PAC Reinforcement Learning in Noisy Continuous Worlds

Emma Brunskill* **Bethany R. Leffler†** **Lihong Li†** **Michael L. Littman†** **Nicholas Roy***
* Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02143
† Department of Computer Science
Rutgers University
Piscataway, NJ 08854

Continuous state spaces and stochastic, switching dynamics characterize a number of rich, real-world domains, such as robot navigation across varying terrain. We describe a reinforcement-learning algorithm for learning in such domains. We prove that this algorithm is provably approximately correct for certain environments. Unfortunately, no optimal planning techniques exist in general for such problems; we instead use fitted value iteration to solve the learned MDP, and extend the error bounds from prior work on policy performance to include the error due to approximate planning. Finally, we provide a robotic car experiment over varying terrain to demonstrate that these dynamics representations adequately capture real world dynamics and that our algorithm can be used to efficiently solve such problems.

1 INTRODUCTION

Reinforcement learning (RL) has had some impressive successes, such as model helicopter flying (Ng 2003) and expert software backgammon players (Tesauro 1994). Two of the key challenges in reinforcement learning are scaling to large worlds, which generally involves a form of generalization, and efficiently handling the exploration/exploitation tradeoff. Many real-world problems involve real-valued state variables: discretizing such environments causes an exponential growth in the number of states as the state dimensionality increases, and so solutions that directly reason with continuous-states are of important consideration.

In this paper, we build on recent work on provably efficient reinforcement learning (Brafman & Tennenholtz, 2002; Strehl et al., 2006; Keans & Singh, 2002) and focus on continuous-state discrete action environments. We focus on the case when the dynamics can be described as switching noisy offsets where the parameters of the dynamics depend on the state's "type" t and the action taken a. More formally,

$$s' = s + \beta_{at} + \varepsilon_{at} \quad (1)$$

where s is the current state, s' is the next state, $\varepsilon_{at} \sim \mathcal{N}(0, \Sigma_{at})$ is drawn from a Gaussian with covariance Σ_{at} and β_{at} is the offset.

An example where we expect such dynamics to arise is during autonomous traversal of varying terrain. Here, types represent the ground surface, such as dirt or rocks. The dynamics of the car may be approximated by an offset from the prior state plus some noise, where the offset and noise depend on the surface underneath the car. These models could be useful approximations in a number of other problems, including transportation planning (learning the mean speed and variance of interstate highways and local streets and using this for path planning to a goal location), and packet routing (learning that wireless and ethernet have different usage patterns and bandwidth and routing accordingly).

We present a new RL algorithm for learning in continuous-state, discrete action Markov decision process (MDP) environments with switching noisy offset dynamics and show that this algorithm is provably approximately correct (PAC) in certain environments. We perform planning using fitted value iteration (FVI) and incorporate the error due to approximate planning into our bounds.

Finally, we present experiments on a small robot task that involves navigation over varying terrains. These experiments demonstrate that these dynamics models can accurately capture real-world dynamics and our algorithm can quickly learn good policies in such environments.

2 CONTINUOUS-STATE TYPED Rmax

This section introduces terminology and then presents our algorithm.

2.1 BACKGROUND

The world is characterized by a continuous-state discounted MDP $M = \langle S, A, p(s'|s,a), R, \gamma \rangle$ where $S = \mathbb{R}^{N_{dim}}$ is the N_{dim}-dimensional state space, A is a set of discrete actions, $p(s'|s,a)$ is the unknown transition dynamics that satisfy the parametric form of Equation 1, $\gamma \in [0,1)$ and $R : S \times A \to \mathbb{R}$ is the known reward model, which is bounded by 1. In addition to the standard MDP formulation, each state s is associated with a unique observable type $t \in T$ and define $N_T = |T|$. The dynamics of the agent are determined by the current state type t and action a taken:

$$p(s'|s,a) = p(s'|s,t_s,a) = \mathcal{N}(s'|s + \beta_{at}, \Sigma_{at}). \quad (2)$$

In this work, we focus on the known reward, unknown dynamics model situation. The parameters of the dynamics model β_{at}, Σ_{at} are assumed to be unknown for all types t and actions a at the start of learning. This model is a departure from prior related work (Abbeel & Ng, 2005; Strehl & M.Littman, 2008), which focuses on a more general linear dynamics model but assumes a single type and that the variance of the noise Σ_{at} is known. We argue there exist interesting problems where the variance of the noise is unknown and estimating this noise may provide the key distinction between the dynamics models of different types.

In the reinforcement learning, the agent must learn to select an action a based on its current state s. at each time step, it receives an immediate reward r also based on its current state[1]. The agent then moves to a next state s' according to the dynamics model. The goal is to learn a policy $\pi : S \to A$ that allows the agent to choose actions. The value of a particular policy is the expected discounted sum of future rewards that will be received from following this policy, and is denoted $V^\pi(s) = E_\pi[\sum_{j=0}^{\infty} \gamma^j r_j | s_0 = s]$, where r_j is the reward received on the j-th time step and s_0 is the initial state of the agent. Let π^* be the optimal policy, and its associated value function be $V^*(s)$.

2.2 ALGORITHM

Our algorithm draws from the R-max algorithm of Brafman and Tennenholtz (2002). We first form a set of $\langle t, a \rangle$ tuples, one for each type-action pair. Note that each tuple corresponds to a particular pair of dynamics model parameters, $\langle \beta_{at}, \Sigma_{at} \rangle$. A tuple is considered to be "known" if if the agent has been in type t and taken action a N_{TA} times. At each timestep, we construct a new MDP M_K as follows. If a tuple has been experienced N_{TA} or more times, then we estimate the parameters for this dynamics model using maximum-likelihood estimation:

$$\tilde{\beta}_{at} = \frac{\sum_{i=1}^{T_1} (s'_i - s_{at,i})}{T_1}$$

$$\tilde{\Sigma}_{at} = \frac{\sum_{i=1}^{T-1} (s'_i - s_{at,i})'(s'_i - s_{at,i})}{T_1 - 1}.$$

Otherwise, we set the dynamics model for this type-action tuple to be a transition with probability 1 back to the same state. We also modify the reward function for all unknown state-action tuples $\langle t_u, a_u \rangle$ so that all state-action values $Q(s_{t_u}, a_u)$ have a reward of $V_{\max}$ (the maximum value possible, $1/(1-\gamma)$). We then seek to solve M_K. This MDP involves switching dynamics with continuous states, and (to the authors' knowledge) there exist no exact optimal planners for such MDPs. Instead, we will use fitted value iteration to approximately solve the MDP.

