

348 A Proofs and Derivations

349 A.1 Proof of Proposition 1

350 *Proof.* We begin with the original definition of the objective function $\mathcal{J}(\phi)$:

$$\mathcal{J}(\phi) = \mathbb{E}_{p_\phi(z_{0:T}, a_{0:T-1}, s_{0:T})} \left[\sum_{t=1}^T R_t(s_t, a_{t-1}) \right],$$

351 which defines the expected cumulative reward over a fixed horizon, where the expectation is over full
352 trajectories $(z_{0:T}, a_{0:T-1}, s_{0:T})$ given a policy π_ϕ . By linearity of expectation, we can interchange
353 the expectation and the sum,

$$\mathcal{J}(\phi) = \sum_{t=1}^T \mathbb{E}_{p_\phi(z_{0:T}, a_{0:T-1}, s_{0:T})} [R_t(s_t, a_{t-1})].$$

354 Now, consider the term $\mathbb{E}_{p_\phi(\cdot)} [R_t(s_t, a_{t-1})]$. The reward R_t depends only on the state s_t and the
355 previous action a_{t-1} . We can thus marginalize out variables corresponding to future times $t' > t$ and
356 past states $s_{0:t-1}$:

$$\begin{aligned} \mathcal{J}(\phi) &= \sum_{t=1}^T \mathbb{E}_{p_\phi(z_{0:t}, a_{0:t-1}, s_t)} [R_t(s_t, a_{t-1})] \\ &= \sum_{t=1}^T \mathbb{E} p(s_t | z_{0:t}, a_{0:t-1}) p_\phi(z_{0:t}, a_{0:t-1}) [R_t(s_t, a_{t-1})], \end{aligned}$$

357 where $p(s_t | z_{0:t}, a_{0:t-1})$ is the belief state and $p_\phi(z_{0:t}, a_{0:t-1})$ is the marginal probability of the
358 history. Now, we can apply the law of total expectation to decouple the expectation over the history
359 $(z_{0:t}, a_{0:t-1})$ from the expectation over the state s_t leading to:

$$\mathcal{B}(\phi) := \mathbb{E}_{p_\phi(z_{0:T}, a_{0:T-1})} \left[\sum_{t=1}^T \mathbb{E} p(s_t | z_{0:t}, a_{0:t-1}) [R_t(s_t, a_{t-1})] \right]$$

360 The outer expectation with respect to $p_\phi(z_{0:T}, a_{0:T-1})$ averages over all possible histories of obser-
361 vations and actions generated under the policy π_ϕ . The inner expectation with respect to $p(s_t | \cdot)$
362 averages the reward R_t over the possible states s_t , given a specific history $(z_{0:t}, a_{0:t-1})$. This ob-
363 jective is consistent with the definition of POMDP objectives according to Kaelbling et al. [1998].
364 To simplify the expression, we define an auxiliary function ℓ_t representing the expected immediate
365 reward at time t given the history up to that point as:

$$\ell_t(z_{0:t}, a_{0:t-1}) := \mathbb{E} p(s_t | z_{0:t}, a_{0:t-1}) [R_t(s_t, a_{t-1})].$$

366 This definition effectively integrates out the dependency on the latent state s_t by averaging according
367 to the belief state distribution. Substituting this definition back yields

$$\mathcal{B}(\phi) = \mathbb{E}_{p_\phi(z_{0:T}, a_{0:T-1})} \left[\sum_{t=1}^T \ell_t(z_{0:t}, a_{0:t-1}) \right],$$

368 with $p_\phi(z_{0:T}, a_{0:T-1})$ being the belief-space generative process

$$p_\phi(z_{0:T}, a_{0:T-1}) = p(z_0) \prod_{t=0}^{T-1} p(z_{t+1} | z_{0:t}, a_{0:t}) \pi_\phi(a_t | z_{0:t}, a_{0:t-1}),$$

369 where $p(z_0) = \mathbb{E}_{p(s_0)} [g(z_0 | s_0)]$ and

$$p(z_{t+1} | z_{0:t}, a_{0:t}) = \iint_{\mathcal{S}^2} g(z_{t+1} | s_{t+1}) f(s_{t+1} | s_t, a_t) p(s_t | z_{0:t}, a_{0:t-1}) \mathrm{d}s_t \mathrm{d}s_{t+1},$$

370 as required. □

371 A.2 Proof of Proposition 2

372 *Proof.* We begin with the definition of the risk-sensitive belief-space objective

$$\mathcal{B}_\eta(\phi) := \frac{1}{\eta} \log p_\phi(\mathcal{O}_{1:T}).$$

373 Since $\eta > 0$ is just a scalar, we focus on the gradient of the log-marginal likelihood $\log p_\phi(\mathcal{O}_{1:T})$

$$\begin{aligned} \nabla_\phi \mathcal{B}_\eta(\phi) &\propto \nabla_\phi \log p_\phi(\mathcal{O}_{1:T}) \\ &= \frac{1}{p_\phi(\mathcal{O}_{1:T})} \nabla_\phi p_\phi(\mathcal{O}_{1:T}) \\ &= \frac{1}{p_\phi(\mathcal{O}_{1:T})} \nabla_\phi \int p_\phi(z_{0:T}, a_{0:T-1}) p(\mathcal{O}_{1:T} \mid z_{0:T}, a_{0:T-1}) \mathrm{d}z_{0:T} \mathrm{d}a_{0:T-1}. \end{aligned}$$

