Learning Interpretable Policies in Hindsight-Observable POMDPs through Partially Supervised Reinforcement Learning

Michael Lanier, Ying Xu, Yevgeniy Vorobeychik, Nathan Jacobs, Chongjie Zhang
Washington University in St. Louis
Department of Computer Science
St. Louis, Missouri, USA
lanier.m@wustl.edu, x.ying1@wustl.edu, yvorobeychik@wustl.edu, jacobsn@wustl.edu, chongjie@wustl.edu

Abstract—**Deep reinforcement learning has demonstrated remarkable achievements across diverse domains such as video games, robotic control, autonomous driving, and drug discovery. Common methodologies in partially observable domains largely lean on end-to-end learning from high-dimensional observations, such as images, without explicitly reasoning about true state. We suggest an alternative direction, introducing the Partially Supervised Reinforcement Learning (PSRL) framework. At the heart of PSRL is the fusion of both supervised and unsupervised learning. The approach leverages a state estimator to distill supervised semantic state information from high-dimensional observations which are often fully observable at training time. This yields more interpretable policies that compose state predictions with control. In parallel, it captures an unsupervised latent representation. These two—the semantic state and the latent state—are then fused and utilized as inputs to a policy network. This juxtaposition offers practitioners a flexible and dynamic spectrum: from emphasizing supervised state information to integrating richer, latent insights. Extensive experimental results indicate that by merging these dual representations, PSRL offers a balance, enhancing interpretability while preserving, and often significantly outperforming, the performance benchmarks set by traditional methods in terms of reward and convergence speed.**

I. Introduction

The realm of deep reinforcement learning has been transformed through a series of advances that utilize deep neural networks for policy or value function approximations. Dominating the current Reinforcement Learning (RL) landscape are methodologies that view the world as a Markov Decision Process (MDP)—treating sequences of observations as states and often learning policies directly from observable data, such as images translating to controls [1]. Such an end-to-end (E2E) approach has indeed set the benchmark in many domains, like robot navigation [2], Atari games [3], human-level driving in Gran Turismo [4], and numerous others.

However, many environments, especially those with high-dimensional sensory inputs, are better modeled as partially observable Markov decision processes (POMDPs). The so-called "true" or "semantic" state in these settings is often a relatively low-dimensional representation which encapsulates semantic information. For example, in autonomous driving, the true state includes critical variables such as the ego-vehicle position, velocity, acceleration, or locations of nearby obstacles. If we are to learn policies that map such semantic information to control, they are naturally more interpretable. However, we do not observe this state directly, but instead have a high-dimensional sensory data stream, such as from camera, LiDAR, and radar. An alternative to typical end-to-end learning that aims to capture some of this intuition has been to learn a low-dimensional latent state representation that is implicit in the raw sensory data [5]. However, such representations are typically not interpretable.

Significantly, what all conventional end-to-end approaches, including those which infer latent state, fail to capture is that in many domains true semantic state information is *available during training, even while it is unobservable at execution time*. For example, this is a common setting in robotics and control research [6]–[14], and true state information is available by construction in any simulation-based learning environment. Several recent efforts have attempted to exploit this structure to improve the efficiency of reinforcement learning in POMDPs. [15] consider a closely related model of *hindsight observability* in which information about semantic state is observed at the end of each episode, and develop model-based reinforcement learning approaches that achieve sample complexity that is linear in the number of actions and logarithmic in the size of the transition and observation function spaces. [16] and [17] term this problem *asymmetric reinforcement learning* and leverage state information by training critics—but not actors—on states rather than observations. However, while [15] offer a general-purpose algorithm, theirs is a custom model-based approach; further, they assume that once the POMDP has been learned, they can quickly solve it to optimality, making their approach impractical in settings with high-dimensional observations, such as images. On the other hand, while both [16] and [17] offer practical algorithms, these are specific to actor-critic reinforcement learning frameworks. Recently, [18] proposed an asymmetric RL algorithm for deep Q-learning. However, this approach is specific to DQN-style algorithms [19], [20]. Thus, asymmetric RL methods are at the moment distinct for Q-learning and actor-critic methods, and no practical unified (general-purpose) method exists for making use of known state information in reinforcement learning for POMDPs with high-dimensional observations in an interpretable way. Moreover, existing asymmetric RL approaches ultimately still train policies that are end-to-end in the sense of mapping raw sensory inputs to controls.When these models err it is impossible to know whether the models have not learned useful representations, or they have not learned good policies with respect to those representations. Consequently, concerns about the lack of interpretability of conventional E2E approaches, which yield complex "black-box" policies, apply to these as well.

To address this gap, we introduce a Partially Supervised Reinforcement Learning (PSRL) framework. In PSRL, we assume that training time trajectories include both observation and true state information, whereas only the former is available at execution time. PSRL has two main building blocks: state predictor g, which maps (finite) observation history to predicted state, and policy π, which maps predicted state to action. At training time, both g and π are jointly trained using a combination of supervised and reinforcement learning loss. Crucially, we train π with *predicted*, rather than actual state as an input, ensuring that the learned policy is robust to prediction errors. These models are interpretable for two reasons. First, it enables designers to develop policies that map actual state to actions independently of raw data, and "plug" these in directly, in addition to the more natural framework of jointly learning predictions and policies as we do in PSRL. This makes the state estimation an "explanation" in the sense defined by [21]. Second, we can now explicitly determine whether the cause of poor behavior is due to poor state predictions or poor state-dependent decisions of designed or learned policies. In other words, state prediction provides a kind of counterfactual information, i.e., we can determine what the correct behavior should have been in a given state, so long as it is predicted well. Notably, this notion critically relies on the state definition being semantically relevant. Our framework can't claim these benefits if the state itself is poorly specified. Fortunately, in many practical applications, there is already an extensive literature devoted to specifying meaningful semantics.

