Analyzing Generalization in Policy Networks: A Case Study with the Double-Integrator System

Ruining Zhang[1,2*], Haoran Han[1*], Maolong Lv[3], Qisong Yang[4], Jian Cheng[1†]

[1]School of Information and Communication Engineering, University of Electronic Science and Technology of China
[2]Glasgow School, University of Electronic Science and Technology of China
[3]Air Traffic Control and Navigation College, Air Force Engineering University
[4]Xi'an Institute of High-Tech
{zhangruining0621, hanadam, maolonglv}@163.com, q.yang@tutanota.com, chengjian@uestc.edu.cn

Abstract

Extensive utilization of deep reinforcement learning (DRL) policy networks in diverse continuous control tasks has raised questions regarding performance degradation in expansive state spaces where the input state norm is larger than that in the training environment. This paper aims to uncover the underlying factors contributing to such performance deterioration when dealing with expanded state spaces, using a novel analysis technique known as state division. In contrast to prior approaches that employ state division merely as a post-hoc explanatory tool, our methodology delves into the intrinsic characteristics of DRL policy networks. Specifically, we demonstrate that the expansion of state space induces the activation function $\tanh$ to exhibit saturability, resulting in the transformation of the state division boundary from nonlinear to linear. Our analysis centers on the paradigm of the double-integrator system, revealing that this gradual shift towards linearity imparts a control behavior reminiscent of bang-bang control. However, the inherent linearity of the division boundary prevents the attainment of an ideal bang-bang control, thereby introducing unavoidable overshooting. Our experimental investigations, employing diverse RL algorithms, establish that this performance phenomenon stems from inherent attributes of the DRL policy network, remaining consistent across various optimization algorithms.

Introduction

Reinforcement learning (RL) stands as a pivotal technique for attaining artificial general intelligence. Its integration with deep learning, called DRL, empowers agents to effectively tackle intricate tasks in high-dimensional state space (Li 2018; Wang et al. 2022). DRL has demonstrated remarkable performance in diverse gaming scenarios and exhibited promising potential in addressing real-world challenges (Li 2018; François-Lavet et al. 2018). Nevertheless, empirical observations indicate that DRL agents encounter difficulties in generalizing learned behaviors to environments with variants (Cobbe et al. 2019; Wu et al. 2022; Kirk et al. 2023). As the testing state space expands, a prevalent outcome is the performance degradation of DRL agents. For example, an agent proficiently navigating a 32×32 environment might struggle to reach the target in a 64×64 environment (Hodge, Hawkins, and Alexander 2021). Despite the ubiquity of this performance degradation and the existence of various proposed remedies, the root cause remains noninterpretable and has scarcely been discussed in prior literature.

Existing solutions to generalization problems can be classified into two categories (Packer et al. 2019; Hansen et al. 2021). The first one revolves around fortifying the intrinsic robustness of policies (Morimoto and Doya 2005; Rajeswaran et al. 2017; Chen and Li 2020). These learned policies can be directly applied in unfamiliar testing environments without updating. A prominent strategy to cultivate policies impervious to environmental changes is domain randomization, which exposes agents to diverse training environments endowed with random properties (Tobin et al. 2017; Peng et al. 2018; Lee et al. 2019; Huang et al. 2021). While this always leads to suboptimal policies characterized by high variance, techniques such as regularization can alleviate these issues and enhance efficacy (Aractingi et al. 2020; Slaoui et al. 2020). Data augmentation has also demonstrated effectiveness in bolstering policy robustness (Laskin et al. 2020; Kostrikov, Yarats, and Fergus 2021; Raileanu et al. 2021). Specialized techniques such as adversarial data augmentation (Volpi et al. 2018; Bai et al. 2021; Hsu et al. 2022) and $mixreg$ (WANG et al. 2020) excel in this realm. The second category of solutions entails dynamically adapting policies to accommodate various environments. When confronted with unseen environments, agents undergo retraining to update learned policies (Xin et al. 2017; Bayerlein, De Kerret, and Gesbert 2018; Dobrevski and Skočaj 2020). In cases where the reward function in testing environments is readily available, supervised learning proves advantageous for swiftly fine-tuning existing policies (Hinton and Salakhutdinov 2006; Julian et al. 2020; Rusu et al. 2022). However, formulating an appropriate reward function often remains infeasible (Hansen et al. 2021). As an alternative, self-supervised learning is introduced to enhance the generalization of DRL policies through optimizing representation learning (Karia and Srivastava 2022; Wu et al. 2022; Ericsson et al. 2022). For example, Hansen et al. (2021) devise a joint-objective system that integrates a self-supervised objective to enhance intermediate representation

*These authors contributed equally.
†Corresponding author.

optimization through self-supervised learning.

Apart from supervised and self-supervised learning, approaches such as meta-learning (Kirsch, van Steenkiste, and Schmidhuber 2020; Guo, Wu, and Lee 2022), transfer learning (Guo et al. 2019; Vrbančič and Podgorelec 2020) and curriculum learning (Bengio et al. 2009; Park and Park 2022; Hu et al. 2022) have also demonstrated their capacity to adapt pre-trained policies to novel environments through retraining (Packer et al. 2019). Integration of techniques such safe RL is recommended within the training process to allow the agent to avoid hazardous states, thereby ensuring consistent normal performance and enhancing training efficiency (Yang et al. 2021, 2022). When confronted with expanded state space, the latter category of solutions that retrain learned policies is often favored. However, crafting and operating such an extensive state space proves inefficient and resource-intensive.

The application of state division serves as an intuitive and powerful method for interpreting DRL policies, which currently functions as an auxiliary tool for mimic learning and explicating policy performance. In the context of mimic learning, (Soares et al. 2021; Dhebar et al. 2022; Liu et al. 2023) visualize state-action patterns to refine approximations. For instance, (Liu et al. 2023) define critical experience points around decision boundaries, establishing their significance in interpretable policy distillation. In contrast, (Pascanu, Montufar, and Bengio 2014; Serra, Tjandraatmadja, and Ramalingam 2018; Hanin and Rolnick 2019; Cohan et al. 2022) delve deeper into the origins and potential impact of state division, focusing on the piecewise linear mapping from states to actions induced by activation in each node. (Pascanu, Montufar, and Bengio 2014; Serra, Tjandraatmadja, and Ramalingam 2018; Hanin and Rolnick 2019) build theories concerning on linear region counting and numbers, based on which (Cohan et al. 2022) investigate the relationship between region density along the policy trajectory and policy performance. Notably, both applications of state division solely provide post-hoc explanations for currently learned policies (Du, Liu, and Hu 2019; Kamath and Liu 2021), as division lines exhibit irregular variations across different cases. They lack generalizability to an expanded state space and diverse input conditions.

