Informed POMDP: Leveraging Additional Information in Model-Based RL

Gaspard Lambrechts [1] Adrien Bolland [1] Damien Ernst [1 2]

Abstract

In this work, we generalize the problem of learning through interaction in a POMDP by accounting for eventual additional information available at training time. First, we introduce the informed POMDP, a new learning paradigm offering a clear distinction between the training information and the execution observation. Next, we propose an objective for learning a sufficient statistic from the history for the optimal control that leverages this information. We then show that this informed objective consists of learning an environment model from which we can sample latent trajectories. Finally, we show for the Dreamer algorithm that the convergence speed of the policies is sometimes greatly improved on several environments by using this informed environment model. Those results and the simplicity of the proposed adaptation advocate for a systematic consideration of eventual additional information when learning in a POMDP using model-based RL.

1. Introduction

Reinforcement learning (RL) aims to learn to act optimally through interaction with environments whose dynamics are unknown. A major challenge in this field is partial observability, where only incomplete observation o of the Markovian state of the environment s is available for taking action a. Such an environment can be formalized as a partially observable Markov decision process (POMDP). In this context, an optimal policy $\eta(a|h)$ generally depends on the history h of observations and past actions, which grows linearly with time. Fortunately, it is theoretically possible to find a statistic $f(h)$ from the history h that summarizes all relevant information to act optimally, and that is recurrent. Formally, a recurrent statistic is updated according to $f(h') = u(f(h), a, o')$ each time that an action a is taken and a new observation o' is received, with $h' = (h, a, o')$. Such a statistic $f(h)$ for which there exists an optimal policy $\eta(a|h) = g(a|f(h))$ is called a sufficient statistic from the history for the optimal control. Standard approaches have thus relied on learning a recurrent policy $\eta_{\theta,\phi}(a|h) = g_\phi(a|f_\theta(h))$, using a recurrent neural network (RNN) f_θ for the statistic. Those policies are simply trained by stochastic gradient ascent of a RL loss using backpropagation through time (Bakker, 2001; Wierstra et al., 2010; Hausknecht & Stone, 2015; Heess et al., 2015; Zhang et al., 2016; Zhu et al., 2017). In this case, the RNN learns a sufficient statistic $f_\theta(h)$ as it learns an optimal policy (Lambrechts et al., 2022; Hennig et al., 2023). Although those standard approaches are theoretically able to implicitly learn a statistic that is sufficient for the optimal control, sufficient statistics can also be learned explicitly. Notably, many works (Igl et al., 2018; Buesing et al., 2018; Han et al., 2019; Gregor et al., 2019; Guo et al., 2020; Lee et al., 2020; Hafner et al., 2019; 2020; 2021; 2023; Guo et al., 2018; Gregor et al., 2019) have focused on learning a recurrent statistic that is predictive sufficient (Bernardo & Smith, 2009) for the reward and next observation given the action: $p(r, o'|h, a) = p(r, o'|f(h), a)$. A recurrent and predictive sufficient statistic is indeed proven to provide a sufficient statistic for the optimal control (Subramanian et al., 2022). It can be noted that in those works, this sufficiency objective is pursued jointly with the RL objective.

Whereas those methods allow one to learn sufficient statistics and optimal policies in the context of POMDP, they learn solely from the partial observations. However, assuming the same partial observability at training time and execution time is too pessimistic for many environments, notably for those that are simulated. We claim that additional information about the state s, be it partial or complete, can be leveraged during training for learning sufficient statistics, in order to increase the supervision of policies. To this end, we generalize the problem of learning from interaction in a POMDP by introducing the informed POMDP. This formalization introduces the training information i about the state s, which is available at training time, but keeps the execution POMDP unchanged. Importantly, this training information is designed such that the observation is conditionally independent of the state given the information. Note that it is

[1] Montefiore Institute, University of Liège, Belgium [2] LTCI, Telecom Paris, Institut Polytechnique de Paris. Correspondence to: Gaspard Lambrechts <gaspard.lambrechts@uliege.be>.

Workshop on New Frontiers in Learning, Control, and Dynamical Systems at the International Conference on Machine Learning (ICML), Honolulu, Hawaii, USA, 2023. Copyright 2023 by the author(s).

always possible to design such an information i, possibly by concatenating the observation o with the eventual additional observations o^+, such that $i = (o, o^+)$. This formalization offers a new learning paradigm where the training information is used along the reward and observation to supervise the learning of the policy.

In the context of informed POMDP, we show that recurrent statistics are sufficient for the optimal control of the execution POMDP when they are predictive sufficient for the reward and next information given the action: $p(r, i'|h, a) = p(r, i'|f(h), a)$. We then derive a convenient objective for finding a predictive sufficient statistic, which amounts to approximating the conditional distribution $p(r, i'|h, a)$ through likelihood maximization using a model $q_\theta(r, i'|f_\theta(h), a)$, where f_θ is a recurrent statistic. Compared to the classic objective for learning sufficient statistics (Igl et al., 2018; Buesing et al., 2018; Han et al., 2019; Hafner et al., 2019), this objective approximates $p(r, i'|h, a)$ instead of $p(r, o'|h, a)$. In addition, we show that this learned generative model $q_\theta(r, i'|f_\theta(h), a)$ is an environment model from which latent trajectories can be generated. Consequently, policies can be optimized in a model-based RL fashion using those generated trajectories. This proposed approach boils down to adapting model-based algorithms, such as PlaNet or Dreamer (Hafner et al., 2019; 2020; 2021; 2023), by relying on a model of the information instead of a model of the observation. We consider several standard environments that we formalize as informed POMDPs (Mountain Hike, Flickering Atari, Velocity Control and Flickering Control). Our informed adaptation of Dreamer is shown to provide a significant convergence speed and performance improvement on some environments, while hurting performances in others, especially in the flickering environments.

Other methods were proposed to account for additional information available at training time. Those approaches, referred to as asymmetric learning, usually learn policies for the POMDP by imitating an expert policy conditioned on the state (Choudhury et al., 2018). Alternatively, asymmetric actor-critic approaches use a critic conditioned on the state (Pinto et al., 2018). However, those heuristic approaches lack a theoretical framework, and the resulting policies are known to be suboptimal for the POMDP (Warrington et al., 2021; Baisero & Amato, 2022; Baisero et al., 2022). Intuitively, under partial observability, optimal policies might indeed need to consider actions that reduce the state uncertainty or that corresponds to safer trajectories. To address those limitations, Warrington et al. (2021) proposes to constrain the expert policy such that its imitation results in an optimal policy in the POMDP. Baisero & Amato (2022) proposed an unbiased state-conditioned critic for asymmetric actor-critic approaches, by introducing the history-state value function $V(h, s)$. Baisero & Amato (2022) adapted this method to value-based RL, where the history-dependent value function $V(h)$ uses from the history-state value function $V(h, s)$ in its temporal difference target. Alternatively, Nguyen et al. (2022) modified the RL objective by trading off the expert imitation objective with respect to the return, resulting in an imitation bonus akin to the entropy in soft actor-critic methods. Finally, in the work that is the closest to ours, Nguyen et al. (2021) proposed, under the strong assumption that beliefs $b(s) = p(s|h)$ are available at training time, to enforce that the statistic $f(h)$ encodes the belief, a sufficient statistic for the optimal control (Åström, 1965). In contrast, we introduce a novel approach that is guaranteed to provide a sufficient statistic for the optimal control, and that leverages the additional information only through the objective. Moreover, our new learning paradigm is not restricted to state supervision, but support any level of additional information. Finally, to the best of our knowledge, our method is the first to exploit additional information for learning an environment model in model-based RL for POMDPs.

This work is structured as follows. In Section 2, the informed POMDP is presented along with the underlying execution POMDP, and its optimal policies. In Section 3, the learning objective for sufficient statistic is presented in the context of informed POMDP. In Section 4, the model-based RL algorithm that is used, Dreamer, is introduced along with our proposed adaptation to informed POMDPs. In Section 5, we compare the performance and convergence speed of the Uninformed Dreamer and the Informed Dreamer in several environments. Finally, in Section 6, we conclude by summarizing the contributions and limitations of this work.

2. Informed Partially Observable Markov Decision Process

In Subsection 2.1, we introduce the informed POMDP and the associated training information, along with the underlying execution POMDP. In Subsection 2.2, we introduce the optimal policies and the reinforcement learning objective in the context of informed POMDPs.

