EQUIVARIANT MUZERO

Andreea Deac[*1,2] **, Théophane Weber**[3]**, George Papamakarios**[3]
[1] Mila – Québec Artificial Intelligence Institute, [2] Université de Montréal, [3] DeepMind
deacandr@mila.quebec, {theophane,gpapamak}@deepmind.com

ABSTRACT

Deep reinforcement learning repeatedly succeeds in closed, well-defined domains such as games (Chess, Go, StarCraft). The next frontier is real-world scenarios, where setups are numerous and varied. For this, agents need to learn the underlying rules governing the environment, so as to robustly generalise to conditions that differ from those they were trained on. Model-based reinforcement learning algorithms, such as the highly successful MuZero, aim to accomplish this by learning a world model. However, leveraging a world model has not consistently shown greater generalisation capabilities compared to model-free alternatives. In this work, we propose improving the data efficiency and generalisation capabilities of MuZero by explicitly incorporating the *symmetries* of the environment in its world-model architecture. We prove that, so long as the neural networks used by MuZero are equivariant to a particular symmetry group acting on the environment, the entirety of MuZero's action-selection algorithm will also be equivariant to that group. We evaluate Equivariant MuZero on procedurally-generated MiniPacman and on Chaser from the ProcGen suite: training on a set of mazes, and then testing on unseen rotated versions, demonstrating the benefits of equivariance. Further, we verify that our performance improvements hold even when only some of the components of Equivariant MuZero obey strict equivariance, which highlights the robustness of our construction.

1 INTRODUCTION

Reinforcement learning (RL) is a potent paradigm for solving sequential decision making problems in a dynamically changing environment. Successful examples of its uses include game playing (Vinyals et al., 2019), drug design (Segler et al., 2018), robotics (Ibarz et al., 2021) and theoretical computer science (Fawzi et al., 2022). However, the generality of RL often leads to data inefficiency, poor generalisation and lack of safety guarantees. This is an issue especially in domains where data is scarce or difficult to obtain, such as medicine or human-in-the-loop scenarios.

Most RL approaches do not directly attempt to capture the regularities present in the environment. As an example, consider a grid-world: moving down in a maze is equivalent to moving left in the 90° clock-wise rotation of the same maze. Such equivalences can be formalised via Markov Decision Process homomorphisms (Ravindran, 2004; Ravindran & Barto, 2004), and while some works incorporate them (e.g. van der Pol et al., 2020; Rezaei-Shoshtari et al., 2022), most deep reinforcement learning agents would act differently in such equivalent states if they do not observe enough data. This becomes even more problematic when the number of equivalent states is large. One common example is 3D regularities, such as changing camera angles in robotic tasks.

In recent years, there has been significant progress in building deep neural networks that explicitly obey such regularities, often termed geometric deep learning (Bronstein et al., 2021). In this context, the regularities are formalised using symmetry groups and architectures are built by composing transformations that are equivariant to these symmetry groups (e.g. convolutional neural networks for the translation group, graph neural networks and transformers for the permutation group).

As we are looking to capture the symmetries present in an environment, a fitting place is within the framework of model-based RL (MBRL). MBRL leverages explicit world-models to forecast the

[*] Work performed while the author was at DeepMind.

effect of action sequences, either in the form of next-state or immediate reward predictions. These imagined trajectories are used to construct plans that optimise the forecasted returns. In the context of state-of-the-art MBRL agent MuZero (Schrittwieser et al., 2020), a Monte-Carlo tree search is executed over these world-models in order to perform action selection.

In this paper, we demonstrate that equivariance and MBRL can be effectively combined by proposing Equivariant MuZero (EqMuZero, shown in Figure 2), a variant of MuZero where equivariance constraints are enforced by design in its constituent neural networks. As MuZero does not use these networks directly to act, but rather executes a search algorithm on top of their predictions, it is not immediately obvious that the actions taken by the EqMuZero agent would obey the same constraints—is it guaranteed to produce a rotated action when given a rotated maze? One of our key contributions is a proof that guarantees this: as long as all neural networks are equivariant to a symmetry group, all actions taken will also be equivariant to that same symmetry group. Consequently, EqMuZero can be more data-efficient than standard MuZero, as it knows by construction how to act in states it has never seen before. We empirically verify the generalisation capabilities of EqMuZero in two grid-worlds: procedurally-generated MiniPacman and the Chaser game in the ProcGen suite.

2 BACKGROUND

Reinforcement Learning The reinforcement learning problem is typically formalised as a Markov Decision Process (S, A, P, R, γ) formed from a set of states S, a set of actions A, a discount factor $\gamma \in [0, 1]$, and two functions that model the outcome of taking action a in state s: the transition distribution $P(s'|s, a)$—specifying the next state probabilities—and the reward function $R(s, a)$—specifying the expected reward. The aim is to learn a *policy*, $\pi(a|s)$, a function specifying (probabilities of) actions to take in state s, such that the agent maximises the (expected) cumulative reward $G(\tau) = \sum_{t=0}^{t=T} \gamma^t R(s_t, a_t)$, where $\tau = (s_0, a_0, s_1, a_1, \ldots, s_T, a_T)$ is the trajectory taken by the agent starting in the initial state s_0 and following the policy to decide a_t based on s_t.

