcal{M}$.* We compute MRF potentials $\psi$ from multi-dimensional array $\mathcal{M}$ after $e$ episodes, by normalizing each cell (with $(i,j) \in E$) using the following formula

$$\psi_{X_i,X_j}^e(l,h) = \frac{\mathcal{M}_e(i,j,l,h)}{\sum_{w,y=1,\dots,k} \mathcal{M}_e(i,j,w,y)}. \tag{4}$$

*4.3.6 Computation of probabilities of state-variables equalities $\mathcal{P}$ from $\psi$.* These probabilities are finally computed for each $(i,j) \in E$

$$\mathcal{P}_e(i,j) = \sum_{l=1,\dots,k} \psi_{X_i,X_j}^e(l,l) \tag{5}$$

namely, as the sum of potentials on equal values of $X_i$ and $X_j$.

## 4.4 Stopping criterion

The methodology here presented aims to stop the learning process when the MRF is informative enough to bring an improvement in planning performance. We use a statistical approach based on confidence intervals (CI) of state-variable equality probabilities $\mathcal{P}_e(i,j),\ (i,j) \in E$. Algorithm 1 formalizes the approach. It receives the matrix of equality probabilities $\mathcal{P}_e(i,j)$, the significance level $\alpha$, and the episode index $e$. It returns true if, for every edge, the sample size is large enough and the CI of the probability does not include value 0.5. According to a known rule of thumb [2, 16], when a probability $p$ is sampled, the sample size $N$ is large enough if $(N \cdot p > 5) \wedge (N \cdot (1-p) > 5)$. This condition is checked in line 2 of Algorithm 1. The condition on the CI is instead checked in line 5. The lower and upper bounds, respectively, $L^e_{i,j}$ and $U^e_{i,j}$, are computed using the formula of the CI for population proportion [2] $p^e_{i,j} \pm Z_{\alpha/2}\sqrt{\frac{p^e_{i,j}(1-p^e_{i,j})}{e}}$, in which $Z_{\alpha/2}$ represents the $Z$ value that cuts off a right-tail area of $\alpha/2$ under the standard normal curve [2].

The rationale of this algorithm is that a MRF must provide quality information for all state-variable relationships in order to be informative for planning (i.e., to improve planning performance). We translate the concept of quality into a statistical form using CIs. Namely, given a confidence level $(1-\alpha)100\%$, our approach guarantees that every *equality constraint* (i.e., edge with equality probability greater than 0.5) has only a small probability $\alpha$ to refer to an inequality relationship (i.e., an edge with equality probability less than 0.5). In the other way around, every *inequality constraint* has only a small probability $\alpha$ to actually refer to an equality relationship[1].

**Algorithm 1:** Stopping criterion

**Data:** $\mathcal{P}_e(i,j)$: equality probabilities at episode $e$;
$\alpha$: significance level; $e$: episode index
**Result:** True if learning should stop, False otherwise

1 **for** $i,j \mid (i,j) \in E$ **do** `// Iterate on all edges (i,j)`
2  **if** $(e \cdot \mathcal{P}_e(i,j) > 5) \wedge (e \cdot (1-\mathcal{P}_e(i,j)) > 5)$ **then** `// Sz`
3  $L^e_{i,j} \leftarrow \mathcal{P}_e(i,j) - Z_{\alpha/2}\sqrt{\frac{\mathcal{P}_e(i,j)(1-\mathcal{P}_e(i,j))}{e}}$ `// Lower`
4  $U^e_{i,j} \leftarrow \mathcal{P}_e(i,j) + Z_{\alpha/2}\sqrt{\frac{\mathcal{P}_e(i,j)(1-\mathcal{P}_e(i,j))}{e}}$ `// Upper`
5  **if** $(L^e_{i,j} < 0.5) \wedge (U^e_{i,j} > 0.5)$ **then** `// Chk CI`
6  **return** False
7  **else**
8  **return** False
9 **end**
10 **return** True

## 4.5 Performance Measures

We introduce here three measures to evaluate our MRF-learning strategy. One for evaluating planning performance, one for the goodness of the learnt MRF, and one for the goodness of the belief.

[1]This has an interesting parallelism with hypothesis testing conducted using Student's t-test on linear regression coefficients, in which a coefficient is said to be not significant if its confidence interval contains the zero value.