2.1. Informed POMDP and Execution POMDP

Formally, an informed POMDP $\widetilde{\mathcal{P}}$ is defined as a tuple $\widetilde{\mathcal{P}} = (\mathcal{S}, \mathcal{A}, \mathcal{I}, \mathcal{O}, T, R, \widetilde{I}, \widetilde{O}, P, \gamma)$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{I}$ is the information space, and $\mathcal{O}$ is the observation space. The initial state distribution P gives the probability $P(s_0)$ of $s_0 \in \mathcal{S}$ being the initial state of the decision process. The dynamics are described by the transition distribution T that gives the probability $T(s_{t+1}|s_t, a_t)$ of $s_{t+1} \in \mathcal{S}$ being the state resulting from action $a_t \in \mathcal{A}$ in state $s_t \in \mathcal{S}$. The reward function R gives the immediate reward $r_t = R(s_t, a_t)$ obtained at each transition. The information distribution $\widetilde{I}$ gives the probability $\widetilde{I}(i_t|s_t)$ to get information $i_t \in \mathcal{I}$ in state $s_t \in \mathcal{S}$. The observation distribution $\widetilde{O}$ gives the probability $\widetilde{O}(o_t|i_t)$ to get observation $o_t \in \mathcal{O}$ given information i_t. Finally, the discount factor $\gamma \in [0, 1[$ gives the relative importance of future rewards. The main assumption about an informed POMDP is that the observation o_t is conditionally independent of the state s_t given the information i_t: $p(o_t|i_t, s_t) = \widetilde{O}(o_t|i_t)$. In other words, the random variables s_t, i_t and o_t satisfy the Bayesian network $s_t \longrightarrow i_t \longrightarrow o_t$. In practice, it is always possible to define such a training information i_t. For example, the information $i_t = (o_t, o_t^+)$ always satisfies the aforementioned conditional independence, whatever o_t^+ is. Taking a sequence of t actions in the informed POMDP conditions its execution and provides samples $(i_0, o_0, a_0, r_0, \dots, i_t, o_t)$ at training time, as illustrated in Figure 1.

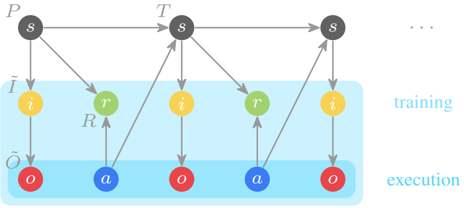

Figure 1. Informed POMDP: Bayesian network of its execution, arrows represent conditional dependencies.

For each informed POMDP, there is an underlying execution POMDP that is defined as $\mathcal{P} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, T, R, O, P, \gamma)$, where $O(o_t|s_t) = \int_{\mathcal{I}} \widetilde{O}(o_t|i)\widetilde{I}(i|s_t)\, \mathrm{d}i$. Taking a sequence of t actions in the execution POMDP conditions its execution and provides the history $h_t = (o_0, a_0, \dots, o_t) \in \mathcal{H}$ at execution time, where $\mathcal{H}$ is the set of histories of arbitrary length. Note that the information samples $i_0, \dots, i_t$ and reward samples $r_0, \dots, r_{t-1}$ are not included in the history, since they are not available at execution time, as illustrated in Figure 1.

2.2. Reinforcement Learning Objective

A policy $\eta \in H$ is defined as a mapping from histories to probability measures over the action space, where $H = \mathcal{H} \rightarrow \Delta(\mathcal{A})$ is the set of such mappings. A policy is said to be optimal for an informed POMDP when it is optimal in the underlying execution POMDP, i.e., when it maximizes the expected return $J(\eta)$, defined as,

$$J(\eta) = \mathop{\mathbb{E}}_{\substack{s_0 \sim P(\cdot) \\ o_t \sim O(\cdot|s_t) \\ a_t \sim \eta(\cdot|h_t) \\ s_{t+1} \sim T(\cdot|s_t, a_t)}} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right]. \tag{1}$$

The RL objective for an informed POMDP is thus to find an optimal policy $\eta^* \in \arg\max_{\eta \in H} J(\eta)$ for the execution POMDP from interaction with the informed POMDP.

3. Optimal Control with Recurrent Sufficient Statistics

In Subsection 3.1, we introduce sufficient statistics for the optimal control and discuss their relation with optimal policies. In Subsection 3.2, we derive an objective for learning in an informed POMDP a recurrent statistic that is sufficient for the optimal control. In Subsection 3.3, we propose a joint objective for learning an optimal recurrent policy with a sufficient statistic. For the sake of conciseness, in this section, we simply use x to denote a random variable at the current time step and x' to denote it at the next time step. Moreover, we use the composition notation $g \circ f$ to denote the history-dependent policy $g(\cdot|f(\cdot))$.

3.1. Recurrent Sufficient Statistics

Let us first define the concept of sufficient statistic, from which a necessary condition for optimality can be derived.

Definition 1 (Sufficient statistic). In an informed POMDP $\widetilde{\mathcal{P}}$ and in its underlying execution POMDP $\mathcal{P}$, a statistic from the history $f \colon \mathcal{H} \rightarrow \mathcal{Z}$ is sufficient for the optimal control if, and only if,

$$\max_{g \colon \mathcal{Z} \rightarrow \Delta(\mathcal{A})} J(g \circ f) = \max_{\eta \colon \mathcal{H} \rightarrow \Delta(\mathcal{A})} J(\eta). \tag{2}$$

Corollary 1 (Sufficiency of optimal policies). In an informed POMDP $\mathcal{P}$ and in its underlying execution POMDP $\widetilde{\mathcal{P}}$, if a policy $\eta = g \circ f$ is optimal, then the statistic $f : \mathcal{H} \rightarrow \mathcal{Z}$ is sufficient for the optimal control.

In this work, we focus on learning recurrent policies, i.e., policies $\eta = g \circ f$ for which the statistic f is recurrent. Formally, we have,

$$\eta(a|h) = g(a|f(h)), \ \forall (h, a), \tag{3}$$
$$f(h') = u(f(h), a, o'), \ \forall h' = (h, a, o'). \tag{4}$$

This allows to process the history iteratively each time that a new action is taken and a new observation is received. According to Corollary 1, when learning a recurrent policy $\eta = g \circ f$, the objective can thus be decomposed into two problems: finding a sufficient statistic f and an optimal conditional distribution g conditioned on this statistic,

$$\max_{\substack{f \colon \mathcal{H} \rightarrow \mathcal{Z} \\ g \colon \mathcal{Z} \rightarrow \Delta(\mathcal{A})}} J(g \circ f). \tag{5}$$

3.2. Learning Recurrent Sufficient Statistics

Below, we provide a sufficient condition for a statistic to be sufficient for the optimal control of an informed POMDP.

Theorem 1 (Sufficiency of recurrent predictive sufficient statistics). In an informed POMDP $\widetilde{\mathcal{P}}$, a statistic $f \colon \mathcal{H} \rightarrow \mathcal{Z}$ is sufficient for the optimal control if it is (i) recurrent and

(ii) predictive sufficient for the reward and next information given the action,

$$\text{(i) } f(h') = u(f(h), a, o'),\ \forall h' = (h, a, o'), \tag{6}$$

$$\text{(ii) } p(r, i'|h, a) = p(r, i'|f(h), a),\ \forall (h, a, r, i'). \tag{7}$$

We provide the proof for this theorem in Appendix A, generalizing earlier work by Subramanian et al. (2022).

Now, let us consider a distribution over the histories and actions whose probability density function writes $p(h, a)$. For example, we consider the stationary distribution induced by the current policy η in the informed POMDP $\widetilde{\mathcal{P}}$. Let us also assume that the probability density function $p(h, a)$ is non-zero everywhere. As shown in Appendix B, under mild assumption, any statistic satisfying the following objective,

$$\max_{\substack{f\colon \mathcal{H}\to\mathcal{Z} \\ q\colon \mathcal{Z}\times\mathcal{A}\to\Delta(\mathbb{R}\times\mathcal{I})}} \mathop{\mathbb{E}}_{p(h,a,r,i')} \log q(r, i'|f(h), a), \tag{8}$$

also satisfies (ii). This variational objective jointly optimizes the statistic function $f\colon \mathcal{H} \to \mathcal{Z}$ with the conditional probability density function $q\colon \mathcal{Z} \times \mathcal{A} \to \Delta(\mathbb{R} \times \mathcal{I})$. According to Theorem 1, a recurrent statistic satisfying objective (8) is thus sufficient for the optimal control.

