Equivariant Reinforcement Learning under Partial Observability

Anonymous Author(s)
Affiliation
Address
email

Abstract: Incorporating inductive biases is a promising approach for tackling challenging robot learning domains with sample-efficient solutions. This paper identifies partially observable domains where symmetries can be a useful inductive bias for efficient learning. Specifically, by encoding the equivariance regarding specific group symmetries into the neural networks, our actor-critic reinforcement learning agents can reuse solutions in the past for related scenarios. Consequently, our equivariant agents outperform non-equivariant approaches significantly in terms of sample efficiency and final performance, demonstrated through experiments on a range of robotic tasks in simulation and real hardware.

Keywords: Partial Observability, Equivariant Learning, Symmetry

1 Introduction

A key challenge in robot learning is to improve sample efficiency, i.e., to reduce the number of experiences or demonstrations needed to learn a good policy. One way to do this is to identify domain symmetries that can structure the policy space. Recent works have demonstrated that symmetry-preserving (equivariant) neural network models are a particularly effective way of accomplishing this [1, 2, 3, 4, 5]. However, these works have focused primarily on fully observable Markov decision processes (MDPs) rather than partially observable systems encoded as partially observable MDPs (POMDPs) [6]. The question arises whether symmetric neural models can also be used to solve Partially Observable Reinforcement Learning (PORL) problems. This paper identifies the theoretical conditions under which this is indeed the case and describes an equivariant recurrent model that is shown to work well in practice.

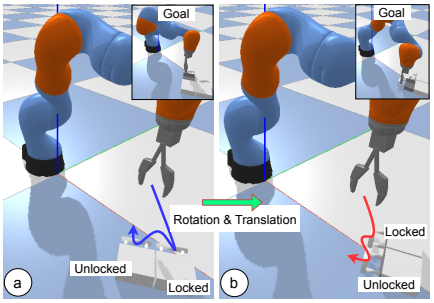


Figure 1: `Drawer-Opening`: This POMDP is rotationally symmetric in the sense that an optimal solution to the problem on the left (in blue) can be rotated to obtain an optimal solution to a rotated version of the problem on the right (in red).

To motivate, Figure 1 illustrates the `Drawer-Opening` problem where a robot is presented with a chest containing two drawers, one locked and one unlocked. To solve this task, the robot must determine which drawer is unlocked and then open that drawer, relying only on top-down image observations. This task reflects a common POMDP when physical properties (whether a drawer is unlocked) are hidden from the visual input. The only way for the robot to distinguish between the two drawers is to attempt to open one of them. This is a classic feature of a POMDP – that the agent must perform *information gathering actions* to obtain information needed to solve the task. Notice that this problem is rotationally symmetric in the sense that its optimal solution (the blue end-effector trajectory in Figure 1a) rotates (the red trajectory in Figure 1b) when the scene itself rotates and is an example of the type of symmetry that we want our agents to embed in their architectures.

We make three contributions in this work. First, we extend the framework of group-invariant Markov decision processes [1] to the partially observable setting, resulting in a new theory and solution

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

method. Specifically, we prove the optimal policy and the value function must be equivariant and invariant functions in this new setting. Second, backed by the proof, we introduce equivariant actor-critic agents that inherently embed the domain symmetry in their architectures. Finally, we apply the agents in realistic robot manipulation tasks with sparse rewards, where our agents are shown to significantly outperform non-equivariant approaches in both sample efficiency and final performance. Our approach's effectiveness is shown through simulated and real-robot experiments with equivariant and recurrent versions of Advantage Actor-Critic (A2C) [7] and Soft Actor-Critic (SAC) [8].

2 Related Works

Learning under Partial Observability Unlike classical planning-based methods [9, 10, 11] that impractically require the complete dynamics of the environment, learning-based methods [12, 13, 14, 15, 16, 17, 18, 19] utilize recurrent versions of common reinforcement learning (RL) algorithms for policy learning by directly interacting with the environment. To speed up learning, some methods leverage privileged information assumed available during training, such as the states, the belief about the environment states, or the fully observable policy [20, 21, 22, 23]. Only a few prior works exploited domain symmetries under partial observability. Kang and Kim [24], Doshi and Roy [25] leveraged the invariance of the value function of some certain POMDPs given a state permutation and experimented on a classical planning-based method [11] with the above limitations. Recently, Muglich et al. [26] used equivariant networks to enforce symmetry when multiple agents coordinate. In contrast to these works, we use model-free RL agents in a single-agent setting.

Equivariant Learning Equivariant networks have been successfully applied to a range of tasks such as point cloud analysis [27] and molecular dynamics [28, 29]. A common approach is to build networks with group equivariant convolutions [30] which are equivariant to arbitrary symmetry groups, such as 2D [31, 32] and 3D [27, 33, 34, 35] transformations. Recently for MDPs, equivariant networks have been applied to robotics [2, 1, 5] and reinforcement learning [36, 37] to improve sample efficiency. Closest to our work is [1], which formalized group-invariant MDPs and used equivariant networks to perform robotic manipulation tasks. In contrast, this work extends equivariant reinforcement learning to partially observable environments, resulting in a new theory and method.

