Conditional Bisimulation for Generalization in Reinforcement Learning

Anuj Mahajan [1] **Amy Zhang** [2]

Abstract

Learning policies that are robust to changes in the environment are critical for real world deployment of Reinforcement Learning (RL) agents. They are also necessary for achieving good generalization across environment shifts. Bisimulation provides a powerful means for abstracting task relevant components of the observation and learning a succinct representation space for training the RL agent in high dimensional spaces by exploiting the rich metric structure induced by the RL dynamics. In this work, we extend the bisimulation framework to also account for context dependent observation shifts. We use simulator based learning as an exemplary setting to demonstrate the use alternate observations to learn a representation space which is invariant to observation shifts using a novel bisimulation based objective. This allows us to deploy the agent to varying observation settings during test time and generalize to unseen scenarios. Empirical analysis on the high-dimensional image based control domains demonstrates the efficacy of our method.

1. Introduction

Many practical scenarios in reinforcement learning (RL) applications require the agent to be robust to changes in the observations space between training and deployment. Such changes can occur due to lack of complete information about the deployment environment which often happens as the training environment is usually highly controlled or simulators are used for training the agent, both of these scenarios seldom capture the complexity and noisiness of the real world. Moreover, these changes can also occur due to various practical errors and constraints under which autonomous agents need to be deployed (e.g. variations in sensor position and fitting on automobiles, change in calibration settings of visual input, change in sensor types due to upgrades, calibration changes due to wear and tear etc.). In this work, we propose a novel solution to the aforementioned problem using the concept of conditional bisimulation and application of simulator/specialized setup during train time which help explicitly teach the agent, the similarities across changes in the observation space. Our method leverages the MDP level isomorphism (Ravindran, 2004) in the observation shift setting for obtaining a richer representation loss. Our methods offers two-fold advantage:

- We can learn representations which are robust to shifts in observation space in a sample efficient manner.
- We learn to ignore task irrelavant features as our metric is grounded in rewards.

[1]Amazon [2]University of Texas at Austin. Correspondence to: Anuj Mahajan <anuj.mahajan.ai@gmail.com>.

Presented at the 2^{nd} Annual Workshop on Topology, Algebra, and Geometry in Machine Learning (TAG-ML) at the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. 2023. Copyright 2023 by the author(s).

2. Background

Reinforcement Learning: A Markov Decision Process (MDP) is formally defined as a tuple $\langle S, U, P, r, \gamma, \rho\rangle$. Here S is the state space of the environment and ρ is the initial state distribution. At each time step t, an agent observes the state $s \in S$ and chooses an action $a \in U$ using its policy $\pi : S \rightarrow \mathcal{P}(U)$, where $\mathcal{P}(\cdot)$ represents the space of distributions on the argument set. This leads to a state transition governed by the distribution $P(s'|s,a) : S \times U \times S \rightarrow [0,1]$, and the agent receives reward $r(s,a) : S \times U \rightarrow [0, R_{max}]$ which can be potentially stochastic. We consider the discounted infinite horizon setting, where the discount factor is given by $\gamma \in [0,1)$. The state-action trajectory of the agent is represented by $\tau \in T \equiv (S \times U)^*$. The value of a policy is defined as: $J^\pi = \mathbb{E}_{\pi,\rho}\left[\sum_{t=0}^{\infty} \gamma^t r_\tau(s_t)\right]$ where the expectation on the RHS is well defined given bounds on rewards and γ. The goal of MDP problem is to find the optimal policy π^* for the optimal policy value J^*.

Rich observations and Contextual MDPs: We now discuss the RL setting used in this work which considers the presence of an underlying parametrized context θ (Hallak et al., 2015; Mahajan et al., 2022), which governs the rewards and transitions in the MDP. Formally, we have $\mathcal{M} \triangleq \langle S, U, P, r, \gamma, \rho, \Theta, P_\Theta, Z, f\rangle$, where Θ defines a space of context parameters, P_Θ is a fixed distribution over the contexts, Z is the set of observations emitted as $f : S \times \Theta \rightarrow Z$. Thus fixing a particular context θ gives us a richly observed MDP (Krishnamurthy et al., 2016; Mahajan, 2023) indexed by θ: $\mathcal{M}_\theta$. We assume that the agent observes θ in our setting. Without loss of generality, we assume $S \subset [0,1]^n$, $Z \subset [0,1]^l$, where typically $n << l$. We will use $f(s,\theta), f_\theta(s)$ interchangeably to highlight the corresponding (un)-curried versions of the observation function.

