Properly Acting under Partial Observability with Action Feasibility Constraints

Caroline Ponzoni Carvalho Chanel[1,2] and Florent Teichteil-Königsbuch[1]

[1] Onera – The French Aerospace Lab, Toulouse, France
{caroline.carvalho,florent.teichteil}@onera.fr
[2] ISAE – Institut Supérieur de l'Aéronautique et de l'Espace, Toulouse, France

Abstract. We introduce Action-Constrained Partially Observable Markov Decision Process (AC-POMDP), which arose from studying critical robotic applications with damaging actions. AC-POMDPs restrict the optimized policy to only apply feasible actions: each action is feasible in a subset of the state space, and the agent can observe the set of applicable actions in the current hidden state, in addition to standard observations. We present optimality equations for AC-POMDPs, which imply to operate on α-vectors defined over many different belief subspaces. We propose an algorithm named PreCondition Value Iteration (PCVI), which fully exploits this specific property of AC-POMDPs about α-vectors. We also designed a relaxed version of PCVI whose complexity is exponentially smaller than PCVI. Experimental results on POMDP robotic benchmarks with action feasibility constraints exhibit the benefits of explicitly exploiting the semantic richness of action-feasibility observations in AC-POMDPs over equivalent but unstructured POMDPs.

Keywords: sequential decision-making, partially observable Markov decision processes, safe robotics, action feasibility constraints, action preconditions

1 Introduction

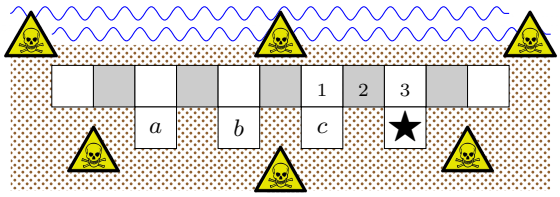
In automated planning, dealing with action preconditions – those feasibility constraints modeling the set of states where a given action is applicable – is an usual standard [1–4]. They allow planning problems' designers to explicitly express properties about feasible actions as logic formulas, which is of first importance in real-life systems or robots. Feasible actions are defined as [5]: neither physically impossible (e.g. flying to Prague from a city without airport), nor forbidden for safety reasons (taking off without sufficient fuel), nor suboptimal and thus useless (flying from Toulouse to Prague via São Borja).

When constructing a solution plan, deciding whether an action is feasible in the current state of the system is obvious if states are fully observable, by testing if the current state is in the set of states where the action is feasible. However, if the agent cannot know its current state with perfect precision, it must rather reason about its *belief state*, that encodes all the different possible states in which the agent can be [6–8]. Thus, solution strategies are defined over belief states but not states, whereas action feasibility constraints are still defined over states. Therefore, additional information

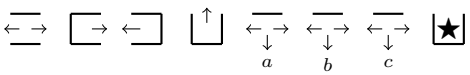
from the environment is required to disambiguate the belief state insomuch the set of feasible actions to insert in the plan can be deduced [9].

For example, consider an autonomous coast guard robot navigating along a cliff with abysses, as shown in Figure 1(a). This example is a slight variation of the Hallway problem [10], where surrounding walls are replaced by cliffs from which the robot may fall down. The goal is to reach the star while being certain (i.e. with probability 1) not to fall down a cliff: in the states near abysses, actions that might make the robot fall down *have to* be prohibited because they are unsafe. Imagine that the belief state of the robot includes states 1, 2, 3 depicted in Figure 1(a). Without sensing the configuration of surrounding abysses, i.e. the feasible actions in the current hidden state, there is no way for the robot to go south in order to sense the presence of the goal. The modeling solution that guarantees to reach the goal while applying only safe actions is well-known by researchers on planning under partial observability: it consists in adding information about applicable actions in the agent's observations, and in assigning near-infinite costs to infeasible state-action pairs. Doing this, we are guaranteed that the maximum-reward policy will: (i) sufficiently disambiguate the belief state in order to reach high interesting rewards ; (ii) only apply feasible actions.

Despite the existence of well-known modeling principles to deal with action feasibility constraints in partially observable planning, there is place for improvements by noting that the set of observations has a specific structure in many real-life or robotic applications: namely, the set of observations is factored in the form of $\Omega = \mathcal{O} \times \Theta$, where, in the current hidden state, the agent can receive "standard" observations randomly from $\mathcal{O}$, and "feasibility" observations deterministically from Θ. For instance in the coast guard problem, a laser or camera sensor will imperfectly locate the agent in the grid, but provide a unique deterministic configuration of surrounding abysses, i.e. set of feasible actions. Thus, this paper aims at benefiting from the specific structure of the observation set to significantly reduce the complexity of finding an optimal policy. We conduct this study with probabilistic settings, in the context of Partially Observable Markov Decision Processes (POMDPs) [6, 7]. Our proposal is oriented towards the exploitation of the specific structure of the problem, whereas standard algorithms can still solve the problem without using this information but far less efficiently.



(a) coastal environment



(b) observations of abyss configurations; arrows represent feasible actions in each configuration

Fig. 1. Coast guard robotic problem.

The remainder of the paper is organized as follows. First, we make explicit and formalize action feasibility constraints in a new model named Action-Constrained POMDPs (AC-POMDPs), which is a subset of POMDPs. This richer model provides structured observation sets and functions, as well as a sound optimization criterion, which properly selects only policies whose actions are feasible in the current hidden state of the system. Most importantly, this criterion reveals that *optimizing AC-POMDPs can be reduced to handle α-vectors that are defined over many different small belief subspaces*, thus significantly reducing computations. Then, we present a point-based algorithm named PreCondition Value Iteration (PCVI), which takes advantage of the specific structure of AC-POMDPs by implementing α-vector procedures that operate over different small belief subspaces. In comparison, standard algorithms like PBVI - Point Based Value Iteration - [11] operate on the full belief space. Finally, we propose a relaxed version of PCVI, which computes a lower bound on the value function that totally removes the set of action feasibility observations from computations, yielding additional exponential-time speedups. Our experimental results on many benchmarks and on a real aerial robotic problem, where action feasibility constraints are essential for safety reasons, highlight the computational benefits of explicitly dealing with action feasibility semantics in POMDPs.

1.1 Related Work

Recently, researchers proposed a structured POMDP model, named Mixed-Observable Markov Decision Processes (MOMDPs, see [12, 13]), which divides the observation space Ω in visible and hidden parts: $\Omega = \Omega_v \times \Omega_h$. MOMDPs exploit the specific structure of the observation set to reduce the dimension of the belief space, resulting in significant computation gains. However, in our approach, the semantics of observation variables are totally different: we assume $\Omega = \mathcal{O} \times \Theta$, with $\Theta \subseteq 2^A$ being a set of applicable actions from the set A of actions. Among algorithmic differences, MOMDPs' α-vectors are all defined on the same subspace, whereas AC-POMDPs' α-vectors are each defined on different subspaces (see later). This noticeable difference suggests that AC-POMDPs can not be simply viewed as MOMDPs for which visible observations would be sets of feasible actions. Further work is needed to explicitly exploit the specific semantics of action-feasibility observations, as actually proposed in this paper.