Equivariance v.s. Data Augmentation Both methods leverage known domain symmetry to improve learning but in different ways. On the one hand, data augmentation artificially expands the training data distribution with transformed versions of the data using the symmetry (e.g., rotating, cropping, or translating images [38, 39]); then training a non-equivariant model. On the other hand, an equivariant approach bakes the domain symmetry directly into the model's weights, so an equivariant model can automatically generalize across input transformations even before training. Compared to an equivariant approach, a model trained using data augmentation alone is often less sample efficient [32, 34], generalizes worse [40], and requires a bigger architecture and longer training time for the same performance due to the extra work of learning symmetry injected in the data.

3 Background

Here, we review some background about POMDPs, some specific group theories used in our work, and finally, the basis of our approach — the framework of group-invariant MDPs [1].

3.1 Partially Observable Markov Decision Processes

A POMDP is defined by a tuple $(\mathcal{S}, \mathcal{A}, \Omega, b_0, T, R, O)$, where $\mathcal{S}$, $\mathcal{A}$, and Ω are the state space, the action space, and the observation space, respectively. $b_0 \in \Delta\mathcal{S}$ is the starting state distribution (a.k.a. the initial belief), states change according to the stochastic dynamics function $T(s, a, s')$, and observations are emitted according to the stochastic observation function $O(a, s', o)$. Generally, an optimal agent may need to choose actions based on the entire observable action-observation history $h_t = (o_0, a_0, \ldots, a_{t-1}, o_t)$ [41]. Denoting the space of all histories as $\mathcal{H}$, the goal is to find a history-policy $\pi\colon \mathcal{H} \to \Delta\mathcal{A}$ which maximizes the expected discounted return $J = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$, where $\gamma \in [0, 1]$ is a discounting factor. An important concept in POMDPs is the belief $b(s) = \Pr(s \mid h)$, which is the probability that the true state is s given an observed history h. The belief state is a sufficient statistic of the history, sufficient for optimal control. However, updating the belief state requires complete knowledge of the POMDP dynamic models, which are often hard to obtain.

3.2 C_n and SO(2) Symmetry Groups

In this work, we are mainly concerned with the symmetry group $G = \text{SO}(2)$ of continuous planar rotation, defined as $\text{SO}(2) = \text{Rot}_\theta : \{0 \leq \theta < 2\pi\}$. For a reduced computation complexity, we use the cyclic subgroup $C_n \leq \text{SO}(2)$ to approximate SO(2), which is defined as $C_n = \{\text{Rot}_\theta : \theta \in \{\frac{2\pi i}{n} \mid 0 \leq i < n\}\}$. In other words, C_n defines n rotations (i.e., group elements), which are multiples of $\frac{2\pi}{n}$. For instance, $C_4 = \{0, \pi/2, 2\pi/2, 3\pi/2\}$ and $C_8 = \{0, \pi/8, \ldots, 6\pi/8, 7\pi/8\}$.

3.3 Group Representations

A *group representation* is a mapping from a group G to a d-dimensional general linear (GL) group, i.e., $\rho : G \to \text{GL}_d$ by assigning each group element $g \in G$ with an invertible matrix $\rho(g) \in \mathbb{R}^{d \times d}$.

When $G = C_n$, the effect of a rotation $g \in C_n$ on a signal x (i.e., gx) starts with a *fiber-wise* rotation $\rho_f(g)^{-1}x$ (with a *fixed* group representation ρ_f), followed by a *channel-wise* rotation, i.e., $gx = \rho(g)(\rho_f(g)^{-1}x)$ (with the choice of group representation ρ). In this work, we consider three choices of the channel-wise representation ρ:

Trivial Representation ($\rho = \rho_t$): For $\forall g \in G$, ρ_t associates g with an identity matrix. For example, in Figure 2a when g is a $\pi/2$ counter-clockwise (CCW) rotation and x is a 1-channel feature map, p_f rotates the pixels of x while ρ_t does not change the pixel values (the colors are unchanged).

Standard Representation ($\rho = \rho_s$): For $\forall g \in G$, ρ_s associates g with a rotational matrix, i.e., $\rho_s(g) = g$. As in Figure 2b, when g is a $\pi/2$ CCW rotation and x is a vector field input, ρ_f rotates the positions of vectors (denoted as colored arrows), and ρ_s rotates their orientations.

Regular Representation ($\rho = \rho_r$): For each $g \in G$, when acting on an input x, p_r will cyclically permute the coordinates of x. Figure 2c illustrates when g is a $\pi/2$ CCW rotation and x is a 2-channel feature map, ρ_f rotates each channel's pixels and ρ_r permutes the orders of the channels.

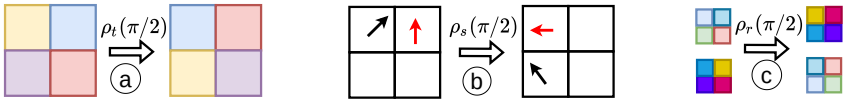


Figure 2: Illustration of a fiber-wise rotation (characterized by a *fixed* representation ρ_f) and a channel-wise rotation (characterized by the representation ρ). When g is a $\pi/2$ CCW rotation, ρ_f *always* rotates the pixels while the effect of ρ varies, e.g., the effect when ρ: **(a)** being a trivial representation (ρ_t) acting on a 1-channel feature map, **(b)** being a standard representation (ρ_s) acting on a vector field, and **(c)** being a regular representation (ρ_r) acting on a 2-channel feature map.

An Illustrative Example Combining the group and the group representation fully characterizes how a signal will be transformed. For an illustrative example in a grid-world domain, see Appendix A.