At execution time, actions are chosen by composing π and g, thereby effecting an end-to-end policy that relies only on observations. Nevertheless, this explicit composition preserves a high degree of interpretability, as the policy makes a clear distinction between the prediction of a semantically meaningful state, and actions taken with such predictions as an input. Further, the PSRL framework admits a natural generalization that allows for state to be incompletely observed even at training time, whereby we additionally learn a very low-dimensional representation of the latent part of the state which is concatenated with predicted state as an input to π. We refer to this as PSRL-K, where K is the number of such latent dimensions. This generalization, in turn, induces a space of algorithmic approaches that span from fully end-to-end (ignoring supervised loss in training, with K just the dimension of latent state in end-to-end RL) to PSRL-0, with its added advantage of being interpretable due to the semantic nature of the state predicted by g. Thus, varying K induces a tradeoff between interpretability and ability to capture additional information from sensor data that may be relevant for control. In general as K grows the amount of information the policy can use to make decision increases. This diminishes the counterfactual explainability efficacy as previously mentioned. Other interpreability techniques such as [22] would be required to understand how much of the behavior was dependent on the state estimation vs. the latent features.

We investigate the efficacy of PSRL-K for varying K for both deep Q-learning (DQN) and actor-critic (PPO) approaches that instantiate the policy learning architecture and loss, on six domains in OpenAI Gym. Broadly, we find that PSRL-0 is usually, although not always, more sample efficient than state-of-the-art end-to-end counterparts, and the added value of increasing K is typically low. Moreover, our experiments demonstrate that both pre-training the state predictor g or the policy π—both common ways state information is used in specific applications—tends to perform poorly, in the former case because PSRL makes more efficient use of supervised data, and in the latter case because of fragility of policies learned using true, rather than predicted, state to prediction errors. Finally, our experiments demonstrate interpretability of composite policies learned by PSRL, showing that state prediction errors are typically far lower than unsupervised embeddings learned in conventional end-to-end reinforcement learning paradigms.

II. Related Work

End-to-end deep reinforcement learning has demonstrated remarkable success in settings such as [3], [4], [23]–[25]. However, in numerous other applied settings, such as involving robotics and control applications, it is often natural to decompose end-to-end control into state inference and state-based control [6]–[14], [26], although this decomposition is typically ad hoc and domain specific. For example, one may simply learn to predict state from observational data first, and then uses such predictions to learn a controller. However, such approaches often fail due to prediction errors [16]. The proposed PSRL framework can be viewed as unifying these disparate application-driven approaches, as well as providing a systematic means for studying them. Moreover, the joint supervised and RL training in PSRL overcomes the fragility of many such approaches to prediction errors.

Moreover, several lines of RL research have noted that one commonly observes true state of a POMDP during training. On the more theoretical side, this has been modeled as *hindside-observability* in POMDPs (HOMDPs), that is, true state can be observed after an action has been taken, or after an episode, during training, inference, or both [15], [27]. This work has demonstrated that observability of state (at least) during training can yield theoretical advantages, but did not yield practical algorithms for high-dimensional settings (e.g., most approaches assume a finite set of states). On the more practical side, several recent approaches proposed *asymmetric RL* [17], [18], [28]. The key idea in asymmetric RL is to train a critic that depends directly on state, and an actor that is end-to-end. However, such approaches need to be designed separately for deep Q-learning and actor-critic RL frameworks, and still ultimately rely on learning end-to-end actors. PSRL provides a simple unified framework that is conceptually independent of the particular flavor of model-free RL, making use of supervised information provided by observability of state at training time.

Our work also builds on ideas in current interpretability literature such as [29] and [30]. These works seek to develop model artifacts that would allow a developer or end user to understand agent actions at each step. Moreover, our work is a practical approach to addressing the anticipatory gap detailed in [31]. The key insight detailed is to formally view interpretability in reinforcement learning as a policy satisfaction problem between the agent and some outside observer (presumed to be a human who needs the explanation). We address this by mapping observations directly to a semantic state. Moreover, our approach is connected to the representation learning literature [32]–[34], which seeks to understand the theoretical and practical impacts of the way networks use underlying representations of their input data, and the way these representations are leveraged by downstream tasks. Of particular relevance is [35], which attempts to solve the POMDP problem using VAE constructed representations over observations. While this method can be useful for comparing underlying abstract belief states from actions, it does not capture the intreptability notions detailed here.

Finally, our work is loosely related to model-based reinforcement learning, such as DREAMER, which also uses supervised techniques [36]–[40]. However, these methods rely on learning dynamics and rewards, whereas the propose PSRL framework uses supervision as a means to take advantage of observed state at training time.

III. Preliminaries

Partially Observable MDPs A partially-observable Markov decision process (POMDP) is characterized by a tuple: $\{S, A, T, O, \Omega, r, \gamma, \rho\}$, where S is the state space, A the action space, T the transition function where $T(s'|s,a)$ is the probability distribution over $s' \in S$ given true state s and action a, Ω the set of observations, O the observation function with $O(o|s)$ the distribution of observations $o \in \Omega$ given true state $s \in S$, $r(s,a)$ the reward function, $\gamma \in [0,1)$ the temporal discount factor, and ρ the distribution over initial state s_0. Throughout, we assume that $S \subseteq \mathbb{R}^n$ and $\Omega \subseteq \mathbb{R}^m$, and that typically $m \gg n$, that is, the dimension of the raw sensory observation space is far greater than the dimension of the true state s, which furthermore is semantically meaningful (for example, position, velocity, and acceleration of a car). We allow A to be either finite or infinite and multi-dimensional.