Our work is also related to POMDP models that incorporate constraints on states or on execution paths. Goal POMDPs [14] require the optimized policy to reach a set of goal states from a given initial state. Constrained POMDPs [15] search for a policy maximizing the value function for a given reward function subject to inequality constraints on value functions for different reward functions, which can be often interpreted as constraining the optimized policy to some areas of the belief space. Thus, these models put state-based constraints on the optimized policy, which are not directly related to properties of feasible actions. On the contrary, our AC-POMDP model forces the optimized policy to apply only actions that are feasible in a given belief state, knowing constraints on feasible state-action pairs. Our action-feasibility constraints are weaker than Constrained POMDPs' ones, which allows us to use a modified dynamic programming schema that does not include the cumulative cost in the state space, contrary to Constrained POMDPs. As a result, the complexity of solving Constrained POMDPs is much higher than in AC-POMDPs.

2 Theoretical Backgrounds

Our work is built upon Partially Observable Markov Decision Processes (POMDPs), which offer a sound mathematical model for sequential decision-making under probabilistic partial observability. A POMDP [6, 7] is a tuple $\langle \mathcal{S}, \mathcal{A}, \Omega, T, O, R, b_0 \rangle$, where: $\mathcal{S}$ is the set of states; $\mathcal{A}$ is the set of actions; Ω is the set of observations; $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the transition function, such that: $T(s, a, s') = p(s_{t+1} = s' | s_t = s, a_t = a)$; $O : \Omega \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the observation function such that: $O(o, a, s') = p(o_{t+1} = o | s_{t+1} = s', a_t = a)$; $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ is the reward function associated with transitions; b_0 is the initial probability distribution over states. We denote $\Delta \subset [0; 1]^{|\mathcal{S}|}$ the (continuous) set of probability distributions over states, named *belief space*. Figure 2(a) depicts the dynamic influence diagram of a POMDP.

At each time step, the agent updates its current *belief* b according to the performed action and the received observation, using Bayes' rule :

$$b_a^o(s') = \frac{O(o, a, s') \sum_{s \in S} T(s, a, s') b(s)}{\sum_{s \in S} \sum_{s'' \in S} O(o, a, s'') T(s, a, s'') b(s)} \tag{1}$$

Solving a POMDP consists in finding a policy function $\pi : \Delta \rightarrow \mathcal{A}$ that maximizes a performance criterion. The expected discounted reward from any initial belief $V^\pi(b) = E_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(b_t, \pi(b_t)) \mid b_0 = b \right]$ is usually optimized. The value of an optimal policy π^* is defined by the optimal value function V^* that satisfies the Bellman optimality equation:

$$V^*(b) = \max_{a \in A} \left[r(b, a) + \gamma \sum_{o \in \Omega} p(o|a, b) V^*(b_a^o) \right] \tag{2}$$

where $r(b, a) = \sum_{s \in \mathcal{S}} b(s) \sum_{s' \in \mathcal{S}} T(s, a, s') R(s, a, s')$. This value function is proven to be piecewise linear and convex over the belief space [6], so that at n^{th} optimization stage, the value function V_n can be parametrized as a set of hyperplanes over Δ named α-*vectors*. An α-vector and the associated action $a(\alpha_n^i)$ define a region of the belief space for which this vector maximizes V_n. Thus, the value of a belief b can be defined as: $V_n(b) = \max_{\alpha_n^i \in V_n} b \cdot \alpha_n^i$. The optimal policy at this step is then: $\pi_n(b) = a(\alpha_n^b)$.

3 Action-Constrained POMDPs

In this section, we propose a more expressive POMDP model, named Action-Constrained POMDP (AC-POMDP), which makes explicit the semantics of feasible actions in the model. Namely, as common in robotic applications, it assumes that observation symbols are factored in 2 parts: probabilistic observations informing about the hidden state, and deterministic observations informing about the set of actions that are feasible in the current hidden state. We will see that the second part implies to maximize the value function over different belief subspaces, yielding computational savings over traditional POMDP solvers.

3.1 AC-POMDPs: Model and Optimization Criterion

An Action-Constrained POMDP is defined as a tuple $\langle \mathcal{S}, (\mathcal{A}_s)_{s\in\mathcal{S}}, \Omega, T, O, \mathbb{I}, R, b_0, \Theta_0 \rangle$, where, in contrast to POMDPs: $(\mathcal{A}_s)_{s\in\mathcal{S}}$ is the set of applicable action sets, such that $\mathcal{A}_s$ is the set of actions that are feasible in a given state s; $\Omega = \mathcal{O} \times \Theta$ is the set of observations, such that $\Theta \subseteq 2^{\mathcal{A}}$; observations in $\mathcal{O}$ and in Θ are independent given any next state and applied action ; $O : \mathcal{O} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the observation function such that: $O(o, a, s') = p(o_{t+1} = o | s_{t+1} = s', a_t = a)$; $\mathbb{I} : \Theta \times \mathcal{S} \to \{0, 1\}$ is the feasibility function: $\mathbb{I}(\theta, s') = p(\theta_{t+1} = \theta \mid s_{t+1} = s') = 1$ if $\theta = \mathcal{A}_{s'}$, otherwise 0 ; Θ_0 is the initial set of applicable actions, observed before applying the first action. Like similar approaches in non-deterministic settings [9], Θ_0 is required to safely apply the first action. For convenience, we also define the action feasibility function $\mathbb{F}$ such that $\mathbb{F}(a, s) = 1$ if and only if $a \in \mathcal{A}_s$.

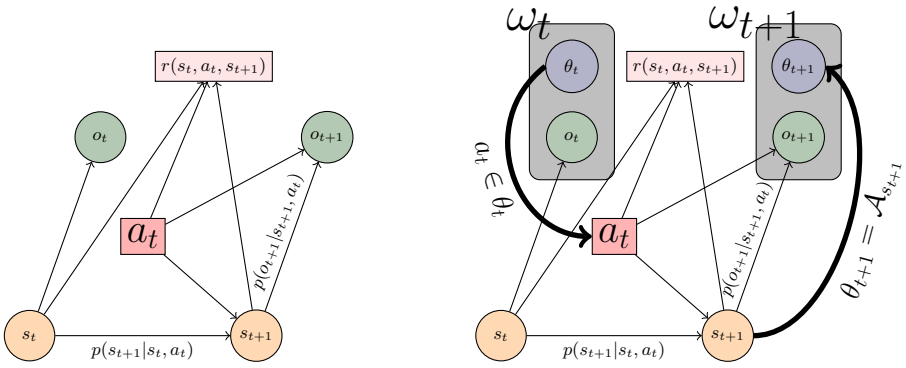
Figure 2(b) represents an AC-POMDP as a controlled stochastic process. The action a_t executed at time t is constrained to belong to the observed set of feasible actions θ_t. The next observed set θ_{t+1} is equal to the set of actions $\mathcal{A}_{s_{t+1}}$ that are feasible in the hidden state s_{t+1}, which stochastically results from applying action a_t in state s_t. The other part of observations (o_t and o_{t+1}) are stochastically received in the same way as in POMDPs.

