2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Equivariant Ensembles and Regularization for Reinforcement Learning in Map-based Path Planning

Mirco Theile[1], Hongpeng Cao[1], Marco Caccamo[1], and Alberto L. Sangiovanni-Vincentelli[2]

***Abstract*— In reinforcement learning (RL), exploiting environmental symmetries can significantly enhance efficiency, robustness, and performance. However, ensuring that the deep RL policy and value networks are respectively equivariant and invariant to exploit these symmetries is a substantial challenge. Related works try to design networks that are equivariant and invariant by construction, limiting them to a very restricted library of components, which in turn hampers the expressiveness of the networks. This paper proposes a method to construct equivariant policies and invariant value functions without specialized neural network components, which we term equivariant ensembles. We further add a regularization term for adding inductive bias during training. In a map-based path planning case study, we show how equivariant ensembles and regularization benefit sample efficiency and performance.**

I. Introduction

Reinforcement learning (RL) is a rapidly advancing methodology for learning policies through interactions with environments, as it promises to address complex real-world problems that were previously unsolvable. Its application breadth is continuously widening with applications in control, planning, and general optimization domains. Many of these environments contain symmetries that could be exploited for improved training efficiency, robustness, and performance. Symmetries in the environments result in equivariance (see Figure 1 for an example) and invariance properties of the optimal policy and value function, respectively. Therefore, imposing these properties on the learning components should be beneficial.

Inspired by the success of convolutional neural networks that make neural networks equivariant to translations [1], earlier research focused on designing neural networks such that they are equivariant or invariant to the symmetry transformations [2], [3]. However, this endeavor can be daunting as it limits the neural network components to only small subsets of equivariant and invariant layers, opposing the trend toward ever-increasing complexity of neural networks for performance improvements. Recently, related research has shifted towards adding a regularization term to the training loss that nudges the networks towards equivariance and invariance without constraining the design choices for the networks [4], [5].

Marco Caccamo was supported by an Alexander von Humboldt Professorship endowed by the German Federal Ministry of Education and Research.

[1]Mirco Theile, Hongpeng Cao, and Marco Caccamo are with TUM School of Engineering and Design, Technical University of Munich, Germany mirco.theile,cao.hongpeng,mcaccamo@tum.de

[2]Alberto L. Sangiovanni-Vincentelli is with Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley, USA alberto@berkeley.edu

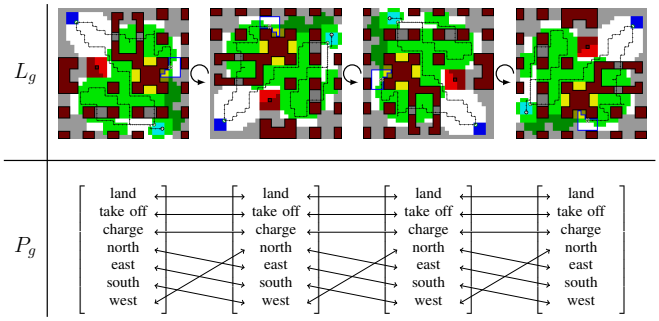


Fig. 1: Visualization of the equivariances in a UAV coverage path planning application showing the input and output transformations L_g and P_g for all rotations in G.

In this paper, we incorporate symmetries into the learning process by constructing equivariant policies and invariant value functions without the need for special neural network designs. We introduce *equivariant ensembles* that average over the networks' outputs for all symmetry transformations. We prove that policy and value ensembles are equivariant and invariant, respectively, and show how they enrich the gradients in policy optimization algorithms such as proximal policy optimization (PPO) [6]. We further use regularization to push the individual components toward the ensembles, adding inductive bias.

To showcase the benefits of the equivariant ensembles and regularization, we evaluate their performance in a challenging, long-horizon, map-based planning application, the unmanned aerial vehicle (UAV) Coverage Path Planning (CPP) problem [7]. In this case study, the environment state can be represented as a map ([8]), and rotational symmetries can be exploited, as visualized in Figure 1. The results show that the ensemble makes the policy equivariant and that combining the ensemble and regularization improves performance significantly. We further show that regularization on the policy does not guarantee equivariance, which should be considered when regularizing the value estimate towards invariance.

To summarize, the contributions of this paper are as follows:

- Introduction of equivariant ensembles to enforce equivariance and invariance of policies and value functions without special neural network designs;
- Combination of equivariant ensembles and regularization to enrich the gradients in policy optimization algorithms through implicit data augmentation and providing inductive bias;

- Implementation[1] of ensembles and regularization in the long-horizon problem of UAV coverage path planning;
- Analyzing the effects of ensembles and regularization on sample efficiency, performance, and out-of-distribution generalization.

II. Preliminaries

A. Invariance and Equivariance

A function $f : \mathcal{X} \to \mathcal{Y}$ is defined to be *invariant* to transformations from a group of linear transformations G if for all $g \in G$ and their corresponding transformation operator $L_g : \mathcal{X} \to \mathcal{X}$ the following equality holds:

$$f(x) = f(L_g[x]), \quad \forall g \in G, x \in \mathcal{X}. \tag{1}$$

In this case, the function output does not change when applying the transformations on the function domain. The group G is characterized by containing the identity, the inverse of each element, and the compositions of all pairs of elements, i.e., it is closed.