In general, a policy in a POMDP can be a function of an arbitrary observation history. However, in practice it is typical to keep track of only a finite sequence of observations. Since an observation o_t can embed any finite sequence of observations and actions, we aim w.l.o.g. to learn a policy $\pi(o_t)$ that maps an observation o_t to an action $a \in A$ to solve the following problem:

$$\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | a_t = \pi(\cdot)\right],$$

where the expectation is with respect to ρ, as well as the distribution of trajectories induced by π and T.

Deep Reinforcement Learning In relatively general terms, deep reinforcement learning algorithms can be viewed as minimizing a composite loss function comprised of up to three elements: 1) critic loss $\mathcal{L}_c(\theta_c)$, 2) actor loss $\mathcal{L}_a(\theta_a)$, and 3) entropy $\mathcal{H}_e$:

$$\mathcal{L}_{RL}(\theta_c, \theta_a) = \mathbb{E}_{r,s,a,s',a'}[\mathcal{L}_c(\theta_c) + \alpha_1 \mathcal{L}_a(\theta_a) + \alpha_2 \mathcal{H}]. \quad (1)$$

In deep Q-learning, $\alpha_1 = \alpha_2 = 0$, and in the most basic variant of deep Q network (DQN) [41], we learn the parametric action-value function $Q_{\theta_c}(s,a)$ using loss $\mathcal{L}_c(\theta_c) = (\bar{Q} - Q_{\theta_c}(s,a))^2$, where $\bar{Q} = r + \max_{a'} Q_\theta(s,a')$. Variations, such as double DQN (DDQN), involve a distinct Q network as a target in $\bar{Q}$ [42]. In the case of actor-critic methods, one commonly learns a value function $V_{\theta_c}(s)$ as the critic. In particular, in this case it is common to define the advantage function $A(s,a) = (\bar{V} - V_{\theta_c})$, where $\bar{V} = r + V_{\theta_c}(s')$, and the critic loss is just $\mathcal{L}_c(\theta_c) = A(s,a)^2$, while the actor loss is $\mathcal{L}_a(\theta_a) = A(s,a)\log \pi_{\theta_a}$, where π_{θ_a} is the policy (often represented as a neural network), as in the case of approaches as such advantage actor critic (A2C) [43], or

$$\mathcal{L}_a(\theta_a) = -A(s,a)\min\left\{\frac{\log \pi_{\theta_a}}{\log \pi_{\theta_{old}}}, \operatorname{clip}\left(\frac{\log \pi_{\theta_a}}{\log \pi_{\theta_{old}}}, 1-\eta, 1+\eta\right)\right\},$$

as in PPO [44]. Finally, the entropy term $\mathcal{H}$ aims to ensure that π remains stochastic during training to ensure exploration.

IV. Approach

In a broad array of domains, it is reasonable to assume that even though the domain is partially observable, true state is observed *during training*, though not at execution time. For example, any simulation, by construction, must generate both true state and observations, while only emitting observations to mimic whatever domain is being simulated. Moreover, common robotic settings involve highly instrumented training runs (besides training using simulations) in which one can indeed closely approximate whatever actual state information is important. Formally, we assume that training trajectories $\tau = (s_0, o_0, a_0, s_1, o_1, a_1, \ldots, s_t, o_t, a_t, \ldots)$ include *both* observations o_t and states s_t at each time step t.

We propose to leverage knowledge of true state at training time by combining reinforcement learning, which learns how to act, with supervised learning, which learns to predict state s from observations o. Let $g_\phi(o_{t-i}, \ldots o_t)$ denote a parametric model that predicts state given an observation sequence of the last $i+1$ observations. In general, g_ϕ can be either a point prediction or a distribution. Here, we assume it makes a point prediction of a state s given an observation sequence to predict s at time t ie the last observation in the sequence corresponds to the predicted state. Recall that policy $\pi_{\theta_a}(\cdot)$ takes an observation sequence (potentially a single observation) as an input. We refer to this as an *end-to-end* policy. The key idea, which is either implicitly or explicitly common in numerous particular cases [], but has not previously been systematically investigated on its own, is to have a policy with an architecture that explicitly composes state prediction with decision. Formally, let $\tilde{\pi}_{\theta_a}(s)$ be a policy that maps the *true* state s (albeit, unobserved at decision time) to actions. We can then define $\pi(o_{t-i}, ..., o_t) = \tilde{\pi}_{\theta_a}(g_\phi(o_{t-i}, ..., o_t))$.

If both $\tilde{\pi}$ and g are differentiable, we can train the resulting policy end-to-end using the conventional RL loss $\mathcal{L}_{RL}(\theta_c, \theta_a)$ in Equation (1). And, in conventional RL settings, there is little else to do. However, when we observe state s during training, we can do more. Recent asymmetric RL approaches take advantage of such information by learning a critic value or action-value (Q) function as a function of true state, rather than observation, with the actor (policy) [16]–[18]. However, actor policies in these methods are still learned end-to-end. We suggest that this is a missed opportunity for two reasons: first, because it does not take full advantage of the knowledge of true state at training time, and second, because distinct approaches are needed for actor-critic and DQN settings. We propose *partially supervised reinforcement learning (PSRL)* as a general framework for RL that leverages knowledge of true state of a POMDP during training. In its most basic variant, we jointly train $\tilde{\pi}$ and g using a combination of supervised loss (for g) and RL loss (for both):

$$\mathcal{L}_{PSRL}(\theta_c, \theta_a, \phi) = \mathcal{L}_{RL}(\theta_c, \theta_a) + \beta \mathcal{L}_S(\phi), \tag{2}$$

where $\mathcal{L}_S(\phi)$ is the supervised (e.g., cross-entropy) loss and β a parameter that determines the relative weight of the supervised loss to RL loss. Since the policy architecture composes the state-level policy $\tilde{\pi}$ and predictions g, effectively both the actor and the critic in actor-critic method are still trained with respect to observation input o, unlike asymmetric actor-critic, in which the critic takes true state as an input. A corresponding asymmetric variation of PSRL is straightforward. A key advantage of PSRL over asymmetric actor-critic, however, is its simplicity and generality, as it applies equally directly to both Q learning and policy gradient methods.