*4.5.1 Discounted return.* The discounted return of an episode $e$, called $\rho_e$, is the sum of the discounted rewards collected in all steps of that episode. The difference between the discounted return obtained using the learnt MRF and the discounted return obtained by the standard POMCP on episode $e$ is called $\Delta\rho_e$ and the average of this difference over all episodes of all runs is called $\overline{\Delta\rho_e}$.

*4.5.2 MRF distance.* Assuming to know the true matrix of probabilities of state-variables equalities $\mathcal{P}^*(i,j), (i,j) \in E$ and that computed after learning episode $e$ by our method $\mathcal{P}_e(i,j),\ (i,j) \in E$, the MRF distance $d^e_M$ is computed as the Euclidean distance normalized by the number of edges in the MRF $\|\mathcal{P}^* - \mathcal{P}_e\|_2/|E|$ between the two matrices. Usually we compute this measure after the last learning episode and we call it $d_M$. The average of this difference over all learning stages of different runs is called $\overline{d_M}$.

*4.5.3 Belief-state distance.* This measure, introduced in [9], allows to quantify the discrepancy between the agent belief about the true state and the true state itself. The prior knowledge introduced by the learnt MRF is expected to improve the planning performance (i.e., discounted return) by improving the belief, hence it is expected to reduce the belief-state distance. This distance, called $d_{SB}$ in the following, is the weighted averaged Manhattan distance between the state-variable configuration in the true hidden state and the state-variable configuration of all states $s \in S$ weighted by their belief probability $b(s)$. In our tests we compute the difference of the belief-state distance reached at each step by POMCP with the MRF and the standard POMCP, and we name it $\Delta d_{SB}$. Then we average these differences over all steps of all episodes of all runs, and we name this $\overline{\Delta d_{SB}}$.

*4.5.4 Time Complexity.* The learning strategies together with the stopping criterion have a complexity $O((|S| + |E|) \cdot NE)$, where $|S|$ is the number of states (to scan the belief), $|E|$ is the number of edges (to update $\mathcal{M}$) and $NE$ is the number of learning episodes.

# 5 RESULTS

In this section we present the results of our empirical analysis. We perform two different types of tests. In Section 5.2 we compare the performance achieved by the three MRF-learning strategies after a fixed number of learning episodes, we identify the best learning strategy and analyze its performance depending on the number of training episodes. In Section 5.3, we analyze the performance of the best method when it is used with the stopping criterion. Experiments are performed on two application domains described in Section 5.1, using *JuliaPOMDP* and *C++* simulators respectively. The extensive analysis of performance considers different MRF topologies and several repetitions of the experiments to guarantee the statistical significance of the results.

## 5.1 Domains

*5.1.1 Rocksample.* This domain is explained in Section 4.2.

*5.1.2 Velocity regulation.* In the velocity regulation problem [8, 9] a mobile robot traverses a pre-defined path split into segments $g_i$ and subsegments $g_{i,j}$. Every segment is characterized by a difficulty $f_i$ depending on the density of obstacles in the segment. The robot has to reach the end of the path in the shortest possible time, tuning its

speed $v$ to avoid collisions with obstacles. A time penalty is given to the robot each time it collides. The robot does not know in advance the true difficulty of each segment (which is the hidden part of the state) but it can only infer its value from the readings of a sensor (Figure 2). This problem can be formalized as a POMDP. The *state* contains: *i)* the (hidden) true configuration of segment difficulties $(f_1, \ldots, f_m)$, where $f_j \in \{L, M, H\}$, $L$ (low), $M$ (medium) and $H$ (high), *ii)* the position $p = (i, j)$ of the robot in the path, where $i$ and $j$ are the indexes of the segment and the subsegment, *iii)* $t$ is the time elapsed from the beginning of the path. *Actions* correspond to the speed of the robot in a subsegment, which can be $L$ (low), $M$ (medium) or $H$ (high). *Observations* are related to subsegment occupancy, namely, $p(o = 1|f = L) = 0.0$, $p(o = 1|f = M) = 0.5$ and $p(o = 1|f = H) = 1.0$. The current time is updated depending on both the action performed by the agent and the collision penalty. The agent needs 1 time unit to traverse a subsegment at high speed, 2 time units at medium speed, and 3 time units at low speed. The collision probability is governed by the *collision model* of Table 1. The reward function is $R = -(t_1 + t_2)$, where $t_1$ is the action time and $t_2$ is the penalty due to collision. In our test we use $t_2 = 10$. Finally, we use a discount factor $\gamma = 0.95$.