Bisimulation: MDP Bisimulation defines a notion of state abstraction which groups states that are behaviorally equivalent (Li et al., 2006). Two states s_i and $s_j$are bisimilar if they both share the same immediate reward and equivalent distributions over the next bisimilar states for all possible actions (Givan et al., 2003). Formally:

Definition 1 (Bisimulation Relations (Givan et al., 2003))**.** *Given an MDP $\mathcal{M}$, an equivalence relation B between states is a bisimulation relation if, for all states $s_i, s_j \in S$ that are equivalent under B (denoted $s_i \equiv_B s_j$) the following conditions hold:*

$$\begin{aligned} r(s_i, a) &= r(s_j, a) \qquad \forall a \in U, \\ P(G|s_i, a) &= P(G|s_j, a) \quad \forall a \in U, \quad \forall G \in S_B, \end{aligned}$$

where S_B is the partition of S under the relation B (the set of all groups G of states equivalent under B), and $P(G|s, a) = \sum_{s' \in G} P(s'|s, a)$.

Finding the coarsest bisimulation relation is known to be an NP-hard problem (Givan et al., 2003). Further, the exact partitioning induced from a bisimulation relation is generally impractical as it a very strict notion of equivalence and seldom leads to meaningful compression of the original MDP, this is especially true in continuous domains, where infinitesimal changes in the reward function or dynamics can break the bisimulation relation but still imply exploitable aggregation. Thus towards addressing this, Bisimulation Metrics (Ferns et al., 2011) relaxes the concept of exact bisimulation, and instead define a pseudometric space (S, d), where a distance function $d : S \times S \mapsto \mathbb{R}_{\geq 0}$ measures the behavioral similarity between two states. The bisimulation metric is formally defined as a convex combination of the reward difference added to the Wasserstein distance between transition distributions:

Definition 2 (Bisimulation Metric)**.** *From Theorem 2.6 in (Ferns et al., 2011) with $c \in [0, 1)$:*

$$d(s_i, s_j) = \max_{a \in U} (1 - c) \cdot |r^a_{s_i} - r^a_{s_j}| + c \cdot W_1(P^a_{s_i}, P^a_{s_j}; d).$$

W refers to the Wasserstein-p metric between two probability distributions P_i and P_j, defined as $W_p(P_i, P_j; d) = \inf_{\gamma' \in \Gamma(P_i, P_j)} [\int_{S \times S} d(s_i, s_j)^p \, d\gamma'(s_i, s_j)]^{1/p}$, where $\Gamma(P_i, P_j)$ is the set of all couplings of P_i and P_j. The metric has intuitive interpretations depending on the exact value of p when viewed from the dual perspective, for example $W_1(P_i, P_j; d)$ denotes the cost of transporting mass from distribution P_i to another distribution P_j where the cost is given by the distance metric d. This is known as the earth-mover distance. The above definition can also be modified to include scenarios involving stochastic rewards, where a similar metric is chosen between reward distributions. To account for state similarities arising from following a particular policy, the π-bisimulation metric (Castro, 2020) is similarly defined by fixing a policy π and replacing the rewards and transitions used by their policy based expectations:

$$d^\pi(s_i, s_j) = (1 - c) \cdot |r^\pi_{s_i} - r^\pi_{s_j}| + c \cdot W_1(P^\pi_{s_i}, P^\pi_{s_j}; d^\pi).$$

In this work we will consider the max entropy RL framework as it ensures a unique optimal policy π^*_{merl}(referred as π^* for brevity). Our goal is to leverage generalization and transfer obtained from informing the agent representation with the bisimulation similarity metric under π^*.