An important limitation of the PSRL approach is that it assumes decodability of the true state from the observation [45], that is, that each observation is associated with a unique state. To address this, we propose a generalization of PSRL, PSRL-K, in which the policy $\tilde{\pi}$ takes as input (predicted) state s along with K latent variables trained solely using the RL loss. Formally, let $\tilde{\pi}(s, z)$, where z represents a latent low-dimensional observation embedding, and let $z = h_\psi(o)$ be a neural network architecture that captures this embedding. The full policy architecture is then $\pi(\cdot) = \tilde{\pi}_{\theta_a}(g_\phi(o), h_\psi(o))$, still trained using the PSRL loss in Equation (2).

Note that PSRL-K generalizes both PSRL and end-to-end RL. In the former case, we omit the dependence on h_ψ (equivalently, set $K = 0$), while in the latter case, we omit the dependence on g_ϕ (and, therefore, the supervised part of PSRL loss). Consequently, it enables us to modulate between the two extremes, which our experiments show can be beneficial.

Finally, the proposed joint supervised and RL training in PSRL and its generalization is unlike a rather natural idea common in applied settings, where we train a state predictor g and independently a state-conditional policy $\tilde{\pi}$, and compose these at decision time. However, this approach is sensitive to state prediction errors. An improvement is to first pre-train g, and then train $\tilde{\pi} \circ g$ using either RL or PSRL loss. We explore these alternatives in the experiments below.

V. Experiments

We evaluate the performance of PSRL-K on several common benchmark environments from OpenAI Gym, comparing to state of the art end-to-end and asymmetric actor critic approaches, as detailed below. In three environments featuring small action sets, we evaluate the implementation of PSRL-K in deep Q-learning-based approaches (specifically, double deep Q-network (DDQN) [42]). In three others that feature continuous actions, we combine PSRL-K with PPO.

A. Experiment Setup

Environments We consider five OpenAI Gym environments: Acrobot, Cart Pole, Mountain Car, Reacher, and Pendulum [46]. Of these, Acrobot and Mountain Car have finite action sets, Reacher and Pendulum involve continuous actions, and we used two versions of Cart Pole, one with finite and another with continuous action sets. In Acrobot, (true) state is a 6-dimensional vector providing information about the two rotational joints and two links. In Mountain Car, state is represented by a 4-dimensional vector, comprising the cart position, cart velocity, pole angle, and pole angular velocity. In Pendulum state is a 3-dimensional vector, representing the cosine and sine of the pendulum's angle and its angular velocity. In Cart Pole is a 4-dimensional vector, comprising the cart position, cart velocity, pole angle, and pole angular velocity. Finally, in Reacher the state is given by an 11-dimensional vector comprising the cosine and sine values for the angles of both arms, the coordinates of the target, the angular velocities of the arms, and the vector between the target and the reacher's fingertip. In all cases, observations o are short sequences of 2D images simulated by OpenAI Gym.
Baselines Our first *true state* baseline is for calibration, and assumes that the environment is fully observable. This baseline learns $\pi(s)$ using either DDQN (in finite-action settings) or PPO (in continuous-action settings). The second baseline is end-to-end (*E2E*), for which we learn policies $\pi(o)$ directly mapping image inputs to control using either DDQN or PPO as conventionally done. For each environment, we use the best performing E2E baseline available. To ensure a fair comparison with PSRL-K, we additionally provide results for E2E-$K + n$, where the architecture of $\pi(o)$ mirrors that of PSRL-K, except the policy is learned using solely the RL loss. Finally, we compare with a state-of-the-art asymmetric

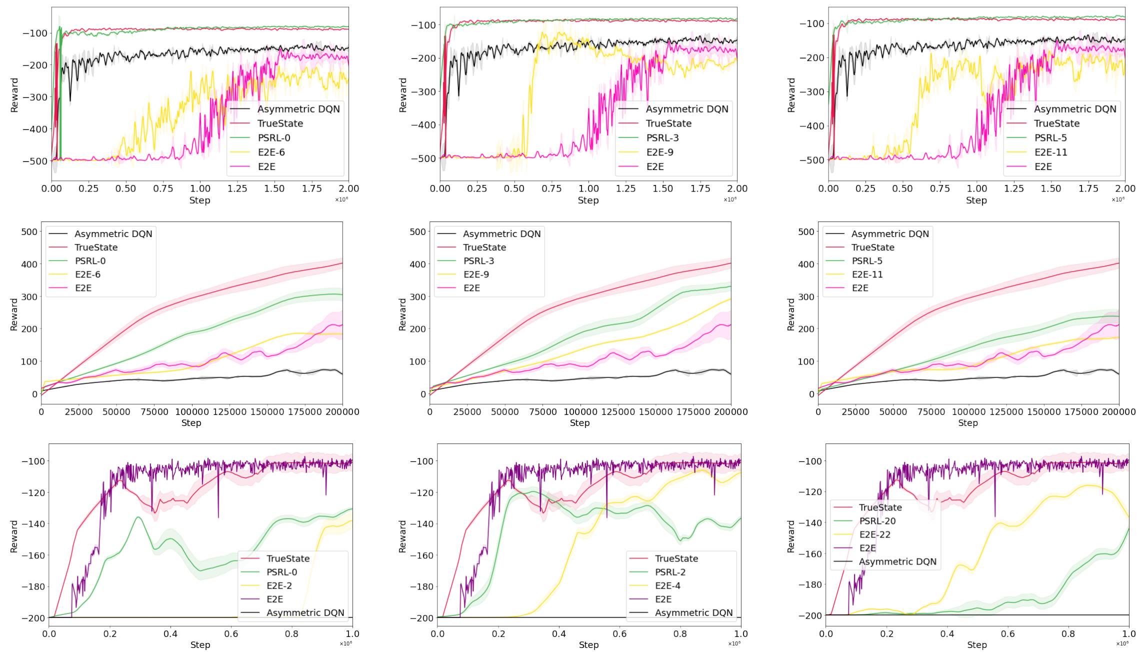

Fig. 1. Experiments in Acrobot (top row), Cart Pole (with finite action sets; middle row), and Mountain Car (bottom row), in finite-action environments using DDQN approaches.