MuZero Reinforcement learning agents broadly fall into two categories: *model-free* and *model-based*. The specific agent we extend here, MuZero (Schrittwieser et al., 2020), is a model-based agent for deterministic environments (where $P(s'|s, a) = 1$ for exactly one s' for all $s \in S$ and $a \in A$). MuZero relies on several neural-network components that are composed to create a *world model*. These components are: the *encoder*, $E : S \to Z$, which embeds states into a latent space Z (e.g. $Z = \mathbb{R}^k$), the *transition model*, $T : Z \times A \to Z$, which predicts embeddings of next states, the *reward model*, $R : Z \times A \to \mathbb{R}$, which predicts the immediate expected reward after taking an action in a particular state, the *value model*, $V : Z \to \mathbb{R}$, which predicts the value (expected cumulative reward) from this state, and the *policy model* $P : Z \to [0, 1]^{|A|}$, which predicts the probability of taking each action from the current state. To plan its next action, MuZero executes a Monte Carlo tree search (MCTS) over many simulated trajectories, generated using the above models.

MuZero has demonstrated state-of-the-art capabilities over a variety of deterministic or near-deterministic environments, such as Go, Chess, Shogi and Atari, and has been successfully applied to real-world domains such as video compression (Mandhane et al., 2022). Although here we focus on MuZero for deterministic environments, we note that extensions to stochastic environments also exist (Antonoglou et al., 2021) and are an interesting target for future work.

Groups and Representations A *group* $(\mathfrak{G}, \circ)$ is a set $\mathfrak{G}$ equipped with a *composition* operation $\circ : \mathfrak{G} \times \mathfrak{G} \to \mathfrak{G}$ (written concisely as $\mathfrak{g} \circ \mathfrak{h} = \mathfrak{gh}$), satisfying the following axioms: *(associativity)* $(\mathfrak{gh})\mathfrak{l} = \mathfrak{g}(\mathfrak{hl})$ for all $\mathfrak{g}, \mathfrak{h}, \mathfrak{l} \in \mathfrak{G}$; *(identity)* there exists a unique $\mathfrak{e} \in \mathfrak{G}$ satisfying $\mathfrak{eg} = \mathfrak{ge} = \mathfrak{g}$ for all $\mathfrak{g} \in \mathfrak{G}$; *(inverse)* for every $\mathfrak{g} \in \mathfrak{G}$ there exists a unique $\mathfrak{g}^{-1} \in \mathfrak{G}$ such that $\mathfrak{gg}^{-1} = \mathfrak{g}^{-1}\mathfrak{g} = \mathfrak{e}$.

Groups are a natural way to describe *symmetries*: object transformations that leave them unchanged. They can be reasoned about in the context of linear algebra by using their *real representations*: functions $\rho_{\mathcal{V}} : \mathfrak{G} \to \mathbb{R}^{N \times N}$ that give, for every group element $\mathfrak{g} \in \mathfrak{G}$, a real matrix demonstrating how this element *acts* on a vector space $\mathcal{V}$. For example, for the rotation group $\mathfrak{G} = \mathrm{SO}(n)$, the representation $\rho_{\mathcal{V}}$ would provide an appropriate $n \times n$ rotation matrix for each rotation $\mathfrak{g}$.

Equivariance and Invariance As symmetries are assumed to not change the essence of the data they act on, we would like to construct neural networks that adequately represent such symmetry-transformed inputs. Assume we have a neural network $f : \mathcal{X} \to \mathcal{Y}$, mapping between vector spaces $\mathcal{X}$ and $\mathcal{Y}$, and that we would like this network to respect the symmetries within a group $\mathfrak{G}$. Then we

can impose the following condition, for all group elements $\mathfrak{g} \in \mathfrak{G}$ and inputs $\mathbf{x} \in \mathcal{X}$:

$$f(\rho_{\mathcal{X}}(\mathfrak{g})\mathbf{x}) = \rho_{\mathcal{Y}}(\mathfrak{g})f(\mathbf{x}). \tag{1}$$

This condition is known as $\mathfrak{G}$-*equivariance*—for any group element, it does not matter whether we act with it on the input or on the output of the function f—the end result is the same. A special case of this, $\mathfrak{G}$-*invariance*, is when the output representation is trivial ($\rho_{\mathcal{Y}}(\mathfrak{g}) = \mathbf{I}$):

$$f(\rho_{\mathcal{X}}(\mathfrak{g})\mathbf{x}) = f(\mathbf{x}). \tag{2}$$

In geometric deep learning, equivariance to reflections, rotations, translations and permutations has been of particular interest (Bronstein et al., 2021).

