Multi-agent active perception with prediction rewards
(Supplementary material)

Mikko Lauri
Department of Computer Science
Universität Hamburg
Hamburg, Germany
lauri@informatik.uni-hamburg.de

Frans A. Oliehoek
Department of Computer Science
TU Delft
Delft, The Netherlands
f.a.oliehoek@tudelft.nl

1 Introduction

This supplementary document is structured as follows. We present proofs of the lemmas and the observation omitted from the main paper in Section 2. Additional details on the APAS algorithm and computation of the α-vectors are provided in Section 3. We give details of our experimental setup in Section 4. Finally, additional experimental results are provided in Section 5.

2 Proofs

In this section, we present the proofs omitted from the main paper. Lemmas 1, 2, 3, and 4 are proven in Subsections 2.1, 2.2, 2.3, and 2.4, respectively. Finally, Observation 1 is proven in Subsection 2.5.

For ease of referencing, before stating the proofs we repeat required definitions from the main paper. First, recall the definitions of a Dec-POMDP and a Dec-ρPOMDP, and of the proposed conversion between the two.

Definition 1. *A Dec-POMDP is a tuple $\mathcal{M} = \langle h, I, S, b_0, \mathcal{A}, \mathcal{Z}, T, \mathcal{R}\rangle$, where*

- *$h \in \mathbb{N}$ is the horizon of the problem,*
- *$I = \{1, 2, \dots, n\}$ is a set of n agents,*
- *S is the finite set of states s,*
- *$b_0 \in \Delta(S)$ is the initial state distribution at time step $t = 0$,*
- *$\mathcal{A}$ is the collection of individual action spaces $A_{i,t}$ for each agent $i \in I$ and time step $t = 0, \dots, h-1$. The tuple $a_t = \langle a_{1,t}, a_{2,t}, \dots, a_{n,t}\rangle$ of individual actions is called the joint action at time t,*
- *$\mathcal{Z}$ is the collection of individual observation spaces $Z_{i,t}$ for each agent $i \in I$ and time step $t = 1, \dots, h$. The tuple $z_t = \langle z_{1,t}, z_{2,t}, \dots, z_{n,t}\rangle$ of individual observations is called the joint observation at time step t,*
- *T is the dynamics function specifying the conditional probability $\mathbb{P}(z_{t+1}, s_{t+1} \mid s_t, a_t)$, and*
- *$\mathcal{R}$ is the collection of reward functions $R_t(s_t, a_t)$ for time steps $t = 0, \dots, h-1$.*

Definition 2 (Dec-ρPOMDP)**.** *A Dec-ρPOMDP is a pair $\langle \mathcal{M}, \Gamma\rangle$, where $\mathcal{M}$ is a Dec-POMDP and Γ is a set of tangent hyperplanes that determine the centralized prediction reward $\rho : \Delta(S) \to \mathbb{R}$ defined as $\rho(b) \triangleq \max_{\alpha\in\Gamma} \sum_s b(s)\alpha(s)$.*

Definition 3. *Given a Dec-ρPOMDP $\langle \mathcal{M}, \Gamma\rangle$ with $\mathcal{M} = \langle h, I, S, b_0, \mathcal{A}, \mathcal{Z}, T, \mathcal{R}\rangle$, convert it to a standard Dec-POMDP $\mathcal{M}^+ = \langle h+1, I, S, b_0, \mathcal{A}^+, \mathcal{Z}^+, T^+, \mathcal{R}^+\rangle$ where the horizon is incremented by one and*

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

- *in $\mathcal{A}^+$, the individual action space $A_{i,h}$ for each agent $i \in I$ at time h is a set of **individual prediction actions** $a_{i,h}$, with one individual prediction action for each tangent hyperplane $\alpha \in \Gamma$; for other time steps $A_{i,t}$ are as in $\mathcal{A}$,*

- *in $\mathcal{Z}^+$, the individual observation space $Z_{i,h+1}$ for each agent $i \in I$ at time $h+1$ contains a single null observation; for other time steps $Z_{i,t}$ are as in $\mathcal{Z}$,*

- *$T^+(z_{h+1}, s_{h+1} \mid s_h, a_h)$ has probability of one for the joint null observation and $s_{h+1} = s_h$, and is otherwise zero; for other time steps T^+ is equal to T, and*

- *in $\mathcal{R}^+$, the reward function R_h at time step h is a linear combination of **individual prediction rewards** $R_{i,h}$ of each agent, such that for a joint prediction action $a_h = \langle a_{1,h}, \ldots, a_{n,h} \rangle$ the **decentralized prediction reward** is*

$$R_h(s_h, a_h) = \frac{1}{n} \sum_{i=1}^{n} R_{i,h}(s_h, a_{i,h}), \tag{1}$$

with $R_{i,h}(s_h, a_{i,h}) = \alpha_{a_{i,h}}(s_h)$, where $\alpha_{a_{i,h}} \in \Gamma$ is the tangent hyperplane corresponding to the individual prediction action $a_{i,h}$; for other time steps R_t are as in $\mathcal{R}$.