3. Methodology

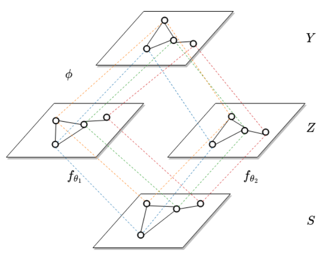
As previously discussed, it is important that agent policies in RL are robust to observation shifts for deployment in real world scenarios. In this work we wish to learn policies which can generalize well across the support set of the context distribution P_Θ. Our goal specifically would be to learn an effective representation function for the RL task set, $\phi : Z \times \Theta \mapsto Y$ which enables robust learning and deployment of autonomous agents to potentially unseen observation shifts (governed by a change in θ), see Fig. 1(a). Under suitable notions of invertibility (eg. see Appendix A), the problem of generalizing across parameterized observation shifts (Section 2) lends itself naturally to the notion of MDP isomorphism (Ravindran, 2004; Mahajan & Tulabandhula, 2017). This is because given two contexts θ_i, θ_j, there is always a one to one mapping between the observations in $\mathcal{M}_{\theta_i} \iff \mathcal{M}_{\theta_j}$ as directed by the underlying state, this is illustrated in Fig. 1(a). This inter-context correspondence helps us inform the representation more efficiently. Concretely, we specify the desiderata which the representation function ϕ must follow, as shown in Fig. 1(b):

- **Base Bisimulation (BB)**: Given a $\theta \in \Theta$, the representation should accurately preserve bisimulation distances between states, thus providing robustness to unimportant noise in observations. Concretely $\forall s_i, s_j \in \mathcal{S}$:
$$d(s_i, s_j) = d_Y(\phi(f_\theta(s_i), \theta), \phi(f_\theta(s_j), \theta)),$$
where d_Y is a metric on Y (we use $Y = \mathbb{R}^m$ and L1 distance for our experiments).

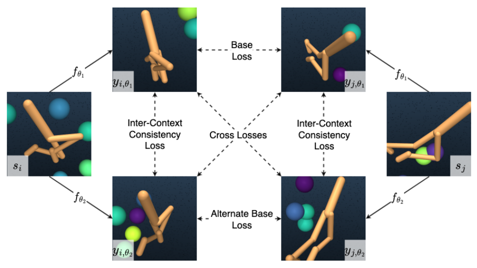
- **Inter-context consistency (ICC)**: The representation should remain invariant under a fixed state as the context changes. Concretely: $\forall s \in \mathcal{S}$ and $\theta_1, \theta_2 \in \Theta$,
$$d_Y(\phi(f_{\theta_1}(s), \theta_1), \phi(f_{\theta_2}(s), \theta_2)) = 0.$$

- **Cross consistency (CC)**: This requires that the representation distance between two states are consistent across observation shifts:
$$d(s_i, s_j) = d_Y(\phi(f_{\theta_1}(s_i), \theta_1), \phi(f_{\theta_2}(s_j), \theta_2)),$$
$$d(s_i, s_j) = d_Y(\phi(f_{\theta_2}(s_i), \theta_2), \phi(f_{\theta_1}(s_j), \theta_1)).$$

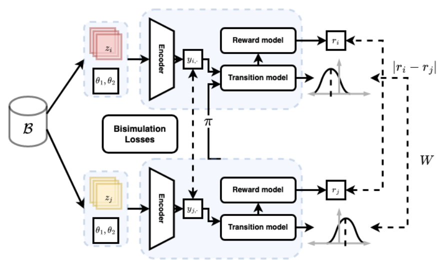
Fig. 1(b) depicts the above representation criteria for 2 different contexts(θ_1, θ_2) on the Mujoco control domain with 3D



(a)



(b)



(c)

Figure 1: (a) Learning representation invariant to observation shifts. (b) Various bisimulation losses. (c) RCB Network.

background objects acting as noise. Towards ensuring the above desiderata, we propose Robust Conditional Bisimulation (RCB, Algorithm 1), a data-efficient approach to learn control policies from unstructured, high-dimensional observations. As evident from Fig. 1(b), for n parallel simulation calls, our method captures $\binom{2n}{2} \sim O(n^2)$ interactions for representation learning using the above conditional bisimulation terms as opposed to $O(n)$ interactions in existing representation learning methods. For instance data augmentation methods based on contrastive learning (like (Laskin et al., 2020b)) focus only on $(f_{\theta_1}(s), f_{\theta_2}(s))$ pairs whereas as plain bisimulation methods (like (Zhang et al., 2021)) focus only on $(f_\theta(s_1), f_\theta(s_2))$ pairs. This order of magnitude increase in utilization of metric information in RCB allows for fast and efficient convergence to an observation invariant representation space.