In practice, both the recurrent statistic and the probability density function are implemented with neural networks f_θ and q_θ, respectively. They are both parametrized by $\theta \in \mathbb{R}^d$, such that the objective can be maximized by stochastic gradient ascent. Regarding f_θ, it is implicitly implemented by an RNN whose update function $z_t = u_\theta(z_{t-1}; x_t)$ is parametrized by θ. The inputs are $x_t = (a_{t-1}, o_t)$, with a_{-1} the null action, which is typically chosen to zero. The hidden state of the RNN $z_t = f_\theta(h_t)$ is thus a statistic from the history that is recurrently updated using u_θ. Regarding q_θ, it is implemented by a parametrized probability density function estimator. The objective writes,

$$\max_{\theta} \underbrace{\mathop{\mathbb{E}}_{p(h,a,r,i')} \log q_\theta(r, i'|f_\theta(h), a)}_{L(f_\theta)}. \tag{9}$$

We might wonder whether this informed objective is better than the classic objective, where $i = o$. In this work, we hypothesize that regressing the information distribution instead of the observation distribution is a better objective in practice. Indeed, according to the data processing inequality applied to the Bayesian network $s' \longrightarrow i' \longrightarrow o'$, the information i' is more informative than the observation o' about the Markovian state s' of the environment,

$$I(s', i'|h, a) \geq I(s', o'|h, a). \tag{10}$$

We thus expect the statistic $f_\theta(h)$ to converge faster towards a sufficient statistic, and the policy to converge faster towards an optimal policy.

3.3. Optimal Control with Recurrent Sufficient Statistics

As seen from Corollary 1, sufficient statistics are needed for the optimal control of POMDPs. Moreover, as we focus on recurrent policies implemented with RNNs, we can exploit objective (9) to learn a sufficient statistic f_θ. In practice, we jointly optimize the RL objective $J(\eta_{\theta,\phi}) = J(g_\phi \circ f_\theta)$ and the statistic objective $L(f_\theta)$. This allows to use the information i to guide the statistic learning through $L(f_\theta)$. This joint objective writes,

$$\max_{\theta,\phi} J(g_\phi \circ f_\theta) + L(f_\theta). \tag{11}$$

A policy $\eta_{\theta,\phi}$ satisfying objectives (11) is guaranteed to satisfy (5) and the policy is thus optimal for the informed and execution POMDP. Note however that there may exist policies satisfying (5) that do not satisfy (11).

The objective $L(f_\theta)$ provides a recurrent model of the reward and next information given the history and action. In the following, we show that we can exploit this model to generate artificial trajectories, called imagined trajectories, under conditions on q_θ. Those imagined trajectories can then be used to maximize the imagined return of the policy, which in turn maximizes $J(g_\phi \circ f_\theta)$ if the model is accurate.

4. Model-Based Reinforcement Learning through Informed World Models

Model-based RL focuses on learning a model of the dynamics $p(r, o'|h, a)$ of the environment, known as a world model. Since this approximate model allows one to generate imagined trajectories, a near-optimal behaviour is usually derived either by online planning or by optimizing a policy based on those trajectories (Sutton, 1991; Ha & Schmidhuber, 2018; Chua et al., 2018; Zhang et al., 2019; Hafner et al., 2019; 2020). In the following, we show that our informed model $q_\theta(r, i'|f_\theta(h), a)$ can be slightly modified to provide an informed world model from which latent trajectories can be sampled. We then propose the Informed Dreamer algorithm, adapting to informed POMDPs the DreamerV3 algorithm (Hafner et al., 2023). In Subsection 4.1, we introduce this informed world model and its training objective. In Subsection 4.2, we present the Informed Dreamer algorithm exploiting this informed world model to train its policy.

4.1. Informed World Model

In this work, we implement the probability density function q_θ with a variational autoencoder (VAE) conditioned on the statistic of the RNN. Together, they form a variational RNN (VRNN) as proposed in (Chung et al., 2015), also known as a recurrent state-space model (RSSM) in the RL context

(Hafner et al., 2019). Formally, we have,

$$\hat{e} \sim q^p_\theta(\cdot|z, a), \qquad \text{(prior, 12)}$$
$$\hat{r} \sim q^r_\theta(\cdot|z, \hat{e}), \qquad \text{(reward decoder, 13)}$$
$$\hat{i}' \sim q^i_\theta(\cdot|z, \hat{e}), \qquad \text{(information decoder, 14)}$$

where $\hat{e}$ is the latent variable of the VAE. The prior q^p_θ and the decoders q^i_θ and q^r_θ are jointly trained with the encoder,

$$e \sim q^e_\theta(\cdot|z, a, o'), \qquad \text{(encoder, 15)}$$

to maximize the likelihood of reward and next information samples. The latent representation $e \sim q^e_\theta(\cdot|z, a, o')$ of the next observation o' can be used to update the statistic to z',

$$z' = u_\theta(z, a, e). \qquad \text{(recurrence, 16)}$$

Note that the statistic z is no longer deterministically updated to z' given a and o', instead we have $z \sim f_\theta(\cdot|h)$, which is induced by u_θ and q^e_θ. This key design choice allows sampling imagined trajectories without reconstructing the imagined observation $\hat{o}'$ by using the latent $\hat{e}$ in update (16), as shown in the next subsection. This requirement of latent representation sampling restricts the class of model-based algorithm that can be adapted using our method.

In practice, we maximize the evidence lower bound (ELBO), a tight variational lower bound on the likelihood of reward and next information samples (Chung et al., 2015),

$$\begin{aligned} \underset{\substack{p(h,a,r,i') \\ f_\theta(z|h)}}{\mathbb{E}} \log q_\theta(r, i'|z, a) \geq \underset{\substack{p(h,a,r,i',o') \\ f_\theta(z|h)}}{\mathbb{E}} \Bigg[\underset{q^e_\theta(e|z,a,o')}{\mathbb{E}} \Big[\\ \log q^i_\theta(i'|z, e) + \log q^r_\theta(r|z, e) \Big] - \\ \mathrm{KL}\left(q^e_\theta(\cdot|z, a, o') \parallel q^p_\theta(\cdot|z, a)\right) \Bigg]. \qquad (17) \end{aligned}$$

Despite the statistic $f_\theta(\cdot|h)$ being stochastic, the ELBO objective ensures that the stochastic statistic $f_\theta(\cdot|h)$ becomes predictive sufficient for the reward and next information. Note that when $i = o$, it corresponds to Dreamer's world model and learning objective. Figure 2 shows, for a sample trajectory $(i_0, o_0, a_0, r_0, \ldots, i_T, o_T)$, the update of the statistic z according to the update function u_θ and the encoder q^e_θ. Maximizing the ELBO maximizes the conditional log-likelihood $q^r_\theta(r|z, e)$ and $q^i_\theta(i|z, e)$ of r and i' for a sample of the encoder $e \sim q^e_\theta(\cdot|z, a, o')$, and minimises the KL divergence from $q^e_\theta(\cdot|z, a, o')$ to the prior distribution $q^p_\theta(\cdot|z, a)$, as highlighted in orange.

4.2. Informed Dreamer

While our informed world model does not learn the observation distribution, it can still generate imagined trajectories. Indeed, the VRNN only uses the latent representation

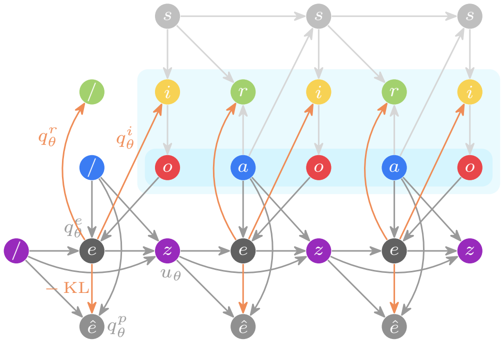

Figure 2. Variational RNN: Bayesian graph of its evaluation for a given trajectory at training time (dependence of q^r_θ and q^i_θ on z is omitted). The loss components are illustrated in orange.