3.4 Equivariance, Invariance, and Group-invariant MDPs

Given $\phi\colon \mathcal{X} \to \mathcal{Y}$ and a symmetric group G that acts on $\mathcal{X}$ and $\mathcal{Y}$, we say that ϕ is *G-equivariant* if $\phi(gx) = g\phi(x)$, and *G-invariant* if $\phi(gx) = \phi(x)$. For the remainder of this document, we drop the prefix G and simply refer to these properties as invariance and equivariance.

These notions have been adopted in the framework of group-invariant MDPs [1]. Specifically, an MDP $M_G = (\mathcal{S}, \mathcal{A}, T, R)$ is invariant if the transition and the reward function are invariant, i.e., $T(gs, ga, gs') = T(s, a, s')$ and $R(gs, ga) = R(s, a)$. Group-invariant MDPs are associated with an invariant optimal Q-function, i.e., $Q^*(gs, ga) = Q^*(s, a)$, and at least one equivariant deterministic optimal policy, i.e., $\pi^*(gs) = g\pi^*(s)$. These properties were exploited to build very sample-efficient equivariant agents under full observability [2, 1, 5, 3].

4 Group-Invariant POMDPs

In this section, we extend the ideas from [1] to POMDPs and identify the basic set of assumptions that a POMDP needs to satisfy in order to have analogous invariance properties. We also note that while other assumptions might also lead to an invariant POMDP, ours are probably the most natural.

Definition 1. *We say a POMDP $P_G = (\mathcal{S}, \mathcal{A}, \Omega, b_0, T, R, O)$ is group-invariant with respect to group G if it satisfies the following invariant properties for all $g \in G$:*

$$\begin{aligned} T(gs, ga, gs') &= T(s, a, s') \quad R(gs, ga) = R(s, a) \\ O(ga, gs', go) &= O(a, s', o) \quad b_0(gs) = b_0(s)\,. \end{aligned} \tag{1}$$

This extends the definition of the group invariant MDP from [1] by incorporating additional constraints on the observation function and the initial belief distribution. We also extend the group operations on histories.

Definition 2. *Group operation g acts on history h_t according to $gh_t := (go_0, ga_0, \ldots, ga_{t-1}, go_t)$.*

Finally, we show that group-invariant POMDPs exhibit similar properties and benefits as group-invariant MDPs.

Theorem 1. *A group-invariant POMDP has an invariant optimal Q-function $Q^*(gh, ga) = Q^*(h, a)$, an invariant optimal value function $V^*(gh) = V^*(h)$, and at least one equivariant deterministic optimal policy $\pi^*(gh) = g\pi^*(h)$.*

Proof. See Appendix B. □

The above analysis allows us to constrain the value function and policy for a G-invariant POMDP to be invariant and equivariant, respectively, without eliminating optimal solutions.

5 Equivariant Actor-Critic RL for POMDPs

In this section, we introduce an equivariant agent that directly exhibits the desired properties of the optimal value function and policy, backed by the analysis in Theorem 1. Figure 3 shows the agent, which takes a typical memory-based agent [13, 42, 43, 44] but has an equivariant actor and an invariant critic, each consisting of equivariant models. We later show that this very general agent, when embedded with the domain symmetry, can outperform significantly strong POMDP methods.

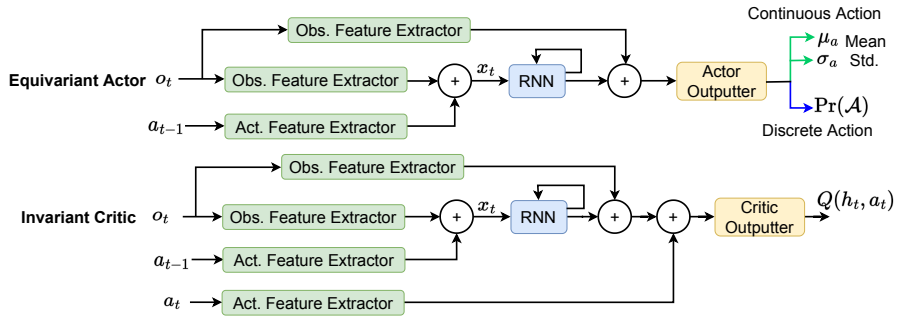


Figure 3: Our equivariant agent takes the general structure of a memory-based actor-critic agent [13, 42, 43, 44] but consists of an equivariant actor and an invariant critic, each constructed by equivariant modules. The actor's output can be learned means and standard deviations (for continuous action spaces) or a categorical distribution over the action space (for discrete action spaces).

5.1 Equivariant Modules

We describe the details of the equivariant modules within our agent below, with the core components being *equivariant CNNs* [45, 32]. For the implementation details, please see Appendix D.

Equivariant Feature Extractor This module takes observations or actions and outputs intermediate features for further processing. It comprises multiple equivariant CNN components chained sequentially as shown in Figure 4a. Its input representation is the observation representation ρ_o or the action representation ρ_a. The intermediate and output representations are chosen to be the regular representation ρ_r, which empirically outperforms other representations [32]. The input representation can

be a single representation type or *mixed*, i.e., a sum of different representations. A mixed representation is necessary when the input has different components that transform differently under a group transformation, e.g., one component rotates with the transform, and one component is unchanged. Such a case can be seen in Appendix D.2 and is simplified in Figure 4b, where $\rho_a = \rho_s + \rho_t$.