**Table 1: Collision model of velocity regulation**

| p(c\|f,a) | | |
|---|---|---|
| $p(1\|L, L) = 0.0$ | $p(1\|L, M) = 0.0$ | $p(1\|L, L) = 0.0$ |
| $p(1\|L, L) = 0.0$ | $p(1\|M, M) = 0.5$ | $p(1\|M, H) = 0.9$ |
| $p(1\|H, L) = 0.0$ | $p(1\|H, M) = 1.0$ | $p(1\|H, H) = 1.0$ |

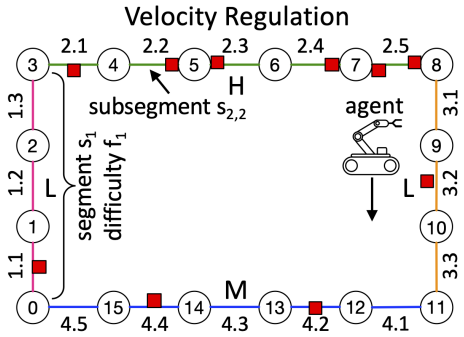


**Figure 2: Path traveled by the agent in the velocity regulation environment. Nodes are subsegment starting points and red blocks represent obstacles.**

## 5.2 Comparison of learning methods

We perform two tests on rocksample(5,8). In the first test (called *test 1* in the following) we compare the performance of the three learning methods presented in Section 4.3. In the second test (called *test 2*) we investigate the dependence of the MRF performance on the number of learning episodes.

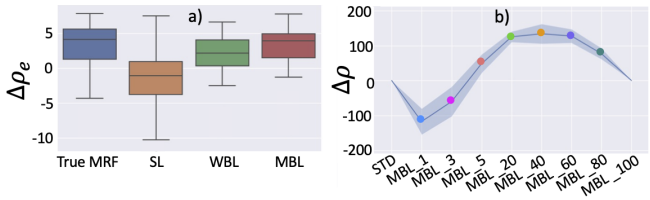


**Figure 3: *a)* Density of difference in discounted return from STD. *b)* Difference in final discounted return from STD (that is equivalent to $\text{MBL}_{100}$).**

*5.2.1 Experimental setting.* In both tests we perform *NR=5 runs*. Each *run* is composed of *NE=100 episodes* of rocksample, and in each episode the agent performs *NS=100 steps*. In each episode rock values change but they always satisfy the constraints of the MRF of Figure 1.b. POMCP always uses 100000 particles and performs the same number of simulations. For each run, the learnt MRF is then tested by performing the same 100 episodes. Performance are evaluated in terms of discounted return and distance between the true MRF and the learnt MRF. Notice that during the learning phase the information gathered in the MRF is not used in POMCP.

*5.2.2 Test 1: Identification of the best MRF learning method.* In this test, for each run we learn the MRF first with SL, then with MBL and finally with WBL. Afterwards, we insert each learnt MRF in POMCP and compute its performance. Figure 3.a shows the difference $\Delta\rho_e$ between the discounted return obtained by standard POMCP [23] (STD in the following) and that obtained using the MRF learnt by the three methods in all the runs (i.e, each box plot represents data of 5 runs). The worst performance is achieved by SL, with a median $\Delta\rho_e$ of -1.11. This method reaches worse performance than STD because the MRF is computed using only information from collected rocks (not from observed rocks), and POMCP tends to collect more valuable rocks than valueless rocks, hence the information is partial and the MRF is not accurate. This is clear also analyzing the final distance between the true MRF and the learnt MRF, namely, $\overline{d_M} = 0.11$. MBL and WBL instead achieve a performance improvement, with median $\Delta\rho_e$ 3.92 and 2.19, respectively. This is because these methods use the belief as a source of information, hence they consider the information coming from observations (i.e., rock sampling) that are performed on both valuable and valueless rocks. MBL reaches an average distance to the true MRF of 0.01 and WBL of 0.03. This analysis clearly shows that MBL reaches the best performance. Figure 3.a finally shows that the median $\Delta\rho_e$ achieved by using the true MRF is 4.10, higher than all the other methods, as expected.