$e \sim q^e_\theta(\cdot|z, a, o')$ of the observation o', trained to reconstruct the information i', in order to update z to z'. Consequently, we can use the prior distribution $\hat{e} \sim q^p_\theta(\cdot|z, a)$, trained to minimise the KL divergence from $q^p_\theta(\cdot|z, a, o')$ in expectation, to generate latent trajectories. The Informed Dreamer algorithm uses this informed world model, a critic $v_\psi(z)$, and a latent policy $a \sim g_\phi(\cdot|z)$. Figure 3 illustrates the generation of a latent trajectory, along with imagined rewards $\hat{r} \sim q^r_\theta(\cdot|z, e)$ and approximate values $\hat{v} = v_\psi(z)$. During generation, the actions are sampled according to $a \sim g_\phi(\cdot|z)$, and any RL algorithm can be used to maximize the imagined returns. Note that the mean imagined reward and estimated values are given by functions that are differentiable with respect to ϕ, such that the imagined return can be directly maximized by stochastic gradient ascent. In the experiments, we use an actor-critic approach for discrete actions and direct maximization for continuous actions, following DreamerV3 (Hafner et al., 2023).

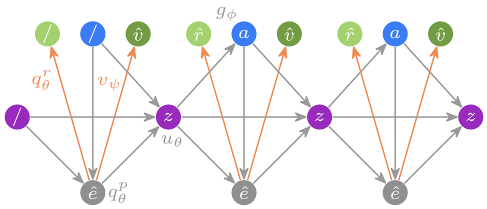

Figure 3. Variational RNN: Bayesian graph of its evaluation when imagining a latent trajectory using policy g_ϕ (dependence of q^r_θ and v_ψ on z is omitted).

A pseudocode for the adaptation of the Dreamer algorithm using this informed world model is given in Appendix C. We also detail some divergences of our formalization with respect to the original Dreamer algorithm (Hafner et al., 2023). Like in DreamerV3, we uses symlog predictions, a discrete VAE, KL balancing, free bits, reward normalisation, a distributional critic, and entropy regularization.

Finally, as shown in Figure 4, when deployed in the execution POMDP, the encoder q^e_θ is used to compute the latent representations of the observations and to update the statistic. The actions are then selected according to $a \sim g_\phi(\cdot|z)$.

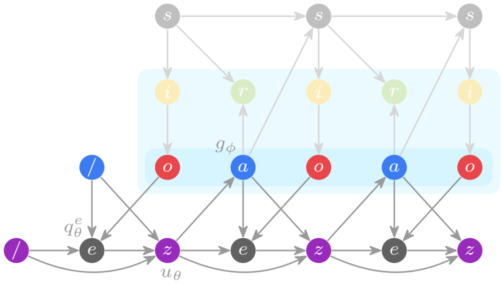

Figure 4. Execution policy: Bayesian graph of its execution in the POMDP using the VRNN encoder q_θ^e and update function u_θ^e to condition the latent policy g_ϕ.

5. Experiments

In this section, we compare Dreamer to the Informed Dreamer on several control problems, formalized as informed POMDPs. We use the implementation of DreamerV3 released at github.com/danijar/DreamerV3 by the authors, and release our adaptation to informed POMDPs at github.com/glambrechts/informed-dreamer. For all environments, we use the same unique set of hyperparameters as in DreamerV3, including for the Informed Dreamer.

5.1. Varying Mountain Hike

In the Varying Mountain Hike environments, the agent is tasked with walking throughout a mountainous terrain. There exists four versions of this environment, depending on the initial state distribution and the type of observation that is available. The agent has a position on a two-dimensional map and can take actions to move relative to its initial orientation. The initial orientation is either always North, or a random cardinal orientation, depending on the environment version. The initial orientation is never available to the agent, but the agent receives a noisy observation of its position or its altitude, depending on the environment version. The reward is given by its altitude relative to the mountain top, such that the goal of the agent is to obtain the highest cumulative altitude. Around the mountain top, states are terminal. The optimal therefore consists in going as fast as possible towards those terminal states while staying on the crests in order to get less negative rewards than in the valleys. We refer the reader to (Lambrechts et al., 2022) for a formal description of this environment, heavily inspired from the Mountain Hike of (Igl et al., 2018).

For this environment, we consider the position and initial orientation to be available as additional information. In other words, we consider the state-informed POMDP with $i = s$. As can be seen from Figure 5, the speed of convergence of the policies is greatly improved when using the Informed Dreamer in this informed POMDP. Moreover, as shown in Table 1, the final performance of the policy is always better than or similar to the Dreamer algorithm.

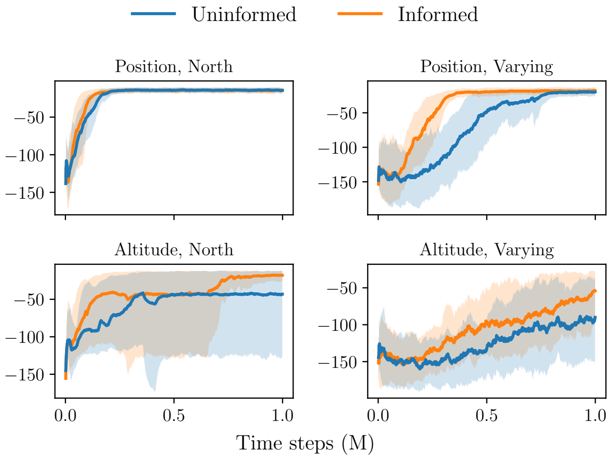

Figure 5. Uninformed Dreamer versus Informed Dreamer ($i = s$) on the Varying Mountain Hike environments: non-discounted return with respect to the number of million steps. Results show the mean, minimum and maximum values over four runs.

Table 1. Final non-discounted reward of Dreamer and Informed Dreamer on the Varying Mountain Hike environments.

ALTITUDE	VARYING	UNINFORMED	INFORMED
FALSE	FALSE	$\mathbf{-14.47 \pm 03.27}$	-14.56 ± 03.45
FALSE	TRUE	-19.84 ± 03.91	$\mathbf{-17.87 \pm 01.18}$
TRUE	FALSE	-43.11 ± 59.89	$\mathbf{-18.04 \pm 11.94}$
TRUE	TRUE	-90.04 ± 35.57	$\mathbf{-54.07 \pm 54.87}$

5.2. Flickering Atari

In the Flickering Atari environments, the agent is tasked with playing the Atari games (Bellemare et al., 2013) on a flickering screen. The dynamics are left unchanged, but the agent may randomly observe a blank screen instead of the game screen, with probability $p = 0.5$. While the classic Atari games are known to have low stochasticity and few partial observability challenges (Hausknecht & Stone, 2015), their flickering counterparts have constituted a classic benchmark in the partially observable RL literature (Hausknecht & Stone, 2015; Zhu et al., 2017; Igl et al., 2018; Ma et al., 2020). Moreover, regarding the recent advances in sample-effiency of model-based RL approaches, we consider the Atari 100k benchmark, where only 100k actions can be taken by the agent for generating samples of interaction.

For these environments, we consider the RAM state of the simulator, a 128-dimensional byte vector, to be available as additional information for supervision. This information vector is indeed guaranteed to satisfy the conditional independence of the informed POMDP: $p(o|i, s) = p(o|i)$. Moreover, we postprocess this additional information by only selecting the subset of variables that are relevant to the game that is considered, according to the annotations provided in (Anand et al., 2019). Depending on the game, this information vector might contain the number of remaining opponents, their positions, the player position, its state, etc.

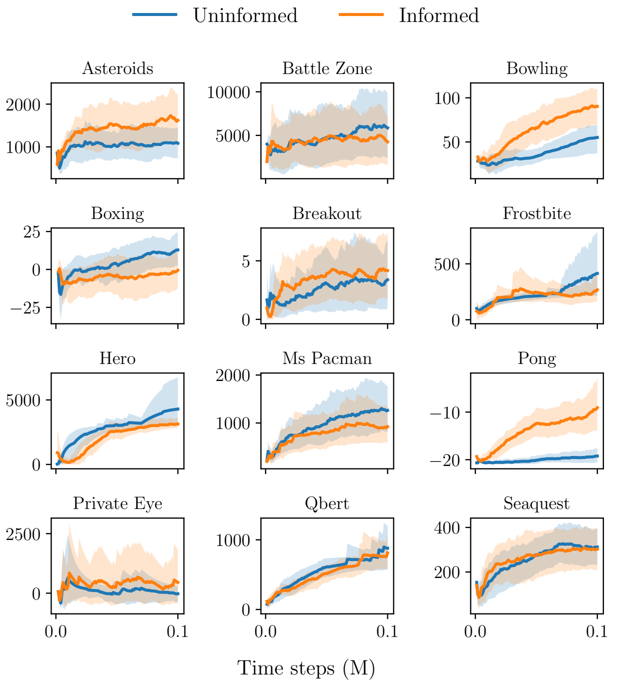

Figure 6. Uninformed Dreamer versus Informed Dreamer ($i = \phi(\text{RAM})$) on the Flickering Atari environments: non-discounted return with respect to the number of million steps. Results show the mean, minimum and maximum values over four runs.