In FVI, the value function is estimated explicitly at only a fixed number of points that are (for example) uniformly spaced over the state space. Planning requires performing Bellman backups for each grid point f_i. Since we are only performing backups of the value function at a set of grid points f_i we need some function approximator to estimate the value of other points that are not in this fixed set. We can use Gaussian kernel functions to interpolate the value at the grid points to other points. The value of a state s that is not a

Algorithm 1 Noisy Offset Continuous-State RL

1: **Input:** N_A,N_{dim},N_T, R, $\Sigma_{\max}$, $\Sigma_{\min}$, γ, ϵ, and δ.
2: Set all type-action tuples $\langle t, a \rangle$ to be unknown and initialize the dynamics models (see text) to create a known MDP model M_K.
3: Select a set of fixed evenly spaced points to use for fitted value iteration.
4: Start in a state s_0.
5: **loop**
6: Solve MDP M_K using fitted value iteration with Gaussian kernels spaced as above.
7: Select action $a = \arg\max_a Q_{M_K}(s,a)$.
8: Transition to the next state s'.
9: Increment the appropriate $n_{t,a}$ count (where t is the type of state s) given the observed transition tuple $\langle s, a, s' \rangle$.
10: If $n_{t,a}$ exceeds N_{TA} where N_{TA} is specified according to the analysis, then mark $\langle t, a \rangle$ as "known" and estimate the dynamics model parameters for this tuple.
11: **end loop**

[1]For simplicity, the reward is assumed to be only a function of state in this paper, but the arguments can be easily extended to where the reward model is also a function of the action chosen.

fixed point is

$$V(s) = \sum_{d=1}^{F} \mathcal{N}(s; f_d, \Sigma_d) Q(f_d, a) \quad (3)$$

where $\mathcal{N}(s; f_d, \Sigma_d)$ represents a Gaussian with mean at grid point f_d and variance Σ_d evaluated at state s. The grid points and variances (f_d, Σ_d) are defined so

$$\sum_{d=1}^{F} w_d \mathcal{N}(s; f_d, \Sigma_d) \approx 1 \quad (4)$$

for all states s of interest. We would like this expression to exactly equal 1 for all states of interest as that guarantees the function approximator is an averager and therefore discounted infinite horizon fitted value iteration is guaranteed to converge (Gordon, 1995). In practice if Gaussians are placed at uniform intervals over the state space of interest then this expression can be extremely close to 1. Indeed, as long as the sum in equation 4 sums to less than or equal to 1 for all states then the approximator operator is guaranteed to be a non-expansion in the max norm and therefore discounted infinite horizon approximate value function is still guaranteed to converge.

Substituting this representation of the value function in place of $V(s')$ and using the dynamics model in the Bellman backup equation, we can perform the integration over future reward in closed form to get

$$\begin{aligned} V(f_i) &= R(f_i) + \gamma \max_a \sum_d w_d \cdot \\ &\quad \mathcal{N}(f_d; f_i + \beta_{at_i}, \Sigma_{at_i} + \Sigma_d) V(f_d). \end{aligned}$$

For a given set of fixed states f_d, the majority of the right side can be computed once and used repeatedly during value iteration; essentially the continuous-state MDP is converted to a new discrete-state MDP where the states are the fixed points.

At each timestep, the agent chooses the action that maximizes the estimate of its current value according to M_K: $a = \arg\max Q_{M_K}(s, a)$. Our complete algorithm is shown in Algorithm 1.

3 LEARNING COMPLEXITY

In Section 4, we will analyze an instance of our learning algorithm and prove it is PAC-MDP (provably approximately correct in Markov decision processes).

When analyzing the performance of a learning algorithm, there are many potential criteria to use. In our work, we will focus predominantly on sample complexity with a brief mention of computational complexity. Computational complexity refers to the number of operations executed by the algorithm for each step taken by the agent in the environment. We will follow Kakade (2003) and use *sample complexity* as shorthand for the *sample complexity of learning*, which Kakade defined as the number of timesteps at which the algorithm's policy at the current state is not ϵ-optimal; that is, $Q^*(s, a) - Q^{\pi_t}(s, a) > \epsilon$ where π_t is the policy of the algorithm at time t. Strehl et al. (2006) defined an algorithm as PAC-MDP if, for a given ϵ and δ, with probability at least $1 - \delta$, the algorithm's sample complexity is less than a polynomial function of the problem's parameters$(|S|, |A|, 1/\epsilon, 1/\delta, 1/(1-\gamma))$. Note that this definition only requires the algorithm to learn and execute a near optimal policy with high probability. As the agent acts in the world, it may be unlucky and experience a series of state transitions that poorly reflect the true dynamics, due to noise.

Strehl et al. (2006) described and proved the conditions for an algorithm to be PAC-MDP. First, they defined an algorithm to be greedy if it chooses its action on timestep t to be the one that maximizes the value of the current state s_t ($a_t = \arg\max_{a \in A} Q_t(s_t, a)$). Their main result goes as follows: let $\mathcal{A}(\epsilon, \delta)$ denote a greedy learning algorithm. Maintain a list K_t of "known" state-action pairs. At each new timestep t, this list stays the same unless during that timestep a new state-action pair becomes known. MDP M_{K_t} is a constructed "known" state-action MDP (where the construction is very similar to what we described earlier) and π_t is the greedy policy of M_{K_t}. Assume that ϵ and δ are given and the following 3 conditions hold for all states, actions and timesteps:

1. $Q^*(s, a) - Q_t(s, a) \leq \epsilon$.
2. $V_t(s) - V^{\pi_t}_{M_{K_t}} \leq \epsilon$.
3. The total number of times the agent visits a state-action tuple that is not in K_t is bounded by $\zeta(\epsilon, \delta)$ (the *learning complexity*).

Then, on any MDP M, $\mathcal{A}(\epsilon, \delta)$ will follow a 4ϵ-optimal policy from its initial state on all but N_{total} timesteps with probability at least $1 - 2\delta$ where where N_{total} is a polynomial in the problems' parameters $(\zeta(\epsilon, \delta), \frac{1}{\delta}, \frac{1}{\epsilon}, \frac{1}{1-\gamma})$.

The majority of our analysis will focus on showing that our algorithm fulfills these three criteria. We will briefly discuss some intuition for these criteria and how we will proceed in proving our algorithm satisfies them. Together, the first and second criteria can be interpreted as saying that the algorithm should produce accurate value estimates of the known MDP, and that it should be optimistic about the values of all state-action pairs. The first criterion is the more challenging to demonstrate. In order to show our estimates of known state-action pairs are close to their

real values, we must consider two potential sources of error that could prevent it. The first is that the model dynamics are only estimated from the samples experienced, and so the dynamics model estimates may be far from the true dynamics. In Proposition 4.1 and Lemmas 4.2, 4.3, and 4.4, we bound the number of samples necessary to ensure the dynamics model parameter estimates are close to the true dynamics. The second source of error comes solving the MDP. We cannot currently perform exact optimal planning for these continuous-state noisy offset MDPs, and therefore we do approximate planning. In Section 4.2, we bound the error due to this approximate planning. We then combine these results in Lemma 4.5 to bound the error between our estimate of the value of the known state MDP and the true optimal values. Theorem 4.6 uses this result to prove the algorithm is PAC-MDP.

Note that our use of an approximate planner is a departure from most related work on PAC RL. Existing work typically assumes the existence of a planning oracle for choosing actions given the estimated model.

To ensure fitted value iteration produces highly accurate results, our algorithm's worst-case computational complexity is exponential in the number of state dimensions. While this fact prevents it from satisfying the conditions to be efficient PAC-MDP (Strehl et al., 2006), our experimental results demonstrate our algorithm performs well compared to related approaches in a real-world robot problem.