374 Under suitable regularity conditions, we can interchange differentiation and integration (see Mohamed
375 et al. [2020] for details on when this is admissible)

$$\nabla_\phi \mathcal{B}_\eta(\phi) \propto \frac{1}{p_\phi(\mathcal{O}_{1:T})} \int \nabla_\phi p_\phi(z_{0:T}, a_{0:T-1}) p(\mathcal{O}_{1:T} \mid z_{0:T}, a_{0:T-1}) \mathrm{d}z_{0:T} \mathrm{d}a_{0:T-1}.$$

376 Next, using the log-ratio trick, $\nabla f = f \nabla \log f$, the expression becomes:

$$\nabla_\phi \mathcal{B}_\eta(\phi) \propto \frac{1}{p_\phi(\mathcal{O}_{1:T})} \int p_\phi(z_{0:T}, a_{0:T-1}, \mathcal{O}_{1:T}) \nabla_\phi \log p_\phi(z_{0:T}, a_{0:T-1}) \mathrm{d}z_{0:T} \mathrm{d}a_{0:T-1}$$

377 Now, let's recall the definition of $p_\phi(z_{0:T}, a_{0:T-1})$ from (2)

$$p_\phi(z_{0:T}, a_{0:T-1}) = p(z_0) \prod_{t=0}^{T-1} p(z_{t+1} \mid z_{0:t}, a_{0:t}) \pi_\phi(a_t \mid z_{0:t}, a_{0:t-1}),$$

378 and the definition of Ψ_T from (5)

$$\Psi_T(z_{0:T}, a_{0:T-1}; \phi) := p_\phi(z_{0:T}, a_{0:T-1} \mid \mathcal{O}_{1:T}) = \frac{p_\phi(z_{0:T}, a_{0:T-1}, \mathcal{O}_{1:T})}{p_\phi(\mathcal{O}_{1:T})}.$$

379 Plugging them back into the gradient expression, we get:

$$\begin{aligned} \nabla_\phi \mathcal{B}_\eta(\phi) &\propto \mathbb{E}_{\Psi_T} \left[\nabla_\phi \log \left\{ p(z_0) \prod_{t=0}^{T-1} p(z_{t+1} \mid z_{0:t}, a_{0:t}) \pi_\phi(a_t \mid z_{0:t}, a_{0:t-1}) \right\} \right] \\ &= \mathbb{E}_{\Psi_T} \left[\nabla_\phi \log \left\{ \prod_{t=0}^{T-1} \pi_\phi(a_t \mid z_{0:t}, a_{0:t-1}) \right\} + \nabla_\phi \log \left\{ p(z_0) \prod_{t=0}^{T-1} p(z_{t+1} \mid z_{0:t}, a_{0:t}) \right\} \right] \\ &= \mathbb{E}_{\Psi_T} \left[\sum_{t=0}^{T-1} \nabla_\phi \log \pi_\phi(a_t \mid z_{0:t}, a_{0:t-1}) \right], \end{aligned}$$

380 as required. \square

381 A.3 Derivation of Equation (8)

382 We omit the dependence on ϕ , and start from the definition of Ψ_T in (5),

$$\begin{aligned}\Psi_T(z_{0:T}, a_{0:T-1}) &= p(z_{0:T}, a_{0:T-1} \mid \mathcal{O}_{1:T}) \\ &= p(z_0 \mid \mathcal{O}_{1:T}) \prod_{t=0}^{T-1} p(a_t \mid z_{0:t}, a_{0:t-1}, \mathcal{O}_{1:T}) p(z_{t+1} \mid z_{0:t}, a_{0:t}, \mathcal{O}_{1:T}) \\ &= p(z_0 \mid \mathcal{O}_{1:T}) \prod_{t=0}^{T-1} p(z_{t+1} \mid z_{0:t}, a_{0:t}, \mathcal{O}_{t+1:T}) p(a_t \mid z_{0:t}, a_{0:t-1}, \mathcal{O}_{t+1:T}).\end{aligned}$$

383 For the final step, we use the fact that z_{t+1} and a_t are conditionally independent of $\mathcal{O}_{1:t}$ given
384 $(z_{0:t}, a_{0:t-1})$. Using Bayes' rule, we have

$$p(a_t \mid z_{0:t}, a_{0:t-1}, \mathcal{O}_{t+1:T}) = \pi(a_t \mid z_{0:t}, a_{0:t-1}) \frac{p(\mathcal{O}_{t+1:T} \mid z_{0:t}, a_{0:t})}{p(\mathcal{O}_{t+1:T} \mid z_{0:t}, a_{0:t-1})},$$

385 where $\pi(a_t \mid z_{0:t}, a_{0:t-1})$ is the prior (unconditioned) policy. Putting everything together, we get

$$\Psi_T(z_{0:T}, a_{0:T-1}) \propto p(z_0 \mid \mathcal{O}_{1:T}) \prod_{t=0}^{T-1} p(z_{t+1} \mid z_{0:t}, a_{0:t}, \mathcal{O}_{t+1:T}) \frac{p(\mathcal{O}_{t+1:T} \mid z_{0:t}, a_{0:t})}{p(\mathcal{O}_{t+1:T} \mid z_{0:t}, a_{0:t-1})},$$

386 as required.