Contrary to POMDPs, policies of AC-POMDPs are constrained to only execute actions that are feasible in the current hidden state. Given the history $h_t = (\omega_0 = (o_0, \theta_0), \cdots, \omega_t = (o_t, \theta_t))$ of observations up to time t, we define the set of feasible policies as:

$$\Pi^{h_t} = \{\pi \in \mathcal{A}^{\Delta} : \forall\, 0 \leqslant i \leqslant t, \pi(b_i(o_0, \cdots, o_i)) \in \theta_i\}$$

where $b_i(o_0, \cdots, o_i)$ is the belief state resulting from observing $(o_0, \cdots, o_i)$. Thus, solving an AC-POMDP consists in finding a policy π^* such that, for all $b \in \Delta$ and $\theta \in \Theta_0$:

$$\pi^*(b, \theta) \in \underset{\pi\in\Pi^{h\infty}}{\operatorname{argmax}} E\left[\sum_{t=0}^{+\infty} \gamma^t r(b_t, \pi(b_t) \mid b_0 = b, \theta_0 = \theta\right] \tag{3}$$



(a) The POMDP model (b) The AC-POMDP model

Fig. 2. Dynamic influence diagrams of POMDPs and AC-POMDPs

3.2 Belief State Update

As in POMDPs, we should compute the new belief state of the agent, noted b_a^{ω}, after applying an action a in belief state b and receiving an observation $\omega = (o, \theta)$.

Theorem 1. *Let b be the belief state at a given time step, a the action applied by the agent at this time step, and $\omega = (o, \theta)$ the pair of observations immediately received. The next belief state, for all possible next state s', is:*

$$b_a^{(o,\theta)}(s') = \frac{\mathbb{I}(\theta, s') b_a^o(s')}{\sum_{s'' \in \mathcal{S}} \mathbb{I}(\theta, s'') b_a^o(s'')} \tag{4}$$

with b_a^o equal to the expression given in eq. 1.

Proof.

$$\begin{aligned} b_a^{(o,\theta)}(s') &= Pr(s_{t+1} = s' | o_{t+1} = o, \theta_{t+1} = \theta, b_t = b, a_t = a) \\ &= \frac{Pr(s_{t+1} = s', o_{t+1} = o, \theta_{t+1} = \theta | b_t = b, a_t = a)}{Pr(o_{t+1} = o, \theta_{t+1} = \theta | b_t = b, a_t = a)} \\ &= \frac{U(s', o, \theta | b, a)}{\sum_{s'' \in \mathcal{S}} U(s'', o, \theta | b, a)} \end{aligned} \tag{5}$$

with $U(s, o, \theta | b, a) = Pr(s_{t+1} = s, o_{t+1} = o, \theta_{t+1} = \theta | b_t = b, a_t = a)$. As observation symbols o and θ are assumed to be independent given any next state and applied action a, we can factorize U in the form of:

$$\begin{aligned} U(s, o, \theta | b, a) \times \frac{1}{Pr(o_{t+1} = o | b_t = b, a_t = a)} = \\ \underbrace{Pr(\theta_{t+1} = \theta | s_{t+1} = s, b_t = b, a_t = a)}_{\mathbb{I}(\theta, s)} \times \\ \underbrace{\frac{Pr(o_{t+1} = o | s_{t+1} = s, b_t = b, a_t = a) \times Pr(s_{t+1} = s | b_t = b, a_t = a)}{Pr(o_{t+1} = o | b_t = b, a_t = a)}}_{b_a^o(s)} \end{aligned}$$

Finally, by replacing s by s' and s'' in resp. the numerator and the denominator of eq. 5, and by multiplying both of them by $Pr(o_{t+1} = o | b_t = b, a_t = a)$, which is assumed to be non-zero exactly as in the standard POMDP theory, we get the intended result.

The previous theorem highlights two important properties. First, Equation 4 clearly shows that the observation of the set of feasible actions in the current hidden state, due to its deterministic nature, acts like a binary mask on the belief state. Intuitively, we can benefit of this property to significantly speedup computations in comparison with a flat observation model (ie. standard POMDPs), by *optimizing the value function only over the relevant belief subspace*. To this purpose, we will actually present later algorithms that manipulate α-vectors over many different belief subspaces.

Second, following from eq. 4, we will prove that at any given time step, all states s' whose belief is non-zero have the same set of feasible actions. This property shows

that executing policies in AC-POMDPs is coherent with the proposed framework. Most importantly, at optimization time, we can deduce without ambiguity the set of actions over which we maximize the value function for the current belief state. Note that *this primordial property would not be true if the agent would not observe the set of feasible actions*, which gives a theoretical justification of observing them in real robotic applications. More formally, let us define the support of the belief state as: $\sigma(b_a^{(o,\theta)}) = \{s' \in \mathcal{S} : b_a^{(o,\theta)}(s') > 0\}$, used in the following theorem.

Theorem 2. *Let $b_a^{(o,\theta)}$ be the belief at a given time step. For any two states s'_1 and s'_2 in $\sigma(b_a^{(o,\theta)})$, we have: $\mathcal{A}_{s'_1} = \mathcal{A}_{s'_2}$.*

Proof. Suppose that $\mathcal{A}_{s'_1} \neq \mathcal{A}_{s'_2}$. Thus, by definition, $\mathbb{I}(\theta, s'_1) \neq \mathbb{I}(\theta, s'_2)$. If $\mathbb{I}(\theta, s'_1) = 0$, then $b_a^{(o,\theta)}(s'_1) = 0$ according to eq. 4, which contradicts $s'_1 \in \sigma(b_a^{(o,\theta)})$. Thus, $\mathbb{I}(\theta, s'_1) = 1$, but then $\mathbb{I}(\theta, s'_2) = 0$, so that $b_a^{(o,\theta)}(s'_2) = 0$: again, this is a contradiction with $s'_2 \in \sigma(b_a^{(o,\theta)})$.

3.3 Optimality Equation

Theorem 2 allows us to adapt dynamic programming equations of POMDPs to AC-POMDPs, by "just" maximizing the value function over the set of feasible actions in the current belief state, instead of considering all actions. This adaptation may seem very simplistic in appearance, but it is absolutely not if we consider that it would definitely not be possible without Theorem 2, ie., by deduction, without observing the set of feasible actions at any decision epoch. Specifically, since all states in the support of the belief state have the same set of feasible actions, we can deduce *the* set of feasible actions from a given belief state b, noted $\mathcal{A}_b$ without ambiguity: $\forall\, s \in \sigma(b), \mathcal{A}_b = \mathcal{A}_s$. Therefore, AC-POMDP policies defined in eq. 3 can be functions of only b by abusing the notation: $\pi(b) = \pi(b, \mathcal{A}_b)$.