Algorithm 1 Robust Conditional Bisimulation (RCB)

```
 1: for Time $t = 0$ to $\infty$ do
 2:     Observe $z_t, \theta$
 3:     Encode observation $y_t = \phi(z_t, \theta)$
 4:     Execute action $a_t \sim \pi(y_t)$
 5:     Record data: $\mathcal{D} \leftarrow \mathcal{D} \cup \{z_t, a_t, z_{t+1}, r_{t+1}\}$
 6:     Sample batch $\mathcal{B} \sim \mathcal{D}$
 7:     Train policy: $\mathbb{E}_{\mathcal{B}}[J^\pi]$
 8:     Train encoder using pairwise loss: $\mathcal{L}_{rep}(\phi)$
 9:     Train dynamics: $J(\hat{P},\phi) = (\hat{P}(\phi(z_t,\theta), a_t) - y_{t+1})^2$
10: end for
```

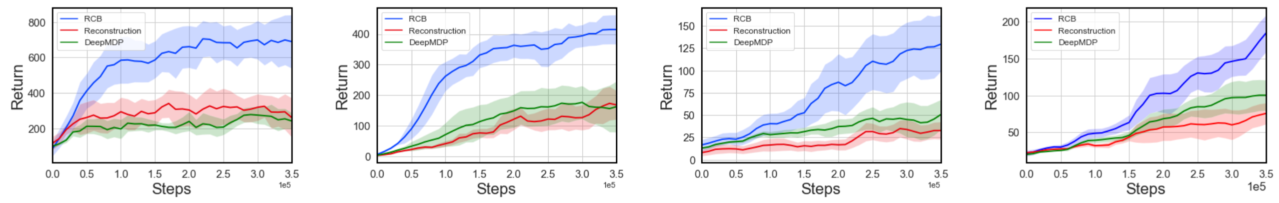
We combine the above three representation conditions into a sum of squared loss components. For this we sample pairs of experiences i, j from the buffer along with base context θ_1 and an alternate context θ_2 both sampled from P_Θ at episode start. We next compute the embedding of the underlying states under the contexts and finally compute the representation loss term as follows:

$$\mathcal{L}_{rep}(\phi) = \lambda_{icc}\big[|\overline{y}_{i,\theta_1} - \overline{y}_{i,\theta_2}|_1^2 + |\overline{y}_{j,\theta_1} - \overline{y}_{j,\theta_2}|_1^2\big]$$
$$+ \lambda_{base}\big[\big(|\overline{y}_{i,\theta_1} - \overline{y}_{j,\theta_1}|_1 - T_{i,j}\big)^2 + \big(|\overline{y}_{i,\theta_2} - \overline{y}_{j,\theta_2}|_1 - T_{i,j}\big)^2\big]$$
$$+ \lambda_{cc}\big[\big(|\overline{y}_{i,\theta_1} - \overline{y}_{j,\theta_2}|_1 - T_{i,j}\big)^2 + \big(|\overline{y}_{i,\theta_2} - \overline{y}_{j,\theta_1}|_1 - T_{i,j}\big)^2\big]$$