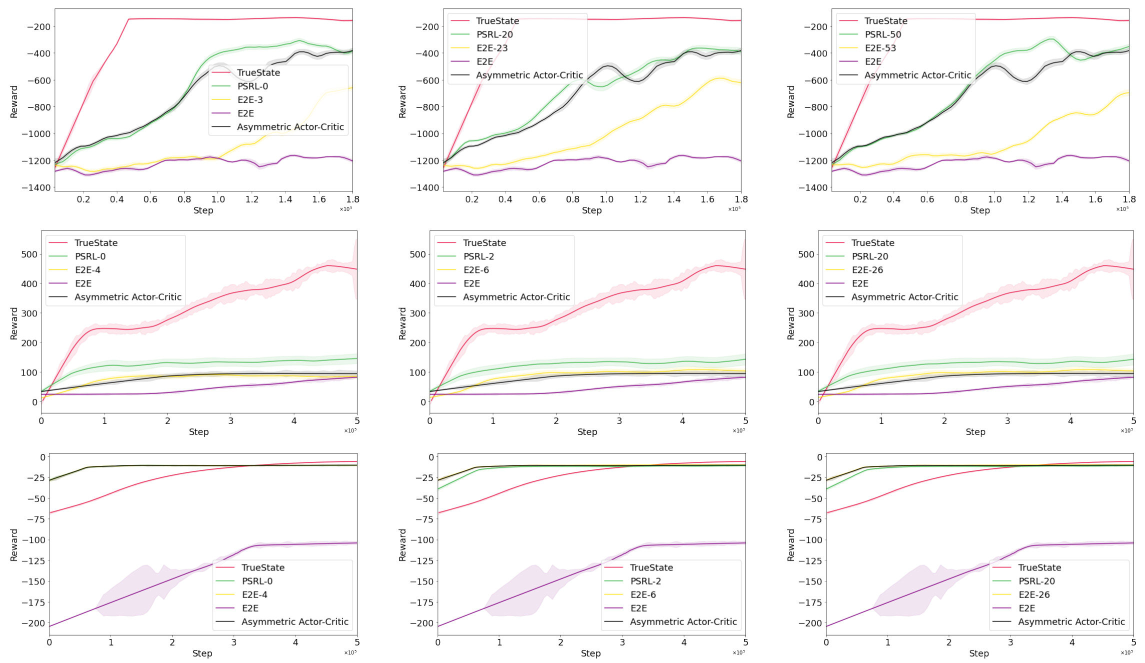

Fig. 2. Experiments in Pendulum (top row), Cart Pole (with continuous action sets; middle row), and Reacher (bottom row), in continuous-action environments using PPO approaches.

DQN (*ADQN*) [18] in the finite-action settings and asymmetric PPO (*APPO*) [17] in the continuous-action settings. We use an identical architecture between APPO and PSRL to ensure a fair comparison.

B. Results

Q-Learning Settings Our first set of experiments considers the DDQN framework, comparing PSRLwith the baselines. The results are presented in Figure 1. Consider first the top row, corresponding to Acrobot. In this setting, PSRL is essentially as effective as learning in its fully observable counterpart. In contrast, both variants of end-to-end learning considerably slower, even as they ultimately approach a near-optimal policy. Finally, we observe that ADQN significantly outperforms E2E approaches, but remains clearly below PSRL