Generally speaking, there are three ways to obtain an equivariant model: a) data augmentation, b) data canonicalisation and c) specialised architectures. Data augmentation creates additional training data by applying group elements $\mathfrak{g}$ to input/output pairs $(\mathbf{x}, \mathbf{y})$—equivariance is encouraged by training on the transformed data and/or minimising auxiliary losses such as $\|\rho_{\mathcal{Y}}(\mathfrak{g})f(\mathbf{x}) - f(\rho_{\mathcal{X}}(\mathfrak{g})\mathbf{x})\|$. Data augmentation can be simple to apply, but it results in only approximate equivariance. Data canonicalisation requires a method to standardise the input, such as breaking the translation symmetry for molecular representation by centering the atoms around the origin (Musil et al., 2021)—however, in many cases, such as the relatively simple MiniPacman environment we use in our experiments, such a canonical transformation may not exist. Specialised architectures have the downside of being harder to build, but they can guarantee exact equivariance—as such, they reduce the search space of functions, potentially reducing the number of parameters and increasing training efficiency.

Equivariance in RL There has been previous work at the intersection of reinforcement learning and equivariance. While leveraging multi-agent symmetries was repeatedly shown to hold promise (van der Pol et al., 2021; Muglich et al., 2022), of particular interest to us are the symmetries emerging from the environment, in a single-agent scenario. Related work in this space can be summarised by the commutative diagram in Figure 1. When considering only the cube at the bottom, we recover Park et al. (2022)—a supervised learning task where a latent transition model T learns to predict the next state embedding. They show that if T is equivariant, the encoder can pick up the symmetries of the environment even if it is not fully equivariant by design. Mondal et al. (2022) build a model-free agent by combining an equivariant-by-design encoder and enforcing the remaining equivariances via regularisation losses. They also consider the invariance of the reward, captured in Figure 1 by taking the decoder to be the reward model and $l = 1$. The work of van der Pol et al. (2020) can be described by having the value model as the decoder, while the work of Wang et al. (2022) has the decoder as the policy model and $l = |A|$.

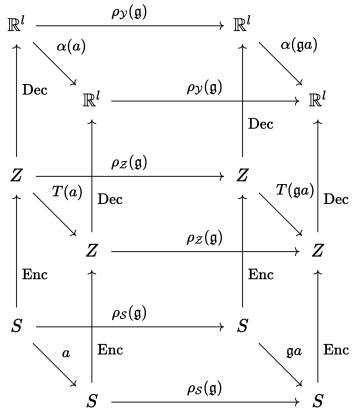

Figure 1: Commutative diagram of symmetries in RL. State transitions due to an action a are back-to-front, transformations due to a symmetry $\mathfrak{g}$ are left-to-right, state encoding and decoding by the model is bottom-to-top.

3 Experiments and Results

Environments We consider two 2D grid-world environments, MiniPacman (Guez et al., 2019) and Chaser (Cobbe et al., 2020), that feature an agent navigating in a 2D maze. In both environments, the state is the grid-world map $\mathbf{X}$ and an action is a direction to move. Both of these grid-worlds are symmetric with respect to $90°$ rotations, in the sense that moving down in some map is the same as moving left in the $90°$ clock-wise rotated version of the same map. Hence, we take our symmetry group to be $\mathfrak{G} = C_4 = \{\mathbf{I}, \mathbf{R}_{90°}, \mathbf{R}_{180°}, \mathbf{R}_{270°}\}$, the 4-element cyclic group, which in our case represents rotating the map by all four possible multiples of $90°$.

Equivariant MuZero In what follows, we describe how the various components of EqMuZero (Figure 2) are designed to obey C_4-equivariance. For simplicity, we assume there are only four directional movement actions in the environment ($A = \{\rightarrow, \downarrow, \leftarrow, \uparrow\}$). Any additional non-movement actions (such as the "do nothing" action) can be included without difficulty.

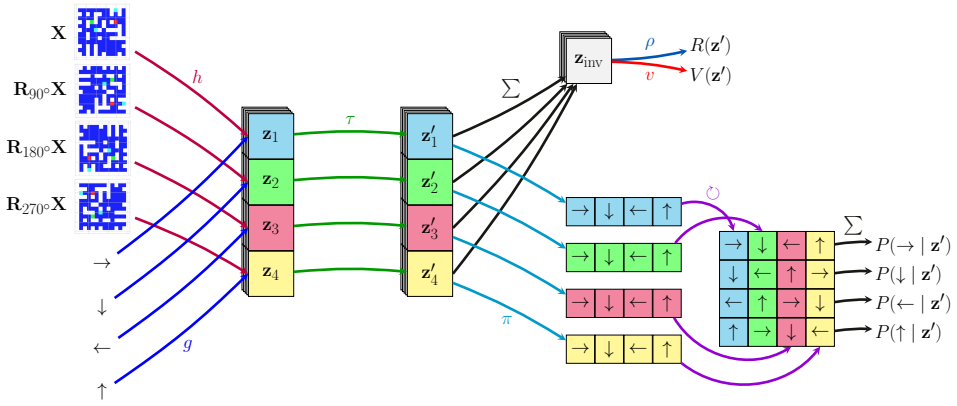

Figure 2: Architecture of Equivariant MuZero, where h, g are encoders, τ is the transition model, ρ is the reward model, v is the value model and π is the policy predictor. Each colour represents an element of the C_4 group $\{\mathbf{I}, \mathbf{R}_{90^\circ}, \mathbf{R}_{180^\circ}, \mathbf{R}_{270^\circ}\}$ applied to the input (observation and action).