Theorem 3.

$$V^*(b) = \max_{a \in \mathcal{A}_b} \left[r(b,a) + \gamma \sum_{\substack{o \in \mathcal{O} \\ \theta \in \Theta}} p(o, \theta \mid a, b) V^* \left(b_a^{(o,\theta)} \right) \right] \tag{6}$$

with $b_a^{(o,\theta)}$ given in eq. 4 and:

$$p(o, \theta \mid a, b) = \sum_{s' \in \mathcal{S}} \mathbb{I}(\theta, s') O(o, a, s') \sum_{s \in \mathcal{S}} T(s, a, s') b(s) \tag{7}$$

Proof. According to Theorem 2, the set of applicable actions observed just before computing b, i.e. θ_{t-1}, can be deduced from b without ambiguity from the support of b: $\theta_{t-1} = \mathcal{A}_b = \mathcal{A}_s$ for any $s \in \sigma(b)$. Since optimal policies are constrained to apply only applicable actions (see eq. 3), the candidate greedy actions that maximize the value function must be chosen in $\mathcal{A}_b$. Then, eq. 6 can be obtained in a similar way to POMDPs, considering (o, θ) as a joint observation. Eq. 7 is proven using a similar reasoning to the proof of Theorem 1:

$$\begin{aligned} p(o,\theta|a,b) &= Pr(\theta|a,b)Pr(o|a,b) = \textstyle\sum_{s' \in \mathcal{S}} Pr(\theta|s')Pr(s'|o,a,b)Pr(o|a,b) \\ &= \textstyle\sum_{s' \in \mathcal{S}} Pr(\theta|s')Pr(s',o|a,b) = \sum_{s' \in \mathcal{S}} Pr(\theta|s')Pr(o|s',a)Pr(s'|a,b) \\ &= \textstyle\sum_{s' \in \mathcal{S}} \mathbb{I}(\theta,s')O(o,a,s') \sum_{s \in \mathcal{S}} T(s,a,s')b(s) \end{aligned}$$

4 PreCondition Value Iteration

We implemented a point-based algorithm to solve AC-POMDPs, which can be viewed as an adaptation of PBVI [11] to the update equations of Theorem 3. The ideas behind this adaptation are yet independent from PBVI, and could have been applied to generalize any modern α-vector-based POMDP planner, like Perseus [16], HSVI2 [17], or SARSOP [18]. The pseudo-code is given in Algorithm 1.

The expansion of $\mathcal{B}$ is performed in a way similar to PBVI (see Line 9 of the Algorithm 1). First, for each point $b \in \mathcal{B}$, a state s is drawn from the belief distribution b. Second, for each action a in $\mathcal{A}_b := \mathcal{A}_s, \forall s \in \sigma(b)$, a successor state s' is drawn from the transition model $T(s, a, s)$, and a pair of observations (θ, o) is drawn using $\mathbb{I}(\theta, s')$, $p(o|s', a)$ and $p(s'|s, a)$. Knowing (b, a, θ, o), we can calculate the new belief set $\{b_{a_0}, ..., b_{a_j}\}$. Finally, the farthest point from all points already in $\mathcal{B}$ is chosen and integrated into $\mathcal{B}$.

Apart from the fact that observations are structured in the form of pairs of "standard" observations o and "feasible action set" observations θ, and that the expansion of $\mathcal{B}$ is performed mostly as in PBVI, there are mainly two differences between our PCVI algorithm and standard point-based algorithms. First, the projections $\Gamma^{a,(o,\theta)}$ are computed *only for actions in* $\mathcal{A}_b, \forall b \in \mathcal{B}$, which can save many projections compared with PBVI that generates them for all $a \in \cup_{s \in \mathcal{S}} \mathcal{A}_s$. In the same vein, the backup value function V_k for a given belief $b \in \mathcal{B}$, is computed only for actions in $\mathcal{A}_b$ (see Line 8), contrary to PBVI that uses all actions of the problem. Remember that, according to Theorem 2, $\mathcal{A}_b$ is computed at optimization time by choosing any state $s \in \sigma(b)$ and assigning $\mathcal{A}_b = \mathcal{A}_s$.

The second difference to standard approaches is much more significant in terms of complexity improvements, and is the biggest benefit of reasoning with explicit action feasibility constraints. By explicitly exploiting the semantics of the AC-POMDP model,

Algorithm 1: PreCondition Value Iteration (PCVI)

1 $k \leftarrow 0$; Initialize $V_{k=0} \leftarrow \emptyset$; Initialize $\mathcal{B}$ with b_0;
2 **repeat**
3 $\quad k \leftarrow k + 1$; $V_k \leftarrow \emptyset$;
4 $\quad$ **for** $a \in (\mathcal{A}_b)_{b \in \mathcal{B}}$ **and** $(o, \theta) \in \mathcal{O} \times \Theta$ **do**
5 $\quad\quad \Gamma^{a,(o,\theta)} \leftarrow \alpha_i^{a,(o,\theta)}(s) = \gamma \mathbb{F}(a, s) \sum_{s': \mathbb{I}(\theta, s') \neq 0} T(s, a, s') O(o, a, s') \alpha'_i(s'), \forall \alpha'_i \in V_{k-1}, a_{\alpha_i} \in \theta$;
6 $\quad$ **for** $b \in \mathcal{B}, a \in \mathcal{A}_b$ **do**
7 $\quad\quad \Gamma_b^a \leftarrow \Gamma^{a,*} + \sum_{\substack{o \in \mathcal{O} \\ \theta \in \Theta}} \operatorname{argmax}_{\alpha \in \Gamma^{a,(o,\theta)}} (\alpha \cdot b), \forall a \in \mathcal{A}_b$;
8 $\quad\quad V_k \leftarrow \operatorname{argmax}_{\alpha_b^a \in \Gamma_b^a, \forall a \in \mathcal{A}_b} (\alpha_b^a \cdot b), \forall b \in \mathcal{B}$;
9 $\quad$ Expand $\mathcal{B}$ as in PBVI [11] ;
10 **until** $\left\| \max_{\alpha_k \in V_k} \alpha_k \cdot b - \max_{\alpha_{k-1} \in V_{k-1}} \alpha_{k-1} \cdot b \right\|_{b \in \mathcal{B}} < \epsilon$;