4 ANALYSIS

This section provides a formal analysis of the algorithm. For simplicity, it assumes a diagonal variance matrix for the noise model: $\Sigma = \text{diag}(\sigma_1^2, \sigma_2^2, \cdots, \sigma_{N_{dim}}^2)$. We believe the algorithm is correct in the non-diagonal case as well, but the analysis is more involved. We also assume that the absolute values of the components in β_{at} and Σ_{at} are upper bounded by B_β and B_σ, respectively, for some known constants B_β and B_σ. This assumption is often true in practice.

4.1 MODEL ACCURACY

We first establish the distance between two dynamics models with different parameters. Following Abbeel and Ng (2005), we use the variational distance

$$d_{var}(P(x), Q(x)) = \frac{1}{2}\int_{\mathcal{X}} |P(x) - Q(x)| dx. \qquad (5)$$

Proposition 4.1 *Assume that both Σ_1 and Σ_2 are diagonal matrices and let $\sigma_{\min}$ be the minimum variance along any of the dimensions. Then,*

$$d_{var}(\mathcal{N}(s'|\beta_1 + s, \Sigma_1), \mathcal{N}(s'|\beta_2 + s, \Sigma_2)) \leq 1 - \left(\prod_{i=1}^{N_{dim}} \frac{\min[\sigma_{1i}^2, \sigma_{2i}^2]}{\sigma_{2i}^2}\right)^{0.5} + \frac{||\beta_2 - \beta_1||_2}{\sqrt{(2\pi)}\sigma_{\min}}$$

where σ_{ki}^2 is the variance of the k-th Gaussian along dimension i.

Proof Assume w.l.o.g. that $|\Sigma_1| \leq |\Sigma_2|$. Then,

$$\begin{aligned} & d_{var}(\mathcal{N}(s'|\beta_1 + s, \Sigma_1), \mathcal{N}(s'|\beta_2 + s, \Sigma_2)) \\ &= \frac{1}{2}\int_{s'} |\mathcal{N}(s'|\beta_1 + s, \Sigma_1) - \mathcal{N}(s'|\beta_2 + s, \Sigma_2)| ds' \\ &= \frac{1}{2}\int_{s'} |\mathcal{N}(s'|\beta_1 + s, \Sigma_1) - \mathcal{N}(s'|\beta_2 + s, \Sigma_1) + \\ &\quad \mathcal{N}(s'|\beta_2 + s, \Sigma_1) - \mathcal{N}(s'|\beta_2 + s, \Sigma_2)| ds' \end{aligned}$$

where we have simply added and subtracted the same term. Using the triangle inequality, we can split the expression into two terms:

$$\begin{aligned} & d_{var}(\mathcal{N}(s'|\beta_1 + s, \Sigma_1), \mathcal{N}(s'|\beta_2 + s, \Sigma_2)) \\ &\leq \frac{1}{2}\int_{s'} |\mathcal{N}(s'|\beta_1 + s, \Sigma_1) - \mathcal{N}(s'|\beta_2 + s, \Sigma_1)| \, ds' \\ &+ \frac{1}{2}\int_{s'} |\mathcal{N}(s'|\beta_2 + s, \Sigma_1) - \mathcal{N}(s'|\beta_2 + s, \Sigma_2)| \, ds', \end{aligned}$$

one where the means are the same and the variances are different, and one where the variances are the same and the means are different. The second term equals

$$\frac{1}{2}(2 - 2A) = 1 - A, \qquad (6)$$

where A is the area of the intersection between two Gaussians with the same mean and different variances.

To upper bound this term, we would like to find a lower bound on A. We can construct a new weighted Gaussian that lies entirely within the intersection area and has the same mean as the two Gaussians $(\beta_2 + s)$ (see Figure 1 for a one dimensional example). We can set the covariance of this new Gaussian by setting its variance along each dimension i to be the smaller of the two Gaussians' variances: $\sigma_{int,i}^2 = \min[\sigma_{i1}^2, \sigma_{i2}^2]$. We then determine the weight on the Gaussian w_{int} by requiring that its height at the mean be no more than the smaller of the two Gaussians. Since we have assumed that $|\Sigma_1| \leq |\Sigma_2|$, then the height at the mean of the smaller Gaussian is simply $1/((2\pi)^{N_{dim}/2}|\Sigma_2|^{0.5})$. Therefore, we can set w_{int} as

$$w_{int}\mathcal{N}_{int}(\beta_2 + s|\beta_2 + s, \Sigma_{int}) = \frac{1}{(2\pi)^{N_{dim}/2}|\Sigma_2|^{0.5}}.$$

Solving for w_{int}, we get

$$w_{int} = \left(\prod_{i=1}^{N_{dim}} \frac{\min[\sigma_{1i}^2, \sigma_{2i}^2]}{\sigma_{2i}^2}\right)^{0.5}.$$

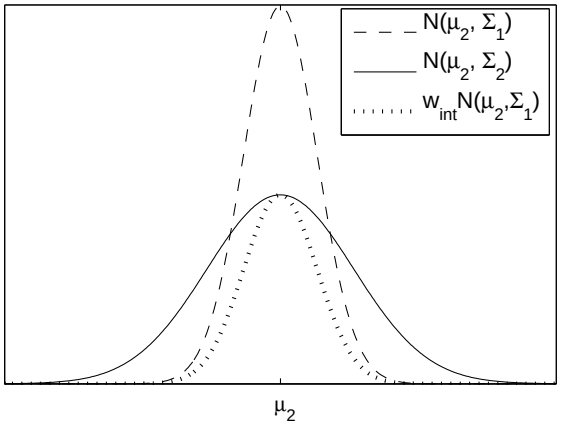

Figure 1: Two Gaussians with identical means and different variances, and a new weighted Gaussian that lies entirely inside their intersection.

This weighted Gaussian always lies within the intersection region and therefore the A is at least

$$\int w_{int}\mathcal{N}_{int}(s'|\beta_2 + s, \Sigma_{int})ds' = w_{int}.$$

Now, substituting this expression back into Equation 6

$$\frac{1}{2}\int_{s'} |\mathcal{N}(s'|\beta_2 + s, \Sigma_1) - \mathcal{N}(s'|\beta_2 + s, \Sigma_2)|\, ds'$$
$$\leq \quad 1 - w_{int} \;=\; 1 - \left(\prod_{i=1}^{N_{dim}} \frac{\min[\sigma_{1i}^2, \sigma_{2i}^2]}{\sigma_{2i}^2}\right)^{0.5}. \quad (7)$$

Next, consider the first term in Equation 6, which looks at the difference between two Gaussians with different means and identical variances. From Abbeel and Ng (2005) (Proposition 7), this expression is upper bounded by

$$\frac{||\beta_2 + s - (\beta_1 + s)||_2}{\sqrt{2\pi}\sigma_{\min}} = \frac{||\beta_2 + \beta_1||_2}{\sqrt{2\pi}\sigma_{\min}}, \quad (8)$$

where $\sigma_{\min}$ is the minimum standard deviation along any of the dimensions. Combining Equations 8 and 7 immediately gives the desired result.

Note this function is 0 when the means and the variances are the same, as one would hope. [2]

We next seek to determine the number of samples necessary to ensure that d_{var} is tightly bounded when evaluated at the estimated model parameters and the true model parameters.