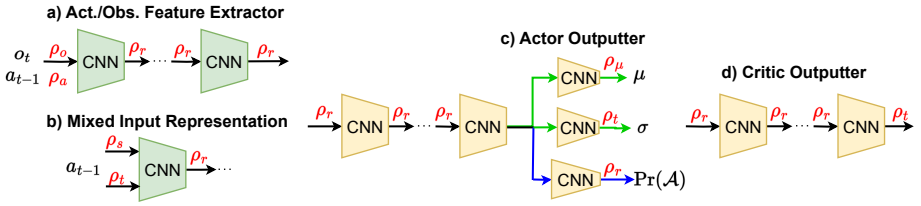


Figure 4: Equivariant Feature Extractor and Actor/Critic Outputter modules.

Equivariant Actor Outputter Figure 4c shows the input representation is regular because the signal coming in (the output of the RNN and the observation feature extractor modules) uses a regular representation. The output representation varies depending on the action type (discrete/continuous) and how a group transformation will affect an action. For *discrete actions*, the module produces a categorical distribution over the action space. In this case, the output representation is the regular one ρ_r as we want to change the (discrete) action when the history is transformed (see Appendix A for an illustration). For *continuous actions*, this module outputs the means with some representation ρ_μ and the standard deviations of actions with some representation ρ_σ (as in A2C [7], PPO [46], or SAC [8]). The representations used for ρ_μ and ρ_σ are mixed, as each action component might change differently under a group transformation (see Appendix D.2).

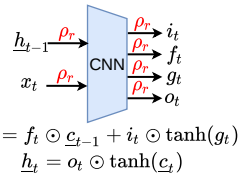


Figure 5: Equivariant LSTM cell.

Equivariant Critic Outputter Because the optimal critic is invariant, this module (Figure 4d) uses the trivial representation ρ_t at the output to keep the output the same under a group transformation. Its input representation is regular, enforced by the output of the RNN and the action feature extractor.

Equivariant Recurrent Neural Network This is our contribution needed for constructing a POMDP equivariant agent (there is another similar component in [26], but only model weights are released). We utilize an LSTM [47] for this module, but the approach can also be modified for other types of RNNs. Specifically, given an input x_t (e.g., the concatenated obs-action feature) and the previous hidden state $\underline{h}_{t-1}$, the input gate i_t, the forget gate f_t, the memory cell candidate g_t, and the output o_t are computed as follows with Ws and bs being learnable weights and biases:

$$\begin{aligned} i_t &= \text{sigmoid}(W_{xi}x_t + W_{hi}\underline{h}_{t-1} + b_i) \quad f_t = \text{sigmoid}(W_{xf}x_t + W_{hf}\underline{h}_{t-1} + b_f) \\ o_t &= \text{sigmoid}(W_{xo}x_t + W_{ho}\underline{h}_{t-1} + b_o) \quad g_t = \tanh(W_{xg}x_t + W_{hg}\underline{h}_{t-1} + b_g)\,. \end{aligned} \tag{2}$$

To implement the above four equations to construct an equivariant module, we compute all at once using an equivariant CNN module (Figure 5), similar to the ConvLSTM network [48]. The input representation ρ_r is determined by the output of the feature extractors (x_t) and the previous hidden state ($\underline{h}_{t-1}$), and the output representation is also regular. Next, we compute the next hidden state $\underline{h}_t$ and cell state $\underline{c}_t$ using the common LSTM equations with $\odot$ denoting the Hadamard product:

$$\underline{c}_t = f_t \odot \underline{c}_{t-1} + i_t \odot \tanh(g_t) \quad \underline{h}_t = o_t \odot \tanh(\underline{c}_t)\,. \tag{3}$$

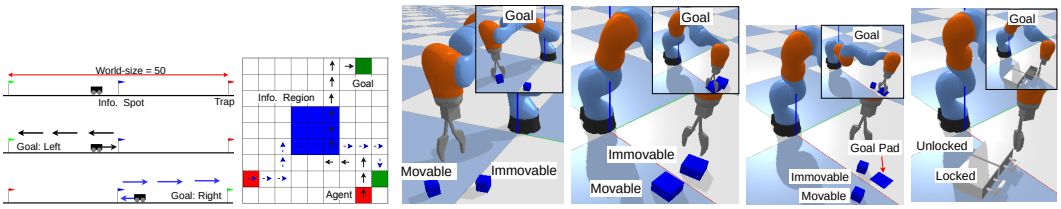
Finally, as the output of the RNN is an approximation of the belief state, to satisfy the condition of an invariant initial belief distribution, we set $\underline{c}_0$ and $\underline{h}_0$ with zero vectors.

6 Experiments

We compare the performance of learning agents on two grid-world domains (discrete actions and feature-based observations) and four robot domains (continuous actions and pixel observations).

6.1 Domains

We briefly describe our domains below. Please refer to Appendix C for more specific details.



`CarFlag-1D` `CarFlag-2D` `Block-Picking` `Block-Pulling` `Block-Pushing` `Drawer-Opening`

Figure 6: Our domains. The first two domains have feature-based observation and discrete action spaces. The last four domains have pixel-based observations and continuous action spaces.