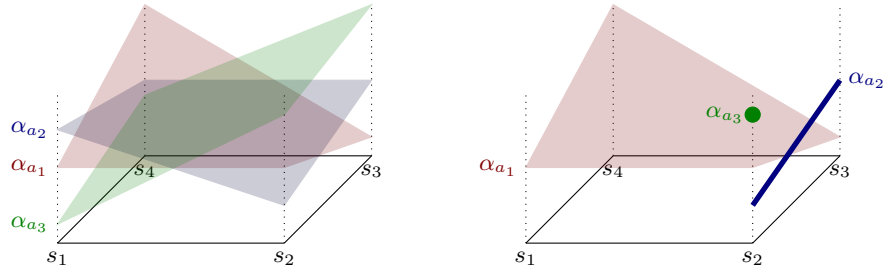
PCVI is able to operate α-vectors defined on many different *reduced* belief subspaces. Namely, since a given action a is defined only over a subset of states $\mathcal{S}_a = \{s \in \mathcal{S} : \mathbb{F}(a,s) = 1\}$, its corresponding α-vectors $\alpha^{a,(o,\theta)}$ are **defined only over a reduced belief subspace** $\Delta_a \subset [0;1]^{\mathcal{S}_a} \subset [0;1]^{\mathcal{S}}$ (see Figure 3(b)). In comparison, PBVI (or any other algorithm for solving POMDPs) works with α-vectors that are defined over the *full* belief space $\Delta \subset [0;1]^{\mathcal{S}}$ (see Figure 3(a)). To this respect, the recent MOMDP model by [12] can be considered as a simpler algorithmic subclass of AC-POMDPs, because MOMDPs deal with α-vectors that are all defined over the *same* belief subspace, whereas AC-POMDPs' α-vectors operate over *different* belief subspaces.

More precisely, the set $\Gamma^{a,(o,\theta)}$ of α-vectors of a given action a and observation (o,θ), knowing the previously computed value function V_{k-1}, is defined as:

$$\Gamma^{a,(o,\theta)} \leftarrow \alpha_i^{a,(o,\theta)}(s) = \gamma \mathbb{F}(a,s) \sum_{s' : \mathbb{I}(\theta,s') \neq 0} T(s,a,s') \times O(o,a,s') \alpha'_i(s'), \forall \alpha'_i \in V_{k-1}, a_{\alpha_i} \in \theta \quad (8)$$

This equation fundamentally differs from standard POMDP algorithms. Action feasibility constraints allow us to apply binary masks on the value function, in order to ensure that values of α-vectors are computed only for the states where the corresponding actions are defined. Equation 8 shows that two masks are applied: (1) the mask $\mathbb{I}(\theta, s')$ restricts the sum over next states s' to the states where the action $a_{\alpha'_i}$ is feasible (the agent observes the set θ of feasible actions, thus necessarily $a_{\alpha'_i} \in \theta$) ; (2) the mask $\mathbb{F}(a,s)$ guarantees to compute the values of $\alpha_i^{a,(o,\theta)}$ only for the states s where it is defined, i.e. where a is feasible.

Note that this masking mechanism is very *different from the masks used in HSVI2* [17]. In HSVI2, so-called masked α-vectors are just sparse vectors that compute and record only the entries of α-vectors corresponding to non-zeros of b. In our case, we explicitly mask (not by just using sparse representations of vectors) the *irrelevant* entries of α-vectors that correspond to infeasible state-action pairs. In AC-POMDPs, the entries of an α-vector, where the corresponding action is not feasible, are not simply equal to zero but can have arbitrary irrelevant values that must be explicitly pruned by



(a) POMDPs: α-vectors are all defined over the entire belief space.

(b) AC-POMDPs: α-vectors are defined over *different* belief **sub**spaces: a_1 feasible in all states; a_2 feasible only in s_2 and s_3; a_3 feasible only in s_2.

Fig. 3. Domain of definition of α-vectors in POMDPs versus AC-POMDPs.

the $\mathbb{F}(a, s)$ masking function. In fact, HSVI2's masking mechanism and ours can be independently applied together.

More precisely, HSVI2's masks automatically prune zero rewards using sparse vectors. Yet, note that modeling infeasible actions in standard POMDP semantics requires to assign near-infinite costs to infeasible actions. Thus, HSVI2 would still have to evaluate these near-infinite costs, which are non-zero and not pruned by its masking mechanism. On the contrary, PCVI's masks automatically prune infeasible actions' costs, which are irrelevant in AC-POMDP semantics, via the function $\mathbb{F}(a, s)$. As a result, masks of HSVI2 and PCVI are totally different, in such a way that infeasible actions' costs are automatically discarded by PCVI but not by HSVI2.

4.1 Relaxed Lower Bound Computation

By studying eq. 6 more in depth, we are able to further benefit from the specific structure of AC-POMDPs' observation model in order to provide a computationally efficient lower bound on the value function. This lower bound, proposed in the following theorem, depends only on the observation set $\mathcal{O}$, instead of the full observation set $\mathcal{O}\times\Theta$. As a result, the computational gains are potentially exponential in the number of actions, since $\Theta \subseteq 2^{\mathcal{A}}$. The idea consists in swapping the $\max$ operator over α-vectors and the sum over action-feasibility observations θ. This is related to – but different from – the fast informed bound method proposed by Hauskrecht [19], which consists in swapping the same $\max$ operator for the sum over states s in standard POMDPs, yielding an upper bound on the value function but not a lower bound (since Hauskrecht's swap is reversed in comparison with ours). Note that Hauskrecht's swap was designed for standard POMDPs, so that it does not reduce the complexity induced by action-feasibility observations, contrary to our swap.

Theorem 4. *Given the value function V_n, we have:*

$$V_{n+1}(b) \geqslant \max_{a\in\mathcal{A}_b} \left[r(b,a) + \gamma \sum_{o\in\mathcal{O}} \max_{\alpha_n\in V_n} \sum_{\substack{s\in\mathcal{S}\\ s'\in\mathcal{S}}} b(s)O(o,a,s')T(s,a,s')\alpha_n(s') \right] \quad (9)$$

Proof.

$$\begin{aligned}
V_{n+1}(b) &= \max_{a\in\mathcal{A}_b} \left[r(b,a) + \sum_{\substack{o\in\mathcal{O}\\ \theta\in\Theta}} p(o,\theta|a,b) V_n\left(b_a^{(o,\theta)}\right) \right] \\
&= \max_{a\in\mathcal{A}_b} \left[r(b,a) + \sum_{\substack{o\in\mathcal{O}\\ \theta\in\Theta}} p(o,\theta|a,b) \max_{\alpha_n\in V_n} \sum_{s'\in\mathcal{S}} b_a^{(o,\theta)}(s')\alpha_n(s') \right] \\
&\geqslant \max_{a\in\mathcal{A}_b} \left[r(b,a) + \sum_{o\in\mathcal{O}} \max_{\alpha_n\in V_n} \sum_{\substack{s'\in\mathcal{S}\\ s\in\mathcal{S}}} O(o,a,s')T(s,a,s')b(s)\alpha_n(s') \underbrace{\sum_{\theta\in\Theta} \mathbb{I}(\theta,s')}_{=1} \right]
\end{aligned}$$