where we use the following notation: $y_{i,\theta} = \phi(f(s_i, \theta), \theta)$ with $\overline{y}_{i,\theta}$ representing embeddings with stopped gradient and the target bisimulation distance $T_{i,j} = |r_i - r_j| + \gamma W_2(\hat{P}(\cdot|y_{i,\theta_1}, a_i), \hat{P}(\cdot|y_{j,\theta_1}, a_j))$. The relative weights for the three loss terms are given by hyper-parameters $\lambda_{base}, \lambda_{icc}, \lambda_{cc}$ respectively. We use a permuted batch of $\mathcal{B}$ for pairwise representation loss computation in step-8 of Algorithm 1. Similarly we a probabilistic dynamics model $\hat{P}$ which outputs a Gaussian distribution. This allows for a simple to compute closed form W_2 metric which is used to replace the W_1 metric in the original formulation: $W_2(\mathcal{N}(\mu_i, \Sigma_i), \mathcal{N}(\mu_j, \Sigma_j))^2 = ||\mu_i - \mu_j||_2^2 + ||\Sigma_i^{1/2} - \Sigma_j^{1/2}||_{\mathcal{F}}^2$, where $||\cdot||_{\mathcal{F}}$ is the Frobenius norm. Fig. 1(c) depicts the overall representation learning process. Finally, for the policy optimization part in step-7, we can use any max entropy policy gradient method. Access to simulator helps us translate a sampled batch from buffer into any randomly sampled contexts from which we can compute the various losses. However, in general this technique can also be extended to non-simulator settings like data augmentation (Laskin et al., 2020a), this could be specially promising as the latter approaches currently only minimize representation distance between two views of same input and not the bisimulation distance which is more aligned with solving the RL task. **Theoretical results for simulator fidelity and performance transfer for the conditional bisimulation setting can be found in** (Mahajan & Zhang, 2023) .

4. Experiments

We perform experiments towards understanding whether our method Robust Conditional Bisimulation (RCB) helps learn representations which generalize better to observation shifts. Towards this, we use the DeepMind control suite (DMC, (Tassa et al., 2018)). We create new tasks for various agent morphologies where we learn to control the agent using image based input. Further, we also modify the simulator to have 3D spheres randomly bouncing in the environment, which contribute towards noise (we call this Modified-DMC). We use two baselines for comparison: (1) DeepMDP (Gelada et al., 2019) which uses reward and forward dynamics predictability for learning a latent representation space. (2) A reconstruction based agent which uses a reward model and an image reconstruction based emission model to inform the representation. We use SAC



(a) Walker Stand (b) Walker Walk (c) Walker Run (d) Cheetah Run

Figure 2: Empirical results on modified-DMC observation generalization tasks for different methods: RCB (our method), DeepMDP, and Reconstruction.

(Haarnoja et al., 2018) as the base algorithm for optimizing the MERL objective in Algorithm 1. The architecture for common modules is kept similar across the methods. For fair comparison, we ensure that all the methods get equal access to the simulator experience and augment the representation learning objective for baselines with any extra simulator calls. **Additional experiments and setup details can be found in Appendix B**.

Modified-DMC: For testing the ability to generalize across observation shifts, we use a uniform distribution over the range $P_\Theta = \mathcal{U}(-\pi/4, \pi/4)$ for the camera angle. At the beginning of each episode, we sample a camera angle context from P_Θ, the agents must adapt to changing image perspectives across training and evaluation. For evaluation, we use a fixed set of camera angles: $\{-\pi/4, -\pi/8, 0, \pi/8, \pi/4\}$ over which we compute the agent performance during the evaluation phase and report the average across the angles as the performance metric. Fig. 2 gives the evaluation performance plots for the agents on five different scenarios averaged over five seeds with one standard error shaded (our method RCB in blue, DeepMDP in green, Reconstruction in red). We see that RCB performs significantly better than the baseline agents on all the scenarios. RCB consistently achieves higher performance across the walker tasks (Figures 2(a) to 2(c)). We also note that the performance for Reconstruction worsens as the task becomes more dynamic, we hypothesize this is due to the lack of focus on the core features of the observation which influence the reward and dynamics. We observe a similar trend on the cheetah domain (Fig. 2(d)) which is slightly easier than walker run. DeepMDP is often unable to perform satisfactorily in the training budget, we posit that this happens as it does not use any inter-context information to inform its representation. Thus, while it may learn close distance embeddings for a fixed context, the embeddings fare poorly across the contexts. RCB alleviates this problem by leveraging both the ICC and cross-consistency objectives in its formulation.

References

Castro, P. S. Scalable methods for computing state similarity in deterministic Markov decision processes. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2020.

Du, S. S., Krishnamurthy, A., Jiang, N., Agarwal, A., Dudík, M., and Langford, J. Provably efficient RL with rich observations via latent state decoding. *Computing Research Repository (CoRR)*, abs/1901.09018, 2019. URL http://arxiv.org/abs/1901.09018.