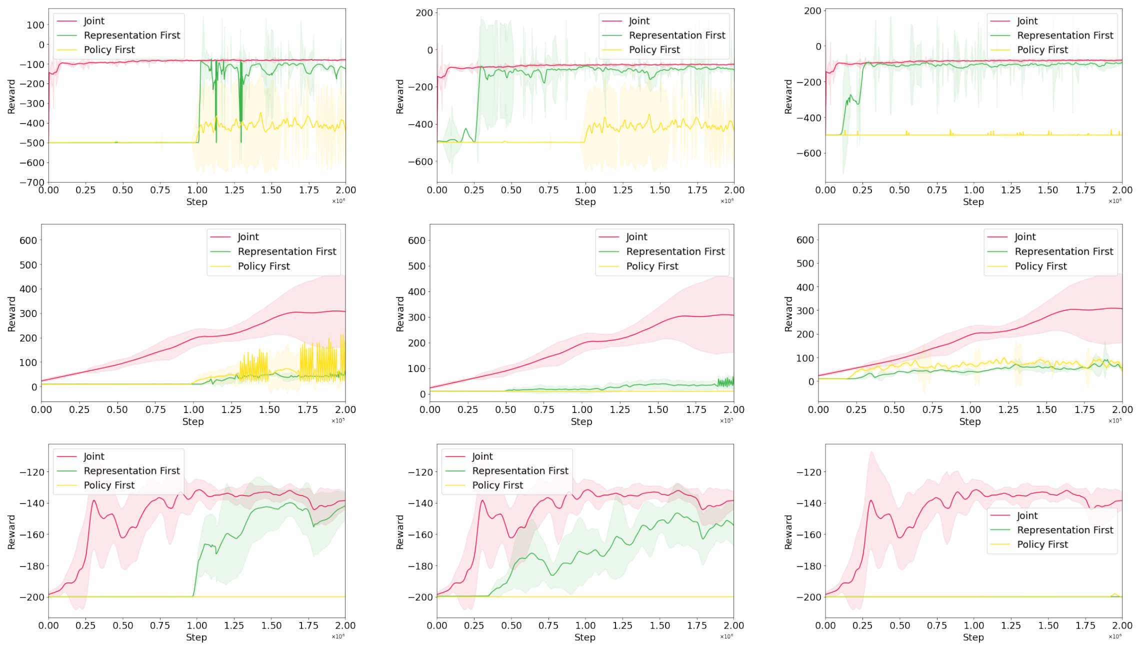

Fig. 3. Experiments comparing PSRL-0 (joint RL and supervised) learning with either representation first or policy first approaches. Top row, left-to-right: Acrobot 50% pretrained, Acrobot 25% pretrained, Acrobot 10% pretrained. Middle row: Cartpole 50% pretrained, Cartpole 25% pretrained, Cartpole 10% pretrained. Bottom row: Mountain Car 50% pretrained, Mountain Car 25% pretrained, Mountain Car 10% pretrained.

in terms of performance.

The second row of Figure 1 presents DDQN results for the finite-action Cart Pole environment. Here, PSRL again exhibits a clear advantage over end-to-end approaches as well as asymmetric DQN, although in this domain there is a clear advantage in the knowledge of true state. Notably, in both the Acrobot and Cart Pole domain, there appears to be little advantage to PSRL-K for $K > 0$, with PSRL-0 already exhibiting strong performance.

Finally, the last row of Figure 1 presents the results for Mountain Car. In this domain, we find that end-to-end performs extremely well, essentially no different from using the true state. PSRL-0 is tangibly worse, but in this case PSRL-2 performs better.

Actor-Critic Settings Next, we consider settings with continuous actions, combining PSRLand baselines with the PPO actor-critic approach. The results are provided in Figure 2. In the Pendulum environment (top row), both asymmetric actor-critic and PSRL outperform end-to-end approaches by a large margin, although learn tangibly slower than true-state PPO. Moreover, PSRL-0 learns somewhat faster than asymmetric actor-critic. In the Cart Pole environment (bottom row), the advantage of PSRL and asymmetric methods over end-to-end is smaller. Indeed, in this case, asymmetric actor critic no longer outperforms one of the end-to-end baselines. PSRL, however, maintains an advantage over both. In both of these domains, PSRL-0 performance is comparable to PSRL-K for several values of K; that is, no added value is provided by including latent features. Finally, in the Reacher environment, PSRL and APPO are both comparable and actually learn somewhat faster than the approach which knows the true state (perhaps because of the implicit increase in exploration that results from imperfect state predictions), whereas end-to-end approaches are considerably worse.

Ablation Experiments Now, we consider two natural ablations of PSRL. The first ablation involves pretraining state prediction (representation network) g_ϕ using only supervised loss for the first T iterations. Thereafter, we freeze the representation network g_ϕ, and train only the policy network $\tilde{\pi}(g_\phi(o))$ using conventional RL loss. This roughly mirrors the typical way that state information in reinforcement learning is used in applied settings, such as robotic control [6], [10], [12], [14]. We refer to this as *representation-first PSRL*. The second ablation first pretrains a policy mapping true state to actions for the first T iterations using RL loss. Thereafter, we freeze the policy network $\tilde{\pi}$, and train the state prediction g_ϕ using supervised loss. We refer to this as *policy-first PSRL*. This corresponds to an effective decomposition of prediction and policy training, also common in applied settings.

The results are provided in Figure 3 for discrete action environments and Figure 4 for continuous action environments. We can observe that neither the representation first nor the policy first baselines are as effective as PSRLin the majority of cases. However, there are a few interesting exceptions. While the policy first baseline is nearly always poor, representation first is effective in Continuous Cart Pole and Reacher, indistinguishable from PSRLin the latter, and actually outperforming it under one configuration in the former. Interestingly, these exceptions are only in the PPO setting, whereas the joint representation and policy training in PSRLappears uniformly more advantageous than these variations in the DQN setting.

Wall Time Experiments We find that our method is com-

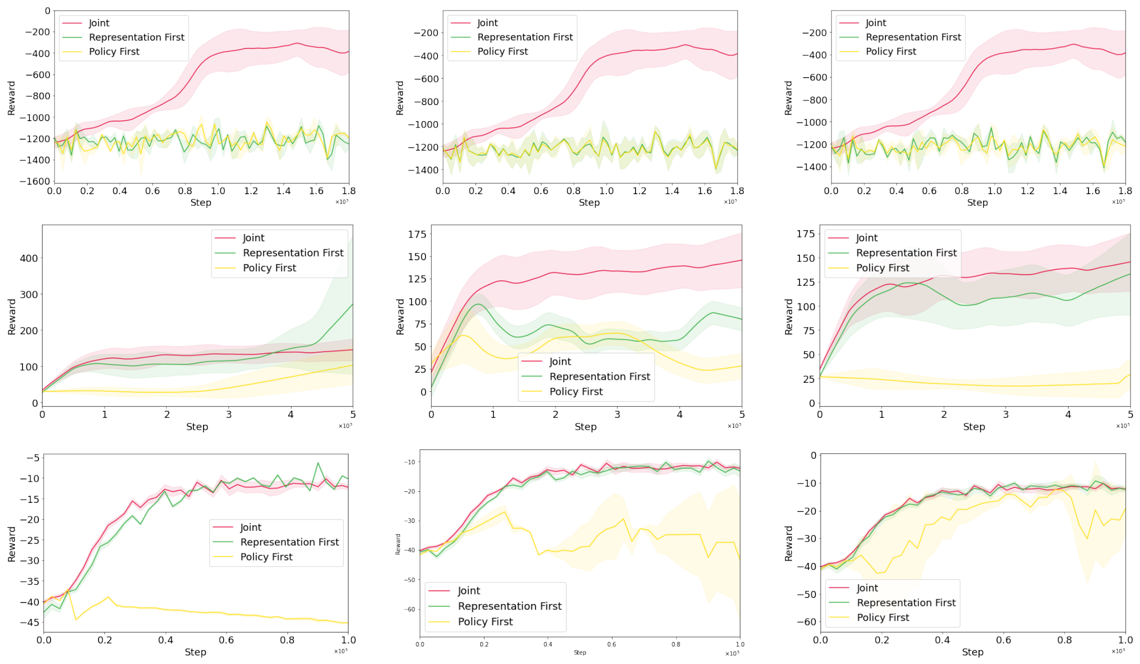
Fig. 4. Experiments comparing PSRL-0 (joint RL and supervised) learning with either representation first or policy first approaches. Top row, left-to-right: Pendulum 50% pretrained, Pendulum 25% pretrained, Pendulum 10% pretrained. Middle row: Continuous Cartpole 50% pretrained, Continuous Cartpole 25% pretrained, Continuous Cartpole 10% pretrained. Bottom row: Reacher 50% pretrained, Reacher 25% pretrained, Reacher 10% pretrained.