The computation of this lower bound is equivalent to ignoring projections on observations θ of feasible actions in Lines 5 and 7 of Algorithm 1, *as if* feasible actions were not observed by the agent. In this way, projections are only computed for the observation set $\mathcal{O}$, instead of the full observation set $\mathcal{O} \times \Theta$, which potentially yields an exponential gain. Note that α-vectors are still defined only for the states where the corresponding actions are defined, so that the relaxed PCVI algorithm is yet not equivalent to the standard PBVI algorithm. To emphasize this point, we give below the update equation of the α-vectors that make up the lower bound value function. The set of α-vectors $\Gamma^{a,o}$ only depends on o, but each α-vector $\alpha^{a,o}$ of the set is computed by using the feasibility function $\mathbb{F}(a, s)$:

$$\Gamma^{a,o} \leftarrow \alpha^{a,o}(s) = \gamma \sum_{s' \in \mathcal{S}} T(s, a, s') \mathbb{F}(a, s) O(o, a, s') \alpha'_i(s'), \forall \alpha_n \in V_n \qquad (10)$$

The lower bound relaxation of PCVI uses Eq. 10 in place of Eq. 8 in Line 5 of Alg. 1. Line 7 is replaced with the following update: $\Gamma^a_b \leftarrow \Gamma^{a,*} + \sum_{o \in \mathcal{O}} \operatorname{argmax}_{\alpha \in \Gamma^{a,o}}(\alpha \cdot b), \forall a \in \mathcal{A}_b$. The resulting algorithm has the same complexity as standard POMDPs, while guaranteeing that α-vectors and the optimized policy use only feasible actions.

5 Experimental Evaluations

We tested various robotic-like planning problems with action feasibility constraints, which we modeled as AC-POMDPs and solved using our PCVI algorithm. In the subsequent figures, the unrelaxed and relaxed versions of PCVI are respectively noted PCVI1 and PCVI2. We compared our approach with equivalent standard POMDP models, as defined by practitioners to deal with action feasibility constraints: observations of feasible actions are incorporated in the set of observations, but the resulting observation set is treated as an unstructured flat observation set; near-infinite costs are assigned to infeasible state-action pairs in order to prevent the optimized policy from containing illegal actions. Otherwise, there are no guarantees that the optimized policies of standard POMDP models do not apply infeasible actions. Standard POMDP models are solved by PBVI [11] or HSVI2 [17]. We first prove that this translation is sound, before presenting the actual experimental comparisons. We studied four performance criteria depending on benchmarks: 1) the size of value function in terms of the number of α-vectors it contains; 2) the evolution of Bellman error during computation of an optimal policy; 3) the planning time up to convergence at $\epsilon = 0.5$; 4) the statistical expected accumulated rewards from the initial belief state by running 1000 simulations of the optimized policy.

5.1 Translating AC-POMDPs into Equivalent POMDPs

Let $\mathcal{M} = \langle \mathcal{S}, (\mathcal{A}_s)_{s \in \mathcal{S}}, \Omega = \mathcal{O} \times \Theta, T, O, \mathbb{I}, R, b_0, \Theta_0 \rangle$ be a given AC-POMDP. Consider POMDP $\widetilde{\mathcal{M}} = \langle \mathcal{S}, \widetilde{\mathcal{A}}, \Omega = \mathcal{O} \times \Theta, T, \widetilde{O}, \widetilde{R}, b_0, \Theta_0 \rangle = \Psi(\mathcal{M})$ where:

- $\widetilde{\mathcal{A}} = \bigcup_{s \in \mathcal{S}} \mathcal{A}_s$;
- $\widetilde{O} : (\mathcal{O} \times \Theta) \times \widetilde{\mathcal{A}} \times S \to [0; 1]$ is the aggregated observation function, such that $\widetilde{O}((o, \theta), a, s') = O(o, a, s') \mathbb{I}(\theta, s')$;

– $\widetilde{R} : \mathcal{S} \times \widetilde{\mathcal{A}} \times \mathcal{S} \to \mathbb{R}$ is the modified reward function, such that $\widetilde{R}(s,a,s') = R(s,a,s')$ if $a \in \mathcal{A}_s$, otherwise $\widetilde{R}(s,a,s') = -\infty$.

Then, based on POMDPs' and AC-POMDPs' optimality equations , we can easily prove that any optimal policy for POMDP $\widetilde{\mathcal{M}}$ is optimal for the original AC-POMDP $\mathcal{M}$.

Theorem 5. *Let $\mathcal{M}$ be an AC-POMDP and $\widetilde{\mathcal{M}} = \Psi(\mathcal{M})$ its POMDP translation. Then any optimal policy for $\widetilde{\mathcal{M}}$ is optimal for $\mathcal{M}$.*

Proof. Let π^* be an optimal policy for $\mathcal{M}$. According to Bellman eq. 2, we have:

$$\pi^*(b) \in \operatorname*{argmax}_{a \in \widetilde{\mathcal{A}}} \left\{ \tilde{r}(b,a) + \gamma \sum_{\substack{o \in \mathcal{O} \\ \theta \in \Theta}} \tilde{p}((o,\theta) \mid a,b) V^{\pi^*}(b_a^{(o,\theta)}) \right\}$$

with $\tilde{r}(b,a) = \sum_{s \in \mathcal{S}} b(s) \sum_{s' \in \mathcal{S}} T(s,a,s') \widetilde{R}(s,a,s')$ and:

$$\begin{aligned} \tilde{p}((o,\theta) \mid a,b) &= \sum_{s' \in \mathcal{S}} \widetilde{O}((o,\theta),a,s') \sum_{s \in \mathcal{S}} T(s,a,s') b(s) \\ &= \sum_{s' \in \mathcal{S}} O(o,a,s') \mathbb{I}(\theta,s') \sum_{s \in \mathcal{S}} T(s,a,s') b(s) \\ &= p(o,\theta \mid a,b) \end{aligned}$$

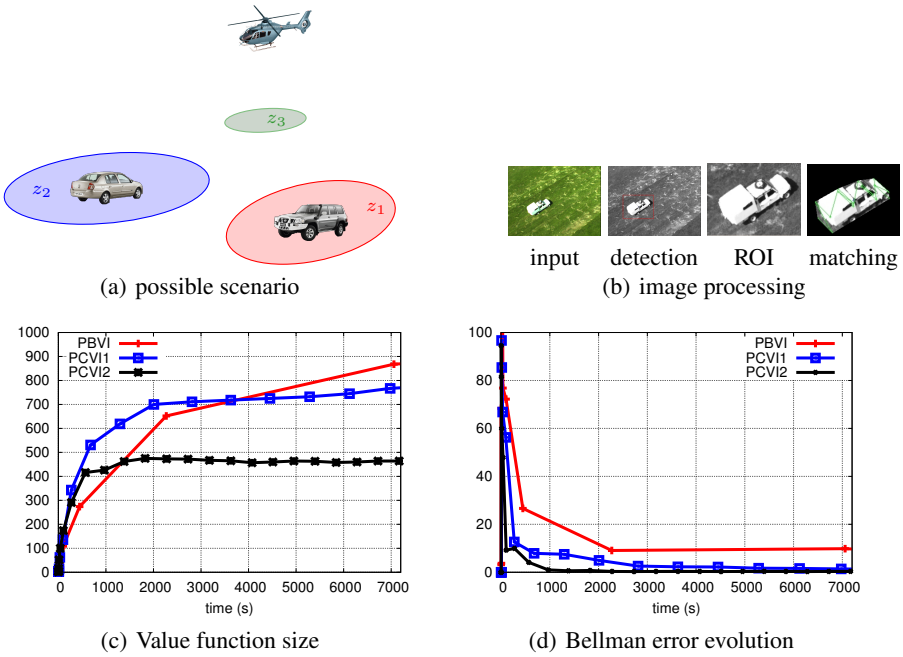
as defined in eq. 7.
Moreover, for $a \notin \mathcal{A}_b$, $\tilde{r}(b,a) = -\infty$: indeed, by definition of $\mathcal{A}_b$, it means that there is a state $s \in \sigma(b)$, i.e. $b(s) > 0$, such that $a \notin \mathcal{A}_s$ and thus $\widetilde{R}(s,a,s') = -\infty$ for all next state s'. Consequently, the maximum value of the above $\max$ operator is necessarily obtained for an action $a^* \in \mathcal{A}_b$. Finally, for all $a \in \mathcal{A}_b$ and states $s \in \sigma(b)$ and $s' \in \mathcal{S}$, $\widetilde{R}(s,a,s') = R(s,a,s')$, so that $\tilde{r}(b,a) = r(b,a)$. Putting it all together, we have:

$$\pi^*(b) \in \operatorname*{argmax}_{a \in \mathcal{A}_b} \left\{ r(b,a) + \gamma \sum_{\substack{o \in \mathcal{O} \\ \theta \in \Theta}} p((o,\theta) \mid a,b) V^{\pi^*}(b_a^{(o,\theta)}) \right\}$$

which means that V^{π^*} is solution of the optimality equation of AC-POMDPs (eq. 6).

5.2 Multi-Target Detection, Identification and Inspection

We first present a real robotic mission, which we solved and actually achieved with real aerial robots. This mission, sketched in Figure 4, is especially interesting because robot's actions are feasible only on a subset of states for safety reasons (accident risk, regulations specific to the test terrain). An autonomous helicopter has to detect, identify then inspect a specific car in an environment composed of different zones, which can possibly contain cars of different models (see Figure 4(a)). The helicopter can receive "standard" observations from an image processing algorithm [20] (see Figure 4(b)): no car detected, car detected but not identified, car identified as another model. Four different actions can be performed: go to a given zone, feasible *only at* altitude 40 meters; land, feasible *only at* altitude 30 meters and *requiring* that the helicopter can



(a) possible scenario (b) image processing

(c) Value function size (d) Bellman error evolution

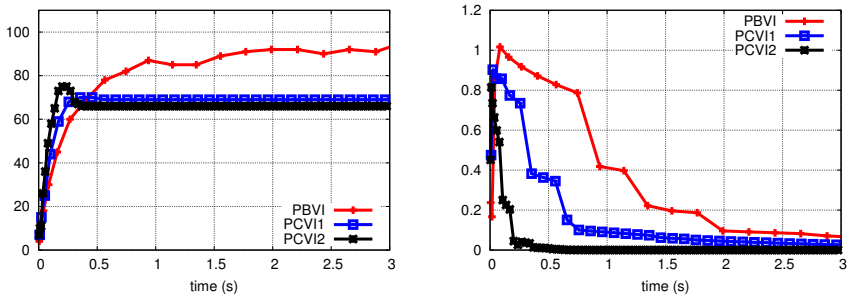
Fig. 4. Multi-target detection, identification & inspection

land in the zone over which it flies; increase the view angle of the observed car by 45 degrees, feasible *only at* altitude 30 meters; change altitude, without constraints. The action feasibility constraints only depend on the helicopter's altitude and on the fact that the zone below the helicopter is safe for landing (no obstacles). Thus, the helicopter is equipped of a laser that gives the current altitude, which is directly interpreted as a set of applicable actions for this altitude. Similarly, a simple image processing algorithm based on texture analysis allows the helicopter to known whether the landing action is feasible or not. Note that these "feasibility" observations are totally independent from the ones that give information about cars' detection and identification.

We ran PBVI and PCVI on the scenario of Figure 4(a), which actually contains the target car, parked in zone z_2. Figure 4(c) shows that PBVI's value function grows faster than PCVI's one, especially PCVI2. The latter ignores observations of feasible actions during optimization, thus producing much less α-vectors than PCVI1 or PBVI. Concerning planning time up to convergence, Figure 4(d) shows that PBVI's rate of convergence is lower than PCVI's ones. This is due to the fact that PCVI backups the value function using a smaller number of α-vectors, and uses masks to restrict α-vectors to be defined only over their relevant belief subspaces.

5.3 Classical Benchmarks: hallway and maze

We modified classical benchmarks from the literature that have a similar structure to our coast guard benchmark (see Figure 1(a)), namely hallway and the 2-floor "4x5x2" maze [21], where we forbade the robot to hit walls in order to prevent damages. These problems have identical actions and observations, but differ in the number of states.



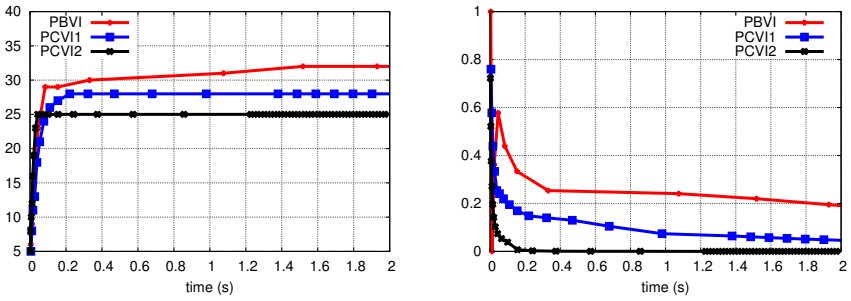
(a) Value function size (b) Bellman error evolution

Fig. 5. The maze domain

Actions consist in going either north, south, east or west. Each observation is composed of 2 symbols: the first one, which is noisy, indicates if the robot is at the goal state; the second is perfectly sensed and informs the robot of the topology of walls around it, which is totally equivalent to informing the robot of the set of feasible actions in its current hidden state, as represented in Figure 1(b).

Results are presented in Figures 5 and 6. On the maze domain, the value function's sizes of PCVI1 and PCVI2 are nearly the same, yet much less than PBVI (see Figure 5(a)). Since PCVI2 ignores observations of feasible actions when operating on α-vectors, this result suggests that there are a few number of such possible observations in this domain. However, Figure 5(b) shows that PCVI2 converges significantly faster than PCVI1, which is itself more efficient than PBVI. Remember that PCVI1 solves the exact same problem as PBVI, yet by explicitly exploiting the semantics of feasible actions, whereas PCVI2 solves a relaxed simpler problem. PBVI can not reason about action feasibility constraints, which are lost in the equivalent but flat unstructured POMDP model.