The sufficient plan-time statistic for initial state distribution b_0 and a past joint policy φ_t is defined as

$$\sigma_t(s_t, \vec{z}_t) \triangleq \mathbb{P}(s_t, \vec{z}_t \mid b_0, \varphi_t), \tag{2}$$

with $\sigma_0(s_0) \triangleq b_0(s_0)$. The sufficient statistic update operator is defined as

$$[U_{ss}(\sigma_t, \delta_t)](s_{t+1}, \vec{z}_{t+1}) \triangleq \sum_{s_t \in S} T(z_{t+1}, s_{t+1} \mid s_t, \delta_t(\vec{z}_t))\sigma_t(s_t, \vec{z}_t). \tag{3}$$

Finally, recall that we write V^π to refer to the expected sum of rewards under a joint policy π, and V^{φ_t} for the expected sum of rewards under a past joint policy φ_t. We use subscripts, for instance $V^\pi_{\mathcal{M}}$, to refer to the expected sum of rewards in a particular Dec-POMDP $\mathcal{M}$.

2.1 Proof of Lemma 1

Lemma 1. *Let $\langle \mathcal{M}, \Gamma \rangle$ be a Dec-ρPOMDP and define $\mathcal{M}^+$ as above. Then, for any past joint policy φ_t with $t \leq h$, the respective plan-time sufficient statistics σ_t in $\langle \mathcal{M}, \Gamma \rangle$ and σ_t^+ in $\mathcal{M}^+$ are equivalent: $\sigma_t \equiv \sigma_t^+$.*

Proof. By induction. As the initial state distributions are equal, clearly $\sigma_0 \equiv \sigma_0^+$. Now assume $\sigma_t \equiv \sigma_t^+$ for some $0 \leq t < h$. By Definition 3 the transition functions T and T^+ are equivalent. Therefore, the respective sufficient statistic update operators U_{ss} and U_{ss}^+ as defined in Eq. (3) are equivalent for the next decision rule δ_t. We conclude that $\sigma_{t+1} \equiv \sigma_{t+1}^+$. □

2.2 Proof of Lemma 2

Lemma 2 (Optimal joint prediction rule)**.** *Let $\langle \mathcal{M}, \Gamma \rangle$ be a Dec-ρPOMDP and define $\mathcal{M}^+$ as above, and let σ_h be a plan-time sufficient statistic for any past joint policy. Then the joint prediction rule $\phi^* = \langle \phi_1^*, \ldots, \phi_n^* \rangle$ where each individual prediction rule ϕ_i^* is defined as*

$$\phi_i^*(\vec{z}_{i,h}, \sigma_h) \triangleq \operatorname*{argmax}_{a_{i,h} \in A_{i,h}} \sum_{\vec{z}_{-i,h}, s_h} \sigma_h(s_h, \vec{z}_{-i,h} \mid \vec{z}_{i,h}) R_{i,h}(s_h, a_{i,h}) \tag{4}$$

maximizes the expected decentralized prediction reward, i.e, $\hat{R}_h(\sigma_h, \phi^*) = \max_\phi \hat{R}_h(\sigma_h, \phi)$.

Proof. We proceed from the definition of the maximum expected decentralized prediction reward:

$$\max_{\phi} \hat{R}_h(\sigma_h, \phi) \triangleq \max_{\phi} \sum_{\vec{z}_h, s_h} \sigma_h(s_h, \vec{z}_h) R_h(s_h, \phi(\vec{z}_h, \sigma_h)) \tag{5a}$$

$$= \max_{\phi_1, \ldots, \phi_n} \sum_{\vec{z}_h, s_h} \sigma_h(s_h, \vec{z}_h) \frac{1}{n} \sum_{i=1}^{n} R_{i,h}(s_h, \phi_i(\vec{z}_{i,h}, \sigma_h)) \tag{5b}$$

$$= \frac{1}{n} \sum_{i=1}^{n} \max_{\phi_i} \sum_{\vec{z}_h, s_h} \sigma_h(s_h, \vec{z}_h) R_{i,h}(s_h, \phi_i(\vec{z}_{i,h}, \sigma_h)) \tag{5c}$$

$$= \frac{1}{n} \sum_{i=1}^{n} \max_{\phi_i} \sum_{\vec{z}_{i,h}, \vec{z}_{-i,h} s_h} \sigma_h(\vec{z}_{i,h}) \sigma_h(s_h, \vec{z}_{-i,h} \mid \vec{z}_{i,h}) R_{i,h}(s_h, \phi_i(\vec{z}_{i,h}, \sigma_h)) \tag{5d}$$

$$= \frac{1}{n} \sum_{i=1}^{n} \sum_{\vec{z}_{i,h}} \sigma_h(\vec{z}_{i,h}) \max_{a_{i,h} \in A_{i,h}} \sum_{\vec{z}_{-i,h}, s_h} \sigma_h(s_h, \vec{z}_{-i,h} \mid \vec{z}_{i,h}) R_{i,h}(s_h, a_{i,h}) \tag{5e}$$

$$= \frac{1}{n} \sum_{i=1}^{n} \sum_{\vec{z}_{i,h}} \sigma_h(\vec{z}_{i,h}) \sum_{\vec{z}_{-i,h}, s_h} \sigma_h(s_h, \vec{z}_{-i,h} \mid \vec{z}_{i,h}) R_{i,h}(s_h, \phi_i^*(\vec{z}_{i,h}, \sigma_h)) \tag{5f}$$

$$= \hat{R}_h(\sigma_h, \phi^*). \tag{5g}$$

Above, 5b follows by definition of R_h as sum of individual prediction rewards (Eq. (1)) and since ϕ is decentralized. Then 5c and 5d follow by rearranging terms and by law of conditional probability, respectively. Then 5e follows since maximizing over ϕ_i is equivalent to finding an individual prediction action for each $\vec{z}_{i,h}$ that maximizes the expected individual prediction reward. Equality 5f follows from the definition of ϕ_i^*, completing the proof. □