6.1.1 Grid-World Domains

There are two versions of `CarFlag` [49] in Figure 6, where an agent must reach a goal (green), whose position is visible *only* when the agent visits an unknown information region. For instance, in `CarFlag-1D`, the agent must visit the central blue flag to get the side (left/right) of the goal; or in `CarFlag-2D`, the agent must visit the central blue region to see the coordinate of the goal cell. We also illustrate the domain symmetry in the figure: in these domains, when the starting position and the goal location are transformed (flipped in `CarFlag-1D` or rotated by $\pi/2$ radians clockwise in `CarFlag-2D`, the optimal trajectories will be transformed similarly, i.e., black $\rightarrow$ blue trajectories.

6.1.2 Robot Manipulation Domains

Figure 6 shows our robot manipulation domains (extended from the BulletArm suite [50]), where a robot arm must perform individual manipulation tasks (i.e., picking, pulling, pushing, and opening) using *top-down* depth images to win a sparse reward. In these domains, only one object is manipulable, but both objects are the same if just relying on the current image. Therefore, the agent must actively check the objects' mobility and remember past interactions with the objects to determine the next action. Specifically, in `Block-Picking`, the agent needs to pick the movable block up. In `Block-Pulling`, the agent needs to pull the movable block to be in contact with the other block. In `Block-Pushing`, the goal is to push the movable block to a goal pad. In `Drawer-Opening`, the agent is tasked to open an unlocked drawer between a locked and an unlocked one.

In these domains, the transition function is invariant because the Newtonian physics applied to the interaction is invariant to the location of the reference frame. The reward function is invariant by definition. Using top-down depth images makes the observation function invariant. If the initial belief is assumed invariant, then according to Definition 1, these domains are group-invariant POMDPs.

6.2 Agents

We compare our proposed agents (instances of the structure in Figure 3 applied to A2C [7] and SAC [8]) against a diverse set of baselines, including on-policy/off-policy, model-based/model-free, and general/specialized POMDP methods (see Appendix D and Appendix E for more details).

6.2.1 Grid-world Domains

RA2C [51] is a recurrent version of A2C [7]. **Equi-RA2C** is our proposed architecture applied to A2C. **DPFRL** [14] is a state-of-the-art model-based POMDP baseline where an A2C agent is given features produced by a differentiable particle filter. **DreamerV2** [52] and **DreamerV3** [53] are strong model-based methods that learn a recurrent world model, thus, can work with POMDPs.

No Data Augmentation for All Since all methods are on-policy algorithms, we do not perform any data augmentation.

6.2.2 Robot Manipulation Domains

While on-policy RA2C or Dreamer-v2 can handle continuous action spaces in these domains, there is no clear way to leverage expert demonstrations necessary to efficiently solve the robot manipulation tasks with sparse rewards in Figure 6. Thus, we switch to SAC [8] as the base RL algorithm, where we can pre-populate its replay buffer with demonstration episodes. In our experiments,

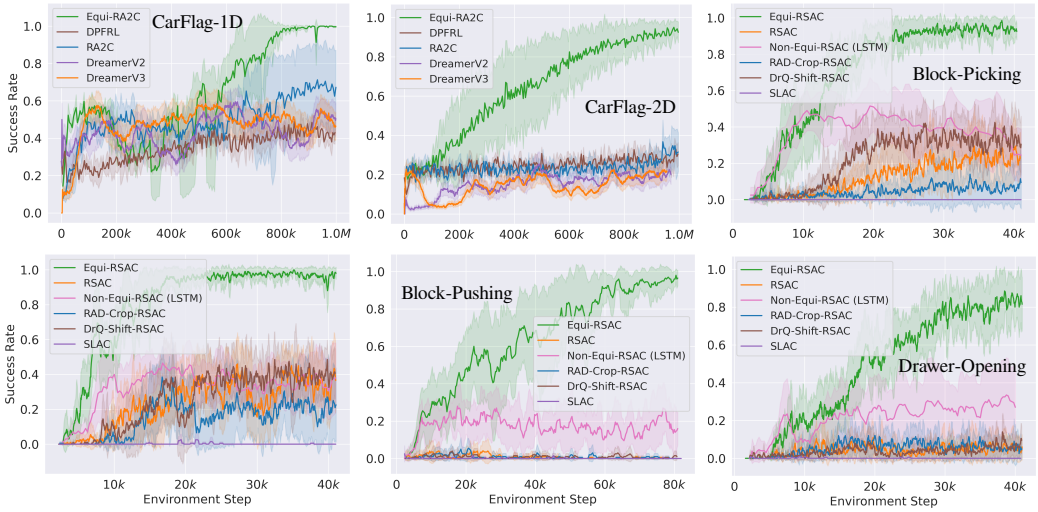


Figure 7: Evaluated success rates (four seeds, shaded areas denote one std.). No data augmentation is used in `CarFlag` domains. Rotational data augmentation is used for *all* agents in robot domains.

RSAC [13] is a non-equivariant recurrent SAC agent. **Equi-RSAC** is our proposed method applied to SAC. We also compare with *recurrent* versions of two strong data augmentation baselines: **RAD-Crop-RSAC** [54] and **DrQ-Shift-RSAC** [55]. These specific data augmentation techniques, i.e., random cropping and shifting (see Appendix G for visualizations), are chosen among others because they were reported to perform best [55]. To train RAD-Crop-RSAC, for each training episode, an auxiliary episode is created by using the same random cropping for every depth image inside the original episode. DrQ-Shift-RSAC applies two random shifts to each depth image in a training episode to create two. The Q-target and the Q-values are then computed by averaging the values computed on the two episodes. Finally, **SLAC** [56] learns a latent model from pixels and then uses SAC on the latent space by using the observation-action history (instead of the latent state) for the actor and the latent state samples for the critic. This enables SLAC to scale to more difficult tasks.