Table 2. Final non-discounted reward of Dreamer and Informed Dreamer on the Flickering Atari environments.

Task	Uninformed	Informed
Asteroids	1085.21 ± 236.29	**1620.98 ± 579.77**
Battle Zone	**5863.99 ± 2081.67**	4258.01 ± 1000.00
Bowling	55.08 ± 13.08	**90.33 ± 04.51**
Boxing	**12.86 ± 03.21**	−0.53 ± 10.69
Breakout	03.38 ± 04.73	**04.17 ± 01.53**
Frostbite	**413.95 ± 377.40**	268.38 ± 490.85
Hero	**4293.33 ± 2534.57**	3133.27 ± 24.66
Ms Pacman	**1262.75 ± 565.18**	923.11 ± 665.01
Pong	−19.24 ± 01.73	**−9.08 ± 15.13**
Private Eye	−23.86 ± 57.74	**448.28 ± 398.36**
Qbert	**879.47 ± 378.32**	812.20 ± 1973.42
Seaquest	**312.08 ± 80.83**	302.60 ± 231.80

Figure 6 shows that the speed of convergence and the performance of the policies is greatly improved by considering additional information for three environments (Asteroids, Bowling, and Pong), while degraded for four others (Boxing, Frostbite, Hero and Ms Pacman) and left similar for the rest. The final non-discounted returns are given in Table 2, offering similar conclusions.

5.3. Velocity Control

In the Velocity Control environments, we consider the standard DeepMind Control task (Tassa et al., 2018) where only the joints velocities are available as observations, and not their absolute positions, which is a standard benchmark in partially observable RL literature (Han et al., 2019; Lee et al., 2020; Warrington et al., 2021). For these environments, we consider the complete state (including the positions) to be available as additional information.

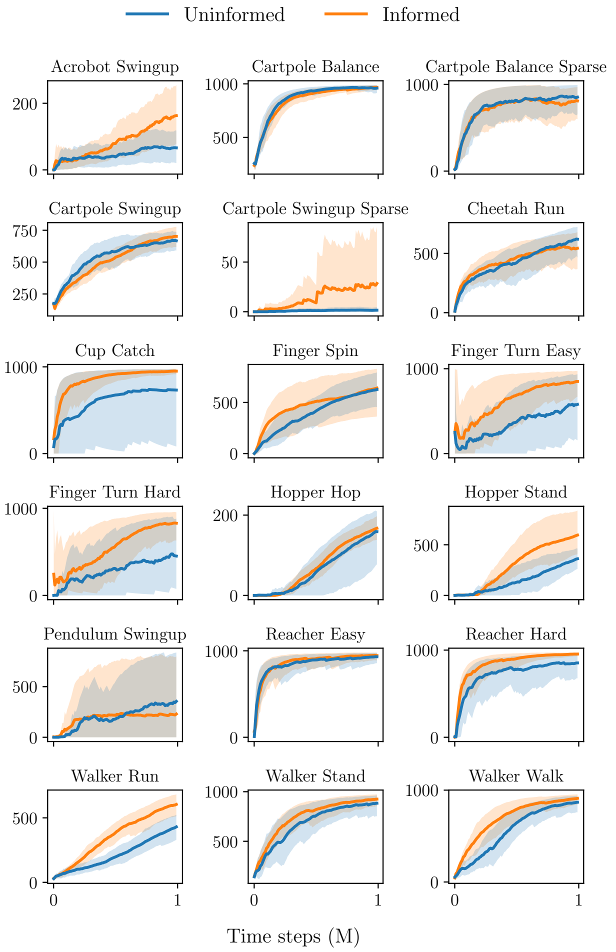

Figure 7. Uninformed Dreamer versus Informed Dreamer ($i = s$) on the Velocity Control environments: non-discounted return with respect to the number of million steps. Results show the mean, minimum and maximum values over four runs.

Figure 7 shows that the speed of convergence and the performance of the policies is greatly improved in this benchmark, for nearly all of the considered games. Moreover, the final non-discounted returns are given in Table 3, and show that the policies obtained after one million time steps are generally better when considering additional information.

Table 3. Final non-discounted reward of Dreamer and Informed Dreamer on the Velocity Control environments.

Task	Uninformed	Informed
Acrobot Swingup	66.21 ± 52.25	**163.01 ± 139.63**
Cartpole Balance	959.60 ± 08.13	**967.45 ± 24.47**
Cartpole Balance Sparse	**852.71 ± 53.15**	810.24 ± 248.14
Cartpole Swingup	667.95 ± 54.72	**701.96 ± 88.14**
Cartpole Swingup Sparse	01.53 ± 03.46	**28.48 ± 109.70**
Cheetah Run	**619.95 ± 241.31**	543.14 ± 136.00
Cup Catch	732.09 ± 477.75	**950.31 ± 48.63**
Finger Spin	626.15 ± 211.54	**640.60 ± 233.99**
Finger Turn Easy	579.49 ± 447.18	**849.73 ± 102.69**
Finger Turn Hard	451.75 ± 479.93	**828.81 ± 132.77**
Hopper Hop	158.88 ± 13.78	**167.22 ± 34.24**
Hopper Stand	361.82 ± 22.89	**595.42 ± 198.96**
Pendulum Swingup	**355.11 ± 406.69**	229.88 ± 479.81
Reacher Easy	931.37 ± 43.92	**944.82 ± 44.94**
Reacher Hard	853.13 ± 102.10	**954.89 ± 14.17**
Walker Run	430.21 ± 83.55	**604.20 ± 75.88**
Walker Stand	883.65 ± 98.58	**925.09 ± 56.47**
Walker Walk	867.97 ± 103.26	**910.38 ± 21.88**

5.4. Flickering Control

In the Flickering Control environments, the agent performs one of the standard DeepMind Control task from images but through a flickering screen. Like for the Flickering Atari environments, the dynamics are left unchanged, except that the agent may randomly observe a blank screen instead of the task screen, with probability $p = 0.5$. For these environments, we consider the state to be available as additional information, as for the Velocity Control environments.

Table 4. Final non-discounted reward of Dreamer and Informed Dreamer on the Flickering Control environments.

Task	Uninformed	Informed
Acrobot Swingup	166.42 ± 117.81	**333.86 ± 147.49**
Cartpole Balance	**988.09 ± 01.57**	943.18 ± 39.97
Cartpole Balance Sparse	971.12 ± 00.00	**979.91 ± 00.00**
Cartpole Swingup	**838.44 ± 23.23**	798.12 ± 28.26
Cartpole Swingup Sparse	485.90 ± 334.90	**677.38 ± 96.19**
Cheetah Run	**683.80 ± 53.87**	590.43 ± 22.62
Cup Catch	**959.79 ± 12.75**	946.11 ± 19.66
Finger Spin	**708.31 ± 397.54**	587.21 ± 188.07
Finger Turn Easy	755.08 ± 483.89	**925.93 ± 20.07**
Finger Turn Hard	568.66 ± 491.80	**887.38 ± 32.84**
Hopper Hop	**279.92 ± 30.22**	213.99 ± 23.51
Hopper Stand	450.49 ± 504.36	**774.22 ± 120.96**
Pendulum Swingup	**797.12 ± 70.80**	741.94 ± 117.27
Reacher Easy	**937.19 ± 16.79**	926.02 ± 67.70
Reacher Hard	**732.34 ± 168.36**	556.36 ± 420.29
Walker Run	**765.40 ± 21.11**	580.77 ± 39.79
Walker Stand	**972.93 ± 39.72**	933.29 ± 96.17
Walker Walk	**957.88 ± 26.84**	898.33 ± 36.68

Regarding this benchmark, considering additional information seem to degrade learning, generally resulting in worse policies. This suggests that not all information is good to learn, some might be irrelevant to the control task and hinders the learning of optimal policies. The final returns are given in Table 4, and offer similar conclusions.