2.3 Proof of Lemma 3

Lemma 3. *The expected decentralized prediction reward $\hat{R}_h(\sigma_h, \phi^*)$ in $\mathcal{M}^+$ is at most equal to the expected centralized prediction reward $\hat{\rho}(\sigma_h)$ in $\langle \mathcal{M}, \Gamma \rangle$, i.e., $\hat{R}_h(\sigma_h, \phi^*) \leq \hat{\rho}(\sigma_h)$.*

Proof. We continue from Eq. (5e):

$$\hat{R}_h(\sigma_h, \phi^*) = \frac{1}{n} \sum_{i=1}^{n} \sum_{\vec{z}_{i,h}} \sigma_h(\vec{z}_{i,h}) \max_{a_{i,h} \in A_{i,h}} \sum_{\vec{z}_{-i,h}, s_h} \sigma_h(s_h, \vec{z}_{-i,h} \mid \vec{z}_{i,h}) R_{i,h}(s_h, a_{i,h}) \tag{6a}$$

$$= \frac{1}{n} \sum_{i=1}^{n} \sum_{\vec{z}_{i,h}} \sigma_h(\vec{z}_{i,h}) \max_{a_{i,h} \in A_{i,h}} \sum_{\vec{z}_{-i,h}} \sigma_h(\vec{z}_{-i,h} \mid \vec{z}_{i,h}) \sum_{s_h} \sigma_h(s_h \mid \vec{z}_{-i,h}, \vec{z}_{i,h}) R_{i,h}(s_h, a_{i,h}) \tag{6b}$$

$$\leq \frac{1}{n} \sum_{i=1}^{n} \sum_{\vec{z}_{i,h}, \vec{z}_{-i,h}} \sigma_h(\vec{z}_{i,h}) \sigma_h(\vec{z}_{-i,h} \mid \vec{z}_{i,h}) \max_{a_{i,h} \in A_{i,h}} \sum_{s_h} \sigma_h(s_h \mid \vec{z}_{-i,h}, \vec{z}_{i,h}) R_{i,h}(s_h, a_{i,h}) \tag{6c}$$

$$= \frac{1}{n} \sum_{i=1}^{n} \sum_{\vec{z}_h} \sigma_h(\vec{z}_h) \max_{a_{i,h} \in A_{i,h}} \sum_{s_h} \sigma_h(s_h \mid \vec{z}_h) R_{i,h}(s_h, a_{i,h}) \tag{6d}$$

$$= \frac{1}{n} \sum_{i=1}^{n} \sum_{\vec{z}_h} \sigma_h(\vec{z}_h) \max_{\alpha \in \Gamma} \sum_{s_h} \sigma_h(s_h \mid \vec{z}_h) \alpha(s_h) \tag{6e}$$

$$= \frac{1}{n} \sum_{i=1}^{n} \hat{\rho}(\sigma_h) = \hat{\rho}(\sigma_h). \tag{6f}$$

Equality 6b follows by law of conditional probability and rearranging the terms. Inequality 6c follows since the expectation of a maximum is greater than or equal to the maximum of the expectation. Then, 6d follows by rearranging terms. Finally, 6e follows due to the one-to-one correspondence between the individual prediction actions $a_{i,h} \in A_{i,h}$ and the tangent hyperplanes $\alpha \in \Gamma$ from Definition 3. □

2.4 Proof of Lemma 4

Lemma 4. *Let $\langle \mathcal{M}, \Gamma \rangle$ and $\mathcal{M}^+$ be as defined above. Let φ_h be a past joint policy for $\mathcal{M}^+$, and let ϕ^* be the optimal joint prediction rule. Then, the value of $\varphi_h \circ \phi^*$ in $\mathcal{M}^+$ is at most equal to the value of φ_h in $\langle \mathcal{M}, \Gamma \rangle$, i.e., $V_{\mathcal{M}^+}^{\varphi_h \circ \phi^*} \leq V_{\langle \mathcal{M}, \Gamma \rangle}^{\varphi_h}$.*

Proof. Suppose that φ_h consists of the joint decision rules $\delta_0, \ldots, \delta_{h-1}$. Now φ_h can be applied as a full joint policy in $\langle \mathcal{M}, \Gamma \rangle$ with value function $V_{\langle \mathcal{M}, \Gamma \rangle}^{\varphi_h}$. Since the value function of a policy is defined as the sum of expected rewards, for any initial plan-time sufficient statistic σ_0,

$$V_{\mathcal{M}^+}^{\varphi_h \circ \phi^*}(\sigma_0) = \sum_{t=0}^{h-1} \hat{R}_t(\sigma_t, \delta_t) + \hat{R}_h(\sigma_h, \phi^*) \leq \sum_{t=0}^{h-1} \hat{R}_t(\sigma_t, \delta_t) + \hat{\rho}(\sigma_h) = V_{\langle \mathcal{M}, \Gamma \rangle}^{\varphi_h}(\sigma_0), \quad (7)$$

where the inequality follows since by Lemma 1 the sufficient plan-time statistics are equivalent and by Lemma 3 the decentralized prediction reward lower bounds the centralized prediction reward. □

2.5 Proof of Observation 1

For convenience, we state again the theorem proven in the main paper, and then give a proof of the observation.