To enhance the policy network generalization in expanded state space, it is imperative to unravel the underlying mechanisms of networks. While state division serves as an intuitive tool, prior works predominantly employ it for emulating noninterpretable policies or treat each piecewise region uniformly, which offers a constrained and superficial perspective. In contrast, our approach harnesses state division to delve into the intrinsic properties of the policy network. This enables us to assess the influence of these properties on network generalization. In contrast, our approach focuses on identifying a siginificant division line in the state space and harness state division to delve into the intrinsic properties of the policy network. This enables us to assess the influence of these properties on network generalization.

This paper introduces state space division theory as a means to interpret DRL policies. The essence of this theory lies in the continuous division line in the state space, whose

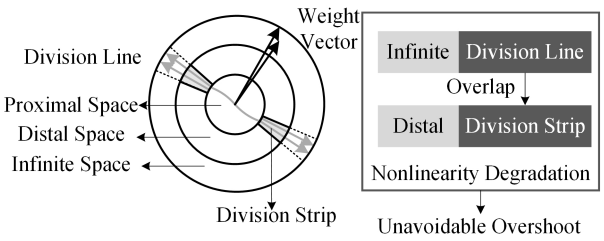

Figure 1: Overview of the state space division theory and article structure.

configuration evolves with varying state norms. While the resulting policy remains nonlinear, distince patterns emerge in the division in the state space division. As shown in Fig. 1, in the proximal state space where the state norm is small, this division line exhibits a nonlinear nature owing to the unsaturated activation function $\tanh$. As the state norm approaches infinity, the policy network approaches saturation, resulting in a predominantly linear division line in the infinite state space. Moreover, accommodating the context of a continuous action space, this division line transforms into a division strip, within which the network outputs are continuous. In the finite distal state space, closely positioned division strips overlap, encompassing a radial boundary. To empirically illustrate these principles, we leverage the double-integrator system as an example, validating that network performance degradation, such as overshoot, stems from the nonlinear decline in policy networks. The source code is available at https://github.com/Han-Adam/GeneralAnalyze.

Background

Deep Reinforcement Learning

The mathematical formulation of the RL problem is the Markov decision process (MDP) described as $(\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma)$. The agent receives the state $s_t \in \mathcal{S}$, and takes an action $a_t \in \mathcal{A}$. Then, the environment generates the next state $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$ and the reward $r_t = R(s_t)$. For any given trajectory, the discounted reward is defined as $G_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i}$. The objective of an RL algorithm is to train its policy μ to maximize the discounted reward the agent can obtain.

Network Based Control

DRL adopts neural networks, whose parameters are denoted as θ, for policy approximation. In this paper, the MLP, a concatenation of m fully connected layers of

$$z^i = \sigma(W^i z^{i-1} + b^i) \tag{1}$$

is adopted. Here, $i = 1, ..., m$ means the layer index, $W^i \in \mathbb{R}^{n_i \times n_{i-1}}$ and $b^i \in \mathbb{R}^{n_i}$ are the weight matrix and bias, n_i is the width of the feature z^i, and σ is the activation function. In particular, $z^0 = s$ is the network input. The weight matrix could be expanded as $W^i = [w_1^{iT}, w_2^{iT}, ..., w_{n_i}^{iT}]^T$, where $w_j^i \in \mathbb{R}^{n_{i-1}}$ is the weight vector.

Previous works introduce a simplified policy network, where the bias b^i in Eq. (1) is eliminated and the activation function should satisfy the condition $\sigma(0) = 0$. This approach has demonstrated enhanced efficiency in attaining stable control performance, as proven through both the Lyapunov method (Sun et al. 2022; Joshi, Virdi, and Chowdhary 2020) and an analysis of system linearization (Han et al. 2023a). Therefore, we leverage the simplified policy network in our investigation, whose fully connected layer should be rewritten as

$$z^i = \tanh(W^i z^{i-1}). \tag{2}$$

Double-Integrator

The double-integrator is described as

$$\dot{p} = v, \dot{v} = a, \tag{3}$$

where $p \in \mathbb{R}$, $v \in \mathbb{R}$, and $a \in [-\bar{a}, \bar{a}]$ respectively represent the position, velocity and acceleration of the agent. $\bar{a}$ is the acceleration bound. Without losing generality, this paper sets $\bar{a} = 5$ for the later statement.

State Space Division

Division Line

If the state is a nonzero vector, it can be expressed as $s = dl$, where $l = ||s||$ is the length of the state and $d = s/l$ is the direction of the state. To demonstrate the linear division of the policy network, we define the function

$$\phi(d) = \lim_{l \to \infty} z^1 \tag{4}$$

as the feature vector of the first layer fed with an infinite state. Here, the domain of ϕ is the unit circle of $\mathcal{C} = \{d \in \mathbb{R}^2 : ||d|| = 1\}$. This function can be expanded as

$$\begin{aligned} \phi(d) &= \lim_{l \to \infty} \tanh(W^1 dl) \\ &= \lim_{l \to \infty} [\tanh(w_1^{1T} dl), ..., \tanh(w_{n_1}^{1T} dl)]^T. \end{aligned} \tag{5}$$

Moreover, each element of $\phi(d)$ is calculated as

$$\phi_i(d) = \begin{cases} -1, & \text{if } w_i^{1T} d < 0. \\ 0, & \text{if } w_i^{1T} d = 0. \\ 1, & \text{if } w_i^{1T} d > 0. \end{cases} \tag{6}$$

This implies that, when the norm of the state goes to infinity, the first layer of the policy network encodes the state into a permutation consisting of -1, 0, and 1. Thus, ϕ is a piecewise-constant function, and the discontinuity occurs if there exist weight vectors w_i^1 such that $w_i^1 \perp d$.

Then, we can define the division direction set associated with the weight vector w_i^1 as $\mathcal{D}_i = \{d \in \mathcal{C} : d \perp w_i^1\}$. These division directions divide the unit circle $\mathcal{C}$ into multiple regions, where each region is associated with a constant feature vector. Therefore, for any given representative direction d that is not perpendicular to any weight vectors, the division region expanded from it can be defined as $\mathcal{R}(d) = \{d' \in \mathcal{C} : \phi(d') = \phi(d)\}$.