[2]The true d_{var} is upper bounded by 1, whereas this expression can go higher, so it is overly pessimistic when the difference between the two Gaussians' parameters is large, but more accurate as their difference goes to 0. Since we need to estimate the parameters fairly precisely, we are more concerned with this second case.

Lemma 4.2 *Given ϵ and δ, let $T_\beta \geq \frac{2N_{dim}B^2}{\epsilon^2}\ln\frac{6N_{dim}}{\delta}$ where $B > 0$ is arbitrary. Then, if there are T_β transition samples (s, a, s') where $\|s' - s\|_\infty < B$ then with probability at least $1 - \frac{\delta}{3}$, the estimated offset parameter $\tilde{\beta}$ is within ϵ of the true offset parameter β^*: $\Pr(\|\tilde{\beta} - \beta^*\|_2 \leq \epsilon) \geq 1 - \frac{\delta}{3}$.*

Proof Since the difference between successive states $\|s' - s\|$ is bounded above and below by B, we can apply Hoeffding's inequality:[3]

$$\Pr\left(|\tilde{\beta}_i - \beta_i^*| \geq \frac{\epsilon}{\sqrt{N_{dim}}}\right) \leq 2\exp\left(\frac{-2T_1^2\frac{\epsilon^2}{N_{dim}}}{T_1(2B)^2}\right). \quad (9)$$

To find the number of samples T_1 needed to ensure this bound holds with probability at least $1 - \frac{\delta}{3N_{dim}}$, solve for $T_1 = \frac{2N_{dim}B^2}{\epsilon^2}\ln\frac{6N_{dim}}{\delta}$. Doing so independently for each dimension and using the fact $\|\tilde{\beta} - \beta^*\|_2 \leq \sqrt{N_{dim}}\max_i|\tilde{\beta}_i - \beta_i^*|$, we can ensure $\|\beta - \beta^*\|_2 \leq \epsilon$ with at least probability $1 - \frac{\delta}{3}$.

We next analyze the number of samples necessary to estimate the variance.

Lemma 4.3 *Let ϵ and δ be given and let $T_\sigma \geq \frac{8B^4}{\epsilon - \epsilon^2}\ln\frac{6N_{dim}}{\delta}$ for arbitrary $B > 0$. Then if there are T_σ samples where $\|s' - s\|_\infty < B$, then with with probability at least $1 - \frac{\delta}{3}$, the estimated variance parameter $\tilde{\sigma_i^2}$ along each dimension i is within ϵ of the true variance parameter σ_i^2 along the same dimension: $\Pr(\max_i|\tilde{\sigma}_i^2 - \sigma_i^2| \leq \epsilon) \geq 1 - \frac{\delta}{3}$.*

Proof We can independently estimate the variance along each dimension. Assume $\max_i|\tilde{\beta}_i - \beta_i^*| \leq \epsilon$. By definition, $\sigma_i^2 = \mathbf{E}[(s_i' - (s_i + \beta_i^*))^2]$, where $\mathbf{E}$ denotes expectation with respect to $s_i' \sim N(s_i + \beta_i^*, \sigma_i^2)$. But, since $\tilde{\beta}_i \neq \beta_i^*$ in general, $\sigma^2 \neq \mathbf{E}[(s_i' - (s_i + \tilde{\beta}_i))^2]$. Define $\bar{\sigma_i^2} = \mathbf{E}[(s_i - (s_i + \tilde{\beta}_i))^2]$. Then,

$$\begin{aligned}\left|\bar{\sigma_i^2} - \sigma_i^2\right| &= \left|\mathbf{E}\left[(s_i' - (s_i + \tilde{\beta}_i))^2 - (s_i' - (s_i + \beta_i^*))^2\right]\right| \\ &= \left|\mathbf{E}\left[\left(\beta_i^* - \tilde{\beta}_i\right)\left(2s_i' - (s_i + \beta_i^*) - (s_i + \tilde{\beta}_i)\right)\right]\right| \\ &\leq \epsilon\left|\mathbf{E}\left[2s_i' - (s_i + \beta_i^*) - (s_i + \tilde{\beta}_i)\right]\right| \\ &= \epsilon\left|\mathbf{E}\left[(s_i + \beta_i^*) - (s_i + \tilde{\beta}_i)\right]\right| \;\leq\; \epsilon^2.\end{aligned}$$

Therefore, $\Pr\left(\left|\tilde{\sigma_i^2} - \sigma_i^2\right| > \epsilon\right)$ is at most $\Pr\left(\left|\tilde{\sigma_i^2} - \bar{\sigma_i^2}\right| + \left|\bar{\sigma_i^2} - \sigma_i^2\right| > \epsilon\right)$, due to the triangle inequality, which in turn is at most

[3]Strictly speaking, the true mean of $s_i' - s_i$ is not β_i^* as we ignore a sample when $|s_i' - s_i| > B$, which can introduce a bias. But, this bias is insignificant when $B \gg \max[B_\beta, B_\sigma]$. In fact, if $B = \Omega(\epsilon^{-0.5}\max[B_\beta, B_\sigma])$, then the introduced bias is at most ϵ, and we will obtain asymptotically identical results.

$\Pr\left(\left|\tilde{\sigma_i^2} - \bar{\sigma_i^2}\right| + \epsilon^2 > \epsilon\right)$ as $|\bar{\sigma_i^2} - \sigma_i^2| \leq \epsilon^2$. Since $(s'_i - (s_i + \tilde{\beta}_i))^2 \in [0, 4B^2]$, we can apply Hoeffding's inequality to upper bound T_σ: solving

$$\begin{aligned}\Pr\left(\left|\tilde{\sigma_i^2} - \bar{\sigma_i^2}\right| + \epsilon^2 > \epsilon\right) &= \Pr\left(\left|\tilde{\sigma_i^2} - \bar{\sigma_i^2}\right| > \epsilon - \epsilon^2\right) \\ &\leq 2\exp\left(-\frac{2T_\sigma^2(\epsilon-\epsilon^2)^2}{T_2(4B^2)^2}\right) \\ &= \frac{\delta}{3N_{dim}},\end{aligned}$$

for T_σ gives $T_\sigma = \frac{8B^4}{(\epsilon-\epsilon^2)^2}\ln\frac{6N_{dim}}{\delta}$. Applying a union bound to all N_{dim} dimensions, T_σ ensures $\Pr(\max_i |\tilde{\sigma_i^2} - \sigma_i^2| \leq \epsilon) \geq 1 - \frac{\delta}{3}$.

One final lemma is used to bound how many samples must be collected until enough "good" samples are collected that fulfill our criteria that $\|s' - s\|_\infty < B$.