Theorem 1 (Loss due to decentralization)**.** *Consider a Dec-ρPOMDP $\langle \mathcal{M}, \Gamma \rangle$ with the optimal value function $V_{\langle \mathcal{M}, \Gamma \rangle}^*$. Let π be an optimal policy for the standard Dec-POMDP $\mathcal{M}^+$ created as in Definition 3, and denote by φ_h the past joint policy consisting of the first h decision rules of π. Then the difference of $V_{\langle \mathcal{M}, \Gamma \rangle}^*$ and the value function $V_{\langle \mathcal{M}, \Gamma \rangle}^{\varphi_h}$ of applying φ_h to $\langle \mathcal{M}, \Gamma \rangle$ is bounded by*

$$|V_{\langle \mathcal{M}, \Gamma \rangle}^*(\sigma_0) - V_{\langle \mathcal{M}, \Gamma \rangle}^{\varphi_h}(\sigma_0)| \leq 2 \max_{\sigma_h} |\hat{\rho}(\sigma_h) - \hat{R}_h(\sigma_h, \phi^*)|, \quad (8)$$

where ϕ^* is the optimal joint prediction rule.

Observation 1. *Consider the setting of Theorem 1. Assume that the observation sequence of each agent is conditionally independent of the observation sequences of all other agents given the past joint policy and initial state distribution, i.e., for every agent i, $\sigma_h(\vec{z}_h) = \sigma_h(\vec{z}_{i,h})\sigma_h(\vec{z}_{-i,h})$. Then π^* is an optimal joint policy for $\langle \mathcal{M}, \Gamma \rangle$ if and only if $\pi^* \circ \phi^*$ is an optimal joint policy for $\mathcal{M}^+$.*

Proof. We show that $\hat{\rho}(\sigma_h) = \hat{R}_h(\sigma_h, \phi^*)$ under the independence condition, which makes the error bound in Theorem 1 zero. Let σ_h be the plan-time sufficient statistic that maximizes the error. Continue from Eq. (6c), and apply the fact that $\sigma_h(\vec{z}_{-i,h} \mid \vec{z}_{i,h}) = \sigma_h(\vec{z}_{-i,h})$ under the independence condition:

$$\hat{R}_h(\sigma_h, \phi^*) = \frac{1}{n} \sum_{i=1}^{n} \sum_{\vec{z}_{i,h}} \sigma_h(\vec{z}_{i,h}) \max_{a_{i,h} \in A_{i,h}} \sum_{\vec{z}_{-i,h}} \sigma_h(\vec{z}_{-i,h}) \sum_{s_h} \sigma_h(s_h \,|\, \vec{z}_{-i,h}, \vec{z}_{i,h}) R_{i,h}(s_h, a_{i,h}) \quad (9a)$$

$$= \frac{1}{n} \sum_{i=1}^{n} \sum_{\vec{z}_{i,h}, \vec{z}_{-i,h}} \sigma_h(\vec{z}_{i,h}) \sigma_h(\vec{z}_{-i,h}) \max_{a_{i,h} \in A_{i,h}} \sum_{s_h} \sigma_h(s_h \mid \vec{z}_{-i,h}, \vec{z}_{i,h}) R_{i,h}(s_h, a_{i,h}) \quad (9b)$$

$$= \frac{1}{n} \sum_{i=1}^{n} \sum_{\vec{z}_h} \sigma_h(\vec{z}_h) \max_{a_{i,h} \in A_{i,h}} \sum_{s_h} \sigma_h(s_h \mid \vec{z}_h) R_{i,h}(s_h, a_{i,h}) \quad (9c)$$

$$= \frac{1}{n} \sum_{i=1}^{n} \sum_{\vec{z}_h} \sigma_h(\vec{z}_h) \max_{\alpha \in \Gamma} \sum_{s_h} \sigma_h(s_h \mid \vec{z}_h) \alpha(s_h) \quad (9d)$$

$$= \frac{1}{n} \sum_{i=1}^{n} \hat{\rho}(\sigma_h) = \hat{\rho}(\sigma_h), \quad (9e)$$

where the second equality follows since the effect of the expectation over $\vec{z}_{-i,h}$ is the same for every $\vec{z}_{i,h}$. Therefore $V_{\langle \mathcal{M}, \Gamma \rangle}^* = V_{\mathcal{M}^+}^{\pi^* \circ \phi^*}$, and the claim follows. □

Algorithm 1 Adaptive prediction action search (APAS) for Dec-ρPOMDP planning

Input: Dec-ρPOMDP $\langle \mathcal{M}, \Gamma \rangle$, convex function $f\colon \Delta(S) \to \mathbb{R}$, number of linearization points K
Output: Best joint policy found, π_{best}