Ferns, N., Panangaden, P., and Precup, D. Bisimulation metrics for continuous Markov decision processes. *Society for Industrial and Applied Mathematics*, 2011.

Gelada, C., Kumar, S., Buckman, J., Nachum, O., and Bellemare, M. G. DeepMDP: Learning continuous latent space models for representation learning. In *International Conference on Machine Learning (ICML)*, 2019.

Givan, R., Dean, T. L., and Greig, M. Equivalence notions and model minimization in Markov decision processes. *Artificial Intelligence*, 147:163–223, 2003.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv:1801.01290*, 2018.

Hallak, A., Castro, D. D., and Mannor, S. Contextual markov decision processes, 2015.

Krishnamurthy, A., Agarwal, A., and Langford, J. Pac reinforcement learning with rich observations. In *Advances in Neural Information Processing Systems*, pp. 1840–1848, 2016.

Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., and Srinivas, A. Reinforcement learning with augmented data. *Advances in Neural Information Processing Systems*, 33: 19884–19895, 2020a.

Laskin, M., Srinivas, A., and Abbeel, P. CURL: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning (ICML)*, pp. 5639–5650. PMLR, 2020b.

Li, L., Walsh, T. J., and Littman, M. L. Towards a unified theory of state abstraction for MDPs. In *International Symposium on Artificial Intelligence and Mathematics (ISAIM)*, 2006.

Mahajan, A. *Reinforcement learning in large state action spaces*. PhD thesis, University of Oxford, 2023.

Mahajan, A. and Tulabandhula, T. Symmetry detection and exploitation for function approximation in deep rl. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pp. 1619–1621. International Foundation for Autonomous Agents and Multiagent Systems, 2017.

Mahajan, A. and Zhang, A. Generalization across observation shifts in reinforcement learning. *arXiv preprint arXiv:2306.04595*, 2023.

Mahajan, A., Samvelyan, M., Gupta, T., Ellis, B., Sun, M., Rocktäschel, T., and Whiteson, S. Generalization in cooperative multi-agent systems. *arXiv preprint arXiv:2202.00104*, 2022.

Ravindran, B. *An algebraic approach to abstraction in reinforcement learning*. University of Massachusetts Amherst, 2004.

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D. d. L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A., Lillicrap, T., and Riedmiller, M. DeepMind Control Suite. *arXiv:1801.00690 [cs]*, 2018. URL http://arxiv.org/abs/1801.00690.

Zhang, A., McAllister, R., Calandra, R., Gal, Y., and Levine, S. Learning Invariant Representations for Reinforcement Learning without Reconstruction. *arXiv:2006.10742 [cs, stat]*, 2021. URL http://arxiv.org/abs/2006.10742.

A. Discussion on invertability

One of the most straight-forward ways to ensure invertibilty would be through block structure:

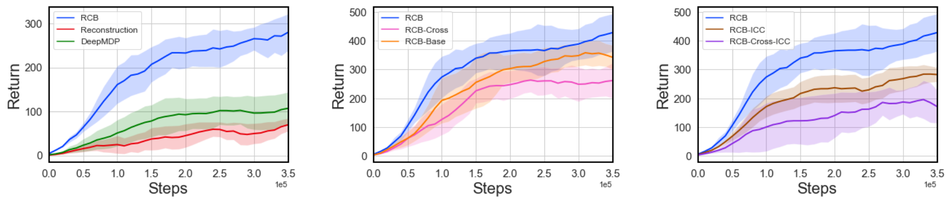
Assumption 1 (Block structure)**.** *We assume that $f_{\theta_1}(s_1) \cap f_{\theta_2}(s_2) \neq \emptyset \implies s_i = s_j, \forall \theta_1, \theta_2$ so that the observation map is invertible.*

This means that the observation space Z can be partitioned into disjoint blocks, each containing the support for a particular value of $s \in S$ (Du et al., 2019). This also ensures that inverse observation map $f_\theta^{-1} : Z \rightarrow S$ exists. Relaxing Assumption 1 can break any guarantees obtainable on value function similarities arising from state similarity. This is because the same observation can get mapped to entirely different states in the latent MDP each with very different values, making the environment only partially observable. Note however that this requirement is not too restrictive, it is possible to consider added noise scenarios (both independent and correlated) which maintain identifiability of the state. Finally, many real-world task observations tend to satisfy this assumption for high dimensional scenarios: e.g. visual projection of non-degenerate objects under different viewing angles.