Lemma 4.4 *Let T be the number of observed samples before $T_0 = \max[T_\beta, T_\sigma]$ good samples are collected. Then, $\Pr(T > \frac{\delta T_0}{\delta - 3N_{dim}p_0}) < \frac{\delta}{3}$, where $p_0 = \sqrt{\frac{8}{\pi}}\frac{B_\sigma^3}{(B-B_\beta)^3}$ and setting $B > B_\beta + \sqrt[6]{\frac{72N_{dim}^2}{\pi\delta^2}}B_\sigma$ ensures $\delta > 3N_{dim}p_0$.*

Proof It follows from the union bound that $\Pr(\|s' - s\|_\infty > B) \leq N_{dim}\Pr(|s'_i - s_i| > B)$ for all i. We will show that $\Pr(|s'_i - s_i| > B)$ is small. Let $\varphi(x)$ and $\Phi(x)$ be the probability density function and cumulative distribution function of the standard Gaussian distribution, respectively. Then,

$$\begin{aligned}\Pr(s'_i - s_i > B) &= 1 - \Phi\left(\frac{B - \beta_i^*}{\sigma_i}\right) \\ &\leq \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(B-\beta_i^*)^2}{2\sigma_i^2}\right)\frac{1}{\frac{B-\beta_i^*}{\sigma_i}} \\ &= \frac{\sigma_i}{\sqrt{2\pi}(B-\beta_i^*)}\exp\left(-\frac{(B-\beta_i^*)^2}{2\sigma_i^2}\right),\end{aligned}$$

where the first equality follows from the definition, and the inequality follows from the fact that $1 - \Phi(y) < \frac{\varphi(y)}{y}$ when $y > 0$. Now, we can apply the inequality $e^{-x} < \frac{1}{1+x}$ to obtain

$$\begin{aligned}\Pr(s'_i - s_i > B) &\leq \frac{\sigma_i}{\sqrt{2\pi}(B-\beta_i^*)}\cdot\frac{1}{1+\frac{(B-\beta_i^*)^2}{2\sigma_i^2}} \\ &< \sqrt{\frac{2}{\pi}}\frac{\sigma_i^3}{(B-\beta_i^*)^3} \leq \sqrt{\frac{2}{\pi}}\frac{B_\sigma^3}{(B-B_\beta)^3}.\end{aligned}$$

Similarly, we may upper bound $\Pr(s'_i - s_i < -B)$ and thus $\Pr(|s'_i - s_i| > B) < p_0$ where p_0 is given in the lemma statement.

Now, return to the full multivariate case:

$$\Pr(\|s' - s\|_\infty > B) \leq N_{dim}p_0 = \sqrt{\frac{8}{\pi}}\frac{B_\sigma^3 N_{dim}}{(B-B_\beta)^3}.$$

This inequality indicates that every sample is a "bad" sample with probability at most $N_{dim}p_0$. Given T i.i.d. samples, let $N(T)$ be the number of bad samples. Our estimation algorithm fails to have T_0 good samples if and only if $N(T) > T - T_0$. By Markov's inequality,

$$\Pr(N(T) > T - T_0) \leq \frac{\mathbf{E}[N(T)]}{T - T_0} < \frac{TN_{dim}p_0}{T - T_0}.$$

Solving for T by letting the last expression equal $\frac{\delta}{3}$ gives $T = \frac{\delta T_0}{\delta - 3N_{dim}p_0}$. We can obtain the minimum value of B by solving $3N_{dim}p_0 = \delta$ for B.

Combining these results with Lemmas 4.2 and 4.3 gives a condition on the minimum number of samples necessary to ensure, with high probability, the estimated parameters of a particular type-action dynamics model are close to the true parameters:

$$T = \max[T_\beta, T_\sigma] = O\left(\frac{N_{dim}B^4}{\epsilon^2}\ln\frac{N_{dim}}{\delta}\right).$$

4.2 PLANNING ERROR

We next bound the error between the value function found by solving our particular continuous state Markov decision process using fitted value iteration compared to the optimal value function V^*. Recall that by performing FVI, we are essentially mapping the original MDP to a new finite-state MDP where the states are the chosen fixed points.

Under a set of four assumptions, Chow and Tsitsiklis (1991) proved that the optimal value function V_ε of a discrete-state MDP formed by discretizing a continuous-state MDP into $O(\varepsilon)$-length (per dimension) [4] grid cells is an ε-close approximation of the optimal continuous-state MDP value function V^*:

$$||V_\varepsilon - V^*|| \leq \varepsilon.$$

The first two assumptions used to prove the above result include that the reward function and probability distribution are Lipschitz-continuous. In our work the

[4] More specifically, the grid spacing h_g must satisfy $h_g \leq \frac{(1-\gamma)^2\varepsilon}{K_1+2KK_2}$ and $h_g \leq \frac{1}{2K}$ where K is the larger of the Lipschitz constants arising from the assumptions discussed in the text, and K_1 and K_2 are constants discussed in Chow and Tsitsiklis (1991). For small ε any h_g satisfying the first condition will automatically satisfy the second condition.

reward function is assumed to be given so this condition is a prior condition on the problem specification. Our probability distributions are Gaussian distributions which are Lipschitz-continuous so the second condition holds. The third key assumption is that the dynamics probablity represent a true probablity measure that sums to 1 ($\int_s' p(s'|s,a) = 1$), though the authors show that this assumption can be relaxed to $\int_s' p(s'|s,a) \leq 1$ and the main results still hold. In our work our dynamics models represent true probability models. Chow and Tsitsiklis's final assumption is that there is a bounded difference between any two controls: in our case we handle finite controls and this holds directly.

In summary, assuming the reward model fulfills the first assumption, our framework satisfied all four assumptions made by Chow and Tsitsiklis. Therefore, by selecting fixed grid points at a regular spacing of $O(\epsilon_{FVI})$ in each dimension, we can ensure that $||\tilde{V}_{FVI} - V^*||_\infty$ is at most ϵ_{FVI} where $\tilde{V}_{FVI}$ is the FVI optimal value function.

4.3 APPROXIMATE REINFORCEMENT LEARNING

The next lemma relates the accuracy in the dynamics model parameters, and the error induced by approximate planning, to the value function of two MDPs. The proof strongly parallels a similar Simulation Lemma in recent work by Strehl and M.Littman (2008).

Lemma 4.5 *Let $M_1 = \langle S, A, p_1(s'|s,a), R, \gamma \rangle$ and $M_2 = \langle S, A, p_2(s'|s,a), R, \gamma \rangle$ be two typed MDPs with dynamics as characterized in Equation 1 and non-negative rewards bounded above by 1. Assume $\frac{||\beta_1 - \beta_2||_2}{\sqrt{2\pi}\sigma_{\min}} \leq F_1$ and $|1 - \frac{|\Sigma_1|^{0.5}}{|\Sigma_2|^{0.5}}| \leq F_2$ for all types t and actions a. Also assume that the difference between the value function $\tilde{V}$ obtained by performing approximate planning by fitted value iteration (FVI) compared to the optimal value function V^*, $||\tilde{V} - V^*||_\infty$ is at most F_3. Let π be a policy that can be applied to both M_1 and M_2. Then, there exists a set of constants C_1, C_2 and C_3 such that for any $0 < \epsilon \leq V_{\max}$ and stationary policy π, if $F_1 = C_1(\frac{(1-\gamma)^2\epsilon}{\gamma})$ $F_2 = C_2(\frac{\epsilon(1-\gamma)^2}{\gamma})$, and $F_3 = C_3(\frac{\epsilon(1-\gamma)}{\gamma})$, then for all states s and actions a, $|Q_1^\pi(s,a) - \tilde{Q}_2^\pi(s,a)| \leq \epsilon$,where $\tilde{Q}_2^\pi$ denotes the state-action value obtained by performing FVI on MDP M_2 and Q_1^π denotes the optimal state-action value for MDP M_1.*

Proof Let $\Delta_Q = \max_{s,a} |Q_1^\pi(s,a) - \tilde{Q}_2^\pi(s,a)|$. Note that since we are taking the max over all actions, Δ_Q is also equal or greater than $\max_s |V_1^\pi(s) - \tilde{V}_2^\pi|$. Let $Lp_2(s'|s,a)$ denotes an approximate backup of FVI.