1: $V_{best} \leftarrow -\infty, \pi_{best} \leftarrow \emptyset$
2: **repeat**
3: *// Policy optimization phase*
4: $\mathcal{M}^+ \leftarrow$ CONVERTDECPOMDP($\mathcal{M}, \Gamma$) ▷ Apply Definition 3
5: $\pi \leftarrow$ PLAN($\mathcal{M}^+$) ▷ Use any Dec-POMDP planner
6: $V \leftarrow$ EVALUATE(π)
7: **if** $V > V_{best}$ **then** $V_{best} \leftarrow V, \pi_{best} \leftarrow \pi$
8: *// Adaptation phase*
9: $\Gamma \leftarrow \emptyset$
10: **for** $k = 1, \ldots, K$ **do**
11: $\vec{z}_h \sim \sigma_h(\vec{z}_h)$ ▷ Sample joint observation sequence $\vec{z}_h$ using π_{best}
12: $b_k \leftarrow \sigma_h(\cdot \mid \vec{z}_h)$ ▷ Final state estimate corresponding to $\vec{z}_h$
13: $\alpha_k \leftarrow \nabla f(b_k) - f^*(\nabla f(b_k))$ ▷ Tangent hyperplane of f at b_k
14: $\Gamma \leftarrow \Gamma \cup \{\alpha_k\}$
15: **end for**
16: **until** converged
17: **return** π_{best}

3 Further details on the APAS algorithm

The APAS algorithm from the main paper is reproduced in Algorithm 1. We provide additional details on how we draw samples from the plan-time sufficient statistic, and how the α-vectors are computed.

To avoid explicitly computing the plan-time sufficient statistic σ_h in the adaptation phase (Lines 9-15), we instead apply rollouts to sample $\vec{z}_h \sim \sigma_h$ as follows. We sample an initial state $s_0 \sim b_0$, and then simulate the Dec-POMDP by taking actions prescribed by π_{best}, sampling the next states s_{t+1} and joint observations z_{t+1} from the dynamics function T until $t = h - 1$. We obtain a sampled sequence $\vec{z}_h$ of joint observations, and a sequence $\vec{a}_h$ of joint actions taken. Then, we apply Bayesian filtering to compute the joint state estimate b_k that is equal to $\sigma_h(\cdot \mid \vec{z}_h)$, i.e., $b_k(s_h) \triangleq \mathbb{P}(s_h \mid \vec{z}_h, \vec{a}_h, b_0)$.

The computation of the tangent hyperplane α_k of f at the joint state estimate b_k is based on the convex conjugate or Fenchel conjugate f^*. We present here a brief overview, details are found in [1, Sect. 3.3.1.]. We derive the special case where f is the negative entropy, but the procedure can be applied to any bounded, convex and differentiable function f. Fix a linearization point $b_k \in \Delta(S)$. Because f is convex and differentiable, the following inequality holds for any $b \in \Delta(S)$:

$$f(b) \geq b^T \left[\nabla f(b_k) - f^*(\nabla f(b_k))\right]. \tag{10}$$

In the problems we consider in the main paper, f is the negative entropy: $f(b) = \sum_s b(s) \ln b(s)$ with $\nabla f(b) = \ln b + 1$.[1] The Fenchel conjugate of f is the log-sum-exp function $f^*(b) = \ln\left(\sum_s e^{b(s)}\right)$. We see that $f^*(\nabla f(b_k)) = \ln\left(\sum_s e^{\ln b(s)+1}\right) = \ln\left(e \sum_s b(s)\right) = \ln(\sum_s b(s)) + \ln e = 1$, since $\sum_s b(s) = 1$. By plugging these values to Eq. (10) we obtain

$$f(b) \geq b^T \left[\nabla f(b_k) - f^*(\nabla f(b_k))\right] = b^T \left(\ln b_k + 1 - 1\right) = b^T \ln b_k = \sum_s b(s)\ \ln b_k(s), \tag{11}$$

from which we identify the α-vector $\alpha_k(s) = \ln b_k(s)$.

A reference implementation of APAS as described here is available in a public repository hosted at `https://github.com/laurimi/multiagent-prediction-reward`.

4 Details of the experimental setup

We present additional details on the algorithms, parameter settings, and the experimental setup we apply.

[1] For a vector b, expressions such as $\ln b$ denote the vector obtained taking the element-wise log of b.

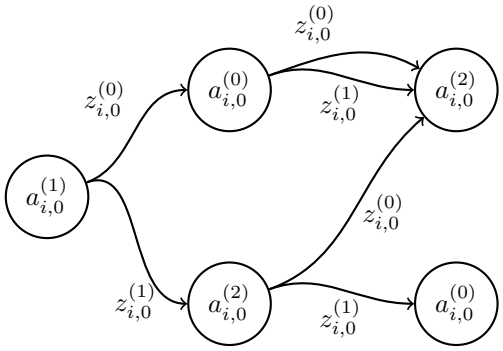

Figure 1: An illustration of a finite state controller representing an individual policy π_i of agent i for horizon $h = 3$. Controller states represented by the circular nodes. There are 2 controller states per time step, except for $t = 0$ where only the leftmost starting node is present. There are three possible individual actions $a_{i,t}^{(j)}$ at $t = 0, 1, 2$. There are two possible individual observations $z_{i,t}^{(j)}$ at $t = 1, 2$.

4.1 Overview of the solution algorithms

We briefly review the NPGI solution algorithm for Dec-ρPOMDPs [2], and the policy graph improvement method of [5] that we use to implement the planning subroutine of our proposed APAS method. The two solution algorithms are closely related, motivating selecting them for our comparison.

NPGI. NPGI [2] represents each agent's policy using a finite state controller (FSC) with a fixed number of controller states. Since NPGI solves a finite horizon problem, each node is identified with a particular time step t in the problem. A conceptual example of a FSC policy is represented in Figure 1. Each controller state is labelled by an individual action $a_{i,t}$ to be taken. At each controller state, a transition function determines the next controller node for each individual observation $z_{i,t+1}$. The transition function is represented by the edges of the directed graph shown in the figure.