Demonstrations + Rotational Data Augmentation for All All replay buffers are pre-populated with 80 expert episodes to overcome the reward sparsity. Moreover, we augment the training data by applying the same random rotation for every action and observation inside a training episode (see Appendix G for visualizations). Note that these rotational data augmentation is in addition to random cropping and random shifting, already in RAD-Crop-RSAC and DrQ-Shift-RSAC.

6.3 Results

Grid-world Domains Figure 7 shows that Equi-RA2C is significantly more sample efficient than the baselines. Moreover, the dominance of our method is also seen with variants of these domains with different sizes (see Appendix I). DPFRL did not perform well potentially because of the reward sparsity, which was also previously reported in [57]. DreamerV2 and DreamerV3 also perform poorly, even with many more learnable parameters of the models, potentially indicating that learning a good model under partial observability and sparse rewards might be more challenging than in the domains that they were originally tested on. For instance, most Atari games and locomotion tasks in the DeepMind Control suite [58] have low levels of partial observability and provide dense rewards.

Robot Manipulation Domains Clearly from Figure 7, Equi-RSAC strongly outperforms other baselines in all domains, with itself being the only agent that can reach a satisfactory performance. Across all domains, without the equivariant LSTM module (denoted as Non-Equi-RSAC (LSTM), which can be considered as a naive extension of [1]), the performance degrades significantly, even though it starts pretty well. SLAC, surprisingly, performs the worst. A possible reason is that SLAC is originally only tested on domains with dense rewards and low levels of partial observability (e.g., locomotion domains in DeepMind Control Suite [58] and OpenAI Gym [59]). Another potential reason is the usage of concatenated feature vectors across an episode for the actor, which can be very

high-dimensional for a long episode. Moreover, we also found that the trained latent model failed to sufficiently reconstruct the observation in `Block-Pulling` (see Appendix F for more details).

Additional Results See Appendix H for the performance when using a different group symmetry (C_8 instead of C_4), utilizing symmetry partially (for either actor or critic only), and $\underline{c}_0$ and $\underline{h}_0$ being random instead of zero vectors. Other additional results are shown in Appendix I.

6.4 Using Equivariant Models on Domains with Imperfect Symmetry

We investigate the performance when the perfect symmetry does not hold in asymmetric variants of `CarFlag`, created by offsetting the information region a distance d from the world center (see Appendix C). From the final success rates shown in Table 1 (see Appendix I.1 for learning curves), we can see that equivariant Equi-RA2C still outperforms non-equivariant RA2C when the domains are close to perfect symmetry, i.e., when $d = d_{1D} = 5$ in `CarFlag-1D` or $d = d_{2D} = 1, 2$ in `CarFlag-2D`. However, a bigger symmetry gap might lead to the sub-optimality of equivariant agents. As evidence, Equi-RA2C performs worse than RA2C when $d = 10, -10$ in `CarFlag-1D`.

d_{1D}	**Equi-RA2C 1M (↑)**	**RA2C 1M (↑)**	d_{2D}	**Equi-RA2C 2M (↑)**	**RA2C 2M (↑)**
-10	0.51 ± 0.06	$\mathbf{0.78} \pm 0.08$	-2	$\mathbf{0.38} \pm 0.11$	$\mathbf{0.31} \pm 0.04$
10	0.44 ± 0.07	$\mathbf{0.70} \pm 0.22$	2	$\mathbf{0.41} \pm 0.12$	0.28 ± 0.02
-5	$\mathbf{0.95} \pm 0.03$	0.72 ± 0.33	-1	$\mathbf{0.58} \pm 0.24$	0.38 ± 0.13
5	$\mathbf{0.99} \pm 0.03$	0.76 ± 0.24	1	$\mathbf{0.71} \pm 0.14$	0.30 ± 0.03

Table 1: The convergent success rates (mean $\pm$ one standard deviation) of Equi-RA2C and RA2C agents in asymmetric variants of `CarFlag` (after 1M and 2M training steps). d_{1D} and d_{2D} refer to the distance from the information region to the world center (see Appendix C for illustrations).

6.5 Zero-shot Transfers to Real Hardware

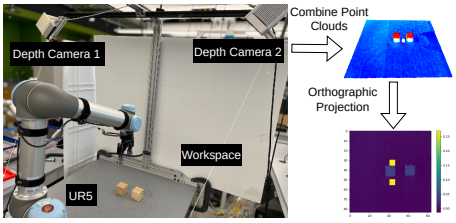


Figure 8: Experimental robot setup.

Domain	**Success Sim / Real (↑)**
`Block-Picking`	1.00 / 0.90
`Block-Pulling`	1.00 / 0.88
`Block-Pushing`	0.96 / 0.92
`Drawer-Opening`	0.95 / 0.80

Table 2: Average success rates of sim2real transfers over 50 episodes.

Because only our agents can perform well in simulation, we transfer their best policies in simulation to a UR5 robot (see Figure 8). We combine the point clouds from two side-view cameras to create a top-down depth image using a projection at the gripper's position. We roll out 50 episodes, divided equally to test cases when the agents first manipulate the immovable or movable objects. Table 2 shows that the learned policies can be zero-shot transferred well in the real world regardless of small performance drops in all domains (see our supplementary video for policy visualizations). The biggest performance drop is in `Draw-Opening`, in which the transferred policies sometimes clumsily move one drawer far away from the other, creating a novel scene never seen in simulation.