Additionally, for the sake of simplicity, this paper assumes that no weight vector is parallel to another. With this assumption, the features in any two adjacent regions have only one different element. Concretely, for any two adjacent regions $\mathcal{R}(d_1), \mathcal{R}(d_2)$, where d_1 and d_2 are the representative directions, if the direction $d \in \mathcal{D}_i$ is their boundary, one can obtain that $\phi_i(d_1) = -\phi_i(d_2)$ and $\phi_j(d_1) = \phi_j(d_2)$ for all $j \neq i$, such as that in Fig. 2.

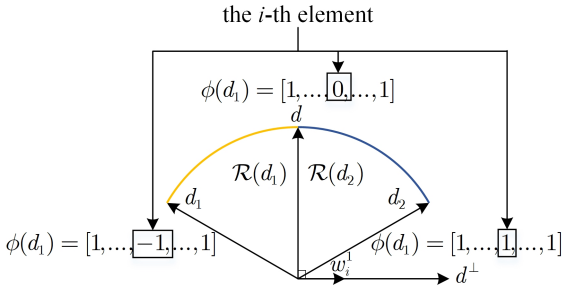

Figure 2: Illustration of ϕ among adjacent regions.

If the directions of two states s and s' fall into the same region and their norms go to infinity, the first layer of the policy network will encode them with the same feature. Therefore, the ultimate output of the policy network,

$$\bar{\phi}(d) = \lim_{l \to \infty} \bar{a} \mu_\theta(s) = \bar{a} \tanh(... \tanh(W^2 \phi(d))...), \tag{7}$$

is also divided by these division directions.

For a real policy network, there are nonlinear division boundaries in the proximal state space. That is because $\tanh$ is far from reaching saturation when the state norm is small. As the state norm grows, $\tanh$ activation gradually tends to saturate, and the feature z^1 converges to $\phi(d)$. Therefore, the division direction defined on the unit circle can be generalized to the division line defined on the whole space as

$$\bar{\mathcal{D}}_i = \{s \in \mathcal{S} : s \perp w_i^1\}. \tag{8}$$

Division Strip

Since the policy network is a continuous function, it is inadequate to characterize its behavior using discontinuous division lines. More realistically, the policy network generates division strips around these lines, indicating that the output changes continuously as the state transforms between adjacent regions. Hence, we define the strip function as

$$\psi(d, d^\perp, x) = \lim_{l \to \infty} z^1 = \lim_{l \to \infty} \tanh(W^1(dl + d^\perp x)), \tag{9}$$

where $d^\perp$ is perpendicular to d. The function ψ assumes the state extends infinitely in the main direction d but has a finite offset of x in the direction of $d^\perp$. For a weight vector such that $w_i^{1T} d \neq 0$, the offset has no effect on the output since $\lim_{l \to \infty} w_i^{1T}(dl + d^\perp x) = \text{sign}(w_i^{1T} d)\infty + w_i^{1T} d^\perp x = \text{sign}(w_i^{1T} d)\infty$. Without loss generality, this paper assumes $w_i^{1T} d^\perp > 0$ and each element of ψ can be calculated as

$$\psi_i(d, d^\perp, x) = \begin{cases} -1, & \text{if } w_i^{1T} d < 0. \\ \tanh(||w_i^1|| x), & \text{if } w_i^{1T} d = 0. \\ 1, & \text{if } w_i^{1T} d > 0. \end{cases} \tag{10}$$

When the main direction d is not perpendicular to any weight vector, the function ψ is equal to ϕ. That means the

offset has no effect on the feature if the main direction is inside a division region. When the main direction d is a division direction, without losing generality, we assume that $w_i^{1T} d_1 > 0$, $w_i^{1T} d_2 < 0$, $\text{sign}(w_j^{1T} d_1) = \text{sign}(w_j^{1T} d_2) = \text{sign}(w_j^{1T} d)$ for all $j \neq i$, and d_1 and d_2 are the representative vectors of the adjacent regions, $\mathcal{R}(d_1)$ and $\mathcal{R}(d_2)$, where d is their boundary, such as that in Fig. 2. Hence, $\psi_j(d, d^\perp, x) = \phi_j(d_1) = \phi_j(d_2)$ for all $j \neq i$, and

$$\begin{aligned} \lim_{x \to \infty} \psi_i(d, d^\perp, x) &= 1 = \phi_i(d_1), \\ \lim_{x \to -\infty} \psi_i(d, d^\perp, x) &= -1 = \phi_i(d_2). \end{aligned} \tag{11}$$

That implies that $\psi(d, d^\perp, x)$ converges to $\phi(d_1)$ or $\phi(d_2)$ as the offset x grows. Considering that $\psi(d, d^\perp, 0) = \phi(d)$, the function shows how the feature signal z^1 gradually changes among the adjacent division regions.

Consequently, the ultimate output of the policy network,

$$\begin{aligned} \bar{\psi}(d, d^\perp, x) &= \lim_{l \to \infty} \bar{a} \mu_\theta(s) \\ &= \bar{a} \tanh(... \tanh(W^2 \psi(d, d^\perp, x))...) \end{aligned} \tag{12}$$

also gradually changes. Therefore, the policy network generates a strip around the division line $\bar{\mathcal{D}}_i$, and the network output gradually changes within this strip.

In infinite space, it can be asserted that the alteration in the direction of the state leads to the modification of solely one element in the feature of the first layer z^1, at any given moment. However, these strips and adjacent division regions may overlap in the finite distal state space. Specifically, if the directions of a cluster of weight vectors are in proximity, their corresponding division lines will also be closely positioned. Hence, the regions demarcated by these lines become considerably narrower. Therefore, these strips and regions collectively form an overlapped division strip wherein multiple elements of z_1 alter as the state direction changes.

Weight Vector Significance

The division line $\bar{\mathcal{D}}_i$ is perpendicular to the weight vector w_i^1. However, not all division lines contribute equally to the network performance. If we assume the direction d is perpendicular to w_i^1, and denote $\delta_i = \tanh(||w_i^1||x)$ as the offset on the feature vector, then the strip function of Eq. (9) can be rewritten as $\psi(d, \delta_i)$, where

$$\psi_j(d, \delta_i) = \begin{cases} -1, & \text{if } i \neq j \wedge w_i^{1T} d < 0. \\ \delta_i, & \text{if } i = j. \\ 1, & \text{if } i \neq j \wedge w_i^{1T} d > 0. \end{cases} \tag{13}$$

Then, the ultimate network output can be rewritten as

$$\bar{\psi}(d, \delta_i) = \tanh(... \tanh(W^2 \psi(d, \delta_i))...), \tag{14}$$

and the significance of the division line $\bar{D}_i$ is defined as

$$\rho_i = |\bar{\psi}(d, 1) - \bar{\psi}(d, -1)|. \tag{15}$$

Noticing that $\bar{\psi}(d, 1)$ and $\bar{\psi}(d, -1)$ are the network outputs of two adjacent division regions, the significance of the weight vector indicates the difference of the network outputs when the state transforms between adjacent regions. Thus, significant disparities across the line imply noticeable changes in network output.