NPGI optimizes the actions to take and the transition function of the FSC of each agent by repeating two phases. First, for each controller state, NPGI computes the expected joint state estimate, marginalizing over the possible controller states of the other agents $-i$. Secondly, NPGI solves a local optimization problem at each controller state, finding an improved action and an improved transition function.

The first step of NPGI requires computing *all joint state estimates reachable under the current policy*, increasing the complexity of the algorithm. NPGI solves Dec-ρPOMDPs with reward functions that are convex functions of the joint state estimate. This means that it is difficult to adapt it to use sampling-based approximations or rollouts instead of explicit computation of joint state estimates. In our experiments, we use the implementation of NPGI provided by the authors of [2].

Policy graph improvement. NPGI is a generalization of the policy graph improvement algorithm of [5]. The policy graph improvement algorithm also represents each agent's policy as a FSC with a fixed number of nodes, and operates using the same two phases as NPGI.

For the PLAN subroutine of APAS, we implement the finite-horizon policy graph improvement algorithm based on the description provided in [5]. We modify the algorithm to use sampling and rollouts to estimate values and expected joint state estimates.

4.2 Conversion to standard Dec-POMDP: implementation details

The conversion from Dec-ρPOMDP to a Dec-POMDP increases the horizon by one, and on the newly added time step modifies the action and observation spaces, the transition and observation models, and the reward function. The actions on the newly added time step are the individual prediction actions. The transition and observation models and the reward function on the newly added time step are trivial.

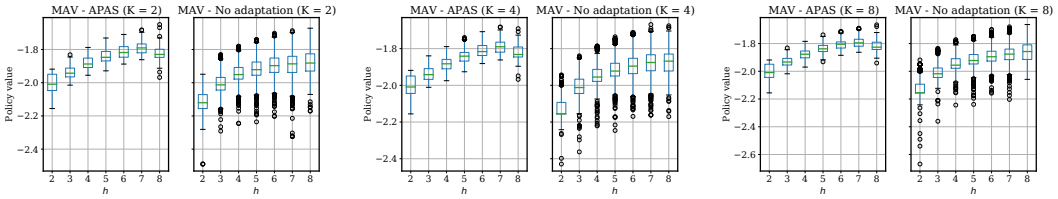

Figure 2: Boxplots of policy values in the MAV domain found by APAS and APAS without the adaptation phase as a function of the horizon h. The plots are arranged in three groups of two plots. From left to right, the groups of two plots report results for $K = 2$, 4, or 8 individual prediction actions. Within each group, the left plot reports the result for APAS, and the right plot the result for APAS without adaptation.

We found it easiest to handle the conversion implicitly, that is, we load the horizon h Dec-ρPOMDP description into memory, and then instruct our solution algorithm to solve the horizon $(h + 1)$ Dec-POMDP while implementing the modifications mentioned above on the final time step directly in the solver. This implicit conversion is fast and its effect on the overall solution time is negligible.

We also experimented with first loading the problem from disk in the `.dpomdp` format [4], then modifying the description to include the changes required before writing it back to disk, thereby explicitly creating the converted standard Dec-POMDP. However, we found this approach to be infeasible, as the `.dpomdp` format is not straightforward to use with time-dependent action and observation spaces and reward functions.

4.3 Parameter settings

As described in the previous subsection, the planning algorithms used in our experiments are closely related. We therefore share the parameters for both of them. In all our experiments, we use FSCs with 2 nodes per time step (see Fig. 1 for an example of the resulting FSCs). 20 policy improvement iterations are executed. We escape local maxima by assigning a random action and a random transition function for a node under optimization with probability 0.1.

4.4 Experimental settings

For APAS and NPGI, we execute 10 runs using each and record in each run the value of the best joint policy found. For APAS without the adaptation phase, using only a single set of randomly sampled α-vectors, we instead run 100 runs. We terminate any run that does not finish within a timeout of 2 hours. Since all algorithms we use are anytime algorithms, we sometimes can obtain results from a partially finished run as well. All experiments were run on a computer with an Intel Core i7-5930K CPU, with 32 GB of memory.

5 Additional experimental results

In this section, we present additional experimental results omitted from the main paper.

Adaptation phase of APAS significantly improves performance. We present here more detailed results on removing the adaptation phase from APAS (Algorithm 1). Instead of executing the adaptation phase, we randomly sample the linearization points and corresponding α-vectors. Figures 2 and 3 shows a comparison of policy values between APAS and APAS without the adaptation phase for the MAV and Rovers domains, respectively. We see that including the adaptation phase consistently improves the average value of policies found in both problem domains. Generally policy values are higher for APAS, while the variance is also lower, indicating usefulness of the adaptation phase. Notably, the worst case performance is much better with the adaptation phase than without it.

We also note that as the horizon increases, both methods experience a decrease in average performance. The greatest policy values found without adaptation sometimes exceed the values of policies found by APAS, e.g., for horizon $h = 10$ with $K = 5$ in the Rovers domain. This suggests that further improvements to APAS might be possible by improving the adaptation phase.