To enforce C_4-equivariance in the encoder, we first need to specify the effect of rotations on the latent state $\mathbf{z}$. In our implementation, the latent state consists of 4 equally shaped arrays, $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4)$, and we prescribe that a 90° clock-wise rotation manifests as a cyclical permutation: $\mathbf{R}_{90^\circ}\mathbf{z} = (\mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4, \mathbf{z}_1)$. Then, our equivariant encoder embeds state $\mathbf{X}$ and action a as follows:

$$E(\mathbf{X}, a) = (h(\mathbf{X})+g(a), h(\mathbf{R}_{90^\circ}\mathbf{X})+g(\mathbf{R}_{90^\circ}a), h(\mathbf{R}_{180^\circ}\mathbf{X})+g(\mathbf{R}_{180^\circ}a), h(\mathbf{R}_{270^\circ}\mathbf{X})+g(\mathbf{R}_{270^\circ}a)) \tag{3}$$

where h is a CNN and g is an MLP. The output of g is accordingly broadcasted across all pixels of h's output. This equation satisfies C_4-equivariance, that is, $E(\mathbf{R}_{90^\circ}\mathbf{X}, \mathbf{R}_{90^\circ}a) = \mathbf{R}_{90^\circ}E(\mathbf{X}, a)$.

We can build a C_4-equivariant transition model by maintaining the structure in the latent space:

$$T(\mathbf{z}) = (\tau(\mathbf{z}_1), \tau(\mathbf{z}_2), \tau(\mathbf{z}_3), \tau(\mathbf{z}_4)). \tag{4}$$

A less constrained T would allow components of $\mathbf{z}$ to *interact*, while still retaining C_4-equivariance:

$$T(\mathbf{z}) = (\tau(\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4), \tau(\mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4, \mathbf{z}_1), \tau(\mathbf{z}_3, \mathbf{z}_4, \mathbf{z}_1, \mathbf{z}_2), \tau(\mathbf{z}_4, \mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3)). \tag{5}$$

In our experiments, we use the more constrained variant for MiniPacman, and the less constrained variant for Chaser, as more data is available for the latter. In either case, we take τ to be a ResNet.

The policy is made C_4-equivariant by combining state and action embeddings from all four latents:

$$P(a \mid \mathbf{z}) = \frac{\pi(a \mid \mathbf{z}_1) + \pi(\mathbf{R}_{90^\circ}a \mid \mathbf{z}_2) + \pi(\mathbf{R}_{180^\circ}a \mid \mathbf{z}_3) + \pi(\mathbf{R}_{270^\circ}a \mid \mathbf{z}_4)}{4} \tag{6}$$

where $\pi(\cdot \mid \mathbf{z}_i)$ is an MLP followed by a softmax, which produces a probability distribution over actions given the map encoded by $\mathbf{z}_i$. It is easy to show that $\sum_{a \in A} P(a \mid \mathbf{z}) = 1$, i.e. $P(\cdot \mid \mathbf{z})$ is properly normalised, and that $P(\mathbf{R}_{90^\circ}a \mid \mathbf{R}_{90^\circ}\mathbf{z}) = P(a \mid \mathbf{z})$, i.e. it satisfies C_4-equivariance.

Lastly, the reward and value networks (R, V), modeled by MLPs ρ and v respectively, should be C_4-invariant. We can satisfy this constraint by *aggregating* the latent space with any C_4-invariant function, such as sum, average or max. Here we use summation:

$$R(\mathbf{z}) = \rho(\mathbf{z}_1 + \mathbf{z}_2 + \mathbf{z}_3 + \mathbf{z}_4), \qquad V(\mathbf{z}) = v(\mathbf{z}_1 + \mathbf{z}_2 + \mathbf{z}_3 + \mathbf{z}_4). \tag{7}$$

Composing the equivariant components described above (Equations 3–7), we construct the end-to-end equivariant EqMuZero agent, displayed in Figure 2. Indeed, we can show that EqMuZero will provably behave in an equivariant manner when selecting actions:

Theorem 1 *If all the relevant neural networks used by MuZero are $\mathfrak{G}$-equivariant, the proposed EqMuZero agent will select actions in a $\mathfrak{G}$-equivariant manner, that is for every state $s \in S$ and for every $\mathfrak{g} \in \mathfrak{G}$, if EqMuZero selects action a while in s, then it must select $\mathfrak{g}a$ while in $\mathfrak{g}s$.*

We prove Theorem 1 in Appendix A.