Adding the corresponding linear transformation operators $K_g : \mathcal{Y} \to \mathcal{Y}$ for all $g \in G$, the function f is *equivariant* to the transformations of G if

$$K_g[f(x)] = f(L_g[x]), \quad \forall g \in G, x \in \mathcal{X} \tag{2}$$

Consider a stochastic function $p : \mathcal{X} \to \mathcal{P}(\mathcal{Y})$ that maps $\mathcal{X}$ to the space of all probability distributions over $\mathcal{Y}$. We define the transformation operator $P_g : \mathcal{P}(\mathcal{Y}) \to \mathcal{P}(\mathcal{Y})$ to which p is equivariant if

$$P_g[p(\cdot|x)] = p(\cdot|L_g[x]), \quad \forall g \in G, x \in \mathcal{X}. \tag{3}$$

Given the equivalence in (3), from equivariance of the stochastic function w.r.t. the transformation P_g it follows that

$$p(y|x) = p(K_g[y] \mid L_g[x]), \quad \forall g \in G, x \in \mathcal{X}, y \in \mathcal{Y}, \tag{4}$$

giving a relationship between P_g and K_g.

B. Reinforcement Learning

1) Fundamentals: Reinforcement learning (RL) ([9]) aims to find a policy π that maximizes the cumulative discounted reward of a Markov Decision Process (MDP). An MDP is defined through the tuple $(\mathcal{S}, \mathcal{A}, \mathrm{P}, \mathrm{R}, \gamma)$, with the state space $\mathcal{S}$, action space $\mathcal{A}$, probabilistic transition function $\mathrm{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$, reward function $\mathrm{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and discount factor $\gamma \in [0, 1]$. The $\mathcal{P}(\mathcal{S})$ in the transition function stands for the space of all probability distributions over $\mathcal{S}$. The cumulative discounted reward, called return or value function, is defined as

$$\mathrm{V}^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathrm{R}(s_t, a_t) \;\middle|\; s_{t+1} \sim \mathrm{P}, a_t \sim \pi, s_0 = s\right], \tag{5}$$

when following a stochastic policy $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$. The state-action value function, the Q-value, is defined recursively as

$$\mathrm{Q}^{\pi}(s, a) = \mathrm{R}(s, a) + \gamma \mathbb{E}_{s' \sim \mathrm{P}, a' \sim \pi}\left[\mathrm{Q}^{\pi}(s', a')\right], \tag{6}$$

[1]Code: `https://github.com/theilem/uavSim.git`

in which s' and a' are the next state and action, respectively, which is the common notation adopted in the following. The Q-value determines the value of a specific action at the given state if following the policy afterward. Therefore, the advantage is defined as

$$\mathrm{A}^{\pi}(s, a) = \mathrm{Q}^{\pi}(s, a) - \mathrm{V}^{\pi}(s), \tag{7}$$

indicating how much better or worse is a specific action at a given state compared to the expectation when following π.

2) Proximal Policy Optimization (PPO): A popular deep RL algorithm is PPO [6], which trains an actor network for the policy π_ϕ with parameters ϕ and a critic network to approximate the state-value function $\mathrm{V}^{\pi}_{\theta}$ with parameters θ. The key underlying equation to train the actor is the policy optimization (PO) objective given by

$$J_{\mathrm{PO}}(\phi) = \mathbb{E}_{(s,a)\sim\tau^{\pi_{\phi_{\mathrm{old}}}}}\left[\frac{\pi_\phi(a|s)}{\pi_{\phi_{\mathrm{old}}}(a|s)} \mathrm{A}^{\pi_{\phi_{\mathrm{old}}}}(s, a)\right], \tag{8}$$

in which $\pi_{\phi_{\mathrm{old}}}$ is the *behavior* policy, which in PPO is the previous π_ϕ that was used to collect trajectories called a rollout $\tau^{\pi_{\phi_{\mathrm{old}}}}$. The advantage $\mathrm{A}^{\pi_{\phi_{\mathrm{old}}}}$ is estimated using the generalized advantage estimate (GAE) [10], which uses the discounted cumulative reward observed during the rollouts and the value estimate from the critic. In essence, the PO objective increases the probability of actions with positive advantage while decreasing it for actions with negative advantage. PPO adapts this objective by constraining the allowed difference between π_ϕ and $\pi_{\phi_{\mathrm{old}}}$. The critic is trained on the discounted cumulative reward observed during the rollouts. Therefore, PPO is an on-policy RL algorithm since it can only be trained on data gathered by the behavior policy.

3) Invariance and Equivariance in RL: Symmetrical MDPs have the following equivalences

$$\mathrm{P}(L_g[s']|L_g[s], K_g[a]) = \mathrm{P}(s'|s, a), \quad \forall g \in G \tag{9}$$

$$\mathrm{R}(L_g[s], K_g[a]) = \mathrm{R}(s, a), \quad \forall g \in G, \tag{10}$$

i.e., invariant transition and reward functions. It is shown that for symmetrical MDPs, the optimal policy π^* is equivariant [3], yielding

$$P_g[\pi^*(\cdot|s)] = \pi^*(\cdot|L_g[s]), \quad \forall g \in G. \tag{11}$$

Additionally, if a policy π_{eq} is equivariant, the corresponding value function is invariant, i.e.,

$$\mathrm{V}^{\pi_{\mathrm{eq}}}(L_g[s]) = \mathrm{V}^{\pi_{\mathrm{eq}}}(s), \quad \forall g \in G. \tag{12}$$

III. Related Research

In this section, we review existing literature on the exploitation of symmetries in RL and highlight our contribution.