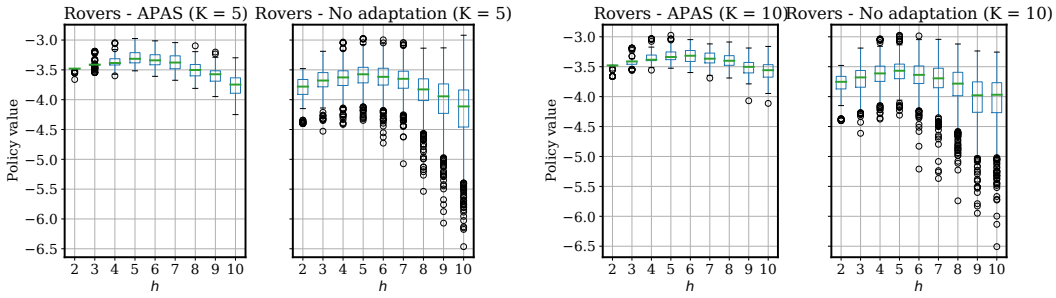

Figure 3: Boxplots of policy values in the Rovers domain found by APAS and APAS without the adaptation phase as a function of the horizon h. From left to right, the groups of two plots report results for $K = 5$ or 10 individual prediction actions. Within each group, the left plot reports the result for APAS, and the right plot the result for APAS without adaptation.

Table 1: Average total runtime of APAS $\pm$ standard deviation in seconds in the MAV domain ($K = 2$) and the Rovers domain ($K = 5$).

Horizon h	MAV (seconds)	Rovers (seconds)
2	1.44 ± 0.15	21.05 ± 0.26
3	2.01 ± 0.12	21.87 ± 0.28
4	2.73 ± 0.14	22.96 ± 0.48
5	3.99 ± 0.18	24.33 ± 0.58
6	14.59 ± 0.21	27.39 ± 0.91
7	190.14 ± 1.65	35.18 ± 2.62
8	2909.11 ± 334.29	65.31 ± 4.63
9	—	177.51 ± 19.41
10	—	607.16 ± 90.82

Effect of the number of α-vectors. The effect of the number K of α-vectors (number of individual prediction actions) on the performance of APAS is shown in Figure 4 for the MAV domain and in Figure 5 for the rovers domain. Each subplot shows for a particular planning horizon h boxplots of the values of policies found by APAS as a function of K. We observe that only in the Rovers domain for $h = 10$ using $K = 10$ individual prediction actions compared to $K = 5$ results in slightly improved performance, although the difference is not very significant.

Timing results. Table 1 shows the average total duration of APAS runs for the MAV domain and Rovers domain using $K = 2$ and $K = 5$ prediction actions, respectively. The average runtime increases strongly as the horizons increases. Upon inspection, we noted that the majority of the time for long horizons was spent evaluating the policies (Line 6 of Algorithm 1). We evaluate policies exactly, by computing all reachable state estimates to evaluate the final reward f at them. This suggests that further scaling in terms of the planning horizon is possible if switching to an approximate evaluation of policy value, e.g., by sampling and evaluating trajectories. We did not explore the effect of the resulting noisy value estimates on the solution quality. Somewhat surprisingly, Table 1 indicates lower runtimes for horizons $h \geq 7$ for the larger Rovers domain. This is due to the domain structure. In any state, most observations in Rovers have zero probability as the agent always correctly observes its own location. These zero-probability observations do not need to be considered in value computation.

Finally, we can compare the time required by APAS and NPGI. While a direct comparison is not fully informative due to differing implementations, we note that in the MAV domain with $h = 5$ a runtime of around 30 seconds per *single backward pass* is reported for NPGI in [3]. As shown in Table 1, for the same domain and horizon, APAS completes the planning (which in our case includes *20 backward passes*) in about 4 seconds.