Since these value functions are the fixed-point solutions to their respective Bellman operator, we have for every (s,a) that

$$
\begin{aligned}
& |Q_1^\pi(s,a) - \tilde{Q}_2^\pi(s,a)| \\
= \; & \left| \left(R(s,a) + \gamma \int_{s' \in S} T_1(s') V_1^\pi(s') ds' \right) - \right. \\
& \left. \left(R(s,a) + \gamma \int_{s' \in S} LT_2(s') \tilde{V}_2^\pi(s') ds' \right) \right| \\
\leq \; & \gamma \left| \int_{s' \in S} \left[T_1(s') V_1^\pi(s') - LT_2(s') \tilde{V}_2^\pi(s') \right] ds' \right| \\
\leq \; & \gamma \left| \int_{s' \in S} \left[T_1(s') V_1^\pi(s') - T_1(s') \tilde{V}_2^\pi(s') \right] ds' + \right. \\
& \left. \int_{s' \in S} \left[T_1(s') \tilde{V}_2^\pi(s') - LT_2(s') \tilde{V}_2^\pi(s') \right] ds' \right| \\
\leq \; & \gamma \left| \int_{s' \in S} T_1(s') (V_1^\pi(s') - \tilde{V}_2^\pi(s')) ds' \right| + \\
& \gamma \left| \int_{s' \in S} T_1(s') \tilde{V}_2^\pi(s') - LT_2(s') \tilde{V}_2^\pi(s') ds' \right|
\end{aligned}
$$

where the last step follows from the triangle inequality. Now add and subtract $\int_{s' \in S} T_2(s') \tilde{V}_2^\pi(s') ds'$ and again apply the triangle inequality:

$$
\begin{aligned}
& |Q_1^\pi(s,a) - \tilde{Q}_2^\pi(s,a)| \\
\leq \; & \gamma \left| \int_{s' \in S} T_1(s') (V_1^\pi(s') - \tilde{V}_2^\pi(s')) ds' \right| + \\
& \gamma \left| \int_{s' \in S} \left[T_1(s') \tilde{V}_2^\pi(s') - T_2(s') \tilde{V}_2^\pi(s') \right] ds' + \right. \\
& \left. \int_{s' \in S} \left[T_2(s') \tilde{V}_2^\pi(s') - LT_2(s') \tilde{V}_2^\pi(s') \right] ds' \right| \\
\leq \; & \gamma \left| \int_{s' \in S} T_1(s') (V_1^\pi(s') - \tilde{V}_2^\pi(s')) ds' \right| + \\
& \gamma \left| \int_{s' \in S} (T_1(s') - T_2(s')) \tilde{V}_2^\pi(s') ds' \right| + \\
& \gamma \left| \int_{s' \in S} T_2(s') \tilde{V}_2^\pi(s') - LT_2(s') \tilde{V}_2^\pi(s') ds' \right| \\
\leq \; & \gamma \left| \int_{s' \in S} T_1(s') (V_1^\pi(s') - \tilde{V}_2^\pi(s')) ds' \right| + \\
& \gamma \left| \int_{s' \in S} (T_1(s') - T_2(s')) \tilde{V}_2^\pi(s') ds' \right| + \\
& \gamma \left| \int_{s' \in S} T_2(s') \tilde{V}_2^\pi(s') - LT_2(s') \tilde{V}_2^\pi(s') ds' \right|.
\end{aligned}
$$

This expression must hold for all states s and actions

a, so it must also hold for Δ_Q:

$$\begin{aligned}\Delta_Q &\leq \gamma\Delta_Q + \gamma\left|\int_{s'\in S}(T_1(s') - T_2(s'))\tilde{V}_2^{\pi}(s')ds'\right| + \\ &\quad \gamma\left|\int_{s'\in S} T_2(s')\tilde{V}_2^{\pi}(s') - LT_2(s')\tilde{V}_2^{\pi}(s')ds'\right| \\ &\leq \gamma\Delta_Q + \gamma V_{max}\left|\int_{s'\in S} T_1(s') - T_2(s')ds'\right| + \\ &\quad \gamma\left|\int_{s'\in S} T_2(s')\tilde{V}_2^{\pi}(s') - LT_2(s')\tilde{V}_2^{\pi}(s')ds'\right| \\ &\leq \gamma\Delta_Q + \gamma V_{max} d_{var}(T_1, T_2) + \gamma\epsilon_{FVI}\end{aligned}$$

where we have again used the triangle inequality. Therefore

$$\begin{aligned}\Delta_Q &\leq \gamma\Delta_Q + \gamma V_{\max} d_{var} + \gamma\epsilon_{FVI} \\ &= \frac{\gamma\frac{dvar}{1-\gamma}}{1-\gamma} + \frac{\gamma\epsilon_{FVI}}{1-\gamma}.\end{aligned}$$

So, we have now expressed the error in the value function as the sum of the error due to the model approximation and the error due to using fitted value iteration for planning. If we can bound the error of each to be less than or equal to $\epsilon/2$, then the overall error $\Delta_Q \leq \epsilon$.

First, note that that we can bound the error due to FVI to be less than or equal to $\frac{\epsilon(1-gamma)}{2\gamma}$ by selecting a grid width of $\mathrm{O}(\frac{\epsilon(1-gamma)^3}{\gamma})$ as discussed in Section 4.2. This ensures that $\frac{\gamma\epsilon_{FVI}}{1-\gamma} \leq \frac{\epsilon}{2}$.

We now wish to bound the error due to the model approximations to be no more than $\epsilon/2$:

$$\frac{\gamma d_{var}}{(1-\gamma)^2} \leq \frac{\epsilon}{2} \implies d_{var} \leq \frac{(1-\gamma)^2\epsilon}{2\gamma}. \tag{10}$$

From Proposition 4.1, $d_{var} \leq 1 - \left(\prod_{i=1}^{N_{dim}} \frac{\min[\sigma_{1i}^2, \sigma_{2i}^2]}{\sigma_{2i}^2}\right)^{0.5} + \frac{||\beta_2 - \beta_1||_2}{\sqrt{(2\pi)}\sigma_{\min}}$. So, in order for Equation 10 to hold, we split the error into two terms and require that

$$\frac{||\beta_2 - \beta_1||_2}{(2\pi)^{N_{dim}}|\Sigma_1|^{0.5}} \leq \frac{(1-\gamma)^2\epsilon}{4\gamma} \tag{11}$$

and

$$1 - \left(\prod_{i=1}^{N_{dim}} \frac{\min[\sigma_{1i}^2, \sigma_{2i}^2]}{\sigma_{2i}^2}\right)^{0.5} \leq \frac{(1-\gamma)^2\epsilon}{4\gamma}. \tag{12}$$

Adding and subtracting $\max[\sigma_{1i}^2, \sigma_{2i}^2]$ from the numerator of the fraction in Equation 12 we get $1 - \left(\prod_{i=1}^{N_{dim}} \frac{\min[\sigma_{1i}^2, \sigma_{2i}^2]}{\sigma_{2i}^2}\right)^{0.5} = 1 - \left(\prod_{i=1}^{N_{dim}} \frac{\min[\sigma_{1i}^2, \sigma_{2i}^2] + \max[\sigma_{1i}^2, \sigma_{2i}^2] - \max[\sigma_{1i}^2, \sigma_{2i}^2]}{\sigma_{2i}^2}\right)^{0.5}$.