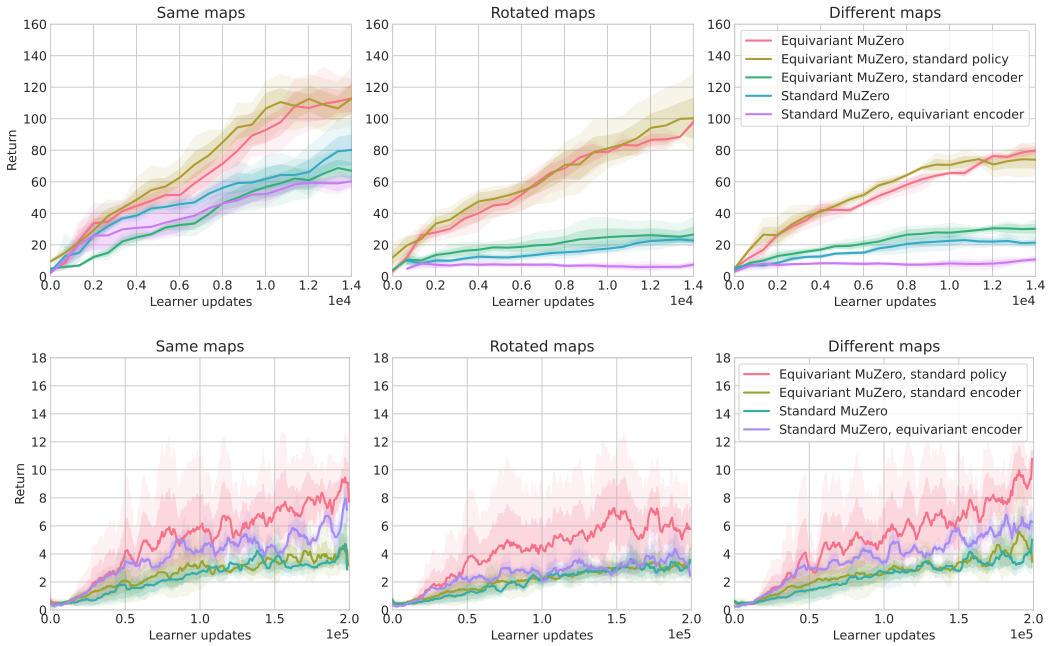

Figure 3: Results on procedurally-generated MiniPacman (top) and Chaser from ProcGen (bottom).

Results We compare EqMuZero with a standard MuZero that uses non-equivariant components: ResNet-style networks for the encoder and transition models, and MLP-based policy, value and reward models, following Hamrick et al. (2020). As the encoder and the policy of EqMuZero are the only two components which require knowledge of how the symmetry group acts on the environment, we include the following ablations in order to evaluate the trade-off between end-to-end equivariance and general applicability: Standard MuZero with an equivariant encoder, equivariant MuZero with a standard encoder and equivariant MuZero with a standard policy model.

We train each agent on a set of maps, $\mathbf{X}$. To test for generalisation, we measure the agent's performance on three, progressively harder, settings. Namely, we evaluate the agent on $\mathbf{X}$, with randomised initial agent position (denoted by *same* in our results), on the set of rotated maps $\mathbf{RX}$, where $\mathbf{R} \in \{\mathbf{R}_{90^\circ}, \mathbf{R}_{180^\circ}, \mathbf{R}_{270^\circ}\}$ (denoted by *rotated*) and, lastly, on a set of maps $\mathbf{Y}$, such that $\mathbf{Y} \cap \mathbf{X} = \varnothing$ and $\mathbf{Y} \cap \mathbf{RX} = \varnothing$ (denoted by *different*).

Figure 3 (top) presents the results of the agents on MiniPacman. First, we empirically confirm that the average reward on layouts $\mathbf{X}$, seen during training, matches the average reward gathered on the rotations of the same mazes, $\mathbf{RX}$, for EqMuZero. Second, we notice that changing the equivariant policy with a non-equivariant one does not significantly impact performance. However, the same swap in the encoder brings the performance of the agent down to that of Standard MuZero—this suggests that the structure in the latent space of the transition model, when not combined with some explicit method of imposing equivariance in the encoder, does not provide noticeable benefits. Third, we notice that Equivariant MuZero is generally robust to layout variations, as the learnt high-reward behaviours also transfer to $\mathbf{Y}$. At the same time, Standard MuZero significantly drops in performance for both $\mathbf{Y}$ and $\mathbf{RX}$. We note that experiments on MiniPacman were done in a low-data scenario, using 5 maps of size 14×14 for training; we observed that the differences between agents diminished when all agents were trained with at least 20 times more maps.

Figure 3 (bottom) compares the performance of the agents on the ProcGen game, Chaser, which has similar dynamics to MiniPacman, but larger mazes of size 64×64 and a more complex action space. Due to the complexity of the action space, we only use EqMuZero with a standard policy, rather than a fully equivariant version. We use 500 maze instances for training. Our results demonstrate that, even when the problem complexity is increased in such a way, Equivariant MuZero still consistently outperforms the other agents, leading to more robust plans being discovered.