387 B Decision-Making in POMDPs

388 B.1 Bellman's Optimality Equations

389 In this section, we briefly review the fundamentals of optimal decision-making in partially observable
390 Markov decision processes. We use $b_t := p(s_t \mid z_{0:t}, a_{0:t-1})$ to denote the belief state. The solution
391 to a POMDP can be formulated in terms of optimal value functions, V_t and Q_t , over beliefs and
392 actions (b_t, a_t) as follows [Thrun et al., 2005, Chapter 15]:

$$V_t(b_t) := \max_{a_t \in \mathcal{A}} Q_t(b_t, a_t), \quad (9)$$

$$\begin{aligned}Q_t(b_t, a_t) &:= \int p(z_{t+1} \mid b_t, a_t) p(b_{t+1} \mid b_t, a_t, z_{t+1}) \left[\ell_{t+1}(b_{t+1}, a_t) \right. \\ &\quad \left. + V_{t+1}(b_{t+1}) \right] dz_{t+1} db_{t+1},\end{aligned} \quad (10)$$

393 where $\ell_{t+1}(b_{t+1}, a_t) = \mathbb{E}_{b_{t+1}}[R_{t+1}(s_{t+1}, a_t)]$ is the expected reward defined in Proposition 1 and
394 b_{t+1} is the updated belief after following action a_t and observing z_{t+1} , retrieved according to a Bayes
395 filter [Särkkä and Svensson, 2023]:

$$b_{t+1}(s_{t+1}) \propto g(z_{t+1} \mid s_{t+1}) \int f(s_{t+1} \mid s_t, a_t) b_t(s_t) ds_t.$$

396 Unlike in fully observable Markov decision processes (MDPs), where value functions are defined
397 over finite-dimensional spaces, POMDP value functions V_t and Q_t are defined over the belief space
398 — an infinite-dimensional space of probability distributions over \mathcal{S} . This makes solving Bellman's
399 optimality equations significantly more challenging. Crucially, the value of information gathering is
400 implicitly encoded into the value function through the predictive distribution over future observations

$$p(z_{t+1} \mid b_t, a_t) := \iint g(z_{t+1} \mid s_{t+1}) f(s_{t+1} \mid s_t, a_t) b_t(s_t) ds_t ds_{t+1}, \quad (11)$$

401 so that the expected utility of an action a_t depends not only on immediate rewards but also its effect
402 on future belief refinement. This way, an agent can account for the long-term value of observations
403 that improve its understanding of the latent state and thus influence subsequent decisions.

B.2 The QMDP Approximation

To simplify Bellman’s optimality equations in POMDPs, Littman et al. [1995] introduced what is known as the QMDP approximation, which replaces (10) with the approximation

$$\begin{aligned} Q_t(b_t, a_t) &\approx \int b_t(s_t) [Q_t(s_t, a_t)] ds_t \\ &= \iint b_t(s_t) f(s_{t+1} | s_t, a_t) [R_{t+1}(s_{t+1}, a_t) + V_{t+1}(s_{t+1})] ds_{t+1} ds_t, \end{aligned}$$

where $Q_t(s_t, a_t)$ is the Q-function of the corresponding fully observable MDP. While this approach is algorithmically simpler, it relies on a strong, crude assumption — action values are estimated as if the latent state will be perfectly observable from the next time step onwards [Littman et al., 1995]. This effectively ignores the need for, and potential value of, future information gathering, as it fails to account for how future observations z_{t+1} might refine the agent’s knowledge of the latent state and improve decision-making. Consequently, this method decouples the procedures of probabilistic inference and decision-making, which are inherently intertwined in POMDPs, making any reduction in belief uncertainty merely incidental, rather than a directed outcome of the policy. For this reason, methods that use QMDP fail to address the core POMDP challenge of actively balancing the need to reduce state uncertainty (exploration) against maximizing expected extrinsic rewards based on the current state of belief (exploitation). Since optimal POMDP policies must achieve this balance, policies derived via the QMDP approximation are inherently sub-optimal [Ross et al., 2008].

B.3 The Belief-Space Generative Process

The Bellman equations in (9) and (10) are computationally tractable only for a limited class of continuous POMDPs. As a result, state-of-the-art methods typically rely on sampling-based approximations. In reinforcement learning for POMDPs, a key difference from fully observable MDPs lies in the generative process used to produce the necessary Monte Carlo samples. This distinction critically affects how belief-space policy or value iteration algorithms are developed.

In MDPs, interaction with the environment directly yields a transition tuple (s, a, r, s') , sampled according to the true dynamics $f(s' | s, a)$ and reward function $R(s, a, s')$. This generative process aligns with the assumptions underlying Bellman’s optimality equations for MDPs [Puterman, 2014, Sutton and Barto, 2018]. In contrast, optimal decision-making in POMDPs must operate in the space of beliefs. The relevant transition tuple becomes (b, a, r, z', b') , as implied by Bellman’s equations for POMDPs. Simulating such a tuple requires sampling the next observation z' from the *predictive observation* distribution under the *current belief* as described in (11). Recent reinforcement learning approaches for POMDPs that aim to learn in the belief-space deviate from this generative process [Igl et al., 2018, Meng et al., 2021, Yang and Nguyen, 2021]. Rather than sampling z' from the distribution implied by (11), these methods sample observations by interacting with the true environment, effectively drawing from the ground-truth conditional $g(z' | \underline{s}')$, where \underline{s}' is the true, but unobserved, state. Additionally, the reward signal $r = R(s, a, \underline{s}')$ registered through this *true* interaction reflects the immediate value of the hidden state transition executed *within* the environment, rather than the expected reward associated with a belief transition as required by (10).