parable or considerably faster than E2E baselines. Below are training times for continuous domains are over 10000 time steps and discrete action over 100000 time steps.

TABLE I
COMPARISON OF TRUESTATE, E2E, AND PSRL TIMINGS FOR DIFFERENT ENVIRONMENTS.

Truestate (s)	**E2E (s)**	**PSRL (s)**	**Env**
5491 ± 191	85343 ± 851	44023 ± 6069	Cartpole
3332 ± 69	70643 ± 8721	55721 ± 1153	Acrobot
8412 ± 19	123892 ± 36304	45008 ± 1997	Mountain Car
67 ± 2	1052 ± 43	2930 ± 54	Continuous Cartpole
69 ± 0.1	1292 ± 3	1292 ± 3	Pendulum
66 ± 0.1	12985 ± 179	13528 ± 87	Reacher

TABLE II
COMPARISON OF PSRL-0 AND E2E TEST MEAN SQUARED ERROR FOR DIFFERENT ENVIRONMENTS.

Environment	**PSRL-0**	**E2E**
Acrobot	0.78 ± 0.27	16260 ± 7435
Mountain Car	0.0003 ± 0.0001	337 ± 37
Cartpole	0.20 ± 0.005	56 ± 5.6
Reacher	0.30 ± 0.13	0.88 ± 0.13
Cont Cartpole	0.05 ± 0.001	805 ± 146
Pendulum	1.04 ± 0.18	134 ± 21

Note that although E2E is able to learn, it does so with embeddings that are not interpretable as state estimates.

Interpretability Finally, we consider the issue of interpretability, which we quantify simply as the quality of true (semantic) state estimation, comparing PSRL (that is, the predictions by g) with E2E with respect to the embedding of the identical dimension. One may expect that since E2E approaches typically learn to be quite effective, they do so by learning a good estimate of true state despite not having any explicit supervision. Our results in Table II dispel this, showing that while PSRL yields highly accurate state predictions, the latent embeddings in E2E are entirely uninterpretable.

VI. Conclusion

The PSRL framework reveals its effectiveness as a compelling alternative for addressing the challenges posed by important classes of partially observable domains, striking a balance between performance and interpretability. PSRL provides interpretability in the inferred state while outperforming state of the art methods.

VII. Acknowledgements

We thank Ce Ma for his contributions to the code base for this work. This work was partially supported by the NSF (IIS-1905558, IIS-1903207, IIS-2214141, CNS-2310470) and ONR (N000142412663).

References

[1] I. Carlucho, M. De Paula, S. Wang, Y. Petillot, and G. G. Acosta, "Adaptive low-level control of autonomous underwater vehicles using deep reinforcement learning," *Robotics and Autonomous Systems*, vol. 107, pp. 71–86, 2018.

[2] H. Shi, L. Shi, M. Xu, and K.-S. Hwang, "End-to-end navigation strategy with deep reinforcement learning for mobile robots," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2393–2402, 2019.

[3] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[4] P. R. Wurman, S. Barrett, K. Kawamoto, J. MacGlashan, K. Subramanian, T. J. Walsh, R. Capobianco, A. Devlic, F. Eckert, F. Fuchs *et al.*, "Outracing champion gran turismo drivers with deep reinforcement learning," *Nature*, vol. 602, no. 7896, pp. 223–228, 2022.

[5] J. Guo, Z. Li, H. Wang, M. Wang, Z. Yang, and X. Zhang, "Provably efficient representation learning with tractable planning in low-rank pomdp," in *International Conference on Machine Learning*, 2023.

[6] S. Bansal, V. Tolani, S. Gupta, J. Malik, and C. Tomlin, "Combining optimal control and learning for visual navigation in novel environments," in *Conference on Robot Learning*, 2020, pp. 420–429.

[7] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, "Probabilistic data association for semantic slam," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 1722–1729.

[8] S. Dean, N. Matni, B. Recht, and V. Ye, "Robust guarantees for perception-based control," in *Learning for Dynamics and Control*, 2020, pp. 350–360.

[9] C. Dong, J. M. Dolan, and B. Litkouhi, "Intention estimation for ramp merging control in autonomous driving," in *2017 IEEE intelligent vehicles symposium (IV)*, 2017, pp. 1584–1589.

[10] A. R. Godbole and K. Subbarao, "Nonlinear control of unmanned aerial vehicles with cable suspended payloads," *Aerospace Science and Technology*, vol. 93, p. 105299, 2019.