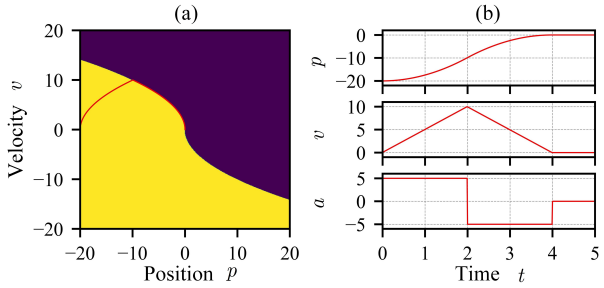

Figure 3: (a) State trajectory (red) and state-action pattern of the ideal bang-bang control. (b) Response of p, v, and a via time under the control law of Eq. (17).

Unavoidable Overshoot

According to the preceding discussion, it is observed that as the input state tends towards infinity, the policy network generates a division line that separates the state space into two regions, wherein the acceleration output approximates either $-\bar{a}$ or $\bar{a}$. This control scheme bears resemblance to a bang-bang control. Nevertheless, an ideal analytic bang-bang control exhibits a division line denoted as

$$2\bar{a}p = -\text{sign}(v)v^2. \tag{16}$$

Thus, its output acceleration can be represented as

$$a = \begin{cases} -\bar{a}, & \text{if } 2\bar{a}p > -\text{sign}(v)v^2, \\ 0, & \text{if } 2\bar{a}p = -\text{sign}(v)v^2, \\ \bar{a}, & \text{if } 2\bar{a}p < -\text{sign}(v)v^2, \end{cases} \tag{17}$$

and its state-action pattern is shown in Fig. 3 (a). This particular control algorithm is time-optimal (Wang et al. 2018) and can attain non-overshoot control performance. However, the policy network is limited in generating a linear division line, thereby unable to approximate the optimal division line specified in Eq. (16). Consequently, time-optimal control cannot be achieved using the policy network. When the initial position error is relatively large, the agent will commence decelerating later, leading to unavoidable overshoots.

Subsequently, we assert that if the policy network confronts considerable initial error, the occurrence of overshoot is unavoidable. This outcome is not primarily attributed to differences in optimization objects (DRL optimizes the discounting reward rather than the time consumption), but is an inherent consequence of the network saturability.

Experiments

Artificially Constructed Examples

To demonstrate how the weight vector and its significance characterize the network performance, we construct a network with parameters of

$$W^1 = \begin{bmatrix} \frac{1}{2} & \frac{\sqrt{3}}{2} \\ 0 & 1 \end{bmatrix}, W^2 = \begin{bmatrix} 2 & -1 \\ 2 & \frac{1}{2} \end{bmatrix}, W^3 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}. \tag{18}$$

As shown in Fig. 4 (a), the unit circle is divided by $\mathcal{D}_1 = \{[\sqrt{3}/2, -1/2]^T, [-\sqrt{3}/2, 1/2]^T\}$ and $\mathcal{D}_2 =$

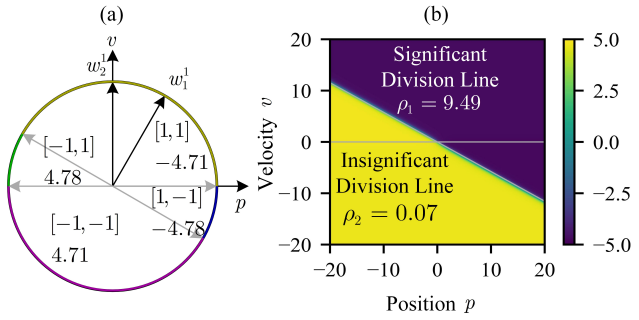

Figure 4: (a) Unit circle divided by the division directions perpendicular to the weight vector. (b) State-action pattern divided by two division lines.

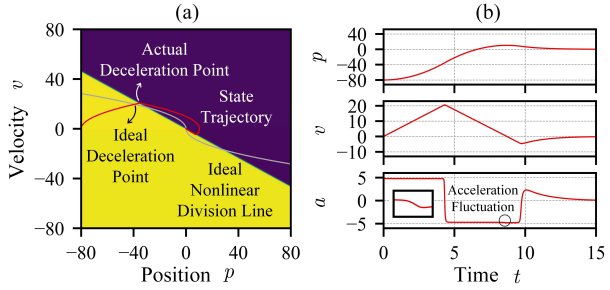

Figure 5: (a) The state trajectory (red) and ideal nonlinear division line (gray). (b) The response of p, v, and a via time.

$\{[1,0]^T, [-1,0]^T\}$ into four regions, each of which is assigned a different ϕ and ψ. However, as shown in Fig. 4 (b), only the division line of $\bar{\mathcal{D}}_1 = \{s : s = [\sqrt{3}/2, -1/2]^T l, l \in \mathbb{R}\}$ is significant. The influence of $\bar{\mathcal{D}}_2 = \{s : v = 0\}$ on the action is unobservable.

Then, we apply the exemplary network to accomplish the control task. The state trajectory and the responses of p, v, and a are shown in Fig. 5. The agent first accelerates and then decelerates after its state crosses $\bar{D}_1$. The acceleration has a slight fluctuation of 0.07 at $t = 8.71$, where the velocity becomes zero and the state crosses the $\bar{D}_2$. This fluctuation is negligible and has little impact on performance.

Therefore, the direction of division lines is determined by the weight vectors of the first layer while the magnitude of the significance is dependent on subsequent layer parameters. Only those significant division lines determine the performance of the policy network.