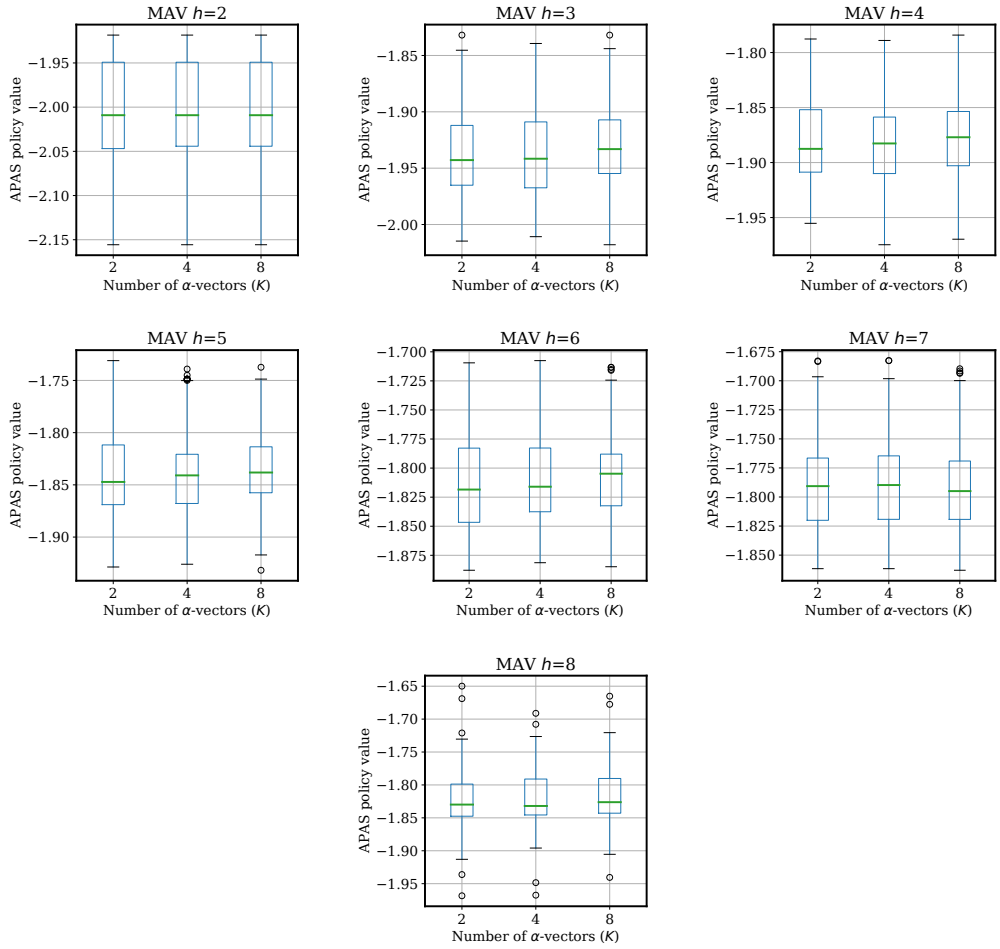

Figure 4: Boxplots of values of policies found by APAS in the MAV domain as a function of the number K of α-vectors.

References

[1] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[2] Mikko Lauri, Joni Pajarinen, and Jan Peters. Information Gathering in Decentralized POMDPs by Policy Graph Improvement. In *Intl. Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1143–1151, 2019.

[3] Mikko Lauri, Joni Pajarinen, and Jan Peters. Multi-agent active information gathering in discrete and continuous-state decentralized POMDPs by policy graph improvement. *Autonomous Agents and Multi-Agent Systems*, 34(42):1–44, 2020. Issue no. 2.

[4] Frans A. Oliehoek, Matthijs TJ Spaan, Bas Terwijn, Philipp Robbel, and João V Messias. The MADP toolbox: an open source library for planning and learning in (multi-) agent systems. *The Journal of Machine Learning Research*, 18(1):3112–3116, 2017.

[5] Joni K Pajarinen and Jaakko Peltonen. Periodic Finite State Controllers for Efficient POMDP and DEC-POMDP Planning. In *Advances in Neural Information Processing Systems*, pages 2636–2644. 2011.

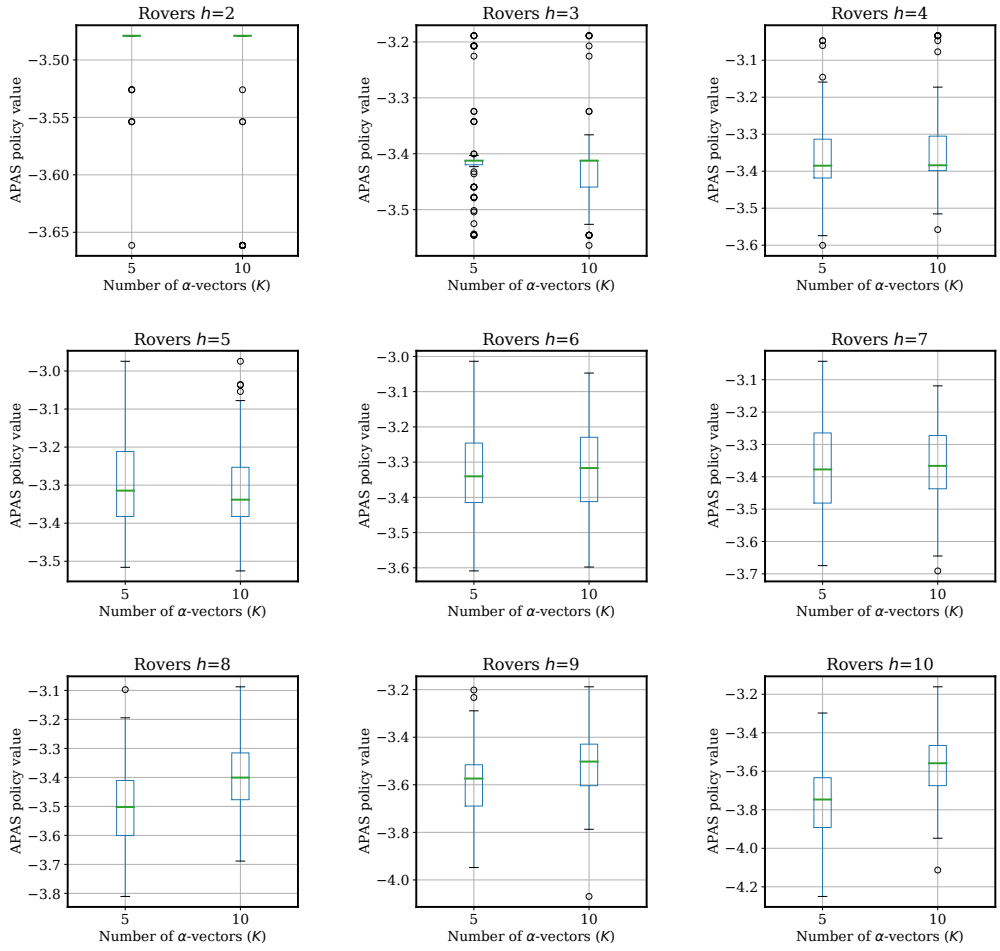

Figure 5: Boxplots of values of policies found by APAS in the Rovers domain as a function of the number K of α-vectors (number of individual prediction actions).