7 Conclusion and Limitations

Conclusion In this work, we introduced group-invariant POMDPs and proposed equivariant actor-critic RL agents as an effective solution method. Through extensive experiments, our proposed equivariant agents can tackle realistic and challenging robotic manipulation domains much better than non-equivariant approaches with learned policies zero-shot transferable to a real robot.

Limitations A limitation of most equivariant approaches, including ours, is the requirement of imperfect symmetry, which might be present when images are affected by non-symmetric factors, e.g., side view instead of top-down view or asymmetric noises. Fortunately, under full observability, recent empirical [60, 61] and theory work [62] show that an equivariant model can still outperform non-equivariant approaches in many such cases. Together with the results in Section 6.4, our approach might still perform better than unstructured agents even under imperfect symmetry.

References

[1] D. Wang, R. Walters, and R. Platt. SO(2) equivariant reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2022.

[2] D. Wang, R. Walters, X. Zhu, and R. Platt. Equivariant Q learning in spatial action spaces. In *Conference on Robot Learning*, pages 1713–1723. PMLR, 2022.

[3] D. Wang, M. Jia, X. Zhu, R. Walters, and R. Platt. On-robot learning with equivariant models. In *Conference On Robot Learning*, 2022.

[4] H. Huang, D. Wang, R. Walters, and R. Platt. Equivariant Transporter Network. In *Proceedings of Robotics: Science and Systems*, 2022.

[5] X. Zhu, D. Wang, O. Biza, G. Su, R. Walters, and R. Platt. Sample efficient grasp learning using equivariant models. In *Proceedings of Robotics: Science and Systems*, 2022.

[6] K. J. Astrom. Optimal control of markov decision processes with incomplete state estimation. *J. Math. Anal. Applic.*, 10:174–205, 1965.

[7] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937. PMLR, 2016.

[8] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870. PMLR, 2018.

[9] H. Kurniawati, D. Hsu, and W. S. Lee. Sarsop: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Robotics: Science and systems*, 2008.

[10] A. Somani, N. Ye, D. Hsu, and W. S. Lee. Despot: Online POMDP planning with regularization. *Advances in neural information processing systems*, 26, 2013.

[11] J. Pineau, G. Gordon, S. Thrun, et al. Point-based value iteration: An anytime algorithm for POMDPs. In *Ijcai*, volume 3, pages 1025–1032, 2003.

[12] M. Hausknecht and P. Stone. Deep recurrent Q-learning for partially observable MDPs. In *2015 aaai fall symposium series*, 2015.

[13] T. Ni, B. Eysenbach, and R. Salakhutdinov. Recurrent model-free RL can be a strong baseline for many POMDPs. In *International Conference on Machine Learning*, pages 16691–16723. PMLR, 2022.

[14] X. Ma, P. Karkus, D. Hsu, W. S. Lee, and N. Ye. Discriminative particle filter reinforcement learning for complex partial observations. In *International Conference on Learning Representations (ICLR)*, 2020.

[15] D. Han, K. Doya, and J. Tani. Variational recurrent models for solving partially observable control tasks. In *International Conference on Learning Representations (ICLR)*, 2020.

[16] Z. Yang and H. Nguyen. Recurrent off-policy baselines for memory-based continuous control. In *Deep RL Workshop NeurIPS 2021*, 2021.

[17] N. Heess, J. J. Hunt, T. P. Lillicrap, and D. Silver. Memory-based control with recurrent neural networks. *arXiv preprint arXiv:1512.04455*, 2015.

[18] M. Igl, L. Zintgraf, T. A. Le, F. Wood, and S. Whiteson. Deep variational reinforcement learning for POMDPs. In *International Conference on Machine Learning*, pages 2117–2126. PMLR, 2018.

[19] L. Meng, R. Gorbet, and D. Kulić. Memory-based deep reinforcement learning for POMDPs. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5619–5626. IEEE, 2021.

[20] H. Nguyen, B. Daley, X. Song, C. Amato, and R. Platt. Belief-grounded networks for accelerated robot learning under partial observability. In *Proceedings of the 2020 Conference on Robot Learning*, volume 155, pages 1640–1653. PMLR, 2021.

[21] H. Nguyen, A. Baisero, D. Wang, C. Amato, and R. Platt. Leveraging fully observable policies for learning under partial observability. In *6th Annual Conference on Robot Learning*, 2022.

[22] A. Baisero, B. Daley, and C. Amato. Asymmetric dqn for partially observable reinforcement learning. In *The 38th Conference on Uncertainty in Artificial Intelligence*, 2022.

[23] A. Baisero and C. Amato. Unbiased asymmetric reinforcement learning under partial observability. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pages 44–52, 2022.

[24] B. K. Kang and K.-E. Kim. Exploiting symmetries for single-and multi-agent partially observable stochastic domains. *Artificial Intelligence*, 182:32–57, 2012.

[25] F. Doshi and N. Roy. The permutable POMDP: fast solutions to POMDPs for preference elicitation. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*, pages 493–500, 2008.

[26] D. Muglich, C. Schroeder de Witt, E. van der Pol, S. Whiteson, and J. Foerster. Equivariant networks for zero-shot coordination. *Advances in Neural Information Processing Systems*, 35: 6410–6423, 2022.