A. Equivariant Neural Networks

Equivariant neural networks directly embed symmetries in structures of the neural networks to constrain inputs and outputs to satisfy equivariance requirements. The critical components of equivariant networks are *equivariant layers*, such as equivariant MLP [2] and special CNN layers [11]

whose weights are designed to satisfy equivariance constraints. The idea of the equivariant network is applied to many state-based tasks such as classical control [2], [12] and vision-based tasks such as robotic manipulation [13], [3], [14], showing significant improvement in sample efficiency. Nguyen et al. [15] extend the application of equivariant neural networks from standard MDPs to Partially Observable Markov Decision Processes (POMDPs). However, designing equivariant models is non-trivial as it needs a deep understanding of neural networks and it constrains the choice of advanced network architectures. Furthermore, equivariant models may introduce instabilities in the training process compared to standard architectures [12].

B. Regularization

Regularization-based approaches incorporate the invariance or equivariance properties as auxiliary terms in the objective functions for the training of the policy and value functions [4], [16], [5]. Invariance regularization for the policy and value is typically conducted in data augmentation [4], [16], [17] with the underlying idea that the value and action should be the same when applying observation transformations. The transformations can include photometric augmentation in vision-based observations or noise injections in state-based observations. The regularization of the policy minimizes the Kullback–Leibler (KL) divergence [4] of the action distribution between the original state and augmented states.

In many cases, the policy should be equivariant under some state transformations, i.e., the policy should also be transformed. To regularize the policy to be equivariant, [5] proposes to minimize the mean-squared error loss between the means of policy on the original state and the transformed policies on transformed states in the SAC algorithm [18] in a robotic manipulation task.

C. Our work

We propose equivariant policy ensembles for policy and value function regularization to exploit symmetries in reinforcement learning. Compared to the equivariant neural networks [2], [13], [3], [14], [15], our approach does not need special neural network designs and can be easily integrated with existing standard reinforcement learning algorithms. Compared with the regularization approaches [4], [16], [5], we construct an equivariant policy through our addition of the equivariant ensembles, such that the value regularization is theoretically sound.

IV. Methodology

Given that the optimal policy for a symmetric MDP is equivariant and the corresponding value function is invariant, one approach is to design neural network architectures for the actor and critic to be equivariant and invariant, respectively. However, this can be very challenging and highly dependent on the type of transformations applied. An alternative would be to augment the training data with the equivariant transitions to enrich the transition data. However, the naïve explicit equivariant data augmentation for PO,

$$J_{\text{PO}}(\phi) = \mathbb{E}_{(s,a)\sim\tau^{\pi_{\phi_{\text{old}}}}}\left[\frac{1}{|G|}\sum_{g\in G}\frac{\pi_\phi(K_g[a]|L_g[s])}{\pi_{\phi_{\text{old}}}(a|s)}\mathrm{A}^{\pi_{\phi_{\text{old}}}}(s,a)\right], \tag{13}$$

is not a sound estimate if π_ϕ is not equivariant, i.e., $\pi_\phi(K_g[a]|L_g[s]) \neq \pi_\phi(a|s)$ as was already noted by [4] in the case of invariant data augmentation. Additionally, if π_ϕ is equivariant, the policy optimization objective reduces back to the original one in (8) as all elements of the summation are equal. Therefore, the question is, how can gradients of the PO objective be propagated to the actor for all transformations in G? We achieve this by constructing an equivariant policy that explicitly incorporates all transformations: Equivariant Ensembles.

A. Equivariant Ensembles

By averaging the policy for each transformation, we create the policy ensemble as

$$\bar{\pi}_\phi(\cdot|s) = \frac{1}{|G|}\sum_{g\in G} P_g^{-1}[\pi_\phi(\cdot|L_g[s])]. \tag{14}$$

While the normal policy is likely not equivariant, i.e., $P_h[\pi(\cdot|s)] \not\equiv \pi(\cdot|L_h[s])$ for all $h \in G$, the policy ensemble is equivariant by construction, i.e.,

$$P_h[\bar{\pi}_\phi(\cdot|s)] = \bar{\pi}_\phi(\cdot|L_h[s]), \quad \forall h \in G. \tag{15}$$

Proof: Applying the transformation P_h to the ensemble policy in (14) yields

$$P_h[\bar{\pi}_\phi(\cdot|s)] = P_h\left[\frac{1}{|G|}\sum_{g\in G} P_g^{-1}[\pi_\phi(\cdot|L_g[s])]\right] \tag{16}$$

$$= \frac{1}{|G|}\sum_{g\in G} P_h[P_g^{-1}[\pi_\phi(\cdot|L_g[s])]], \tag{17}$$

interchanging it with the summation and constant terms since P_h is a linear operator. Recall that the composition of the transformations P_h and P_g^{-1} is also in G. Therefore, for each $g \in G$ there exists an $g' = h \circ g^{-1}$ (with $g = g'^{-1} \circ h$) in G, allowing the reindexation of the sum as

$$P_h[\bar{\pi}_\phi(\cdot|s)] = \frac{1}{|G|}\sum_{g'\in G} P_{g'}[\pi_\phi(\cdot|L_{g'^{-1}\circ h}[s])] \tag{18}$$