*5.2.3 Test 2: Dependence of the performance on the number of training episodes.* Test 2 considers only the best learning method, namely MBL. We learn the MRF for different number of episodes $z \in \{1, 3, 5, 20, 40, 60, 80\}$, and then we use the learnt MRF in the remaining episodes until the 100th. Each test is repeated 5 times, namely, we perform $NR = 5$ runs, and we analyze the average difference of final discounted return $\Delta\rho$ achieved by MBL compared to STD with related standard deviations. Figure 3.b shows the results. Interestingly, the MRF obtained using one and three episodes, labels

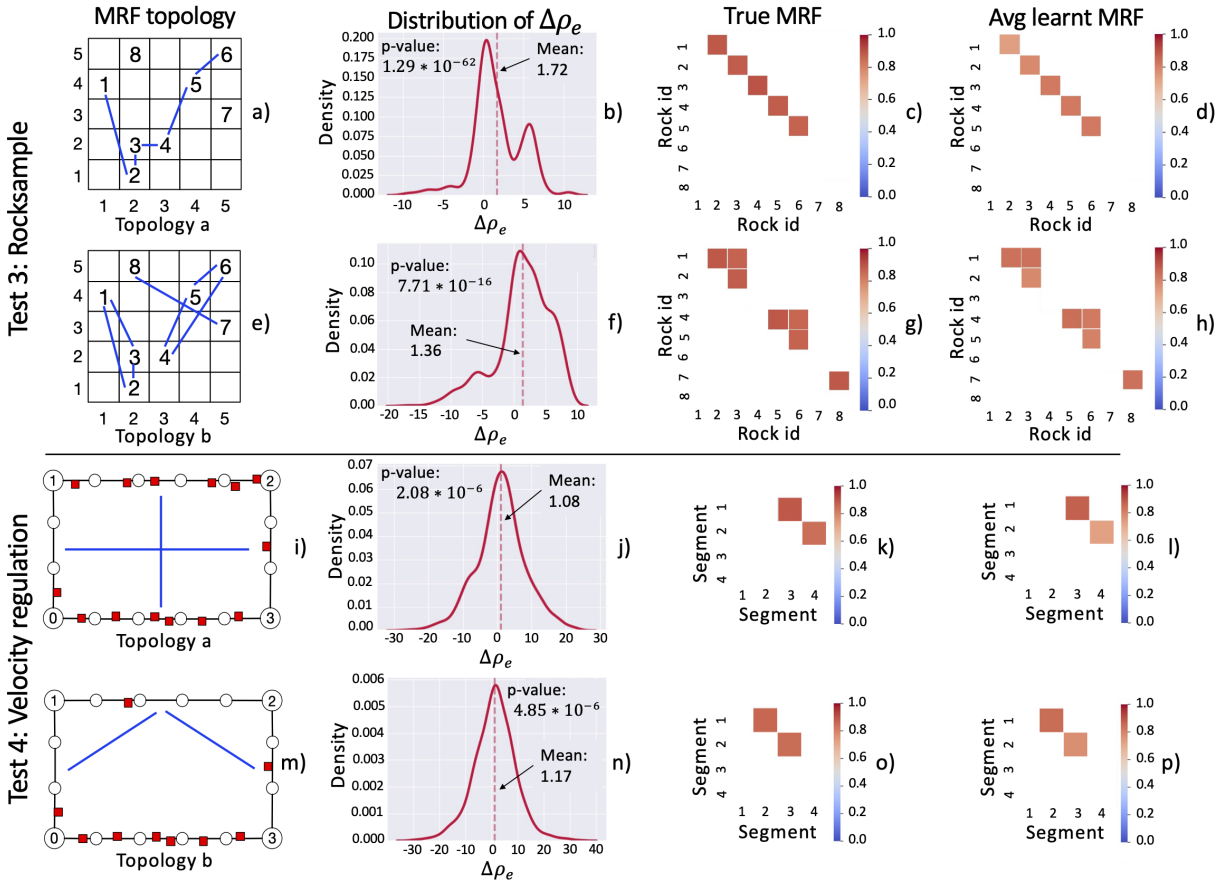


**Figure 4: *a), e), i)* and *m)* MRF topologies. *b), f), j)* and *n)* Density of difference in discounted return from STD. *c), g), k)* and *o)* Heatmap of the true MRF to learn. *d), h), l)* and *p)* Average of the learnt MRFs during experiment 1 and 2.**