There are two cases for each i: either $\sigma_{2i}^2 \geq \sigma_{1i}^2$ or vica versa. If $\sigma_{2i}^2 \geq \sigma_{1i}^2$ then

$$\begin{aligned}\frac{\min[\sigma_{1i}^2, \sigma_{2i}^2 + \max[\sigma_{1i}^2, \sigma_{2i}^2] - \max[\sigma_{1i}^2, \sigma_{2i}^2]]}{\sigma_{2i}^2} &= \\ \frac{\sigma_{1i}^2 - \sigma_{2i}^2 + \sigma_{2i}^2}{\sigma_{2i}^2} &= \\ 1 - \frac{|\sigma_{1i}^2 - \sigma_{2i}^2}{\sigma_{2i}^2}.\end{aligned}$$

If $\sigma_{2i}^2 < \sigma_{1i}^2$ then

$$\begin{aligned}\frac{\min[\sigma_{1i}^2, \sigma_{2i}^2 + \max[\sigma_{1i}^2, \sigma_{2i}^2] - \max[\sigma_{1i}^2, \sigma_{2i}^2]]}{\sigma_{2i}^2} &= \\ \frac{\sigma_{2i}^2 - \sigma_{1i}^2 + \sigma_{1i}^2}{\sigma_{2i}^2} &= \\ \frac{\sigma_{1i}^2}{\sigma_{2i}^2} - \frac{|\sigma_{1i}^2 - \sigma_{2i}^2}{\sigma_{2i}^2} &\geq \\ 1 - \frac{|\sigma_{1i}^2 - \sigma_{2i}^2|}{\sigma_{2i}^2}\end{aligned}$$

So in both cases $1 - \frac{|\sigma_{1i}^2 - \sigma_{2i}^2}{\sigma_{2i}^2}$ is a lower bound to the fraction. Therefore

$$1 - \left(\prod_{i=1}^{N_{dim}} \frac{\min[\sigma_{1i}^2, \sigma_{2i}^2]}{\sigma_{2i}^2}\right)^{0.5} \leq 1 - \left(\prod_{i=1}^{N_{dim}} 1 - \frac{\left|\sigma_{1i}^2 - \sigma_{2i}^2\right|}{\sigma_{2i}^2}\right)^{0.5}.$$

We can further upper bound this by substituting in the minimum variance and maximum difference between variances over any dimension:

$$1 - \left(\prod_{i=1}^{N_{dim}} \frac{\min[\sigma_{1i}^2, \sigma_{2i}^2]}{\sigma_{2i}^2}\right)^{0.5} \leq 1 - \left(1 - \frac{\max_i\left|\sigma_{1i}^2 - \sigma_{2i}^2\right|}{\sigma_{\min}^2}\right)^{N_{dim}}$$

Since $(1-x)^C \geq 1 - \lceil C \rceil x \ \forall C \geq 0$

$$\begin{aligned}1 - \left(\prod_{i=1}^{N_{dim}} \frac{\min[\sigma_{1i}^2, \sigma_{2i}^2]}{\sigma_{2i}^2}\right)^{0.5} &\leq 1 - \left(1 - \lceil\frac{N_{dim}}{2}\rceil\frac{\left|\sigma_{1i}^2 - \sigma_{2i}^2\right|}{\sigma_{\min}^2}\right) \\ &= \lceil\frac{N_{dim}}{2}\rceil\frac{\left|\sigma_{1i}^2 - \sigma_{2i}^2\right|}{\sigma_{\min}^2}\end{aligned}$$

We now set the right hand side to $\frac{(1-\gamma)^2\epsilon}{4\gamma}$ (to upper bound our desired expression) and solve for $|\sigma_{1i}^2 - \sigma_{2i}^2|$:

$$\left|\sigma_{1i}^2 - \sigma_{2i}^2\right| \leq \frac{\sigma_{\min}^2(1-\gamma)^2\epsilon}{4\gamma\lceil N_{dim}/2\rceil}.$$

From Lemma 4.4, we know that after $O(\frac{N_{dim}^3 B^4\gamma^2}{\sigma_{\min}^4(1-\gamma)^4\epsilon^2}\ln\frac{N_{dim}}{\delta})$ samples, this bound is guaranteed to hold with probability at least $1-\delta$. This number of samples is also sufficient to ensure that Equation 11 holds with probability at least $1-\delta$.

4.4 APPROXIMATELY OPTIMAL REINFORCEMENT LEARNING

Theorem 4.6 *For any given δ and ϵ in a continuous-state noisy offset dynamics MDP with N_T types where the variance along each dimension of all the dynamics models is bounded by $[\sigma^2_{\min}, B^2_\sigma]$ and the offset parameter is bounded by $|\beta_i| < B_\beta$ on all but N_{total} timesteps, our algorithm will follow a 4ϵ-optimal policy from its current state with probability at least $1 - 2\delta$, where N_{total} is polynomial in the problem parameters $(N_T, |A|, \frac{1}{\delta}, \frac{1}{\epsilon}, \frac{1}{1-\gamma}, \frac{1}{\sigma_{\min}}, B_\sigma, B_\beta, N_{dim})$.*

Proof In short, we demonstrate that our algorithm fulfills the three criteria outlined earlier. From the analysis done in the prior section, we know that after $N_{TA} = O\left(\frac{N^3_{dim} B^4 \gamma^2}{\sigma^4_{\min}(1-\gamma)^4\epsilon^2}\right)$ samples, with probability $1 - \delta$, the errors $\|\beta_1 - \beta_2\|_2$ and for each state dimension i $|\sigma_i^2 - \tilde{\sigma_i^2}|$ will be $O((1-\gamma)^2\epsilon)$. We also chose the spacing of our fixed grid points such that $\frac{\epsilon_{FVI}\gamma}{1-\gamma} \leq \frac{\epsilon}{2}$. Then, the Simulation Lemma (4.5) guarantees that the approximate value of our known state MDP solved using FVI is ϵ-close to the optimal value of the known state MDP with the true dynamics parameters $||\tilde{V}^\pi_{\tilde{K}} - V^\pi_K||_\infty \leq \epsilon$. All unknown type-action pairs that have not yet been experienced N_M times are considered to be unknown and their value is set to $V_{\max}$. So, condition (1) and (2) (Strehl et al., 2006) hold. The third condition limits the number of times the algorithm may experience an unknown type-action tuple. Since there are a finite number of types and actions, this quantity is bounded above by $N_{TA}N_T|A|$, which is a polynomial in the problem parameters $(N_T, |A|, \frac{1}{\delta}, \frac{1}{\epsilon}, \frac{1}{1-\gamma}, \frac{1}{\sigma_{\min}}, B_\sigma, B_\beta, N_{dim})$. Therefore, our algorithm is PAC-MDP.