Examples Trained with Realistic Conditions

In the subsequent section, we analyze experimental results derived from networks trained under realstic conditions to validate the theory regarding linear state space division and its implications for the generalization of network controller. The neural network employed in experiments is a simplified policy network comprised of three fully connected layers, each containing 32 nodes. To ensure the broad applicability of our experiments, four prominent DRL algorithms, DDPG

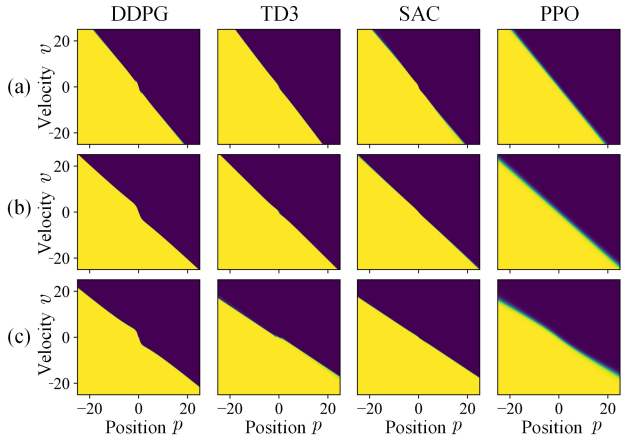

Figure 6: State-action patterns of the best-performing agents using different algorithms.

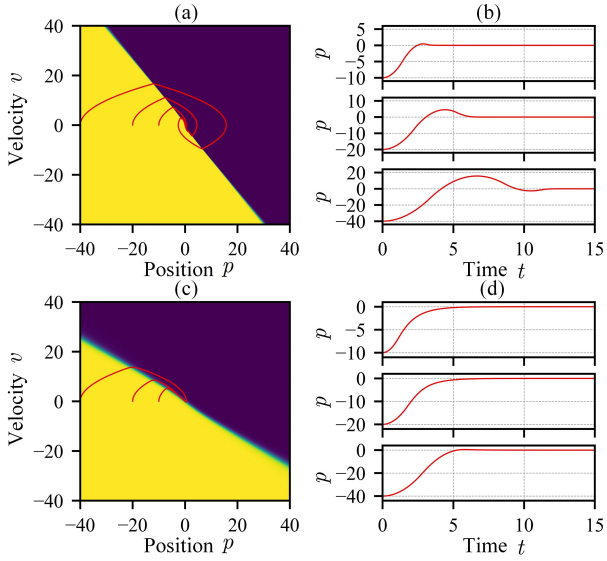

Figure 7: Performance of the network with different slopes of the division line facing different test cases.

(Lillicrap et al. 2019), TD3 (Fujimoto, van Hoof, and Meger 2018), SAC (Haarnoja et al. 2019) and PPO (Schulman et al. 2017) are selected as examples.

Universality of Linear Division Line The state-action patterns of the best-performing agent are shown in Fig. 6. In all cases, the initial velocity is set to zero and the initial position is set to $-10, -20$ and -40. A division line is evident in all testing cases, and it exhibits a nonlinear nature in the proximal state space, where the activation function $\tanh$ is unsaturated. As the state norm grows, $\tanh$ saturates, and these division lines become essentially linear when observed from the whole state space. Consequently, this linear division of the state space is a universal phenomenon that is determined by the intrinsic structure of the neural network and is irrelevant to the optimization algorithm.

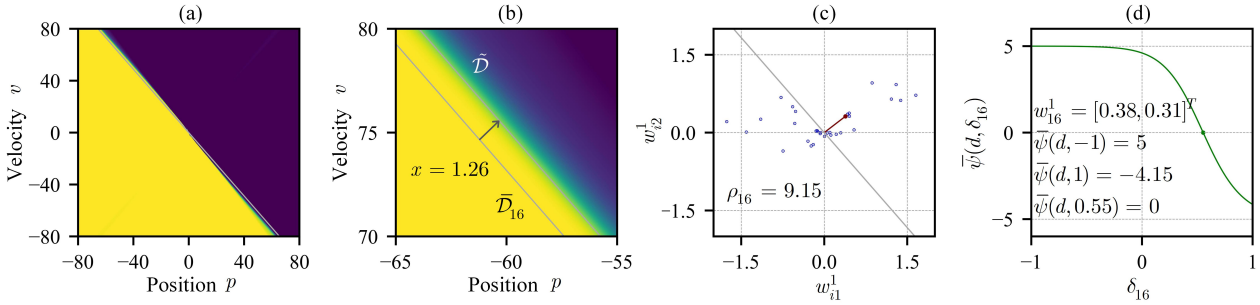

Figure 8: Analysis of network in DDPG (a) in Fig. 6, where $\tilde{\mathcal{D}} = \{s \in \mathcal{S} : \mu_\theta(s) = 0\}$ and $\bar{\mathcal{D}}_{16} = \{s \in \mathcal{S} : s \perp w^1_{16}\}$.

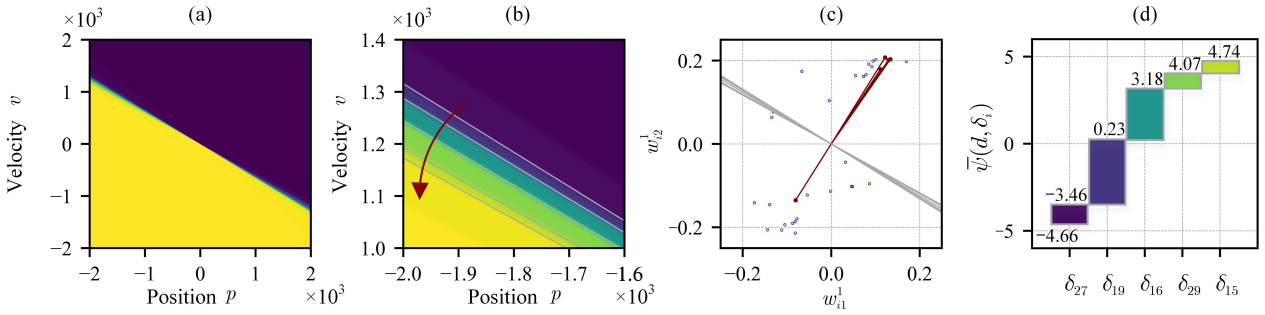

Figure 9: Analysis of network in PPO (c) in Fig. 6.

Impact of Linear Division Another notable phenomenon is the flattening of division lines as the initial position increases. It is analyzed by examining the state-action pattern of a general agent that does not exhibit optimal performance. As depicted in Fig. 5 (a), the state trajectory interacts with both the ideal division line and the actual division line, with their corresponding intersections defined as the ideal deceleration point and the actual deceleration point. The alignment of these two points indicates that the timing of deceleration for an agent is ideal, enabling it to achieve optimal performance. Hence, the division lines for the best-performing cases illustrated in Fig. 6 should intersect both the ideal deceleration point and the origin. Additionally, based on the characteristics of ideal bang-bang control, when the initial velocity is set to zero, the abscissa of the ideal deceleration point should be half the initial position error. Therefore, the ideal deceleration point moves away from the origin along the ideal division line with increasing initial position error, causing a flattening of the actual division line.