MRF_1 and MRF_3 in the chart, have lower performance than STD. This is because they are specialized only on the rock configurations of these episodes but they do not generalize enough on the true probability distribution of rock configurations. In fact, the distances of these two MRFs to the true one are high, i.e., $\overline{d_M} = 0.12$ and $\overline{d_M} = 0.05$, respectively. Figure 3.b shows instead that the performance obtained with the MRFs learnt with 5 and more episodes are higher than that of STD, with a maximum obtained with 40 episodes (see MRF_40). Notice that in our test the performance decrease after 40 episodes not because the MRF gets worse but because it has fewer and fewer episodes to exploit the acquired knowledge.

## 5.3 Performance of MBL with stopping criterion

In these experiments we perform tests using MBL with the stopping criterion described in Section 4.4 on rocksample(5,8) (see *test 3* in the following), and velocity regulation, (see *test 4* in the following).

*5.3.1 Experimental setting.* In test 3 we first select a true MRF (i.e., a set of relationships among rock values, as those shown in Figures 4.a,e). Edge probabilities are always set to 0.9 in the true MRF. We perform *NR=10* runs. In each run we start preparing an empty MRF having the same topology (i.e., set of edges) of the true one (notice that our current method does not learn the topology of the MRF but only the potentials of a MRF with pre-defined topology). We learn the MRF potentials for a number of episodes determined by the stopping criterion in which the information in the MRF is not used in POMCP. The configuration of rock values changes with each episode satisfying the distribution defined by the true MRF. Then we evaluate the performance of the MRF by introducing it in POMCP and performing *NE=100* episodes with and without the MRF, comparing the discounted return of each episode and averaging it over all the runs. In each episode the agent performs *NS=70* steps. POMCP always uses 100000 particles and performs the same number of simulations. Test 4 is performed in the same way but on the velocity regulation domain. The MRF topologies used are displayed in Figures 4.i,m. Edge probabilities are again set to 0.9. The number of steps per episode is in this case *NS=16*, namely, the number of subsegments in the path. To summarize the flow of test 3 and test 4, we show in Figure 5 the MRF learning and usage for MBL with stopping criterion.

To prove that the introduction of the learnt MRF provides a statistically significant improvement with respect to the standard

**Table 2: Performance improvement obtained by the learnt MRF compared to STD in tests 3 and 4.**

| Test | MRF topol. | $\overline{\Delta\rho_e}(\overline{\Delta\rho_e}\%)$ | p-val | $\overline{d_M}$ | $\overline{\Delta d_{SB}}$ |
|---|---|---|---|---|---|
| 3 | a | **1.72(8.35%)** | $1.29 \cdot 10^{-62}$ | 0.04 | -0.16 |
| 3 | b | **1.36 (7.16%)** | $7.71 \cdot 10^{-16}$ | 0.03 | -0.14 |
| 4 | a | **1.08 (3.52%)** | $2.08 \cdot 10^{-6}$ | 0.04 | -0.52 |
| 4 | b | **1.17 (3.92%)** | $4.85 \cdot 10^{-6}$ | 0.06 | -0.33 |

POMCP, we show that the average difference $\overline{\Delta\rho_e}$ between the discounted return obtained with the learnt MRF and the discounted return obtained with STD is significantly larger than zero. Notice that the difference is computed episode by episode to reduce the randomness and the average is computed across all the NE=100 episodes of each run (i.e., over 1000 episodes in total). More precisely, at episode $e$, we compute the difference of discounted return $\rho_e$ as $\Delta\rho_e = \rho_e^{MBL} - \rho_e^{STD}$. Then we compute the average of these values over all the episodes of all the runs *average discounted return* $\overline{\Delta\rho_e}$. Similarly we compute the *average belief-state distance* $\overline{\Delta d_{SB}}$. This is however averaged over all steps of each episode of all runs, which are $70 \cdot 100 \cdot 10 = 70000$ in rocksample and $16 \cdot 100 \cdot 10 = 16000$ in velocity regulation.