REFERENCES

Ioannis Antonoglou, Julian Schrittwieser, Sherjil Ozair, Thomas K Hubert, and David Silver. Planning in stochastic environments with a learned model. In *International Conference on Learning Representations*, 2021.

Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

Karl Cobbe, Chris Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *International Conference on Machine Learning*, pp. 2048–2056. PMLR, 2020.

Alhussein Fawzi, Matej Balog, Aja Huang, Thomas Hubert, Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Francisco J R Ruiz, Julian Schrittwieser, Grzegorz Swirszcz, David Silver, Demis Hassabis, and Pushmeet Kohli. Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, 610(7930):47–53, 2022.

Arthur Guez, Mehdi Mirza, Karol Gregor, Rishabh Kabra, Sébastien Racanière, Théophane Weber, David Raposo, Adam Santoro, Laurent Orseau, Tom Eccles, Greg Wayne, David Silver, and Timothy Lillicrap. An investigation of model-free planning. In *International Conference on Machine Learning*, pp. 2464–2473. PMLR, 2019.

Jessica B Hamrick, Abram L Friesen, Feryal Behbahani, Arthur Guez, Fabio Viola, Sims Witherspoon, Thomas Anthony, Lars Buesing, Petar Veličković, and Théophane Weber. On the role of planning in model-based deep reinforcement learning. *arXiv preprint arXiv:2011.04021*, 2020.

Julian Ibarz, Jie Tan, Chelsea Finn, Mrinal Kalakrishnan, Peter Pastor, and Sergey Levine. How to train your robot with deep reinforcement learning: lessons we have learned. *The International Journal of Robotics Research*, 40(4-5):698–721, 2021.

Amol Mandhane, Anton Zhernov, Maribeth Rauh, Chenjie Gu, Miaosen Wang, Flora Xue, Wendy Shang, Derek Pang, Rene Claus, Ching-Han Chiang, et al. MuZero with self-competition for rate control in VP9 video compression. *arXiv preprint arXiv:2202.06626*, 2022.

Arnab Kumar Mondal, Vineet Jain, Kaleem Siddiqi, and Siamak Ravanbakhsh. EqR: Equivariant representations for data-efficient reinforcement learning. In *International Conference on Machine Learning*, pp. 15908–15926. PMLR, 2022.

Darius Muglich, Christian Schroeder de Witt, Elise van der Pol, Shimon Whiteson, and Jakob Foerster. Equivariant networks for zero-shot coordination. *arXiv preprint arXiv:2210.12124*, 2022.

Felix Musil, Andrea Grisafi, Albert P Bartók, Christoph Ortner, Gábor Csányi, and Michele Ceriotti. Physics-inspired structural representations for molecules and materials. *Chemical Reviews*, 121 (16):9759–9815, 2021.

Jung Yeon Park, Ondrej Biza, Linfeng Zhao, Jan Willem van de Meent, and Robin Walters. Learning symmetric embeddings for equivariant world models. *arXiv preprint arXiv:2204.11371*, 2022.

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

Balaraman Ravindran and Andrew G Barto. Approximate homomorphisms: A framework for non-exact minimization in Markov decision processes. 2004.

Sahand Rezaei-Shoshtari, Rosie Zhao, Prakash Panangaden, David Meger, and Doina Precup. Continuous MDP homomorphisms and homomorphic policy gradient. *arXiv preprint arXiv:2209.07364*, 2022.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

Marwin H S Segler, Mike Preuss, and Mark P Waller. Planning chemical syntheses with deep neural networks and symbolic AI. *Nature*, 555(7698):604–610, 2018.

Elise van der Pol, Daniel Worrall, Herke van Hoof, Frans Oliehoek, and Max Welling. MDP homomorphic networks: Group symmetries in reinforcement learning. *Advances in Neural Information Processing Systems*, 33:4199–4210, 2020.

Elise van der Pol, Herke van Hoof, Frans A Oliehoek, and Max Welling. Multi-agent MDP homomorphic networks. *arXiv preprint arXiv:2110.04495*, 2021.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

Dian Wang, Robin Walters, and Robert Platt. SO(2)-equivariant reinforcement learning. *arXiv preprint arXiv:2203.04439*, 2022.

A Proof of MuZero Equivariance

Assume our neural networks are: h for the encoder, τ for the transition model, π for the policy model, v for the value model and ρ for the reward model. By design, we make h, τ and π be $\mathfrak{G}$-equivariant, and v and ρ be $\mathfrak{G}$-invariant.