As a result, belief updates based on direct interactions with the environment yield a next belief b' that evolves along a trajectory driven by state-space dynamics, rather than the idealized belief-space dynamics specified by Bellman’s equations (10). Using tuples (b, a, r, z', b') generated via interaction with the underlying environment in policy or value iteration schemes can lead to updates that diverge from the optimal POMDP solution. In effect, such algorithms appear to estimate a *belief-space* Q-function based on *state-space* Monte Carlo samples of the reward. This mismatch may hinder an algorithm’s ability to accurately estimate the value of information gathering, as observed rewards are tied to specific latent states rather than reflecting their expected utility under the agent’s belief.

C Backward Sampling

C.1 Overview

Backward sampling is an algorithm used to generate more diverse samples from the smoothing distribution [Godsill et al., 2004, Lindsten and Schön, 2013]. While the smoothing distribution can

be represented by the ancestral lineage of the trajectories generated by the particle filter, backward sampling prescribes an additional step: compute the likelihood that each particle from the previous time step could have led to a given current particle, and sample an ancestor from this distribution. As a result, backward sampling can generate trajectories that were not explicitly formed during the forward pass.

We use the notation for Feynman–Kac models from Section 3.1 and follow the presentation in Corenflos [2024, Chapter 3]. The formula for backward sampling is based on the trivial identity

$$\mathbb{Q}_T(x_{0:t} | x_{t+1:T}) \propto \mathbb{Q}_T(x_{0:T}) = \frac{\mathbb{Q}_T(x_{0:T})}{\mathbb{Q}_t(x_{0:t})} \mathbb{Q}_t(x_{0:t}).$$

After performing particle filtering, we will have weighted particles $\{x_{0:t}^n, w_t^n\}_{n=1}^N$ that form a discrete approximation to the filtering distribution $\mathbb{Q}_t(x_{0:t})$ for all $t \in \{0, \dots, T\}$. We can use these to form an approximation to $\mathbb{Q}_T(x_{0:t} | x_{t+1:T})$ as

$$\mathbb{Q}_T(x_{0:t} | x_{t+1:T}) \approx \sum_{n=1}^N \tilde{w}_t^n \delta_{x_{0:t}^n}(x_{0:t}),$$

where the *smoothing weights* \tilde{w}_t^n are given by

$$\tilde{W}_t^n = \frac{\mathbb{Q}_T(x_{0:t}^n, x_{t+1:T})}{\mathbb{Q}_t(x_{0:t}^n)} w_t^n, \quad \tilde{w}_t^n = \tilde{W}_t^n / \sum_{n=1}^N \tilde{W}_t^n. \quad (12)$$

Thus, we need to compute the ratios $\mathbb{Q}_T(x_{0:t}^n, x_{t+1:T})/\mathbb{Q}_t(x_{0:t}^n)$ in addition to the filtering weights w_t^n . The full backward sampling procedure is as follows:

1. Sample a particle at the final time step: $S_T \sim \text{Categorical}(N, \{w_T^n\}_{n=1}^N)$ and $x_T = x_T^{S_T}$.
2. Then for $t \in \{T-1, \dots, 0\}$, sample an ancestor $S_t \sim \text{Categorical}(N, \{\tilde{w}_t^n\}_{n=1}^N)$ and set $x_{t:T} = (x_t^{S_t}, x_{t+1:T})$, with the smoothing weights \tilde{w}_t^n computed using (12).

C.2 Details for the Nested SMC Algorithm

We now discuss how to compute the smoothing weights for our nested SMC algorithm (Algorithm 2). Observe that one "particle" of the outer particle filter is the set $\{z_t, a_{t-1}, B_{t-1}^{1:M}, s_t^{1:M}\} =: \Theta_t$, defined for every $t \in \{1, \dots, T\}$. When doing backward sampling, at time t , we will have already sampled a trajectory $\Theta_{t+1:T}$, and we need to sample an ancestor $\Theta_{0:t}^n$ for some $n \in \{1, \dots, N\}$. Denoting the distribution of $\Theta_{0:t}^n$ by Ψ_t^M , the fraction we compute to sample an ancestor for time t is given by (12)

$$\begin{aligned} \frac{\Psi_T^M(\Theta_{0:t}^n, \Theta_{t+1:T})}{\Psi_t^M(\Theta_{0:t}^n)} &= \frac{\Psi_T^M(z_{0:t}^n, a_{0:t-1}^n, B_{0:t-1}^{1:M}, s_{0:t}^{1:M}, z_{t+1:T}, a_{t:T}, B_{t:T}^{1:M}, s_{t+1:T}^{1:M})}{\Psi_t^M(z_{0:t}^n, a_{0:t-1}^n, B_{0:t-1}^{1:M}, s_{0:t}^{1:M})} \\ &\propto \underbrace{\prod_{s=t}^{T-1} \pi_\phi(a_s | z_{0:t}^n, a_{0:t-1}^n, z_{t+1:s}, a_{t:s-1})}_{\text{Probability of sampling future actions}} \\ &\quad \prod_{m=1}^M \underbrace{w_t^{nB_t^m}}_{\text{Probability of sampling } B_t^m} \underbrace{f(s_{t+1}^m | s_t^{nB_t^m}, a_t)}_{\text{Transition probability for the state}}. \end{aligned} \quad (13)$$

Note that we have only written down the terms that are unique to each ancestor $\Theta_{0:t}^n$, as the terms that are common for all n will be normalized away. By using (12) and the stored filtering weights, we can now perform backward sampling for Algorithm 2.