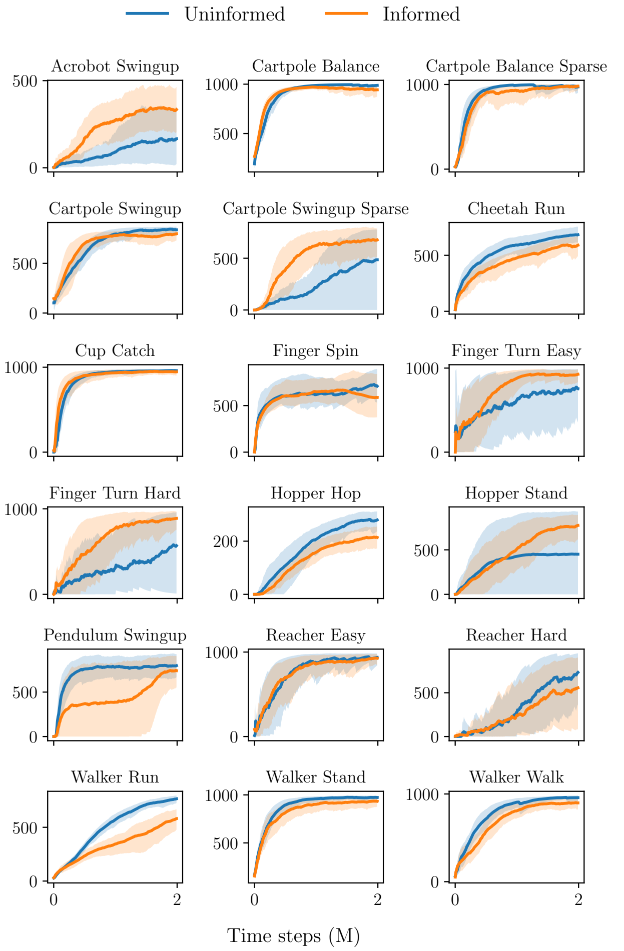

Figure 8. Uninformed Dreamer versus Informed Dreamer ($i = s$) on the Flickering Control environments: non-discounted return with respect to the number of million steps. Results show the mean, minimum and maximum values over four runs.

6. Conclusion

In this work, we introduced a new formalization for considering additional information available at training time for POMDP, called the informed POMDP. In this context, we proposed an objective for learning recurrent sufficient statistic for the optimal control. Next, we showed that this objective can be slightly modified to provide an environment model from which latent trajectories can be generated. We then adapted a successful model-based RL algorithm, known as Dreamer, with this informed world model, resulting in the Informed Dreamer algorithm. By considering several environments from the partially observable RL literature, we showed that this informed learning objective improves the convergence speed and quality of the policies in several environments. However, we also observed that this informed objective hurts performance in some environments, motivating further work in which a particular attention is given to the design of the information i.

Acknowledgements

The authors would like to thank our colleagues Pascal Leroy, Arnaud Delaunoy, Renaud Vandeghen and Florent De Geeter for their valuable comments on this manuscript. Gaspard Lambrechts gratefully acknowledges the financial support of the *Fédération Wallonie-Bruxelles* for his FRIA grant. Adrien Bolland gratefully acknowledges the financial support of the *Fédération Wallonie-Bruxelles* for his FNRS grant. Computational resources have been provided by the *Consortium des Équipements de Calcul Intensif* (CÉCI), funded by the *Fonds de la Recherche Scientifique de Belgique* (F.R.S.-FNRS) under Grant No. 2502011 and by the *Walloon Region*, including the Tier-1 supercomputer of the *Fédération Wallonie-Bruxelles*, infrastructure funded by the *Walloon Region* under Grant No. 1117545.

References

Anand, A., Racah, E., Ozair, S., Bengio, Y., Côté, M.-A., and Hjelm, R. D. Unsupervised State Representation Learning in Atari. *Advances in Neural Information Processing Systems*, 32, 2019.

Åström, K. J. Optimal Control of Markov Processes with Incomplete State Information. *Journal of Mathematical Analysis and Applications*, 10:174–205, 1965.

Baisero, A. and Amato, C. Unbiased Asymmetric Reinforcement Learning under Partial Observability. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pp. 44–52, 2022.

Baisero, A., Daley, B., and Amato, C. Asymmetric DQN for Partially Observable Reinforcement Learning. In *Uncertainty in Artificial Intelligence*, pp. 107–117. PMLR, 2022.

Bakker, B. Reinforcement Learning with Long Short-Term Memory. *Advances in Neural Information Processing Systems*, 14, 2001.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Bernardo, J. M. and Smith, A. F. *Bayesian Theory*, volume 405. John Wiley & Sons, 2009.

Buesing, L., Weber, T., Racaniere, S., Eslami, S., Rezende, D., Reichert, D. P., Viola, F., Besse, F., Gregor, K., Hassabis, D., et al. Learning and Querying Fast Generative Models for Reinforcement Learning. *arXiv preprint arXiv:1802.03006*, 2018.

Choudhury, S., Bhardwaj, M., Arora, S., Kapoor, A., Ranade, G., Scherer, S., and Dey, D. Data-Driven Planning via Imitation Learning. *The International Journal of Robotics Research*, 37(13-14):1632–1672, 2018.

Chua, K., Calandra, R., McAllister, R., and Levine, S. Deep Reinforcement Learning in a Handful of Trials Using Probabilistic Dynamics Models. *Advances in Neural Information Processing Systems*, 31, 2018.

Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C., and Bengio, Y. A Recurrent Latent Variable Model for Sequential Data. *Advances in Neural Information Processing Systems*, 28, 2015.

Gregor, K., Jimenez Rezende, D., Besse, F., Wu, Y., Merzic, H., and van den Oord, A. Shaping Belief States with Generative Environment Models for RL. *Advances in Neural Information Processing Systems*, 32, 2019.

Guo, Z. D., Azar, M. G., Piot, B., Pires, B. A., and Munos, R. Neural Predictive Belief Representations. *arXiv preprint arXiv:1811.06407*, 2018.

Guo, Z. D., Pires, B. A., Piot, B., Grill, J.-B., Altché, F., Munos, R., and Azar, M. G. Bootstrap Latent-Predictive Representations for Multitask Reinforcement Learning. In *International Conference on Machine Learning*, pp. 3875–3886. PMLR, 2020.

Ha, D. and Schmidhuber, J. Recurrent World Models Facilitate Policy Evolution. *Advances in Neural Information Processing Systems*, 31, 2018.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning Latent Dynamics for Planning from Pixels. In *International Conference on Machine Learning*, pp. 2555–2565. PMLR, 2019.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to Control: Learning Behaviors by Latent Imagination. In *International Conference on Learning Representations*, 2020.

Hafner, D., Lillicrap, T., Norouzi, M., and Ba, J. Mastering Atari with Discrete World Models. In *International Conference on Learning Representations*, 2021.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. Mastering Diverse Domains through World Models. *arXiv preprint arXiv:2301.04104*, 2023.

Han, D., Doya, K., and Tani, J. Variational Recurrent Models for Solving Partially Observable Control Tasks. In *Internal Conference on Learning Representations*, 2019.

Hausknecht, M. and Stone, P. Deep Recurrent Q-Learning for Partially Observable MDPs. In *2015 AAAI Fall Symposium Series*, 2015.

Heess, N., Hunt, J. J., Lillicrap, T. P., and Silver, D. Memory-Based Control with Recurrent Neural Networks. *arXiv preprint arXiv:1512.04455*, 2015.

Hennig, J., Romero Pinto, S. A., Yamaguchi, T., Linderman, S. W., Uchida, N., and Gershman, S. J. Emergence of Belief-Like Representations through Reinforcement Learning. *bioRxiv*, pp. 2023–04, 2023.

Igl, M., Zintgraf, L., Le, T. A., Wood, F., and Whiteson, S. Deep Variational Reinforcement Learning for POMDPs. In *International Conference on Machine Learning*, pp. 2117–2126. PMLR, 2018.

Lambrechts, G., Bolland, A., and Ernst, D. Recurrent Networks, Hidden States and Beliefs in Partially Observable Environments. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856.

Lee, A. X., Nagabandi, A., Abbeel, P., and Levine, S. Stochastic Latent Actor-Critic: Deep Reinforcement Learning with a Latent Variable Model. *Advances in Neural Information Processing Systems*, 33:741–752, 2020.

Ma, X., Karkus, P., Hsu, D., Lee, W. S., and Ye, N. Discriminative Particle Filter Reinforcement Learning for Complex Partial Observations. In *International Conference on Learning Representations*, 2020.