The following examples are evaluated to substantiate the aforementioned analysis. Figs. 7 (a) and (c) present DDPG (a) and PPO (c) in Fig. 6 with visualized state trajectories starting at -10, -20 and -40. Figs. 7 (b) and (d) display the corresponding position responses. In the context of a steeper division line, the agent starting at -5 swiftly reaches the origin, approximating optimal performance. However, agents beginning at -20 and -40 both exhibit late deceleration, resulting in noticeable overshoot. Conversely, for a flatter division line, the agent starting at -40 approximates its optimal performance, while those starting at -20 and -10 gradually approach the origin, demonstrating conservative behavior.

Examples of Division Strips This analysis examines cases related to division strips. Fig. 8 involves a policy network where $\bar{\mathcal{D}}_{16}$ represents the perpendiculars of the weight vector $w^1_{16} = [0.38, 0.31]^T$, bearing the highest significance of $\rho = 9.15$. As shown in Fig. 8 (b), the practical division line $\tilde{\mathcal{D}} = \{s \in \mathcal{S} : \mu_\theta(s) = 0\}$ deviates from $\bar{\mathcal{D}}_{16}$ by $x = 1.26$ in the perpendicular direction of $\bar{\mathcal{D}}_{16}$. This deviation causes an offset δ_{16} on the feature weight vector, computed as 0.55 using the strip function $\delta_i = \tanh(||w^1_i||x)$. Notably, $\delta_{16} = 0.55$ precisely corresponds to $\bar{\psi}(d, \delta_{16}) = 0$ in Fig. 8 (d). This implies that altering the direction of a state vector can induce an offset δ_i on the feature weight vector, ultimately influencing the network output.

An intriguing anomaly is observed in Fig. 9 (a), where the linear division line assumes a radial and blurry configuration. This phenomenon arises due to the existence of five closely positioned weight vectors, all of which bear comparable significance. These weight vectors generate five perpendicular division lines, as shown in Fig. 9 (c). Upon closer examination in Fig. 9 (b), the state-action patterns in the distal state space disclose a divergence of the division line cluster, manifesting in a substantial inter-line distance. As the state norm decreases, these strips mutually overlap with the progressive convergence of division lines, leading to the formation of a radial division boundary. In Fig. 9 (c), the network output within these strips is visualized. As state changes in the direction shown in Fig. 9 (b), the output exhibits a noticeable variation across each strip, which corresponds to their significance.

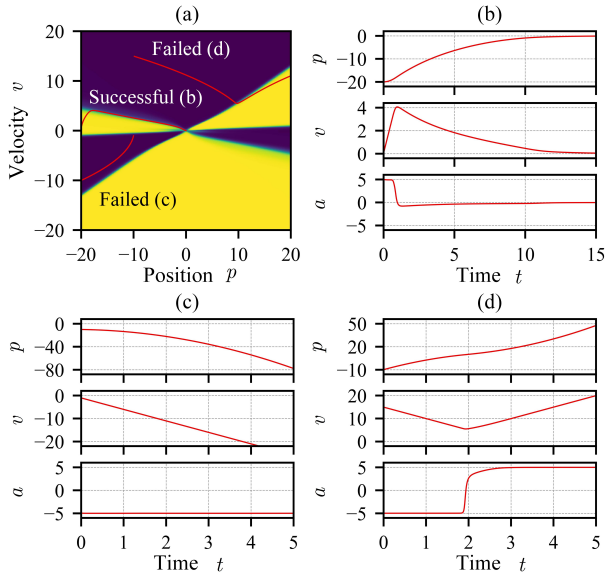

Figure 10: Illustration of a failed policy network. Dead zones exist in the state-action pattern (a). (b) showcases the response of a successful case, while (c) and (d) present the failed ones.

Failed Policy Network Furthermore, an unsuccessful policy network is examined to ascertain the reason for its failure. As depicted in Fig. 10 (a), these state trajectories predominantly align with division lines, with some diverging and failing to converge at the origin. This can be attributed to the fact that the state of these agents falls within a dead zone created by radial division lines. Fig. 10 (a) shows that trajectories (d) and (c) both enter dead zones, where the position, velocity and generated acceleration share the same sign. Additionally, this dead zone uninterruptedly expands according to the definition of the division region $\mathcal{R}(d)$. Hence, once an agent enters a dead zone, its deviation from the target accelerates, leading to a permanent failure to reach it. Therefore, it is confidently asserted that the presence of a dead zone in the state space is the primary catalyst for the failure of a policy network.

General Policy Network To simplify our exploration, we have thus far focused on the performance of a simplified policy network employing $\tanh$ as the activation function. In this section, our investigation shifts to the state space division in general policy networks, which encompasses the bias b^i and adopts ReLU as the activation function. Visualized state-action patterns for this network are shown in Fig. 11, where linear division lines are evident in the state space. However, for a general network, the linear division line is subject to unpredictability, as it remains uncertain whether the division line is perpendicular to weight vectors. While the aforementioned examples may not elucidate the factors affecting division line patterns, they unequivocally establish the prevalence of linear state space division in general policy networks.

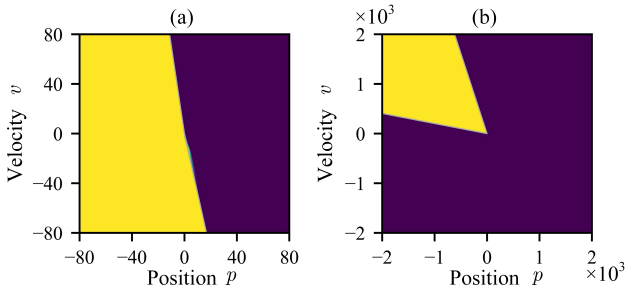

Figure 11: (a) Successful and (b) failed state-action patterns of general policy networks.

Conclusion

In this study, we uncover the emergence of linear state space division as policy networks expand across the entirety of space. Through a comprehensive analysis, this paper investigates the ramifications of this phenomenon on network performance, employing the double-integrator system as an illustrative example. By leveraging empirical data, this paper demonstrates that the linear division of the state space arises from the inherent saturability of the network and is independent of the optimization algorithms utilized. Furthermore, this paper illustrates how this linear division hampers a policy network's ability to accurately approximate the nonlinear ideal bang-bang control, resulting in inevitable overshooting. This revelation not only elucidates the reason for the network performance degradation in expanded state space but also implies the inherent limitations in the network's generalization capabilities. It can only achieve optimal control in specific scenarios while incurring significant performance degradation in others.