Concerning the hallway domain, PCVI2 outperforms PCVI1, which is itself better than PBVI, both in terms of value function size and convergence rates. In comparison with the maze domain, PCVI2 is now able to generate significantly less α-vectors than PCVI1, because the number of different possible sets of feasible observations that can be observed (and ignored by PCVI2) is quite large.



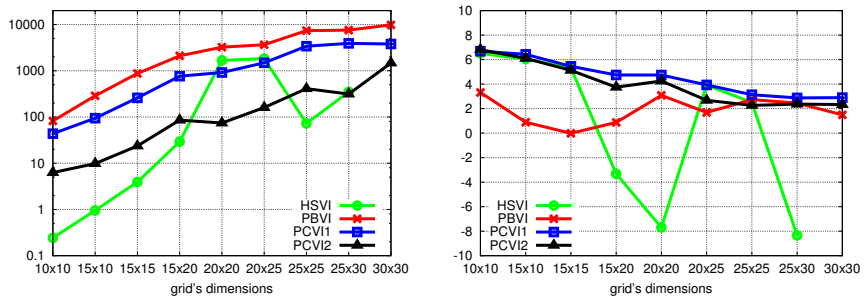
(a) Value function size (b) Bellman error evolution

Fig. 6. The hallway domain

5.4 Larger Navigation Problems

Finally, we tested random navigation problems whose domain is identical to hallway and maze, except that many cells are obstacles that can damage the robot. The problems have also many more states. The robot can observe obstacles around it, using for instance a circular laser sensor. This time, we compared PCVI with HSVI2 [17], which is a heuristic point-based planner that proved to be very efficient in many domains of the literature. As PCVI is adapted from PBVI, which is generally outperformed by HSVI2, we could expect that PVCI would perform poorly in comparison with HSVI2. But Figure 7(a) shows that PCVI2's planning time is actually comparable to HSVI2 for the largest problems, and is even 10 times faster on 2 problems (logarithmic scale). Moreover, HSVI2's policies are very poor on many problems (see Figure 7(b)), especially for the "25x30" problem for which PCVI gets a similar planning time.

Most interestingly, PCVI1 and PCVI2 get similar expected cumulated rewards, as shown in Figure 7(b). Thus, in grid-like problems, on which many robotic applications are based, ignoring the observation of feasible actions at optimization time has no impact on the quality of the optimized policy. Note that PCVI2's solution policy is still guaranteed to contain only feasible actions in each possible belief state: independently from the relaxed lower bound used in place of the optimal value function, PCVI2's α-vectors are anyway defined only over the belief subspace where the corresponding action is feasible. Thus, it can never be the case that the policy is deduced from incoherent α-vectors.



(a) Planning time (seconds) (b) Expected cumulated rewards

Fig. 7. Experiments with various navigation problems

6 Conclusion

We studied a subclass of probabilistic sequential decision-making problems under partial observability, for which the agent's observations contain symbols that represent the set of applicable actions in the current hidden state. This class of problems appears to be very useful at least in autonomous robotics, where such observation symbols are typically obtained from specific or dedicated sensors. Knowing whether action feasibility constraints are only convenient modeling or algorithmic means, is an open question. However, it has been shown in a multi-agent context, that action feasibility constraints can not be equivalently modeled using additional observation symbols and near-infinite costs on infeasible state-action pairs [5]. In any case, exploiting the knowledge of action preconditions can bring a lot, especially in partially observable domains.

References

1. Ghallab, M., Nau, D., Traverso, P.: Automated Planning: theory and practice. Morgan Kaufmann (2004)
2. Fikes, R., Nilsson, N.J.: STRIPS: A new approach to the application of theorem proving to problem solving. Artif. Intell. **2**(3/4) (1971) 189–208
3. Younes, H., Littman, M.: PPDDL1.0: An extension to PDDL for expressing planning domains with probabilistic effects. In: In Proc. of ICAPS. (2003)
4. Sanner, S.: Relational Dynamic Influence Diagram Language (RDDL): Language Description. http://users.cecs.anu.edu.au/∼ssanner/IPPC_2011/RDDL.pdf (2010)
5. Pralet, C., Schiex, T., Verfaillie, G.: Sequential Decision-Making Problems - Representation and Solution. Wiley (2009)
6. Sondik, E.: The optimal control of partially observable Markov processes (1971)
7. Kaelbling, L., Littman, M., Cassandra, A.: Planning and acting in partially observable stochastic domains. AIJ **101**(1-2) (1998)
8. Palacios, H., Geffner, H.: Compiling uncertainty away in conformant planning problems with bounded width. J. Artif. Int. Res. **35**(1) (August 2009) 623–675
9. Pralet, C., Verfaillie, G., Lemaıtre, M., Infantes, G.: Constraint-based Controller Synthesis in Non-Deterministic and Partially Observable Domains. Proc. of ECAI (2010)
10. Littman, M.L., Cassandra, A.R., Kaelbling, L.P.: Learning policies for partially observable environments: Scaling up. In: ICML. (1995) 362–370
11. Pineau, J., Gordon, G., Thrun, S.: Point-based value iteration: An anytime algorithm for POMDPs. In: Proc. of IJCAI. (2003)
12. Ong, S.C.W., Png, S.W., Hsu, D., Lee, W.S.: Planning under uncertainty for robotic tasks with mixed observability. Int. J. Rob. Res. **29**(8) (July 2010) 1053–1068
13. Araya-Lopez, M., Thomas, V., Buffet, O., Charpillet, F.: A Closer Look at MOMDPs. In: Proc. of the 22nd IEEE International Conference on Tools with Artificial Intelligence - Volume 02. ICTAI '10, Washington, DC, USA, IEEE Computer Society (2010) 197–204
14. Bonet, B., Geffner, H.: Solving pomdps: Rtdp-bel vs. point-based algorithms. In: Proceedings of the 21st international jont conference on Artifical intelligence. IJCAI'09, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (2009) 1641–1646
15. Kim, D., Lee, J., Kim, K.E., Poupart, P.: Point-based value iteration for constrained pomdps. In: IJCAI. (2011) 1968–1974
16. Spaan, M., Vlassis, N.: Perseus: Randomized point-based value iteration for POMDPs. JAIR **24** (2005) 195–220
17. Smith, T., Simmons, R.G.: Point-based POMDP algorithms: Improved analysis and implementation. In: Proc. UAI. (2005)
18. Kurniawati, H., Hsu, D., Lee, W.: SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In: Proc. RSS. (2008)
19. Hauskrecht, M.: Value-function approximations for partially observable markov decision processes. J. Artif. Intell. Res. (JAIR) **13** (2000) 33–94
20. Saux, B., Sanfourche, M.: Robust vehicle categorization from aerial images by 3d-template matching and multiple classifier system. In: 7th International Symposium on Image and Signal Processing and Analysis (ISPA). (2011) 466–470
21. Cassandra, A.R.: POMDP's Homepage. http://www.pomdp.org/pomdp/index.shtml (1999)