Since $L_{g'^{-1}\circ h}[s] = L_{g'}^{-1}[L_h[s]]$, it can be rewritten as

$$P_h[\bar{\pi}_\phi(\cdot|s)] = \frac{1}{|G|}\sum_{g'\in G} P_{g'}[\pi_\phi(\cdot|L_{g'}^{-1}[L_h[s]]) \tag{19}$$

Since the inverse of g' is part of G, the summation is simply rearranged when iterating over $g = g'^{-1}$, yielding

$$P_h[\bar{\pi}_\phi(\cdot|s)] = \frac{1}{|G|}\sum_{g\in G} P_g^{-1}[\pi_\phi(\cdot|L_g[L_h[s]])] \tag{20}$$

$$= \bar{\pi}_\phi(\cdot|L_h[s]), \tag{21}$$

concluding the proof of equivariance of the ensemble policy. □

Using the policy ensemble as behavior policy, the PO objective can be written as

$$J_{\text{PO}}(\phi) = \mathbb{E}_{(s,a)\sim\tau^{\bar{\pi}_{\phi_{\text{old}}}}}\left[\frac{\bar{\pi}_{\phi}(a|s)}{\bar{\pi}_{\phi_{\text{old}}}(a|s)}\mathrm{A}^{\bar{\pi}_{\phi_{\text{old}}}}(s,a)\right], \quad (22)$$

which allows the gradient to propagate to the parameters ϕ for all transformations as they are incorporated in the ensemble policy in (14). Similarly, an ensemble critic can be created as

$$\bar{\mathrm{V}}^{\bar{\pi}}_{\theta}(s) = \frac{1}{|G|}\sum_{g\in G} V^{\bar{\pi}}_{\theta}(L_g[s]) \quad (23)$$

to leverage the same effect during critic training. The value ensemble corresponding to the equivariant ensemble policy is invariant by construction, i.e.,

$$\bar{\mathrm{V}}^{\bar{\pi}}_{\theta}(L_g[s]) = \bar{\mathrm{V}}^{\bar{\pi}}_{\theta}(s), \quad \forall g \in G \quad (24)$$

Proof: The invariance of the ensemble critic can be shown through the following derivation:

$$\bar{\mathrm{V}}^{\bar{\pi}}_{\theta}(L_g[s]) = \frac{1}{|G|}\sum_{g'\in G} V^{\bar{\pi}}_{\theta}(L_{g'}[L_g[s]]) \quad |h = g'\circ g \quad (25)$$

$$= \frac{1}{|G|}\sum_{h\in G} V^{\bar{\pi}}_{\theta}(L_h[s]) \quad |g = h \quad (26)$$

$$= \bar{\mathrm{V}}^{\bar{\pi}}_{\theta}(s) \quad (27)$$

concluding the proof. □

Using the ensemble actor and critic thus allows the PO gradients to propagate through the networks for each possible transformation at the same time, which should improve training efficiency and generalization, as the agent observes a wider variety of scenarios at the same time.

B. Equivariant Regularization

Even though the ensemble policy is used as behavior policy and the actor is thus trained for all transformations simultaneously, the individual policy for one transformation is not explicitly trained to be equivariant. However, explicitly training the actor and critic to be equivariant and invariant could give some inductive bias that could further accelerate training. Therefore, we add a regularization loss for the actor and critic that penalizes the difference to the ensemble.

For the actor, the regularization loss is defined as

$$\mathrm{L}_{\pi}(\phi) = \mathbb{E}_{s\sim\tau}\left[\frac{1}{|G|}\sum_{\forall g\in G}\mathrm{D}\left(\pi_{\phi}(\cdot|L_g[s])\middle\|\bar{\pi}_{\phi}(\cdot|L_g[s])\right)\right], \quad (28)$$

where D is a divergence measure such as the Kullback-Leibler (KL) divergence, D_{KL}. This loss pushes the actor to output the equivariant distribution given each individual transformation. Similarly, the critic regularization can be formulated as

$$\mathrm{L}_{V}(\theta) = \mathbb{E}_{s\sim\tau}\left[\frac{1}{|G|}\sum_{\forall g\in G}\left(\mathrm{V}^{\bar{\pi}}_{\theta}(L_g[s]) - \bar{\mathrm{V}}^{\bar{\pi}}_{\theta}(s)\right)^2\right] \quad (29)$$

which is a mean square error on the difference between the value estimates of all transformations to the ensemble critic.

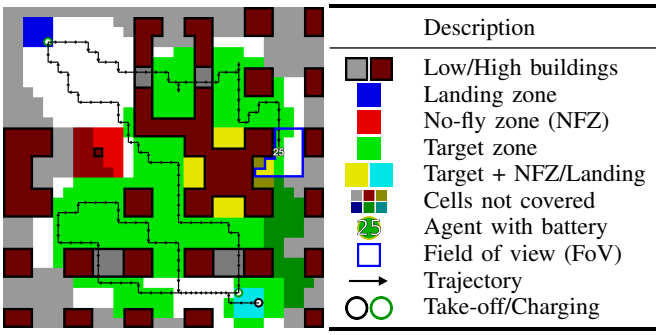


Fig. 2: Example state of a UAV in a coverage path planning grid-world problem on the left, showing the covered area, trajectory, and field of view, with a legend on the right.