A current approach to addressing non-linearity degradation in distal state space involves generating a more conservative policy through the state regulation (Han et al. 2023b). This technique focuses on constraining the agent's velocity within a predefined range. However, this approach compromises policy performance due to imposed velocity limits and the challenge still lies in the absence of an effective method to rectify the underlying network performance degradation due to the intrinsic structure of neural networks.

In addition to devising techniques for enhancing network performance, our focus extends to evaluating the applicability of our analysis method within high-dimensional state space. Specifically, we are intrigued by the possibility of transitioning from a division line to a division hyperplane and the potential impact of this transition on the policy network performance. Moreover, how the degradation impact a nonlinear and high-dimension system will also be discussed in the future work.

Acknowledgments

This work was supported in part by the NNSFC&CAAC under Grants U2233209, in part by the Natural Science Foundation of Sichuan, China under Grant 2023NSFSC0484.

References

Aractingi, M.; Dance, C.; Perez, J.; and Silander, T. 2020. Improving the Generalization of Visual Navigation Policies using Invariance Regularization.

Bai, T.; Luo, J.; Zhao, J.; Wen, B.; and Wang, Q. 2021. Recent Advances in Adversarial Training for Adversarial Robustness. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 4312–4321. International Joint Conferences on Artificial Intelligence Organization.

Bayerlein, H.; De Kerret, P.; and Gesbert, D. 2018. Trajectory Optimization for Autonomous Flying Base Station via Reinforcement Learning. In *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 1–5.

Bengio, Y.; Louradour, J.; Collobert, R.; and Weston, J. 2009. Curriculum Learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, 41–48. New York, NY, USA: Association for Computing Machinery.

Chen, S.; and Li, Y. 2020. An Overview of Robust Reinforcement Learning. In *2020 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, 1–6.

Cobbe, K.; Klimov, O.; Hesse, C.; Kim, T.; and Schulman, J. 2019. Quantifying Generalization in Reinforcement Learning. In Chaudhuri, K.; and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 1282–1289. PMLR.

Cohan, S.; Kim, N. H.; Rolnick, D.; and van de Panne, M. 2022. Understanding the Evolution of Linear Regions in Deep Reinforcement Learning. arXiv:2210.13611.

Dhebar, Y.; Deb, K.; Nageshrao, S.; Zhu, L.; and Filev, D. 2022. Toward Interpretable-AI Policies Using Evolutionary Nonlinear Decision Trees for Discrete-Action Systems. *IEEE Transactions on Cybernetics*, 1–13.

Dobrevski, M.; and Skočaj, D. 2020. Adaptive Dynamic Window Approach for Local Navigation. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 6930–6936.

Du, M.; Liu, N.; and Hu, X. 2019. Techniques for Interpretable Machine Learning. *Commun. ACM*, 63(1): 68–77.

Ericsson, L.; Gouk, H.; Loy, C. C.; and Hospedales, T. M. 2022. Self-Supervised Representation Learning: Introduction, advances, and challenges. *IEEE Signal Processing Magazine*, 39(3): 42–62.

François-Lavet, V.; Henderson, P.; Islam, R.; Bellemare, M. G.; and Pineau, J. 2018. An Introduction to Deep Reinforcement Learning. *Foundations and Trends® in Machine Learning*, 11(3-4): 219–354.

Fujimoto, S.; van Hoof, H.; and Meger, D. 2018. Addressing Function Approximation Error in Actor-Critic Methods. arXiv:1802.09477.

Guo, Y.; Shi, H.; Kumar, A.; Grauman, K.; Rosing, T.; and Feris, R. 2019. SpotTune: Transfer Learning Through Adaptive Fine-Tuning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 4800–4809.

Guo, Y.; Wu, Q.; and Lee, H. 2022. Learning Action Translator for Meta Reinforcement Learning on Sparse-Reward Tasks. In *AAAI*, 6792–6800.

Haarnoja, T.; Zhou, A.; Hartikainen, K.; Tucker, G.; Ha, S.; Tan, J.; Kumar, V.; Zhu, H.; Gupta, A.; Abbeel, P.; and Levine, S. 2019. Soft Actor-Critic Algorithms and Applications. arXiv:1812.05905.

Han, H.; Cheng, J.; Xi, Z.; and Lv, M. 2023a. Symmetric actor–critic deep reinforcement learning for cascade quadrotor flight control. *Neurocomputing*, 559: 126789.

Han, H.; Xi, Z.; Cheng, J.; and Lv, M. 2023b. Obstacle Avoidance Based on Deep Reinforcement Learning and Artificial Potential Field. In *2023 9th International Conference on Control, Automation and Robotics (ICCAR)*, 215–220.

Hanin, B.; and Rolnick, D. 2019. Complexity of Linear Regions in Deep Networks. arXiv:1901.09021.

Hansen, N.; Jangir, R.; Sun, Y.; Alenyà, G.; Abbeel, P.; Efros, A. A.; Pinto, L.; and Wang, X. 2021. Self-Supervised Policy Adaptation during Deployment. arXiv:2007.04309.

Hinton, G. E.; and Salakhutdinov, R. R. 2006. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786): 504–507.

Hodge, V. J.; Hawkins, R.; and Alexander, R. 2021. Deep Reinforcement Learning for Drone Navigation Using Sensor Data. *Neural Comput. Appl.*, 33(6): 2015–2033.

Hsu, C.-Y.; Chen, P.-Y.; Lu, S.; Liu, S.; and Yu, C.-M. 2022. Adversarial Examples Can Be Effective Data Augmentation for Unsupervised Machine Learning. In *AAAI*, 6926–6934.

Hu, Z.; Gao, X.; Wan, K.; Wang, Q.; and Zhai, Y. 2022. Asynchronous Curriculum Experience Replay: A Deep Reinforcement Learning Approach for UAV Autonomous Motion Control in Unknown Dynamic Environments. arXiv:2207.01251.

Huang, J.; Guan, D.; Xiao, A.; and Lu, S. 2021. FSDR: Frequency Space Domain Randomization for Domain Generalization. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 6887–6898.

Joshi, G.; Virdi, J.; and Chowdhary, G. 2020. Design and flight evaluation of deep model reference adaptive controller. In *AIAA Scitech 2020 Forum*, volume 6, 1–17.