[11] P. Kumar, M. Perrollaz, S. Lefevre, and C. Laugier, "Learning-based approach for online lane change intention prediction," in *2013 IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 797–802.

[12] J. Morton and M. J. Kochenderfer, "Simultaneous policy learning and latent state inference for imitating driver behavior," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017, pp. 1–6.

[13] R. Rafailov, T. Yu, A. Rajeswaran, and C. Finn, "Offline reinforcement learning from images with latent space models," in *Learning for Dynamics and Control*, 2021, pp. 1154–1168.

[14] S. Tang, V. Wüest, and V. Kumar, "Aggressive flight with suspended payloads using vision-based control," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1152–1159, 2018.

[15] J. Lee, A. Agarwal, C. Dann, and T. Zhang, "Learning in pomdps is sample-efficient with hindsight observability," in *International Conference on Machine Learning*, 2023, pp. 18 733–18 773.

[16] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," in *Robotics: Science and Systems*, 2018.

[17] A. Baisero and C. Amato, "Unbiased asymmetric reinforcement learning under partial observability," in *International Conference on Autonomous Agents and Multiagent Systems*, 2022.

[18] A. Baisero, B. Daley, and C. Amato, "Asymmetric DQN for partially observable reinforcement learning," in *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, ser. Proceedings of Machine Learning Research, J. Cussens and K. Zhang, Eds., vol. 180. PMLR, 01–05 Aug 2022, pp. 107–117. [Online]. Available: https://proceedings.mlr.press/v180/baisero22a.html

[19] J. Farebrother, M. C. Machado, and M. Bowling, "Generalization and regularization in dqn," 2020. [Online]. Available: https://arxiv.org/abs/1810.00123

[20] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1995–2003.

[21] S. Milani, N. Topin, M. Veloso, and F. Fang, "Explainable reinforcement learning: A survey and comparative review," *ACM Comput. Surv.*, vol. 56, no. 7, apr 2024. [Online]. Available: https://doi.org/10.1145/3616864

[22] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, no. 2, p. 336–359, Oct. 2019. [Online]. Available: http://dx.doi.org/10.1007/s11263-019-01228-7

[23] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[24] Y. Li, "Deep reinforcement learning: An overview," 2018. [Online]. Available: https://arxiv.org/abs/1701.07274

[25] S. Wang, D. Jia, and X. Weng, "Deep reinforcement learning for autonomous driving," 2019, unpublished. [Online]. Available: https://arxiv.org/abs/1811.11329

[26] X. Ma, J. Li, M. J. Kochenderfer, D. Isele, and K. Fujimura, "Reinforcement learning for autonomous driving with latent state inference and spatial-temporal relationships," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 6064–6071.

[27] J. Guo, M. Chen, H. Wang, C. Xiong, M. Wang, and Y. Bai, "Sample-efficient learning of pomdps with multiple observations in hindsight," 2023.

[28] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," 2017.

[29] X. Peng, M. O. Riedl, and P. Ammanabrolu, "Inherently explainable reinforcement learning in natural language," 2022.

[30] R. Luss, A. Dhurandhar, and M. Liu, "Local explanations for reinforcement learning," 2022.

[31] M. Finkelstein, L. Liu, N. L. Schlot, Y. Kolumbus, D. C. Parkes, J. S. Rosenshein, and S. Keren, "Explainable reinforcement learning via model transforms," 2022.

[32] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[33] Q. He, H. Su, J. Zhang, and X. Hou, "Frustratingly easy regularization on representation can boost deep reinforcement learning," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jun 2023, pp. 20 215–20 225. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CVPR52729.2023.01936

[34] C. Le Lan, S. Tu, M. Rowland, A. Harutyunyan, R. Agarwal, M. G. Bellemare, and W. Dabney, "Bootstrapped representations in reinforcement learning," in *Proceedings of the 40th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., vol. 202. PMLR, 23–29 Jul 2023, pp. 18 686–18 713. [Online]. Available: https://proceedings.mlr.press/v202/le-lan23a.html

[35] A. Wang, A. C. Li, T. Q. Klassen, R. T. Icarte, and S. A. Mcilraith, "Learning belief representations for partially observable deep RL," in *Proceedings of the 40th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., vol. 202. PMLR, 23–29 Jul 2023, pp. 35 970–35 988. [Online]. Available: https://proceedings.mlr.press/v202/wang23p.html

[36] C. Jin, Z. Yang, Z. Wang, and M. I. Jordan, "Provably efficient reinforcement learning with linear function approximation," 2019.

[37] C. Jin, Q. Liu, and S. Miryoosefi, "Bellman eluder dimension: New rich classes of rl problems, and sample-efficient algorithms," *Advances in Neural Information Processing Systems*, 2021.

[38] M. Okada and T. Taniguchi, "Dreaming: Model-based reinforcement learning by latent imagination without reconstruction," in *2021 ieee international conference on robotics and automation (icra)*, 2021, pp. 4209–4215.

[39] P. Wu, A. Escontrela, D. Hafner, P. Abbeel, and K. Goldberg, "Daydreamer: World models for physical robot learning," in *Conference on Robot Learning*, 2023, pp. 2226–2240.

[40] L. Yang and M. Wang, "Sample-optimal parametric q-learning using linearly additive features," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6995–7004.

[41] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[42] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," 2015.

[43] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," 2016.

[44] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.

[45] Z. Mhammedi, D. J. Foster, M. Simchowitz, D. Misra, W. Sun, A. Krishnamurthy, A. Rakhlin, and J. Langford, "Learning the linear quadratic regulator from nonlinear observations," *34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada*, 2020.

[46] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016. [Online]. Available: https://arxiv.org/abs/1606.01540