5 EXPERIMENT

To examine the performance of our algorithm, we performed experiments in a real world robotic environment involving a navigation task where a robotic car must traverse multiple surface types to reach a goal location. Our experiments seek to demonstrate both that our dynamics models provide a sufficiently good representation of real world dynamics to allow our algorithm to learn good policies, and that our approach can outperform an alternative related approach.

An alternate model to that suggested in this paper is to discretize the world environment. Recent work by Leffler et al. (2007) provided RAM-Rmax, a provably efficient RL algorithm for learning in discrete-state worlds with types. The authors demonstrated that by explicitly representing the types they could get a significant learning speedup compared to Rmax,

Figure 2: Image of the environment. The start location and orientation is marked with an arrow. The goal location is indicated by the circle.

which learns a separate dynamics model for each state. This algorithm represents the dynamics model using a list of possible next outcomes for a given type. Our approach assumes a fixed parametric distribution that automatically constrains the size of the representation. Though the RAM-Rmax approach can handle a more general set of dynamics models, we expect our approach to outperform RAM-Rmax when our parametric representation is a good approximation of the true dynamics.

5.1 EXPERIMENTAL SETUP

For our experiment, we ran a LEGO® Mindstorms NXT on a multi-surface environment. This domain, shown in Figure 2, consisted of two types: rocks embedded in wax and a carpeted area. The goal was for the agent to begin in the start location (indicated in the figure by an arrow) and end in the goal without going outside the environmental boundaries. The rewards were -1 for going out of bounds, $+1$ for reaching the goal, and -0.01 for taking an action. Reaching the goal and going out of bounds ended the episode and resulted in the agent getting moved back to the start location.

One difficulty of this environment is the difference in dynamics models. Due to the close proximity of the goal to boundary, the agent needs an accurate dynamics model to reliably reach the goal. To make this task even more difficult, the actions were limited to going forward, turning left, and turning right. By removing the ability for the agent to move backwards, it increased the need for the agent to accurately approach the goal reliably. A robot with an inaccurate transition model would be likely to judge this task as

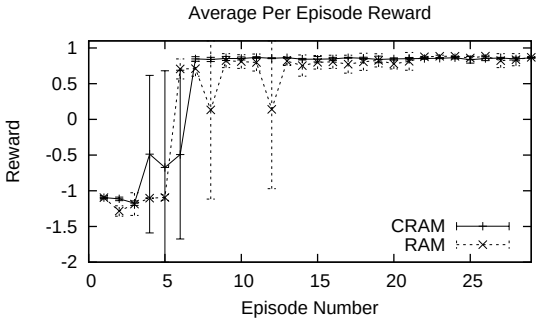

Figure 3: Reward received by algorithms averaged over three runs. Error bars show one standard deviation.

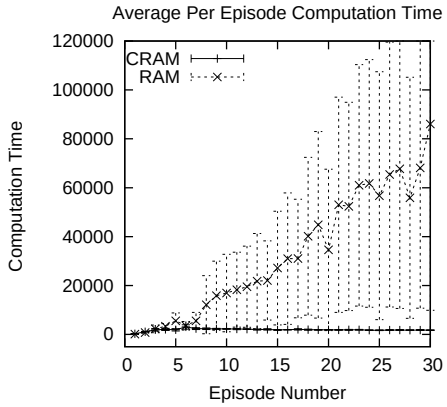

Figure 4: Total time taken by algorithms averaged over three runs. Error bars show one standard deviation.

impossible.

For the experiments, we compared our algorithm (labeled as "CRAM" in the figures) and the RAM-Rmax algorithm (labeled as "RAM"). The fixed points for the fitted value iteration portion of our algorithm were set to the discretized points of the RAM-Rmax algorithm. Both algorithms used an EDISON image segmentation system to uniquely identify the current surface type. The reward function was provided to both algorithms.

The state space is three dimensional: x,y, and orientation. Our algorithm implementation for this domain uses a full covariance matrix to model the dynamic's variance model. For the RAM-Rmax agent, the world was discretized to a forty by thirty by ten state space. In our algorithm we used a function approximator of a weighted sum of Gaussians, as described in Section 2.2. We used the same number of Gaussians to represent the value function as the size of the state space used in the discretized algorithm, and placed these fixed Gaussians at the same locations. The variance over the x and y variables was independent of each other and of orientation, and was set to be 16. In order to average orientation vectors correctly (so that -180 degrees and 180 degrees don't average to 0) we converted orientation vectors to an x,y representation θ_x, θ_y. The variance over these two was set to be 9 for each variable (with no covariance). The value of all variances were set by informal experimentation. For our algorithm and the RAM-Rmax algorithm, the value of N_{TA} was set to four and five, respectively, which was determined after informal experimentation. The discount factor was set to 1.

6 RESULTS

Figure 3 shows the average reward of each of the algorithms with standard deviation over three runs. Both algorithms are able to receive near optimal reward on a consistent basis choosing similar paths to the goal. This demonstrates that our dynamics representation is sufficient to allow our algorithm to learn well in a real-world environment.

In addition, by using a fixed parametric representation, the computational time per episode of our algorithm is roughly constant, compared to the computation time of the RAMRmax algorithm, as shown in Figure 4. This suggests that in addition to our theoretical results, our algorithm is an interesting practical alternative to discretized techniques in certain environments.

7 CONCLUSION

We have presented a new reinforcement-learning algorithm for handling continuous-state typed worlds where the dynamics can be modeled as a noisy offset. We proved that when the noise covariance matrix is diagonal, the algorithm is PAC-MDP. We also demonstrated that these dynamics representations provide a reasonable approximation of real-world dynamics by running our algorithm in a small robotic experiment.

References

Abbeel, P., & Ng, A. Y. (2005). Exploration and apprenticeship learning in reinforcement learning. *Proceedings of the 22nd International Conference on Machine Learning* (pp. 1–8).

Brafman, R. I., & Tennenholtz, M. (2002). R-MAX—a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, *3*, 213–231.

Chow, C., & Tsitsiklis, J. (1991). An optimal multigrid algorithm for continuous state discrete time stochas-

tic control. *IEEE Transactions on Automatic Control*, *36*, 898–914.

Gordon, G. (1995). Stable function approximation in dynamic programming. *Proc. International Conference on Machine Learning*.

Kakade, S. (2003). *On the sample complexity of reinforcement learning*. Doctoral dissertation, University College London.

Keans, M. J., & Singh, S. P. (2002). Near-optimal reinforcement learning in polynomial time. *Machine Learning*, *49*, 209–232.

Leffler, B. R., Littman, M. L., & Edmunds, T. (2007). Efficient reinforcement learning with relocatable action models. *AAAI-07: Proceedings of the Twenty-Second Conference on Artificial Intelligence* (pp. 572–577). Menlo Park, CA, USA: The AAAI Press.

Strehl, A., L.Li, & Littman, M. (2006). Incremental model-based laerners with formal learning-time gaurantees. *Uncertainty in Artificial Intelligence*. Cambridge, USA.

Strehl, A., & M.Littman (2008). Online linear regression and its application to model-based reinforcement learning. *Neural Information Processing Systems 20*.