The reward, value, policy and transition respect the equivariances, as compositions of equivariant functions:

$$\begin{aligned} R &= \rho\tau^k h \\ V &= v\tau^k h \\ P &= \pi\tau^k h \\ T &= \tau^k h. \end{aligned} \tag{8}$$

Then, the return is also a $\mathfrak{G}$-invariant function as it is the sum of two $\mathfrak{G}$-invariant functions:

$$G(s^k) = \sum_{\tau=0}^{l-1-k} \gamma^\tau \rho(s^{k+\tau}, a^{k+1+\tau}) + \gamma^{l-k} v(s^l, a^{l+1}). \tag{9}$$

For proving that one planning step is equivariant, we need to show that the action selection is $\mathfrak{G}$-equivariant.

Since the outcome of MuZero's MCTS function is based on the initial observation, o, we denote MCTS's internal state as $\{Q^o(s,a), N^o(s,a), \ldots\}$. We use identical notation as Schrittwieser et al. (2020) for these states, even though we express the MuZero models R, V, P, T somewhat differently.

Knowing how they are updated:

$$a^k = \operatorname*{argmax}_a \left[Q^o(s^{k-1}, a) + P^o(s^{k-1}, a) \frac{\sqrt{\sum_b N^o(s^{k-1}, b)}}{1 + N^o(s^{k-1}, a)} \left(c_1 + \log\left(\frac{\sum_b N^o(s^{k-1}, b) + c_2 + 1}{c_2} \right) \right) \right] \tag{10}$$

$$\begin{aligned} Q^o_t(s^{k-1}, a^k) &= \frac{N^o_{t-1}(s^{k-1}, a^k) Q^o_{t-1}(s^{k-1}, a^k) + G(s^{k-1})}{N^o_{t-1}(s^{k-1}, a^k) + 1} \\ N^o_t(s^{k-1}, a^k) &= N^o_{t-1}(s^{k-1}, a^k) + 1. \end{aligned} \tag{11}$$

As discussed previously, we need to show that, for each MCTS internal state (e.g. N^o), if we assume π, v, τ, ρ, h to be equivariant functions, the resulting state would also be equivariant under transformations of the initial observation. That is, for all s, a:

$$N^{\mathfrak{g}_o o}(\mathfrak{g}_s s, \mathfrak{g}_a a) = N^o(s, a). \tag{12}$$

To prove this, we will use induction on the number of backups performed by MCTS, t. We proceed:

$$\begin{aligned} \text{Base case } (t=0): N_0^{\mathfrak{g}_o o}(\mathfrak{g}_s s, \mathfrak{g}_a a) &= N_0^o(s, a) = 0 \\ Q_0^{\mathfrak{g}_o o}(\mathfrak{g}_s s, \mathfrak{g}_a a) &= Q_0^o(s, a) = 0. \end{aligned} \tag{13}$$

Assume:

$$\begin{aligned} \text{Case } t: N_t^{\mathfrak{g}_o o}(\mathfrak{g}_s s, \mathfrak{g}_a a) &= N_t^o(s,a) \\ Q_t^{\mathfrak{g}_o o}(\mathfrak{g}_s s, \mathfrak{g}_a a) &= Q_t^o(s,a). \end{aligned} \tag{14}$$

We will start by showing that the states and actions expanded by MCTS under initial $\mathfrak{G}$-transformed observation $\mathfrak{g}_o o$ $(\widetilde{s}^0, \widetilde{a}^1, \widetilde{s}^1, \widetilde{a}^2, \dots)$, would exactly correspond to $(\mathfrak{g}_s s^0, \mathfrak{g}_a a^1, \mathfrak{g}_s s^1, \mathfrak{g}_a a^2, \dots)$, where $(s^0, a^1, s^1, a^2, \dots)$ are states expanded under the non-transformed observation, o.

By equivariance of h, $\widetilde{s}^0 = h(\mathfrak{g}_o o) = \mathfrak{g}_s h(o) = \mathfrak{g}_s s^0$, as expected.

Next, we show that the actions selected by MCTS also obey a $\mathfrak{G}$-equivariance constraint, in the sense that: if $\widetilde{s}^{k-1} = \mathfrak{g}_s s^{k-1}$, then $\widetilde{a}^k = \mathfrak{g}_a a^k$.

As we assumed N_t^o to be $\mathfrak{G}$-equivariant, it must hold that $\sum_b N_t^o(s,b)$ is $\mathfrak{G}$-invariant (as a sum-reduction of equivariant functions). Hence, we can rewrite Equation 10 as:

$$a^k = \arg\max_a \left[Q_t^o(s^{k-1}, a) + P_t^o(s^{k-1}, a) \frac{\epsilon(s^{k-1})}{1 + N_t^o(s^{k-1}, a)} \right] \tag{15}$$

where ϵ is $\mathfrak{G}$-invariant, P^o is $\mathfrak{G}$-equivariant by composition of functions that are $\mathfrak{G}$-equivariant by assumption, and Q^o is $\mathfrak{G}$-equivariant by assumption of Case t.