The algorithm developed thus far, while correct, can be improved in two major ways. First, when we compute the probability of transitioning from a belief state $\{s_t^{nm}, w_t^{nm}\}_{m=1}^M$ to a belief state $\{s_{t+1}^m, 1/M\}_{m=1}^M$, the expression in (13) looks at the pairwise alignment of individual particles (i.e., the probability of sampling s_{t+1}^m from $s_t^{nB_t^m}$). This is likely to yield very low probabilities for all n except the previously traced ancestor index A_{t-1} , because even if the particle sets represent similar

posterior distributions, their pairwise alignment can be arbitrarily poor. To fix this, we use a solution proposed by Iqbal et al. [2024] and integrate over the the resampling indices $B_t^{1:M}$, yielding the following transition probability for belief states

$$\begin{aligned}
p(s_{t+1}^{1:M} | s_t^{1:M}, a_t) &= \int \prod_{m=1}^M f(s_{t+1}^m | s_t^{n_{B_t^m}}, a_t) d\Psi_t^M(B_t^{1:M}) \\
&= \prod_{m=1}^M \int f(s_{t+1}^m | s_t^{n_{B_t^m}}, a_t) d\Psi_t^M(B_t^{1:M}) \\
&= \prod_{m=1}^M \left\{ \sum_{k=1}^M w_t^{n_k} f(s_{t+1}^m | s_t^{n_k}, a_t) \right\}. \tag{14}
\end{aligned}$$

In the second line, we have used the fact that the resampling indices are sampled independently from a categorical distribution. We replace the second line in (13) with (14).

The second improvement we make to the algorithm concerns its computational complexity. Computing the smoothing weights with (13) and 14 for a single n at a single time step t has complexity $\mathcal{O}(M^2T)$, leading to the full backward sampling algorithm having complexity $\mathcal{O}(NM^2T^2)$ to simulate a single trajectory. We improve this to $\mathcal{O}(M^2T^2)$ by using a modified version of backward sampling from Bunch and Godsill [2013], in which the ratio in (13) needs to be computed for only two, rather than N , possible ancestors. For details, we refer to Dau and Chopin [2023].

D Evaluation Details

D.1 Architectures

For history-dependent policies, used in both P3O and SLAC, we use a gated recurrent unit [GRU, Cho et al., 2014], a type of recurrent neural network [RNN, Elman, 1990], to encode the history into a fixed-size vector (Table 1), similar to Yang and Nguyen [2021]. This embedding is passed to a multi-layer perceptron (MLP) to get the mean m and standard deviation σ of the action distribution (Table 2). Actions are then sampled from a Gaussian distribution, $a_t \sim \mathcal{N}(m, \sigma)$.

Table 1: The GRU encoder architecture used for P3O and SLAC.

Layer	Size	Activation
Input	$\dim(\mathcal{Z} \times \mathcal{A})$	-
Dense	256	ReLU
LayerNorm	-	-
Dense	256	ReLU
LayerNorm	-	-
Dense	128	-
LayerNorm	-	-
GRU	128	-
GRU	128	-
Dense	128	-

Table 2: The MLP decoder architecture used for the policies of all algorithms except P3O. For P3O, the noise is independent of the input, and hence the output dense layer has size $\dim(\mathcal{A})$.

Layer	Size	Activation
Input	Variable	-
Dense	256	ReLU
Dense	256	ReLU
Dense	$2 \times \dim(\mathcal{A})$	-

The policy in DVRL uses the belief state, which is a weighted particle set, as input. We concatenate the particles and their weights, flatten them, and use a dense layer to encode the belief state as a vector

of size 32. This encoded vector is then passed to the decoder in Table 2 to generate the mean and standard deviation of the Gaussian action distribution. The process for DualSMC is similar, except we do not include the weights because the particles are resampled before being passed to the policy, and we append the mean of the particles to the belief state before flattening [Wang et al., 2020].

For the belief-dependent policy for P3O, we use the *set transformer* architecture from Lee et al. [2019] to encode the belief state. The set transformer ensures that the network output is invariant to permutations of the particles in the belief state. The encoded belief state is passed to the same MLP decoder in Table 2.

Finally, SLAC, DualSMC, and DVRL also have critic networks. The critic networks for SLAC and DualSMC have the same architecture, given in Table 3. The critic networks for DVRL use the belief state as input (encoded in the same way as for the policy), and the architecture is given in Table 4.

Table 3: Critic network architecture for SLAC and DualSMC. The input is state, action, and time. The input state is randomly sampled from the belief distribution.

Layer	Size	Activation
Input	$\dim(\mathcal{S} \times \mathcal{A}) + 1$	-
Dense	256	ReLU
Dense	256	ReLU
Dense	1	-

Table 4: Critic network architecture for DVRL. The input is the encoded belief state, action, and time.