[27] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

[28] V. G. Satorras, E. Hoogeboom, and M. Welling. E (n) equivariant graph neural networks. In *International Conference on Machine Learning*, pages 9323–9332. PMLR, 2021.

[29] S. Batzner, A. Musaelian, L. Sun, M. Geiger, J. P. Mailoa, M. Kornbluth, N. Molinari, T. E. Smidt, and B. Kozinsky. E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature communications*, 13(1):1–11, 2022.

[30] T. Cohen and M. Welling. Group equivariant convolutional networks. In *International Conference on Machine Learning*, pages 2990–2999. PMLR, 2016.

[31] Y. LeCun, Y. Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[32] M. Weiler and G. Cesa. General e (2)-equivariant steerable cnns. *Advances in Neural Information Processing Systems*, 32, 2019.

[33] H. Chen, S. Liu, W. Chen, H. Li, and R. Hill. Equivariant point network for 3d point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14514–14523, 2021.

[34] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling. Spherical cnns. *arXiv preprint arXiv:1801.10130*, 2018.

[35] C. Deng, O. Litany, Y. Duan, A. Poulenard, A. Tagliasacchi, and L. J. Guibas. Vector neurons: A general framework for so (3)-equivariant networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12200–12209, 2021.

[36] A. K. Mondal, P. Nair, and K. Siddiqi. Group equivariant deep reinforcement learning. *arXiv preprint arXiv:2007.03437*, 2020.

[37] E. van der Pol, D. Worrall, H. van Hoof, F. Oliehoek, and M. Welling. MDP homomorphic networks: Group symmetries in reinforcement learning. *Advances in Neural Information Processing Systems*, 33:4199–4210, 2020.

[38] M. Laskin, A. Srinivas, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, pages 5639–5650. PMLR, 2020.

[39] A. Zhan, R. Zhao, L. Pinto, P. Abbeel, and M. Laskin. A framework for efficient robotic manipulation. In *Deep RL Workshop NeurIPS 2021*, 2021.

[40] R. Wang, R. Walters, and R. Yu. Incorporating symmetry into deep dynamics models for improved generalization. In *International Conference on Learning Representations (ICLR)*, 2021.

[41] S. P. Singh, T. Jaakkola, and M. I. Jordan. Learning without state-estimation in partially observable markovian decision processes. In *Machine Learning Proceedings 1994*, pages 284–292. Elsevier, 1994.

[42] D. Ha and J. Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[43] L. Zintgraf, K. Shiarlis, M. Igl, S. Schulze, Y. Gal, K. Hofmann, and S. Whiteson. Varibad: a very good method for bayes-adaptive deep rl via meta-learning. In *International Conference on Learning Representations (ICLR)*, 2020.

[44] C.-C. Hung, T. Lillicrap, J. Abramson, Y. Wu, M. Mirza, F. Carnevale, A. Ahuja, and G. Wayne. Optimizing agent behavior over long time scales by transporting value. *Nature communications*, 10(1):1–12, 2019.

[45] M. Weiler and G. Cesa. General E(2)-Equivariant Steerable CNNs. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[46] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[47] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[48] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015.

[49] H. Nguyen. POMDP robot domains. https://github.com/hai-h-nguyen/pomdp-domains, 2021.

[50] D. Wang, C. Kohler, X. Zhu, M. Jia, and R. Platt. Bulletarm: An open-source robotic manipulation benchmark and learning framework. In *The International Symposium of Robotics Research*, 2022.

[51] I. Kostrikov. Pytorch implementations of reinforcement learning algorithms. https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail, 2018.

[52] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations (ICLR)*, 2021.

[53] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

[54] M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas. Reinforcement learning with augmented data. *Advances in neural information processing systems*, 33:19884–19895, 2020.

[55] I. Kostrikov, D. Yarats, and R. Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations (ICLR)*, 2021.

[56] A. X. Lee, A. Nagabandi, P. Abbeel, and S. Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *Advances in Neural Information Processing Systems*, 33:741–752, 2020.

[57] H. Nguyen, Z. Yang, A. Baisero, X. Ma, R. Platt, and C. Amato. Hierarchical reinforcement learning under mixed observability. In *Algorithmic Foundations of Robotics XV: Proceedings of the Fifteenth Workshop on the Algorithmic Foundations of Robotics*, pages 188–204. Springer, 2022.

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

[59] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[60] D. Wang, J. Y. Park, N. Sortur, L. L. Wong, R. Walters, and R. Platt. The surprising effectiveness of equivariant models in domains with latent symmetry. *arXiv preprint arXiv:2211.09231*, 2022.

[61] R. Yang, G. Yang, and X. Wang. Neural volumetric memory for visual locomotion control. *arXiv preprint arXiv:2304.01201*, 2023.

[62] D. Wang, X. Zhu, J. Y. Park, R. Platt, and R. Walters. A general theory of correct, incorrect, and extrinsic equivariance. *arXiv preprint arXiv:2303.04745*, 2023.

[63] K. X. Nguyen. Converting POMDPs into MDPs using history representation. 2021.

[64] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

[65] G. Cesa, L. Lang, and M. Weiler. A program to build E(N)-equivariant steerable CNNs. In *International Conference on Learning Representations (ICLR)*, 2022.

[66] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

[67] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.