*5.3.2 Test 3: results on rocksample.* The results on rocksample are summarized in the first two rows of Table 2 and the first two rows of Figure 4. The main result is represented by the average difference in discounted return $\overline{\Delta\rho_e}$ achieved using the learnt MRF compared to not using it, which is 1.72 (i.e, 8.35%) with the MRF topology $a$ displayed in Figure 4.a, and 1.36 (i.e, 7.16%) with the MRF topology $b$ displayed in Figure 4.e. These average differences are computed over 100 episodes and the 10 runs. The distributions of these differences are displayed in Figures 4.b,f. We verified that these average differences are statistically different from zero using the Student's t-test. In both cases the p-values are lower than 0.05, confirming that the learnt MRF produces on average a statistically significant performance improvement.

To explain the motivation of this improvement we show in Figures 4.c,d and in Figures 4.g,h, the differences between the true MRF and the learnt MRF, respectively, for the first and second MRF topology. The similarity is very high, in fact the average MRF distance is $\overline{d_M} = 0.04$ for topology $a$ and $\overline{d_M} = 0.03$ for topology $b$ (see Table 2). This shows that MBL with the proposed stopping criterion manages to generate an accurate MRF. Furthermore, we display in the fifth and sixth column of Table 2 that the usage of the learnt MRF produces a statistically significant decrease in the *average belief-state distance* $\overline{\Delta d_{SB}}$. The negative values of these differences, respectively -0.16 and -0.14, mean that the average belief-state distance obtained using the MRF is smaller than that obtained without using it. This provides insight into the mechanism that generates the improvement in the discounted return. Namely, using the learnt MRF the belief tends to converge faster to the true state (i.e., the true rock value configuration) allowing the agent to collect larger rewards than standard POMCP. Finally, in Figure 6 we show the trends of the lower bounds $L^e_{i,j}$ of probabilities $\mathcal{P}_e(i,j)$, $(i,j) \in E$ of the MRF learnt in the first run of test 3 (topology $a$, shown in Figure 4.a). Episodes $e$ are displayed in the x-axis and lower bound values $L^e_{i,j}$ in the y-axis. Since the equality probabilities in the true MRF are all 0.9, the stopping criterion (Algorithm 1) stops the learning phase (line 5) when all lower bounds are higher than 0.5 (black horizontal line in Figure 6) and the condition on the sample size (line 2) is satisfied. This happens at episode 28 in the chart. We highlight that the lower bounds tend to be low in the first episodes, when the counts of state-variables equalities/inequalities $\mathcal{M}^{MBL}_{e+1}(i,j,l,h)$ are low, and they tend to converge to the true equality probability as the number of episodes increases.

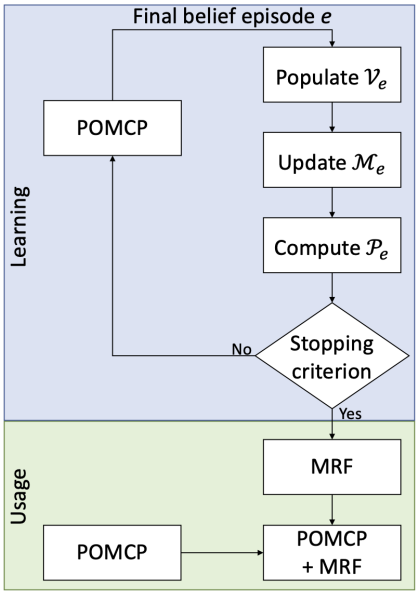


**Figure 5: Main steps of MRF learning and usage with MBL with stopping criterion. The blue background highlights the MRF learning phase, while the green one emphasizes the MRF usage process.**