Layer	Size	Activation
Input	$32 + \dim(\mathcal{A}) + 1$	-
Dense	256	ReLU
Dense	256	ReLU
Dense	1	-

D.2 Training Details

SLAC, DualSMC, and DVRL all use soft actor-critic [SAC, Haarnoja et al., 2018] as the base RL algorithm. Our SAC implementation and the hyperparameters chosen for training are based on the implementations in CleanRL [Huang et al., 2022] and Brax [Freeman et al., 2021]. Please see the code for full hyperparameter specification for all reference algorithms.

For all algorithms considered (including P3O), we use a particle filter with 32 particles to track the belief state. Additionally, for P3O, the outer particle filter has 128 particles. Unlike in Algorithm 1, which would use all 128 trajectories to perform one gradient step, we use mini-batches of 16 trajectories and perform 8 gradient updates per nested SMC run to improve sample efficiency.

For P3O, we use an additional slew rate penalty in the reward function that penalizes large changes in action in adjacent time steps. We found that this was necessary to ensure that the trajectories sampled using Algorithm 2 are smooth-enough for the GRU networks to learn.

All our experiments were carried out on an NVIDIA A100 80GB GPU. For complete details, including the scripts used for the experiments, see the code attached in the supplementary material.

References

- M. Aoki. *Optimization of Stochastic Systems: Topics in Discrete-time Systems*. Mathematics in science and engineering. Academic Press, 1967.
- K. J. Åström. Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174–205, 1965.
- H. Attias. Planning by probabilistic inference. In *International Workshop on Artificial Intelligence and Statistics*, 2003.
- Y. Bar-Shalom and E. Tse. Dual effect, certainty equivalence, and separation in stochastic control. *IEEE Transactions on Automatic Control*, 19(5):494–500, 1974.
- P. Bickel, B. Li, and T. Bengtsson. Sharp failure rates for the bootstrap particle filter in high dimensions. In *Pushing the limits of contemporary statistics: Contributions in honor of Jayanta K. Ghosh*, volume 3, pages 318–330. Institute of Mathematical Statistics, 2008.
- D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.
- C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, 2012.
- P. Bunch and S. Godsill. Improved particle approximations to the joint smoothing distribution using Markov chain Monte Carlo. *IEEE Transactions on Signal Processing*, 61(4):956–963, 2013.
- O. Cappé, E. Moulines, and T. Rydén. *Inference in Hidden Markov Models*. Springer Series in Statistics. Springer, 2005.
- A. Cassandra, M. L. Littman, and N. L. Zhang. Incremental pruning: a simple, fast, exact method for partially observable markov decision processes. In *Conference on Uncertainty in Artificial Intelligence*, page 54–61. Morgan Kaufmann Publishers Inc., 1997.
- X. Chen, Y. M. Mu, P. Luo, S. Li, and J. Chen. Flow-based recurrent belief state learning for POMDPs. In *International Conference on Machine Learning*, 2022.
- K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
- N. Chopin and O. Papaspiliopoulos. *An Introduction to Sequential Monte Carlo*. Springer Cham, 2020.
- N. Chopin, P. E. Jacob, and O. Papaspiliopoulos. SMC²: An efficient algorithm for sequential analysis of state space models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 75(3):397–426, 2013.
- P.-A. Coquelin, R. Deguest, and R. Munos. Particle filter-based policy gradient in POMDPs. In *Advances in Neural Information Processing Systems*, 2008.
- A. Corenflos. *Computationally Efficient Statistical Inference in Markovian Models*. PhD thesis, Aalto University, 2024.
- A. Corenflos, J. Thornton, G. Deligiannidis, and A. Doucet. Differentiable particle filtering via entropy-regularized optimal transport. In *International Conference on Machine Learning*, 2021.
- D. Crisan and J. Míguez. Nested particle filters for online parameter estimation in discrete-time state-space Markov models. *Bernoulli*, 24(4a):3039–3086, 2018.
- H.-D. Dau and N. Chopin. On backward smoothing algorithms. *The Annals of Statistics*, 51(5):2145 – 2169, 2023.

570 P. Dayan and G. E. Hinton. Using expectation-maximization for reinforcement learning. *Neural*
571 *Computation*, 9(2):271–278, 1997.

572 S. Deglurkar, M. H. Lim, J. Tucker, Z. N. Sunberg, A. Faust, and C. Tomlin. Compositional learning-
573 based planning for vision POMDPs. In *Learning for Dynamics and Control Conference*, pages
574 469–482. Pmlr, 2023.

575 P. Del Moral. *Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applica-*
576 *tions*. Springer, 2004.

577 P. Del Moral and L. Miclo. Genealogies and increasing propagation of chaos for Feynman-Kac and
578 genetic models. *The Annals of Applied Probability*, 11(4):1166–1198, 2001.

579 J. L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

580 A. Feldbaum. Dual control theory problems. *IFAC Proceedings Volumes*, 1(2):541–550, 1963.

581 C. D. Freeman, E. Frey, A. Raichuk, S. Girgin, I. Mordatch, and O. Bachem. Brax - A differentiable
582 physics engine for large scale rigid body simulation, 2021. URL [http://github.com/google/](http://github.com/google/brax)
583 [brax](http://github.com/google/brax).

584 S. J. Godsill, A. Doucet, and M. West. Monte Carlo smoothing for nonlinear time series. *Journal of*
585 *the American Statistical Association*, 99(465):156–168, 2004.