V. Map-based Path Planning Case Study

Map-based path planning is a widely spread problem for various mobile robots. In this paper, we consider the UAV coverage path planning (CPP) problem, as it is a challenging problem with rotational symmetries.

A. Problem Formulation

In the UAV CPP problem, a UAV equipped with a face-down camera is tasked to fly over designated target zones while avoiding obstacles and adhering to battery constraints. This paper considers the power-constrained CPP problem with recharge, defined in [7]. The agent is a UAV moving in a grid world, as shown in Figure 2. As the legend describes, the environment consists of obstacles, landing zones, and no-fly zones (NFZs). The obstacles can be low such that the agent can fly over or high so it cannot. The environment contains target zones, which the agent is supposed to cover. The agent is a quadcopter-like UAV that can move north, east, south, or west and take off, land, or recharge in landing zones. It has an onboard battery, indicated by the number of steps the agent can do before it runs out. The UAV is equipped with a camera whose rectangular field of view (FoV) is obstructed by low and high obstacles. The objective is to find the shortest trajectory such that each cell of the target zone is in the FoV at least once, i.e., each cell is covered. After covering all target zones, the task is defined as solved after the agent lands in one of the landing zones.

General CPP problems are proven to be NP-hard [19], with the power-constrained UAV CPP problem with recharge problem adding to the challenge in two ways. First, since the UAV is equipped with a camera, its FoV is larger than its occupancy footprint, meaning it can cover cells by visiting nearby cells. Therefore, formulating it as an optimization problem is more complex as there is more than one position of the UAV from which a target zone can be covered. Second, the power constraint and the ability to recharge in designated areas add significant complexity to the problem. The optimal trajectory hinges on efficiently dividing the target zone for multiple flight segments interleaved with charging periods. Additionally, the allocation of selected target zones for the return journeys exacerbates the challenge. Overall, this

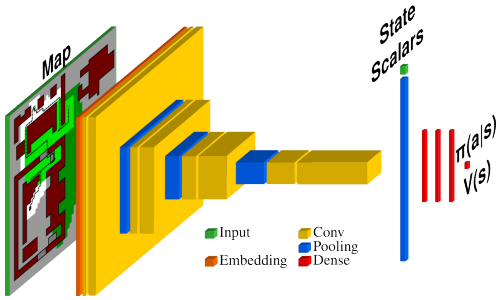


Fig. 3: Neural network architecture where the position is a one-hot representation in the map, and the battery and landing state are fed in as state scalars.

problem is a complex long-horizon problem that we chose to address with deep reinforcement learning.

B. Reinforcement Learning for CPP

As introduced in [8], we address the UAV CPP problem with RL using map-based observations of the environment. The MDP of the problem contains the state space

$$\mathcal{S} = \underbrace{\mathbb{B}^{m\times m\times 3}}_{\substack{\text{Environment}\\ \text{Map}}} \times \underbrace{\mathbb{B}^{m\times m}}_{\substack{\text{Target}\\ \text{Map}}} \times \underbrace{\mathbb{N}^2}_{\text{Position}} \times \underbrace{\mathbb{N}}_{\substack{\text{Battery}\\ \text{Level}}} \times \underbrace{\mathbb{B}}_{\text{Landed}}, \tag{30}$$

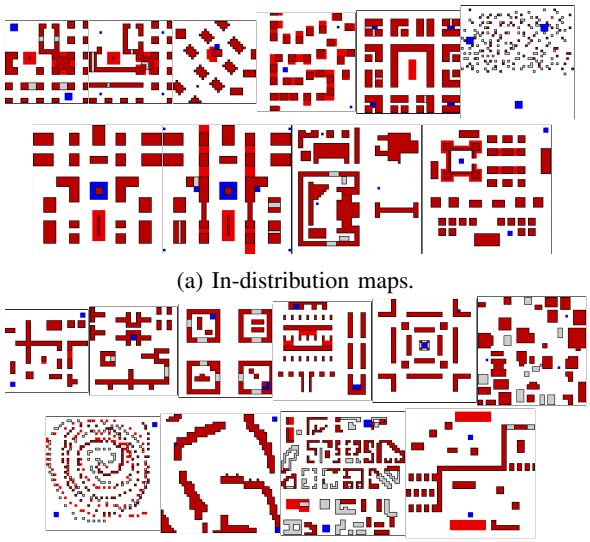
given the number of cells in the environment as $m \times m$. The action space is defined as a discrete set of actions

$$\mathcal{A} = \{\textit{east}, \textit{north}, \textit{west}, \textit{south}, \textit{take off}, \textit{land}, \textit{charge}\}. \tag{31}$$

The neural network to process the state input and create the action of value output is shown in Figure 3. For the details of the transition function P, the interested reader is directed to [7]. The reward function yields the step-wise reward

$$r_t = r_c(|\mathcal{C}_t| - |\mathcal{C}_{t+1}|) - r_m, \tag{32}$$

which gives a reward for the difference in the number of covered target cells $|\mathcal{C}_t| - |\mathcal{C}_{t+1}|$ scaled by r_c after performing the action and a flat penalty r_m for every step. The first part of the reward incentivizes the agent to cover as much of the target zones as possible, while the second part directs the agent to finish as early as possible to stop incurring penalties. As in [7], we schedule the discount factor γ such that the agent first learns to solve the task using a lower γ and then optimizes path length with a $\gamma \to 1$ where the cumulative movement penalty becomes dominant. In this paper, however, we do not schedule the discount factor based on training steps, as it differs based on the algorithm. Instead, we schedule it based on performance, such that the discount factor increases slightly every time the agent successfully solves a scenario. We further adopt the action masking approach from [7], which masks out actions that would lead to a collision or drive the agent too far away from a landing zone.