Hence, using this formula to define $\widetilde{a}^k$, we recover:

$$\begin{aligned} \widetilde{a}^k &= \arg\max_a \left[Q_t^{\mathfrak{g}_o o}(\widetilde{s}^{k-1}, a) + P_t^{\mathfrak{g}_o o}(\widetilde{s}^{k-1}, a) \frac{\epsilon(\widetilde{s}^{k-1})}{1 + N_t^{\mathfrak{g}_o o}(\widetilde{s}^{k-1}, a)} \right] \\ &= \arg\max_a \left[Q_t^{\mathfrak{g}_o o}(\mathfrak{g}_s s^{k-1}, a) + P_t^{\mathfrak{g}_o o}(\mathfrak{g}_s s^{k-1}, a) \frac{\epsilon(\mathfrak{g}_s s^{k-1})}{1 + N_t^{\mathfrak{g}_o o}(\mathfrak{g}_s s^{k-1}, a)} \right] \\ &= \arg\max_a \left[Q_t^{\mathfrak{g}_o o}(\mathfrak{g}_s s^{k-1}, \mathfrak{g}_a \mathfrak{g}_a^{-1} a) + P_t^{\mathfrak{g}_o o}(\mathfrak{g}_s s^{k-1}, \mathfrak{g}_a \mathfrak{g}_a^{-1} a) \frac{\epsilon(\mathfrak{g}_s s^{k-1})}{1 + N_t^{\mathfrak{g}_o o}(\mathfrak{g}_s s^{k-1}, \mathfrak{g}_a \mathfrak{g}_a^{-1} a)} \right] \\ &= \arg\max_a \left[Q_t^o(s^{k-1}, \mathfrak{g}_a^{-1} a) + P_t^o(s^{k-1}, \mathfrak{g}_a^{-1} a) \frac{\epsilon(s^{k-1})}{1 + N_t^o(s^{k-1}, \mathfrak{g}_a^{-1} a)} \right] \\ &= \mathfrak{g}_a \arg\max_a \left[Q_t^o(s^{k-1}, a) + P_t^o(s^{k-1}, a) \frac{\epsilon(s^{k-1})}{1 + N_t^o(s^{k-1}, a)} \right] \\ &= \mathfrak{g}_a a^k. \end{aligned}$$

Note that we have taken the $\mathfrak{g}_a$ out of the $\arg\max$, which is an unambiguous operation only if there is a unique action a^k that maximises the expression in Equation 15. To avoid breaking the symmetry in practice, we propose that tiebreaks for a^k are resolved in a purely randomised fashion.

Showing this, we now only need to verify that the updates to N_t and Q_t (in Equation 11) are equivariant for all state-action pairs along the trajectory. Values of N and Q for all other state-action pairs will be unchanged from N_t, and therefore trivially still $\mathfrak{G}$-equivariant.

First we show this for N:

$$\begin{aligned} N_{t+1}^{\mathfrak{g}_o o}(\widetilde{s}^{k-1}, \widetilde{a}^k) &= N_{t+1}^{\mathfrak{g}_o o}(\mathfrak{g}_s s^{k-1}, \mathfrak{g}_a a^k) \\ &= N_t^{\mathfrak{g}_o o}(\mathfrak{g}_s s^{k-1}, \mathfrak{g}_a a^k) + 1 \\ &= N_t^o(s^{k-1}, a^k) + 1 \\ &= N_{t+1}^o(s^{k-1}, a^k). \end{aligned}$$

Hence, Case $t+1$ still holds for N. Now we turn our attention to Q.

First, by invariance of ρ and w, we can show that $G(s^k)$ is a sum of $\mathfrak{G}$-invariant functions and therefore also invariant. Plugging into the Q update:

$$\begin{aligned}
Q_{t+1}^{\mathfrak{g}_o o}(\widetilde{s}^{k-1}, \widetilde{a}^k) &= Q_{t+1}^{\mathfrak{g}_o o}(\mathfrak{g}_s s^{k-1}, \mathfrak{g}_a a^k) \\
&= \frac{N_t^{\mathfrak{g}_o o}(\mathfrak{g}_s s^{k-1}, \mathfrak{g}_a a^k) Q_t^{\mathfrak{g}_o o}(\mathfrak{g}_s s^{k-1}, \mathfrak{g}_a a^k) + G(\mathfrak{g}_s s^{k-1})}{N_t^{\mathfrak{g}_o o}(\mathfrak{g}_s s^{k-1}, \mathfrak{g}_a a^k) + 1} \\
&= \frac{N_t^o(s^{k-1}, a^k) Q_t^o(s^{k-1}, a^k) + G(s^{k-1})}{N_t^o(s^{k-1}, a^k) + 1} \\
&= Q_{t+1}^o(s^{k-1}, a^k).
\end{aligned}$$

Hence, Case $t+1$ also holds for Q. As discussed before, we assume it holds by composition for all other state stored by MCTS (P, T, R).

Having proved that all internal state of of MCTS consistently remains transformed by $\mathfrak{G}$ under transformed input observations, we can conclude that the final policy given by MCTS will be exactly $\mathfrak{G}$-equivariant.