Nguyen, H., Daley, B., Song, X., Amato, C., and Platt, R. Belief-Grounded Networks for Accelerated Robot Learning under Partial Observability. In *Conference on Robot Learning*, pp. 1640–1653. PMLR, 2021.

Nguyen, H., Baisero, A., Wang, D., Amato, C., and Platt, R. Leveraging Fully Observable Policies for Learning under Partial Observability. In *Conference on Robot Learning*, 2022.

Pinto, L., Andrychowicz, M., Welinder, P., Zaremba, W., and Abbeel, P. Asymmetric Actor Critic for Image-Based Robot Learning. In *14th Robotics: Science and Systems, RSS 2018*. MIT Press Journals, 2018.

Subramanian, J., Sinha, A., Seraj, R., and Mahajan, A. Approximate Information State for Approximate Planning and Reinforcement Learning in Partially Observed Systems. *Journal of Machine Learning Research*, 23(12): 1–83, 2022.

Sutton, R. S. Dyna, an Integrated Architecture for Learning, Planning, and Reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D. d. L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A., et al. Deepmind Control Suite. *arXiv preprint arXiv:1801.00690*, 2018.

Warrington, A., Lavington, J. W., Scibior, A., Schmidt, M., and Wood, F. Robust Asymmetric Learning in POMDPs. In *International Conference on Machine Learning*, pp. 11013–11023. PMLR, 2021.

Wierstra, D., Förster, A., Peters, J., and Schmidhuber, J. Recurrent Policy Gradients. *Logic Journal of the IGPL*, 18(5):620–634, 2010.

Zhang, M., McCarthy, Z., Finn, C., Levine, S., and Abbeel, P. Learning Deep Neural Network Policies with Continuous Memory States. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 520–527. IEEE, 2016.

Zhang, M., Vikram, S., Smith, L., Abbeel, P., Johnson, M., and Levine, S. SOLAR: Deep Structured Representations for Model-Based Reinforcement Learning. In *International Conference on Machine Learning*, pp. 7444–7453. PMLR, 2019.

Zhu, P., Li, X., Poupart, P., and Miao, G. On Improving Deep Reinforcement Learning for POMDPs. *arXiv preprint arXiv:1704.07978*, 2017.

A. Proof of the Sufficiency of Recurrent Predictive Sufficient Statistics

In this section, we prove Theorem 1, that is recalled below.

Theorem 1 (Sufficiency of recurrent predictive sufficient statistics). In an informed POMDP $\widetilde{\mathcal{P}}$, a statistic $f\colon \mathcal{H} \to \mathcal{Z}$ is sufficient for the optimal control if it is (i) recurrent and (ii) predictive sufficient for the reward and next information given the action,

$$\text{(i) } f(h') = u(f(h), a, o'),\ \forall h' = (h, a, o'), \tag{6}$$

$$\text{(ii) } p(r, i'|h, a) = p(r, i'|f(h), a),\ \forall (h, a, r, i'). \tag{7}$$

Proof. From Proposition 4 and Theorem 5 of (Subramanian et al., 2022), we know that a statistic is sufficient for the optimal control of an execution POMDP if it is (i) recurrent and (ii') predictive sufficient for the reward and next *observation* given the action: $p(r, o'|h, a) = p(r, o'|f(h), a)$. Let us consider a statistic $f\colon \mathcal{H} \to \mathcal{A}$ satisfying (i) and (ii). Now, let us show that it also satisfy (ii'). We have,

$$\begin{aligned} p(r, o'|f(h), a) &= \int_{\mathcal{I}} p(r, o', i'|f(h), a)\,\mathrm{d}i' && (18)\\ &= \int_{\mathcal{I}} p(o'|r, i', f(h), a) p(r, i'|f(h), a)\,\mathrm{d}i', && (19)\end{aligned}$$

using the law of total probability and the chain rule. As can be seen from the informed POMDP formalization of Section 2 and the resulting Bayesian network in Figure 1, the Markov blanket of o' is $\{i'\}$. As a consequence, o' is conditionally independent of any other variable given i'. In particular, $p(o'|i', r, f(h), a) = p(o|i')$, such that,

$$p(r, o'|f(h), a) = \int_{\mathcal{I}} p(o'|i') p(r, i'|f(h), a)\,\mathrm{d}i'. \tag{20}$$

From hypothesis (ii), we can write,

$$p(r, o'|f(h), a) = \int_{\mathcal{I}} p(o'|i') p(r, i'|h, a)\,\mathrm{d}i'. \tag{21}$$

Finally, exploiting the Markov blanket $\{i'\}$ of o', the chain rule and the law of total probability again, we have,

$$\begin{aligned} p(r, o'|f(h), a) &= \int_{\mathcal{I}} p(o'|i', r, h, a) p(r, i'|h, a)\,\mathrm{d}i' && (22)\\ &= \int_{\mathcal{I}} p(o', r, i'|h, a)\,\mathrm{d}i' && (23)\\ &= p(r, o'|h, a). && (24)\end{aligned}$$

This proves that (ii) implies (ii'). As a consequence, any statistic satisfying (i) and (ii) is a sufficient statistic from the history for the optimal control of the informed POMDP. □

B. Recurrent Sufficient Statistic Objective

First, let us consider a fixed history h and action a. Let us recall that two density functions $p(r, i'|h, a)$ and $p(r, i'|f(h), a)$ are equal almost everywhere if, and only if, their KL divergence is zero,

$$\mathop{\mathbb{E}}_{p(r,i'|h,a)} \log \frac{p(r, i'|h, a)}{p(r, i'|f(h), a)} = 0. \tag{25}$$

Now, let us consider a probability density function $p(h, a)$ that is non zero everywhere. We have that the KL divergence from $p(r, i'|h, a)$ to $p(r, i'|f(h), a)$ is equal to zero for almost every history h and action a if, and only if, it is zero on expectation over $p(h, a)$, since the KL divergence is non-negative,

$$\mathop{\mathbb{E}}_{p(r,i'|h,a)} \log \frac{p(r, i'|h, a)}{p(r, i'|f(h), a)} \overset{\text{a.e.}}{=} 0 \Leftrightarrow \mathop{\mathbb{E}}_{p(h,a,r,i')} \log \frac{p(r, i'|h, a)}{p(r, i'|f(h), a)} = 0. \tag{26}$$

Rearranging, we have that $p(r, i'|h, a)$ is equal to $p(r, i'|f(h), a)$ for almost every h, a, r and i' if, and only if,

$$\mathop{\mathbb{E}}_{p(h,a,r,i')} \log p(r, i'|h, a) = \mathop{\mathbb{E}}_{p(h,a,r,i')} \log p(r, i'|f(h), a). \tag{27}$$

Now, we recall the data processing inequality, allowing to write, for any statistic f',

$$\mathop{\mathbb{E}}_{p(h,a,r,i')} \log p(r, i'|h, a) \geq \mathop{\mathbb{E}}_{p(h,a,r,i')} \log p(r, i'|f'(h), a). \tag{28}$$

since $h(r, i'|h, a) = h(r, i'|h, f(h), a) \leq h(r, i'|f(h), a)$, $\forall(h, a)$, where $h(x)$ is the differential entropy of random variable x. Assuming that there exists at least one $f \colon \mathcal{H} \to \mathcal{Z}$ for which the inequality is tight, we obtain the following objective for a predictive sufficient statistic f,

$$\max_{f\colon \mathcal{H} \to \mathcal{Z}} \mathop{\mathbb{E}}_{p(h,a,r,i')} \log p(r, i'|f(h), a). \tag{29}$$

Unfortunately, the probability density $p(r, i'|f(h), a)$ is unknown. However, knowing that the distribution that maximizes the log-likelihood of samples from $p(r, i'|f(h), a)$ is $p(r, i'|f(h), a)$ itself, we can write,

$$\mathop{\mathbb{E}}_{p(h,a,r,i')} \log p(r, i'|f(h), a) = \max_{q\colon \mathcal{Z}\times\mathcal{A}\to\Delta(\mathbb{R}\times\mathcal{I})} \mathop{\mathbb{E}}_{p(h,a,r,i')} \log q(r, i'|f(h), a). \tag{30}$$

By jointly maximizing the probability density function $q \colon \mathcal{Z} \times \mathcal{A} \to \Delta(\mathbb{R} \times \mathcal{I})$, we obtain,

$$\max_{\substack{f\colon \mathcal{H}\to\mathcal{Z} \\ q\colon \mathcal{Z}\times\mathcal{A}\to\Delta(\mathbb{R}\times\mathcal{I})}} \mathop{\mathbb{E}}_{p(h,a,r,i')} \log q(r, i'|f(h), a). \tag{31}$$

This objective ensures that the statistic $f(h)$ is predictive sufficient for the reward and next information given the action. If $f(h)$ is a recurrent statistic, then it is also sufficient for the optimal control, according to Theorem 1.