(a) In-distribution maps.

(b) Out-of-distribution maps.

Fig. 4: All maps used during training and testing.

C. Equivariances in the CPP Problem

The equivariances utilized in this paper are shown in Figure 1. We only focus on the four rotations as L_g and leave horizontal and vertical flips to future work. When rotating the input map, the three actions *take off*, *land*, and *charge* remain unaffected, while the four directional actions permute depending on the rotation applied.

VI. Experiment

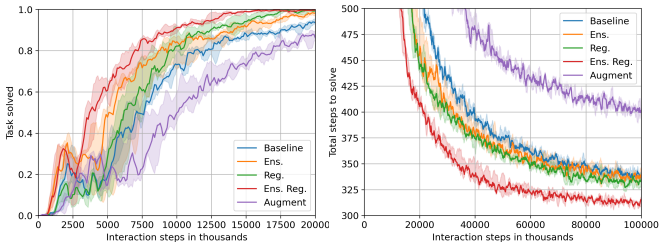
A. Setup

To study the effect of equivariant ensembles and regularization, we trained agents on 10 maps with varying sizes from 32×32 to 50×50 shown in Figure 4a. In each scenario encountered during training, one of the maps is randomly chosen, and a random target zone is generated. Since the range of possible target zones is vast, it is unlikely that an agent will ever encounter the same scenario between two episodes. Each agent is trained using PPO [6] for 100M interaction steps with the environment with rollouts of 40K steps.

We trained five different agent configurations:

1) *Baseline*: Normal agent trained on 10 maps.
2) *Augment*: Agent trained on the 10 maps in all four rotations, yielding 40 training maps.
3) *Ensemble*: Agent with ensemble actor and critic.
4) *Regularized*: Agent with actor and critic regularization only.
5) *Ensemble + Regularized (Ens.+Reg.)*: Agent with ensemble actor and critic and regularization.

The Augment agent was trained to show if the positive effects simply result from a broader set of maps trained on. Note that the Regularization agent regularizes the critic towards invariance even though the actor is not necessarily equivariant. For each agent configuration, three agents were



(a) Task solved ratio (b) Steps to solve

Fig. 5: Training curves of all algorithms showing the task-solved ratio throughout the first 20M steps and the average steps to solve the scenarios for the full 100M training steps.

trained, and their results were aggregated. The total training time for the Baseline and Augment agents is 24 hours and for the others, 60 hours on an Nvidia A100 GPU. The increase in training time stems from the added computation for the forward and backward passes for all transformations.

As metrics for comparison, we use

1) *Coverage ratio (CR)*: The ratio of covered cells when a timeout is reached, or the episode ended successfully, in which case CR=1.
2) *Task solved*: The share of scenarios that were solved, i.e., full coverage and landed, before a timeout (1500 steps) is reached.
3) *Episode steps*: The steps in the episode, which are 1500 if the timeout is reached, or the step of the landing action after the task is solved. The agent is supposed to minimize this metric.
4) *Relative deviation (RD)*: Comparing the steps needed to solve an episode with a heuristic ([7]), showing the difference. Lower is better.

We will first show the training performance of the different agent configurations, followed by an analysis of their performance after training. In the last part, the effect of regularization on equivariance and invariance is visualized.

B. Training Acceleration

The training curves for the task-solved ratio and the steps to solve the episode are shown in Figure 5. The task-solved ratio in Figure 5a, which focuses on the first 20M interaction steps, shows how the ensemble and regularization accelerate the training individually and that their combination accelerates it even further. It further shows that the Augment agent takes longer to learn, as it interacts with 40 seemingly unrelated maps. In Figure 5b, it can be seen that ensemble and regularization individually accelerate training performance at first but converge to a similar final result as the baseline. The combination of ensemble and regularization, however, accelerates training and improves final performance significantly. On the other hand, the augmented agent struggles to find fast solutions, leading to slower and suboptimal learning performance.

C. Performance

This section compares the performance of each agent configuration on three different sets of maps: (1) the in-distribution maps that each agent encountered during training, (2) the in-distribution maps but rotated 90°, 180°, and 270°, and (3) 10 out-of-distribution maps shown in Figure 4b that no agent saw during training. Table I summarizes the results leading to the following observations. Each agent configuration consistently solves the in-distribution maps, with the Augment agent having the lowest RD performance (highest RD). As already seen in the training curves, the Ensemble and Regularized agents have similar performance compared with the baseline, and the combination has the best performance with a significant lead over the Baseline.

For the rotated in-distribution maps, Augment, Ensemble, and Ens.+Reg. show very similar performance as for the non-rotated maps, showing that the equivariant ensemble is indeed equivariant. The Baseline cannot reliably solve these rotated maps. The Regularized agent shows a decreased performance for the rotated maps, which shows that it is not exactly equivariant, emphasizing the need for the ensemble when seeking an equivariant policy.