586 N. J. Gordon, D. J. Salmond, and A. F. Smith. Novel approach to nonlinear/non-Gaussian Bayesian
587 state estimation. In *IEE proceedings F (Radar and Signal Processing)*, volume 140, pages 107–113,
588 1993.

589 T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep
590 reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*,
591 2018.

592 D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent
593 imagination. *arXiv preprint arXiv:1912.01603*, 2019a.

594 D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson. Learning latent
595 dynamics for planning from pixels. In *International Conference on Machine Learning*, 2019b.

596 D. Han, K. Doya, and J. Tani. Variational recurrent models for solving partially observable control
597 tasks. *arXiv preprint arXiv:1912.10703*, 2019.

598 S. Huang, R. F. J. Dossa, C. Ye, J. Braga, D. Chakraborty, K. Mehta, and J. G. Araújo. Cleanrl:
599 High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of*
600 *Machine Learning Research*, 23(274):1–18, 2022.

601 M. Igl, L. Zintgraf, T. A. Le, F. Wood, and S. Whiteson. Deep variational reinforcement learning for
602 POMDPs. In *International Conference on Machine Learning*, 2018.

603 V. Indelman, L. Carlone, and F. Dellaert. Planning in the continuous domain: A generalized belief
604 space approach for autonomous navigation in unknown environments. *The International Journal*
605 *of Robotics Research*, 34(7):849–882, 2015.

606 S. Iqbal, H. Abdulsamad, S. Pérez-Vieites, S. Särkkä, and A. Corenflos. Recursive nested filtering for
607 efficient amortized Bayesian experimental design. In *NeurIPS Workshop on Bayesian Decision-*
608 *making and Uncertainty*, 2024.

609 L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable
610 stochastic domains. *Artificial Intelligence*, 101(1):99–134, 1998.

611 N. Kantas, A. Doucet, S. S. Singh, J. Maciejowski, and N. Chopin. On particle methods for parameter
612 estimation in state-space models. *Statistical Science*, 30(3), 2015.

613 H. J. Kappen. Path integrals and symmetry breaking for optimal control theory. *Journal of statistical*
614 *mechanics: theory and experiment*, 2005(11):P11011, 2005.

- 615 L. Kocsis and C. Szepesvári. Bandit based Monte-Carlo planning. In *European conference on*
616 *machine learning*, pages 282–293. Springer, 2006.
- 617 H. Kurniawati and V. Yadav. An online POMDP solver for uncertainty planning in dynamic environ-
618 ment. In *Robotics Research: The 16th International Symposium ISRR*, pages 611–629. Springer,
619 2016.
- 620 A. X. Lee, A. Nagabandi, P. Abbeel, and S. Levine. Stochastic latent actor-critic: Deep reinforcement
621 learning with a latent variable model. In *Advances in Neural Information Processing Systems*,
622 2020.
- 623 J. Lee, Y. Lee, J. Kim, A. Kosiosek, S. Choi, and Y. W. Teh. Set transformer: A framework for
624 attention-based permutation-invariant neural networks. In *International Conference on Machine*
625 *Learning*, 2019.
- 626 S. Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv*
627 *preprint arXiv:1805.00909*, 2018.
- 628 W. Li and E. Todorov. Iterative linearization methods for approximately optimal control and estimation
629 of non-linear stochastic system. *International Journal of Control*, 80(9):1439–1453, 2007.
- 630 M. H. Lim, C. J. Tomlin, and Z. N. Sunberg. Voronoi progressive widening: Efficient online solvers
631 for continuous state, action, and observation POMDPs. In *Conference on Decision and Control*,
632 pages 4493–4500. Ieee, 2021.
- 633 F. Lindsten and T. B. Schön. Backward simulation methods for Monte Carlo statistical inference.
634 *Foundations and Trends in Machine Learning*, 6(1):1–143, 2013.
- 635 M. L. Littman, A. R. Cassandra, and L. P. Kaelbling. Learning policies for partially observable
636 environments: Scaling up. In *International Conference on Machine Learning*, pages 362–370.
637 Elsevier, 1995.
- 638 X. Ma, P. Karkus, D. Hsu, W. S. Lee, and N. Ye. Discriminative particle filter reinforcement learning
639 for complex partial observations. *arXiv preprint arXiv:2002.09884*, 2020.
- 640 S. I. Marcus, E. Fernández-Gaucherand, D. Hernández-Hernandez, S. Coraluppi, and P. Fard. Risk
641 sensitive Markov decision processes. In *Systems and Control in the Twenty-First Century*, pages
642 263–279. Birkhäuser Boston, 1997.
- 643 L. Meng, R. Gorbet, and D. Kulić. Memory-based deep reinforcement learning for POMDPs. In *2021*
644 *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 5619–5626.
645 IEEE, 2021.
- 646 V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu.
647 Asynchronous methods for deep reinforcement learning. In *International Conference on Machine*
648 *Learning*, 2016.
- 649 S. Mohamed, M. Rosca, M. Figurnov, and A. Mnih. Monte Carlo gradient estimation in machine
650 learning. *Journal of Machine Learning Research*, 21(132):1–62, 2020.
- 651 R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In
652 *International Conference on Machine Learning*, 2013.
- 653 J. Pineau, G. Gordon, S. Thrun, et al. Point-based value iteration: An anytime algorithm for POMDPs.
654 In *IJCAI*, volume 3, pages 1025–1032, 2003.
- 655 R. Platt, R. Tedrake, L. Kaelbling, and T. Lozano-Perez. Belief space planning assuming maximum
656 likelihood observations. In *Robotics: Science and Systems*, volume 2, 2010.
- 657 J. M. Porta, M. T. Spaan, and N. Vlassis. Robot planning in partially observable continuous domains.
658 In *Robotics: Science and Systems*. Massachusetts Institute of Technology, 2005.
- 659 P. Poupart, N. Vlassis, J. Hoey, and K. Regan. An analytic solution to discrete Bayesian reinforcement
660 learning. In *International Conference on Machine Learning*, pages 697–704, 2006.