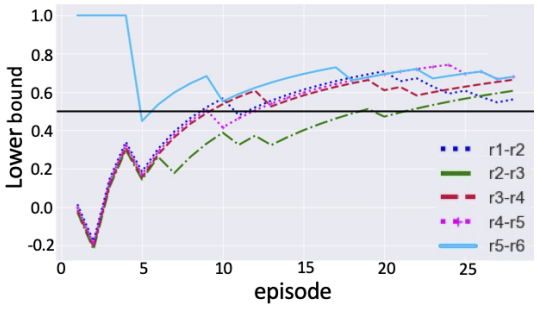


**Figure 6: Lower bound of MRF equality probability for the first run of test 3/a (black line represents value 0.5).**

*5.3.3 Test 4: results on velocity regulation.* The experiments on the velocity regulation domain confirm the positive results achieved with rocksample. The average increase of discounted return obtained using the MRF is in this case of 3.52% with the MRF topology

$a$ displayed in Figure 4.i, and of 3.92% with the MRF topology $b$ displayed in Figure 4.m. The numerical results are summarized in the third and fourth rows of Table 2. Also in this case the p-values of the $\overline{\Delta\rho_e}$, which are lower than 0.05, confirm the statistical significance of the improvement. The motivation of the improvement can be found, again, in the very good approximations of the true MRFs (see Figures 4.l,p,k,o and Table 2) generated by MBL learning with stopping criterion. These good approximations yield a statistically significant reduction of the *average belief-state distance* $\overline{\Delta d_{SB}}$, of -0.52 and -0.33, respectively, in the first and second MRF topology.

## 6 CONCLUSIONS AND FUTURE WORK

In this paper we propose three methods to learn state-variable relationships in POMDPs and introduce them in the POMCP algorithm to improve planning performance. Moreover, we propose a confidence-interval based criterion to decide when to stop the MRF leaning process and to start using it safely i.e., without performance loss. Results show that our approach produces MRFs informative enough to achieve performance improvement. Future research directions involve developing a method able to adapt the learnt MRF to the specific episode characteristics during the application phase. Moreover, we are investigating the possibility to integrate the learning process with planning to optimally balance the exploration-exploitation trade-off and we are also implementing the approach on real robots (Turtlebots) using ROS libraries.

## ACKNOWLEDGMENTS

The research has been partially supported by the projects "Dipartimenti di Eccellenza 2018-2022, funded by the Italian Ministry of Education, Universities and Research (MIUR), "SAFEPLACE, POR-FESR 2014-2020", funded by Regione del Veneto, and "RL-HEAT, Joint project", funded by the University of Verona.

## REFERENCES

[1] Pieter Abbeel, Daphne Koller, and Andrew Y. Ng. 2006. Learning Factor Graphs in Polynomial Time and Sample Complexity. *Journal of Machine Learning Research* 7 (2006), 1743âĂŞ1788.

[2] Amir Aczel and Jayavel Sounderpandian. 2008. *Complete Business Statistics.* McGraw-Hill.

[3] Christopher Amato and Frans A. Oliehoek. 2015. Scalable Planning and Learning for Multiagent POMDPs. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15).* AAAI Press, 1995âĂŞ2002.

[4] Julian Besag. 1977. Efficiency of pseudolikelihood estimation for simple Gaussian fields. *Biometrika* 64, 3 (12 1977), 616–618. https://doi.org/10.1093/biomet/64.3.616

[5] Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer-Verlag, Berlin, Heidelberg.

[6] Cameron B. Browne, Edward Powley, Daniel Whitehouse, Simon M. Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. 2012. A Survey of Monte Carlo Tree Search Methods. *IEEE Transactions on Computational Intelligence and AI in Games* 4, 1 (2012), 1–43. https://doi.org/10.1109/TCIAIG.2012.2186810

[7] Alberto Castellini, Georgios Chalkiadakis, and Alessandro Farinelli. 2019. Influence of State-Variable Constraints on Partially Observable Monte Carlo Planning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019.* ijcai.org, 5540–5546. https://doi.org/10.24963/ijcai.2019/769

[8] Alberto Castellini, Enrico Marchesini, and Alessandro Farinelli. 2021. Partially Observable Monte Carlo Planning with state variable constraints for mobile robot navigation. *Engineering Applications of Artificial Intelligence* 104 (2021), 104382. https://doi.org/10.1016/j.engappai.2021.104382