B. Additional experiments and details

B.1. Additional experiments

Out-of-distribution Generalization: To test the ability of the algorithms in dealing with unseen observation contexts during test time, we train on Modified-DMC where we use a uniform distribution over the range $P_\Theta = \mathcal{U}(-3\pi/16, 3\pi/16)$ for the camera angle and test on the unseen $\{-\pi/4, \pi/4\}$ angles. Fig. 3(a) gives the performance on the unseen angles for walker walk domain across the algorithms. Once again we note that RCB is able to better generalize to the unseen context due to its learning of a more accurate representation space using the inter-context objectives (**ICC** and **CC** terms in Section 3).



(a) Out-of-distribution (P_Θ) (b) Ablation 1 (c) Ablation 2

Figure 3: Out-of-distribution generalization and ablations

Ablations: To understand the effects of the different bisimulation loss components, we perform ablations removing each component. In Fig. 3(b) we remove the base (RCB-Base) and cross consistency (RCB-Cross) bisimulation terms. We see that removing the cross term has a bigger effect on performance. We believe this is because the cross-bisimulation term has a stronger anchoring effect as it also implicitly accounts for both the base and inter-context terms (see Fig. 1(b)). Next, in Fig. 3(c) we remove the inter-context consistency term (RCB-ICC) and both the inter-context consistency and cross consistency terms (RCB-Cross-ICC). We notice a slight decrease in performance arising from dropping the inter context consistency loss. The RCB-Cross-ICC ablation is similar to DBC (Zhang et al., 2021) as it only contains base bisimulation losses (regular and alternate). We observe a significant decrease in performance in this latter ablation as we drop all the inter-context bisimulation terms helpful in generalization across the contexts. Thus it is important to ensure explicit alignment across the representations for the richly observed MDPs defined by different θ context when desiring good generalization across observation shifts.

B.2. Architecture details

We use separate deep networks for actor, critic, transition and reward models. The encoder network for each used 32 filters and a 50 feature dimensions. The actor and critic models each used an MLP trunk of 4 layers and 1024 hidden dimensions on top of the encoder. The reward model used MLP trunk of 2 MLP layers and 512 hidden dimensions on top of the encoder. The transition model type used was a mixture of Gaussians of ensemble size 5. Each component in the transition ensemble uses a 2 MLP layers of 768 hidden dimensions on top of the encoder with the final layer bifurcating for a value for mean and standard deviation per feature dimension. Layer normalization was used for the reward and transition models. Target networks were used for value estimates and were updated every 4 epochs. Relu non-linearity was used for the networks. We exponentially anneal the representation loss with weight $(1.8 - 0.8 * 2^{\frac{\text{steps}}{\text{total steps}}})$. We use identical architectures for the overlapping components of the baselines (Reconstruction and DeepMDP). The reconstruction agent uses an image decoder with an MLP followed by 2 deconvolution layers with the intermediate layer using 32 filters. Adam optimizer was used for training the parameters of the networks used. Grid search was used for tuning the hyperparameters. Our code is based on implementation by (Zhang et al., 2021) for their work. Each seed takes around 4 days to run on an Nvidia V100 GPU.

B.3. Hyper-parameters used: Conditional bisimulation

Table 1: Hyper-parameters used: Conditional bisimulation

PARAMETER	**VALUE**
λ_{base}	0.24
λ_{icc}	0.32
λ_{cc}	0.24
Initial steps	1000
Batch size	512
Action repeat	2
Encoder learning rate	10^{-3}
Encoder τ	$5 \cdot 10^{-3}$
Decoder learning rate	10^{-3}
Frames	1000
Actor learning rate	10^{-3}
Critic learning rate	10^{-3}
Critic τ	10^{-2}
α learning rate	10^{-4}
γ	0.99
Total Steps	$3.5 \cdot 10^{5}$
Temperature	0.1