661 M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John
662 Wiley & Sons, 2014.

663 K. Rawlik, M. Toussaint, and S. Vijayakumar. On stochastic optimal control and reinforcement
664 learning by approximate inference. In *Robotics: Science and Systems*, 2012.

665 K. C. Rawlik. *On probabilistic Inference Approaches to Stochastic Optimal Control*. PhD thesis,
666 University of Edinburgh, 2013.

667 S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa. Online planning algorithms for POMDPs. *Journal*
668 *of Artificial Intelligence Research*, 32:663–704, 2008.

669 S. Särkkä and L. Svensson. *Bayesian Filtering and Smoothing*. Cambridge University Press, 2nd
670 edition, 2023.

671 D. Silver and J. Veness. Monte-Carlo planning in large POMDPs. In *Advances in Neural Information*
672 *Processing Systems*, volume 23, 2010.

673 G. Singh, S. Peri, J. Kim, H. Kim, and S. Ahn. Structured world belief for reinforcement learning in
674 POMDP. In *International Conference on Machine Learning*, 2021.

675 A. Somani, N. Ye, D. Hsu, and W. S. Lee. DESPOT: Online POMDP planning with regularization.
676 *Advances in Neural Information Processing Systems*, 26, 2013.

677 E. J. Sondik. *The Optimal Control of Partially Observable Markov Processes*. Stanford University,
678 1971.

679 R. F. Stengel. *Optimal Control and Estimation*. Courier Corporation, 1994.

680 Z. Sunberg and M. Kochenderfer. Online algorithms for POMDPs with continuous state, action,
681 and observation spaces. In *International Conference on Automated Planning and Scheduling*,
682 volume 28, pages 259–263, 2018.

683 R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition,
684 2018.

685 S. Thrun. Monte Carlo POMDPs. In *Advances in Neural Information Processing Systems*, volume 12,
686 1999.

687 S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

688 E. Todorov. General duality between optimal control and estimation. In *Conference on Decision and*
689 *Control*, pages 4286–4292. IEEE, 2008.

690 M. Toussaint and A. Storkey. Probabilistic inference for solving discrete and continuous state Markov
691 decision processes. In *International Conference on Machine Learning*, pages 945–952, 2006.

692 E. Tse and Y. Bar-Shalom. Adaptive dual control for stochastic nonlinear systems with free end-time.
693 *IEEE Transactions on Automatic Control*, 20(5):670–675, 1975.

694 J. Van Den Berg, S. Patil, and R. Alterovitz. Efficient approximate value iteration for continuous
695 Gaussian POMDPs. In *Conference on Artificial Intelligence*, volume 26, pages 1832–1838, 2012.

696 Y. Wang, B. Liu, J. Wu, Y. Zhu, S. S. Du, L. Fei-Fei, and J. B. Tenenbaum. Dualsmc: Tunneling
697 differentiable filtering and planning under continuous POMDPs. In *Proceedings of the Twenty-*
698 *Ninth International Joint Conference on Artificial Intelligence*, 2020.

699 J. Watson and J. Peters. Inferring smooth control: Monte Carlo posterior policy iteration with
700 Gaussian processes, 2022.

701 J. Watson, H. Abdulsamad, and J. Peters. Stochastic optimal control as approximate input inference.
702 In *Conference on Robot Learning*, pages 697–716. PMLR, 2020.

703 D. Wierstra, A. Foerster, J. Peters, and J. Schmidhuber. Solving deep memory POMDPs with
704 recurrent policy gradients. In *International Conference on Artificial Neural Networks*, pages
705 697–706. Springer, 2007.

- 706 R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement
707 learning. *Machine Learning*, 8(3–4):229–256, 1992.
- 708 Z. Yang and H. Nguyen. Recurrent off-policy baselines for memory-based continuous control. *arXiv*
709 *preprint arXiv:2110.12628*, 2021.
- 710 N. Ye, A. Somani, D. Hsu, and W. S. Lee. DESPOT: Online POMDP planning with regularization.
711 *Journal of Artificial Intelligence Research*, 58:231–266, 2017.
- 712 D. Zhang, A. Courville, Y. Bengio, Q. Zheng, A. Zhang, and R. T. Q. Chen. Latent state marginaliza-
713 tion as a low-cost approach for improving exploration. In *International Conference on Learning*
714 *Representations*, 2023.
- 715 M. Zhang, S. Vikram, L. Smith, P. Abbeel, M. Johnson, and S. Levine. Solar: Deep structured
716 representations for model-based reinforcement learning. In *International conference on machine*
717 *learning*, pages 7444–7453, 2019.
- 718 B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement
719 learning. In *Conference on Artificial Intelligence (AAAI)*, 2008.