[9] Alberto Castellini, Enrico Marchesini, Giulio Mazzi, and Alessandro Farinelli. 2020. Explaining the Influence of Prior Knowledge on POMCP Policies. In *Multi-Agent Systems and Agreement Technologies - 17th European Conference, EUMAS 2020, and 7th International Conference, AT 2020, Thessaloniki, Greece, September 14-15, 2020, Revised Selected Papers (Lecture Notes in Computer Science)*, Vol. 12520. Springer, 261–276. https://doi.org/10.1007/978-3-030-66412-1_17

[10] Rina Dechter. 2003. *Constraint Processing.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[11] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. 1998. Planning and Acting in Partially Observable Stochastic Domains. *Artificial Intelligence* 101, 1âĂŞ2 (May 1998), 99âĂŞ134. https://doi.org/10.1016/S0004-3702(98)00023-X

[12] Sammie Katt, Frans A. Oliehoek, and Christopher Amato. 2017. Learning in POMDPs with Monte Carlo Tree Search. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70 (ICML'17).* JMLR.org, 1819âĂŞ1827.

[13] Sammie Katt, Frans A. Oliehoek, and Christopher Amato. 2019. Bayesian Reinforcement Learning in Factored POMDPs. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS'19).* International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 7âĂŞ15.

[14] Levente Kocsis and Csaba Szepesvári. 2006. Bandit Based Monte-Carlo Planning. In *Proceedings of the 17th European Conference on Machine Learning (ECML'06).* Springer-Verlag, Berlin, Heidelberg, 282âĂŞ293. https://doi.org/10.1007/11871842_29

[15] Jongmin Lee, Geon-Hyeong Kim, Pascal Poupart, and Kee-Eung Kim. 2018. Monte-Carlo Tree Search for Constrained POMDPs. In *Advances in Neural Information Processing Systems, NeurIPS 2018.* 7934–7943.

[16] Douglas C. Montgomery and George C. Runger. 2010. *Applied statistics and probability for engineers.* John Wiley & Sons.

[17] Kevin P. Murphy. 2012. *Machine Learning: A Probabilistic Perspective.* The MIT Press.

[18] Christos H. Papadimitriou and John N. Tsitsiklis. 1987. The Complexity of Markov Decision Processes. *Mathematics of Operations Research* 12, 3 (1987), 441–450.

[19] Patrick Pletscher, Cheng Soon Ong, and Joachim Buhmann. 2009. Spanning Tree Approximations for Conditional Random Fields. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics, AISTATS 2009 (Proceedings of Machine Learning Research)*, Vol. 5. PMLR, 408–415.

[20] Stéphane Ross, Joelle Pineau, Brahim Chaib-draa, and Pierre Kreitmann. 2011. A Bayesian Approach for Learning and Planning in Partially Observable Markov Decision Processes. *Journal of Machine Learning Research* 12 (2011), 1729–1770.

[21] Russ R. Salakhutdinov. 2009. Learning in Markov Random Fields using Tempered Transitions. In *Advances in Neural Information Processing Systems, NeurIPS 2009.*

[22] Abhin Shah, Devavrat Shah, and Gregory W. Wornell. 2021. On Learning Continuous Pairwise Markov Random Fields. In *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, Virtual Event (Proceedings of Machine Learning Research)*, Vol. 130. PMLR, 1153–1161.

[23] David Silver and Joel Veness. 2010. Monte-Carlo Planning in Large POMDPs. In *Advances in Neural Information Processing Systems, NeurIPS 2010.* 2164–2172.

[24] Trey Smith and Reid Simmons. 2004. Heuristic Search Value Iteration for POMDPs. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence (UAI '04).* AUAI Press, 520âĂŞ527.

[25] Edward J. Sondik. 1978. The Optimal Control of Partially Observable Markov Processes over the Infinite Horizon: Discounted Costs. *Operations Research* 26, 2 (April 1978), 282âĂŞ304. https://doi.org/10.1287/opre.26.2.282

[26] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction* (second ed.). The MIT Press.

[27] Graham Upton and Ian Cook. 2008. *A Dictionary of Statistics.* OUP Oxford.

[28] Marc Vuffray, Sidhant Misra, and Andrey Y. Lokhov. 2020. Efficient Learning of Discrete Graphical Models. In *Advances in Neural Information Processing Systems, NeurIPS 2020.*