Finally, the last row shows the average coverage ratio (CR) on the 10 out-of-distribution maps, as it is the best metric if the scenarios are not solved reliably. It can be seen that every agent configuration is better than the baseline, with the best-performing agent being the Ensemble, closely followed by the Augment agent. Regularization is hindering out-of-distribution performance, which we attribute to a specialization problem. The regularizing agents specialize on the in-distribution maps throughout training. This effect can be observed by inspecting the performance of the Ens.+Reg.^{50M} agent, which was only trained for 50M interaction steps. While its performance on in-distribution maps is lower than that of the fully trained counterpart, its CR value on the out-of-distribution maps is higher. To avoid this specialization effect in the future, the agents will probably need to be trained on a significantly more extensive set of maps or based on a procedural map generator.

To conclude, the study shows that the combination of equivariant ensembles and regularization significantly improves the performance for scenarios on the maps that were seen during training (note that the in-distribution *scenarios* were still not seen during training as the optimal path strongly depends on the target zones that are generated randomly for each scenario). On out-of-distribution maps, just using the equivariant ensemble yields the highest performance as it appears to specialize less on the maps it has seen.

D. Equivariance through Regularization

In Table I, we show that the performance of the Regularized agent is lower for the rotated maps, indicating that it is not equivariant. To examine this in more detail, Table II shows the action distribution and value estimation for all transformations and agents in two states, one in-distribution and one out-of-distribution. The action distribution is only

		Baseline	Augment	Ensemble	Regularized	Ens.+Reg.	Ens.+Reg.^{50M}
In-Distribution (Fig 4a)	Solved	$\mathbf{1.00^{0.00}}$	$\mathbf{1.00^{0.00}}$	$\mathbf{1.00^{0.00}}$	$\mathbf{1.00^{0.00}}$	$\mathbf{1.00^{0.00}}$	$\mathbf{1.00^{0.00}}$
	RD	$-0.22^{0.01}$	$-0.15^{0.01}$	$-0.22^{0.00}$	$-0.24^{0.01}$	$\mathbf{-0.28^{0.00}}$	$-0.24^{0.01}$
In-Distribution Rotated	Solved	$0.58^{0.03}$	$\mathbf{1.00^{0.00}}$	$\mathbf{1.00^{0.00}}$	$\mathbf{1.00^{0.00}}$	$\mathbf{1.00^{0.00}}$	$\mathbf{1.00^{0.00}}$
	RD	$0.26^{0.03}$	$-0.14^{0.01}$	$-0.23^{0.01}$	$-0.19^{0.00}$	$\mathbf{-0.28^{0.00}}$	$-0.24^{0.01}$
Out-of-Distribution (Fig 4b)	CR	$0.71^{0.02}$	$0.84^{0.01}$	$\mathbf{0.86^{0.02}}$	$0.79^{0.00}$	$0.78^{0.01}$	$0.80^{0.00}$

TABLE I: Quantitative comparison of the different agents tested on different sets of maps, showing the task-solved ratios and relative deviation (RD) to a heuristic for the in-distribution maps and the collection ratio (CR) for out-of-distribution maps. The Ens.+Reg.^{50M} agent is the performance of Ens.+Reg. after 50M interaction steps. The numbers show the median value of the three agents per configuration, with the maximum deviation in superscript.

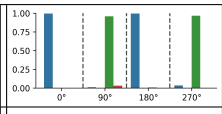
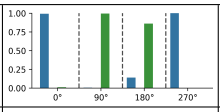
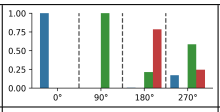
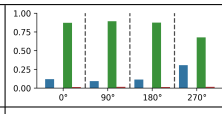
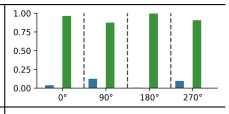
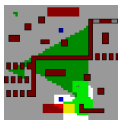
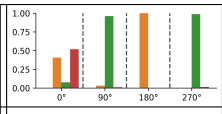
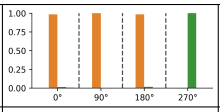
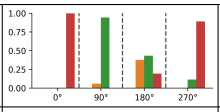
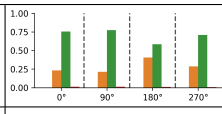
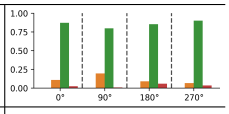
		Baseline	Augment	Ensemble	Regularize	Ensemble + Regularize
	$\pi(\cdot\|s)$					
	$\lvert\Delta V(s)\rvert$	[2.09 0.83 1.23 0.03]	[0.25 0.07 0.17 0.15]	[0.71 1.12 1.02 0.80]	[0.01 0.02 0.03 0.02]	[0.02 0.00 0.03 0.01]
	$\pi(\cdot\|s)$					
	$\lvert\Delta V(s)\rvert$	[0.45 0.24 0.27 0.06]	[0.24 0.19 0.44 0.49]	[0.52 0.21 0.12 0.61]	[0.15 0.16 0.10 0.11]	[0.08 0.06 0.04 0.09]

TABLE II: Qualitative visualization of equivariance and invariance of the different agents' policies and value functions for one in-distribution state (top) and one out-of-distribution state (bottom). The policy rows show the different action distributions over the *directional* actions **east**, **south**, **west**, and **north** for all four rotations after transforming them back to the normal direction, such that an equivariant policy should show the same distribution for each rotation. The value row shows the absolute difference of the value estimate for each rotation to the mean, such that an invariant value function should show values at 0.

shown for the directional actions, as the agent is not in a landing zone in both states, and thus the action mask is masking out the other three actions. In the table, an equivariant policy should show precisely the same distribution for the four rotations indicated through the 0°, 90°, 180°, and 270°.