C. Informed Dreamer

The Informed Dreamer algorithm is presented in Algorithm 1. Differences with the Uninformed Dreamer algorithm (Hafner et al., 2020) are highlighted in blue. In addition, it can be noted that in the original Dreamer algorithm, the statistic z_t encodes $h_t = (o_0, a_0, \dots, o_t)$ and a_t, instead of h_t only. As a consequence, the prior distribution $e_t \sim q^p_\theta(\cdot|z_t)$ can be conditioned on the statistic z_t only, instead of the statistic and last action. Similarly, the encoder distribution $e_t \sim q^p_\theta(\cdot|z_t, o_{t+1})$ can be conditioned on the statistic z_t only, instead of the statistic and last action. On the other hand, the latent policy $a_{t+1} \sim g(\cdot|z_t, e_t)$ should be conditioned on the statistic z_t and the new latent e_t to account for the last observation, and the same is true for the value function $v_\psi(z_t, e_t)$. In the experiments, we follow their implementation for both the Uninformed Dreamer and the Informed Dreamer, according to the code that we released at github.com/glambrechts/informed-dreamer.

Following Dreamer, the algorithm introduces the continuation flag c_t, which indicates whether state s_t is terminal. A terminal state s_t is a state from which the agent can never escape, and in which any further action provides a zero reward. It follows that the value function of a terminal state is zero, and trajectories can be truncated at terminal states since we do not need to learn their value or the optimal policy in those states. Alternatively, c_t can be interpreted as an indicator that can be extracted from the observation o_t, but we have decided to make it explicit in the algorithm.

Algorithm 1 Informed Dreamer - Direct Reward Maximization

```
Hyperparameters: Environment steps $S$, steps before training $F$, train ratio $R$, backpropagation horizon $W$, imagination horizon $K$, batch size $N$, replay buffer capacity $B$.
Initialise neural network parameters $\theta$, $\phi$, $\psi$ randomly, initialise empty replay buffer $\mathcal{B}$.
Let $g = 0$, $t = 0$, $a_{-1} = 0$, $r_{-1} = 0$, $z_{-1} = 0$.
Reset the environment and observe $o_0$ and $c_0$ (true at reset).
for $s = 0 \dots S - 1$ do
    // Environment interaction
    Encode observation $o_t$ to $e_{t-1} \sim q^e_\theta(\cdot|z_{t-1}, a_{t-1}, o_t)$.
    Update $z_t = u_\theta(z_{t-1}, a_{t-1}, e_{t-1})$.
    Given the current history $h_t$, take action $a_t \sim g_\phi(\cdot|z_t)$.
    Observe reward $r_t$, information $i_{t+1}$, observation $o_{t+1}$ and continuation flag $c_{t+1}$.
    if $c_{t+1}$ is false (terminal state) then
        Reset $t = 0$.
        Reset the environment and observe $o_0$ and $c_0$ (true at reset).
    end if
    Update $t = t + 1$.
    Add trajectory of last $W$ time steps $(a_{w-1}, r_{w-1}, i_w, o_w, c_w)_{w=t-W+1}^{t}$ to the replay buffer $\mathcal{B}$.
    // Learning
    while $|\mathcal{B}| \geq F \wedge g < Rs$ do
        // Environment learning
        Draw $N$ trajectories of length $W$ $\left\{(a^n_{w-1}, r^n_{w-1}, i^n_w, o^n_w, c^n_w)_{w=0}^{W-1}\right\}_{n=0}^{N-1}$ uniformly from the replay buffer $\mathcal{B}$.
        Compute statistics and encoded latents $\left\{(z^n_w, e^n_w)_{w=-1}^{W-2}\right\}_{n=0}^{N-1} = \text{Encode}\left(u_\theta, q^e_\theta, \left\{(a^n_{w-1}, o^n_w)_{w=0}^{W-1}\right\}_{n=0}^{N-1}\right)$.
        Update $\theta$ using $\nabla_\theta \sum_{n=0}^{N} \sum_{w=-1}^{W-2} L^n_w$, where $a^n_{-1} = 0$ and,
            $L^n_w = \log q^i_\theta(i^n_{w+1}|z^n_w, e^n_w) + \log q^c_\theta(c^n_{w+1}|z^n_w, e^n_w) + \log q^r_\theta(r^n_w|z^n_w, e^n_w) - \text{KL}\left(q^e_\theta(\cdot|z^n_w, a^n_w, o^n_{w+1}) \parallel q^p_\theta(\cdot|z^n_w, a^n_w)\right).$
        // Behaviour learning
        Sample latent trajectories $\left\{\left\{(z^{n,w}_k, \hat{e}^{n,w}_k)_{k=0}^{K-1}\right\}_{w=-1}^{W-2}\right\}_{n=0}^{N-1} = \text{Imagine}\left(u_\theta, q^p_\theta, g_\phi, \left\{(z^n_w, e^n_w, a^n_w)_{w=-1}^{W-2}\right\}_{n=0}^{N-1}\right)$.
        Predict rewards $r^{n,w}_k \sim q^r_\theta(\cdot|z^{n,w}_k, \hat{e}^{n,w}_k)$, continuation flags $c^{n,w}_{k+1} \sim q^c_\theta(\cdot|z^{n,w}_k, \hat{e}^{n,w}_k)$, and values $v^{n,w}_k = v_\psi(z^{n,w}_k)$.
        Compute value targets using $\lambda$-returns, with $G^{n,w}_{K-1} = v^{n,w}_{K-1}$ and
            $G^{n,w}_k = r^{n,w}_k + \gamma c^{n,w}_k \left((1-\lambda) v^{n,w}_{k+1} + \lambda G^{n,w}_{k+1}\right).$
        Update $\phi$ using $\nabla_\phi \sum_{n=0}^{N-1} \sum_{w=-1}^{W-2} \sum_{k=0}^{K-1} G^{n,w}_k$.
        Update $\psi$ using $\nabla_\psi \sum_{n=0}^{N-1} \sum_{w=-1}^{W-2} \sum_{k=0}^{K-1} \|v_\psi(z^{n,w}_k) - \text{sg}(G^{n,w}_k)\|^2$, where sg is the stop-gradient operator.
        Count gradient steps $g = g + 1$
    end while
end for
```

Algorithm 2 Encode

```
inputs: Update function $u_\theta$, encoder $q^e_\theta$, and histories $\left\{(a^n_{w-1}, o^n_w)_{w=0}^{W-1}\right\}_{n=0}^{N-1}$.
Let $z^n_{-1} = 0$.
for $w = 0 \dots W - 1$ do
    Let $e^n_{w-1} \sim q^e_\theta(\cdot|z^n_{w-1}, a^n_{w-1}, o^n_w)$.
    Let $z^n_w = u_\theta(z^n_{w-1}, a^n_{w-1}, e^n_{w-1})$.
end for
returns: $\left\{(z^n_w, e^n_w)_{w=-1}^{W-2}\right\}_{n=0}^{N-1}$.
```

Algorithm 3 Imagine

```
inputs: Update function $u_\theta$, prior $q^p_\theta$, policy $g_\phi$, statistics, encoded latents and actions $\left\{(z^n_w, e^n_w, a^n_w)_{w=-1}^{W-2}\right\}_{n=0}^{N-1}$.
Let $z^{n,w}_{-1} = z^n_w$, $\hat{e}^{n,w}_{-1} = e^n_w$, $a^{n,w}_{-1} = a^n_w$.
for $k = 0 \dots K - 1$ do
    Let $z^{n,w}_k = u_\theta(z^{n,w}_{k-1}, a^{n,w}_{k-1}, \hat{e}^{n,w}_{k-1})$.
    Let $a^{n,w}_k \sim g_\phi(\cdot|z^{n,w}_k)$.
    Let $\hat{e}^{n,w}_k \sim q^p_\theta(\cdot|z^{n,w}_k, a^{n,w}_k)$.
end for
returns: $\left\{\left\{(z^{n,w}_k, \hat{e}^{n,w}_k)_{k=0}^{K-1}\right\}_{w=-1}^{W-2}\right\}_{n=0}^{N-1}$.
```