Julian, R.; Swanson, B.; Sukhatme, G. S.; Levine, S.; Finn, C.; and Hausman, K. 2020. Never Stop Learning: The Effectiveness of Fine-Tuning in Robotic Reinforcement Learning. arXiv:2004.10190.

Kamath, U.; and Liu, J. 2021. *Post-Hoc Interpretability and Explanations*, 167–216. Springer International Publishing.

Karia, R.; and Srivastava, S. 2022. Relational Abstractions for Generalized Reinforcement Learning on Symbolic Problems. arXiv:2204.12665.

Kirk, R.; Zhang, A.; Grefenstette, E.; and Rocktäschel, T. 2023. A Survey of Zero-shot Generalisation in Deep Reinforcement Learning. *Journal of Artificial Intelligence Research*, 76: 201–264.

Kirsch, L.; van Steenkiste, S.; and Schmidhuber, J. 2020. Improving Generalization in Meta Reinforcement Learning using Learned Objectives. arXiv:1910.04098.

Kostrikov, I.; Yarats, D.; and Fergus, R. 2021. Image Augmentation Is All You Need: Regularizing Deep Reinforcement Learning from Pixels. arXiv:2004.13649.

Laskin, M.; Lee, K.; Stooke, A.; Pinto, L.; Abbeel, P.; and Srinivas, A. 2020. Reinforcement Learning with Augmented Data. arXiv:2004.14990.

Lee, K.; Lee, K.; Shin, J.; and Lee, H. 2019. Network Randomization: A Simple Technique for Generalization in Deep Reinforcement Learning. In *International Conference on Learning Representations*.

Li, Y. 2018. Deep Reinforcement Learning: An Overview. arXiv:1701.07274.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2019. Continuous control with deep reinforcement learning. arXiv:1509.02971.

Liu, X.; Liu, S.; An, B.; Gao, Y.; Yang, S.; and Li, W. 2023. Effective Interpretable Policy Distillation via Critical Experience Point Identification. *IEEE Intelligent Systems*, 1–10.

Morimoto, J.; and Doya, K. 2005. Robust Reinforcement Learning. *Neural Computation*, 17(2): 335–359.

Packer, C.; Gao, K.; Kos, J.; Krähenbühl, P.; Koltun, V.; and Song, D. 2019. Assessing Generalization in Deep Reinforcement Learning. arXiv:1810.12282.

Park, J.; and Park, K. 2022. Indoor Path Planning for Multiple Unmanned Aerial Vehicles via Curriculum Learning. In *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*, 1238–1241. IEEE.

Pascanu, R.; Montufar, G.; and Bengio, Y. 2014. On the number of response regions of deep feed forward networks with piece-wise linear activations. arXiv:1312.6098.

Peng, X. B.; Andrychowicz, M.; Zaremba, W.; and Abbeel, P. 2018. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*.

Raileanu, R.; Goldstein, M.; Yarats, D.; Kostrikov, I.; and Fergus, R. 2021. Automatic Data Augmentation for Generalization in Deep Reinforcement Learning. arXiv:2006.12862.

Rajeswaran, A.; Ghotra, S.; Ravindran, B.; and Levine, S. 2017. EPOpt: Learning Robust Neural Network Policies Using Model Ensembles. arXiv:1610.01283.

Rusu, A. A.; Rabinowitz, N. C.; Desjardins, G.; Soyer, H.; Kirkpatrick, J.; Kavukcuoglu, K.; Pascanu, R.; and Hadsell, R. 2022. Progressive Neural Networks. arXiv:1606.04671.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347.

Serra, T.; Tjandraatmadja, C.; and Ramalingam, S. 2018. Bounding and Counting Linear Regions of Deep Neural Networks. arXiv:1711.02114.

Slaoui, R. B.; Clements, W. R.; Foerster, J. N.; and Toth, S. 2020. Robust Visual Domain Randomization for Reinforcement Learning. arXiv:1910.10537.

Soares, E.; Angelov, P. P.; Costa, B.; Castro, M. P. G.; Nageshrao, S.; and Filev, D. 2021. Explaining Deep Learning Models Through Rule-Based Approximation and Visualization. *IEEE Transactions on Fuzzy Systems*, 29(8): 2399–2407.

Sun, R.; Greene, M. L.; Le, D. M.; Bell, Z. I.; Chowdhary, G.; and Dixon, W. E. 2022. Lyapunov-Based Real-Time and Iterative Adjustment of Deep Neural Networks. *IEEE Control Systems Letters*, 193–198.

Tobin, J.; Fong, R.; Ray, A.; Schneider, J.; Zaremba, W.; and Abbeel, P. 2017. Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World. arXiv:1703.06907.

Volpi, R.; Namkoong, H.; Sener, O.; Duchi, J. C.; Murino, V.; and Savarese, S. 2018. Generalizing to Unseen Domains via Adversarial Data Augmentation. In Bengio, S.; Wallach, H.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Vrbančič, G.; and Podgorelec, V. 2020. Transfer Learning With Adaptive Fine-Tuning. *IEEE Access*, 8: 196197–196211.

Wang, G.; Wang, L.; Xu, Y.; and Zhang, Y. 2018. *Time Optimal Control of Evolution Equations*, chapter 6. Cham, Switzerland: Birkhäuser.

WANG, K.; Kang, B.; Shao, J.; and Feng, J. 2020. Improving Generalization in Reinforcement Learning with Mixture Regularization. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 7968–7978. Curran Associates, Inc.

Wang, X.; Wang, S.; Liang, X.; Zhao, D.; Huang, J.; Xu, X.; Dai, B.; and Miao, Q. 2022. Deep Reinforcement Learning: A Survey. *IEEE Transactions on Neural Networks and Learning Systems*, 1–15.

Wu, K.; Wu, M.; Chen, Z.; Xu, Y.; and Li, X. 2022. Generalizing Reinforcement Learning through Fusing Self-Supervised Learning into Intrinsic Motivation. In *AAAI*, 8683–8690.

Xin, J.; Zhao, H.; Liu, D.; and Li, M. 2017. Application of deep reinforcement learning in mobile robot path planning. In *2017 Chinese Automation Congress (CAC)*, 7112–7116.

Yang, Q.; Simão, T. D.; Tindemans, S. H.; and Spaan, M. T. J. 2022. Safety-Constrained Reinforcement Learning with a Distributional Safety Critic. *Mach. Learn.*, 112(3): 859–887.

Yang, Q.; Simão, T. D.; Tindemans, S. H.; and Spaan, M. T. J. 2021. WCSAC: Worst-Case Soft Actor Critic for Safety-Constrained Reinforcement Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12): 10639–10646.