Focussing on the policies in both states, the Baseline, Augment, and Ensemble agents show significantly different action distributions. On the other hand, the Regularized and Ens.+Reg. agents show very similar distributions for each rotation, which can be seen for the in-distribution and out-of-distribution states. However, the distributions are not exactly the same, confirming that the added regularization loss does not create exact equivariant policies but rather "nearly-equivariant" ones. The same can be observed for the value estimate, which is shown as the absolute difference to the mean of the value estimate for each rotation. The agents with regularization produce values significantly closer to the mean than the others, showing that they are "more invariant".

This result yields the critical insight that regularization brings the policy toward being equivariant but not to exact equivariance. Since the value estimate should only be invariant if the policy is equivariant, it is essential to consider that the regularization of the critic, as done in [4], [5], may not be as effective as expected if the actor is only regularized and no ensemble is used.

VII. Discussion

In this paper, we proposed the usage of equivariant ensembles and regularization to exploit symmetries in MDPs. We proved that ensemble policies and value functions are respectively equivariant and invariant, providing theoretical soundness to value regularization. We demonstrated the benefits of the proposed approach in a map-based path planning case study, showing a decrease in sample complexity and an increase in performance and generalization.

This paper assumes that symmetries in the MDP are perfect. However, this assumption may not be valid in some real-world applications. In asymmetrical cases, our approach may still work as long as the asymmetry is observable to the agent; this we will study in future work. The additional computational overhead needs to be considered, especially when the group of transformations grows large, e.g., by adding the flip symmetries in the case study. In that case, we would need to investigate whether we can randomly sample a subset of the transformations, which could yield equivariance in expectation. Furthermore, in this specific case study, we saw the limits on out-of-distribution generalization. These limits could be overcome by training the agent on procedurally generated maps, for which the equivariant ensembles and regularization would play an important role.

In future work, we will also investigate the effect of equiv-

ariant ensembles and regularization for different problem settings, specifically for continuous action spaces. Further, to make it more broadly applicable, we will study the effects when applying different RL algorithms such as the off-policy SAC algorithm [18].

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[2] 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*, vol. 33, pp. 4199–4210, 2020.

[3] D. Wang, R. Walters, and R. Platt, "So(2) equivariant reinforcement learning," in *International Conference on Learning Representations*, 2021.

[4] R. Raileanu, M. Goldstein, D. Yarats, I. Kostrikov, and R. Fergus, "Automatic data augmentation for generalization in reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 5402–5415, 2021.

[5] H. Nguyen, T. Kozuno, C. C. Beltran-Hernandez, and M. Hamaya, "Symmetry-aware reinforcement learning for robotic assembly under partial observability with a soft wrist," *arXiv preprint arXiv:2402.18002*, 2024.

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

[7] M. Theile, H. Bayerlein, M. Caccamo, and A. L. Sangiovanni-Vincentelli, "Learning to recharge: UAV coverage path planning through deep reinforcement learning," *arXiv preprint arXiv:2309.03157*, 2023.

[8] M. Theile, H. Bayerlein, R. Nai, D. Gesbert, and M. Caccamo, "Uav coverage path planning under varying power constraints using deep reinforcement learning," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 1444–1449.

[9] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: The MIT Press, 1998.

[10] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[11] T. S. Cohen, M. Geiger, and M. Weiler, "A general theory of equivariant cnns on homogeneous spaces," *Advances in neural information processing systems*, vol. 32, 2019.

[12] M. Finzi, G. Benton, and A. G. Wilson, "Residual pathway priors for soft equivariance constraints," *Advances in Neural Information Processing Systems*, vol. 34, pp. 30 037–30 049, 2021.

[13] D. Wang, R. Walters, X. Zhu, and R. Platt, "Equivariant q learning in spatial action spaces," in *Conference on Robot Learning*. PMLR, 2022, pp. 1713–1723.

[14] 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," in *The Eleventh International Conference on Learning Representations*, 2022.

[15] H. H. Nguyen, A. Baisero, D. Klee, D. Wang, R. Platt, and C. Amato, "Equivariant reinforcement learning under partial observability," in *Conference on Robot Learning*. PMLR, 2023, pp. 3309–3320.

[16] D. Yarats, I. Kostrikov, and R. Fergus, "Image augmentation is all you need: Regularizing deep reinforcement learning from pixels," in *International conference on learning representations*, 2020.

[17] M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas, "Reinforcement learning with augmented data," *Advances in neural information processing systems*, vol. 33, pp. 19 884–19 895, 2020.

[18] 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*. PMLR, 2018, pp. 1861–1870.

[19] E. M. Arkin, S. P. Fekete, and J. S. Mitchell, "Approximation algorithms for lawn mowing and milling," *Computational Geometry*, vol. 17, no. 1-2, pp. 25–50, 2000.