Diffusion Guided Adaptive Augmentation for Generalization in Visual Reinforcement Learning

Jeong Woon Lee and Hyoseok Hwang*
Department of Software Convergence, Kyung Hee University, Republic of Korea
{everyman123,hyoseok}@khu.ac.kr

Abstract

Reinforcement learning (RL) has proven its potential in complex decision-making tasks. Yet, many RL systems rely on manually crafted state representations, requiring effort in feature engineering. Visual Reinforcement Learning (VRL) offers a way to address this challenge by enabling agents to learn directly from raw visual input. Nonetheless, VRL continues to face generalization issues, as models often overfit to specific domain features. To tackle this issue, we propose Diffusion Guided Adaptive Augmentation (DGA2), an augmentation method that utilizes Stable Diffusion to enhance domain diversity. We introduce an Adaptive Domain Shift strategy that dynamically adjusts the degree of domain shift according to the agent's learning progress for effective augmentation with Stable Diffusion. Additionally, we employ saliency as the mask to preserve the semantics of data. Our experiments on the DMControl-GB, Adroit, and Procgen environments demonstrate that DGA2 improves generalization performance compared to existing data augmentation and generalization methods. Project page: https://airlabkhu.github.io/DGA2/.

1. Introduction

In recent years, the integration of Reinforcement Learning (RL) with Deep Neural Networks (DNNs) has led to significant advancements in fields that demand complex decision-making [17]. However, RL systems often depend on manually crafted state representations, which necessitates effort in feature engineering. To overcome this issue, Visual Reinforcement Learning (VRL) can be a promising approach. VRL enables agents to learn directly from raw visual inputs, minimizing the need for predefined state representations [22]. VRL expands the applicability of RL to a wide range of tasks where visual information is abundant, including complex game-playing [24] and robotic manipulation [21].

*Corresponding author (hyoseok@khu.ac.kr)

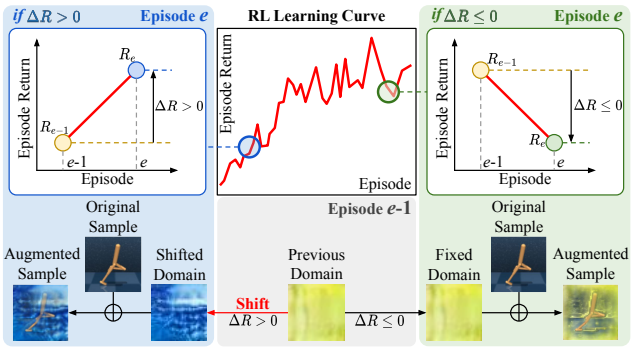


Figure 1. Overview of Adaptive Domain Shift. Adaptive Domain Shift (ADS) adjusts the extent of domain shift based on episode return changes. If the episode return improves, ADS shifts the learning domain according to the change in episode return. Otherwise, the agent continues learning in the previous domain.

Despite these advances, VRL faces significant challenges in generalization [9]. Agents frequently overfit to specific domain features during training, resulting in degraded performance when encountering novel environments [7]. Addressing these generalization challenges is crucial for deploying VRL in real-world applications [32]. While various approaches such as distillation [5, 9] and self-supervised learning [6, 35, 40] have shown promise, they primarily focus on feature extraction rather than increasing domain diversity. In this context, data augmentation emerges as a compelling solution by introducing the necessary domain variability during training.

Established data augmentation techniques, such as flipping, shifting, and random convolution, have been shown to effectively increase the diversity of training samples [16]. A further enhancement in diversity can be achieved by incorporating images from external datasets, a strategy that significantly broadens domain coverage [11]. More advanced approaches leverage normalization techniques [18, 19, 41] and Fourier transform-based methods [14], which introduce variations while preserving essential structural properties. Despite their effectiveness, these methods primarily depend

on predefined transformations or limited dataset distributions, which inherently constrain the extent of domain variability.

Stable Diffusion [27], a generative model trained on large-scale datasets such as LAION [28], offers a compelling alternative to these conventional augmentation strategies. With its capacity to generate highly diverse and contextually rich samples, Stable Diffusion effectively enhances domain diversity [15, 20, 34]. This characteristic is particularly beneficial in VRL, where agents must develop policies that generalize well across a wide array of unseen domains.

However, directly applying Stable Diffusion to VRL presents a challenge: while it enables flexible domain generation, the degree of domain shift must be carefully regulated to align with the agent's learning progress and ensure stable training. Uncontrolled variations in domain characteristics lead to abrupt or inconsistent shifts, potentially disrupting the learning process. Thus, it is crucial to design a method that not only introduces diverse domain variations but also adaptively adjusts the extent of domain shift based on the agent's evolving learning dynamics. To effectively leverage Stable Diffusion in VRL, domain variations must be systematically controlled to maintain a balance between diversity and training stability.

In this study, we propose Diffusion Guided Adaptive Augmentation (DGA2), a novel method that leverages Stable Diffusion for domain augmentation in VRL. As shown in Figure 1, DGA2 introduces an Adaptive Domain Shift (ADS) strategy that dynamically adjusts the domain variations induced by Stable Diffusion based on fluctuations in the agent's episode return. Specifically, when the episode return improves, ADS shifts the learning domain according to the change in episode return. Conversely, in the absence of improvement, the agent continues training in the same domain. By adjusting the domain shift based on performance, ADS promotes training stability while enhancing domain diversity. Additionally, DGA2 employs a saliency-based preservation mechanism that retains task-relevant information by leveraging gradients from the agent's value network inspired by [3]. We evaluate DGA2 on DMControl-GB [12], Adroit [39], and Procgen [8], demonstrating its superior generalization performance compared to existing augmentation and generalization methods. The main contributions of this study can be summarized as follows:

- We propose Diffusion Guided Adaptive Augmentation (DGA2), an augmentation method that harnesses Stable Diffusion to generate diverse domain variations.
- We introduce the Adaptive Domain Shift strategy, which dynamically modulates the domain variations generated by Stable Diffusion based on the agent's episode return to ensure stable training.
- We conduct comprehensive experiments on DMControl-GB, Adroit, and Procgen, demonstrating that DGA2 surpasses existing augmentation techniques and generalization methods.

2. Related Work

2.1. Augmentation in Visual RL

Data augmentation techniques have been introduced to expand the training domain [23]. For instance, Reinforcement Learning with Augmented Data (RAD) [16] uses pixel-level transformations and improves robustness across different domains. However, simple image transformations can risk losing task-relevant information critical for decision-making. To mitigate this, there are approaches that enhance data diversity while maintaining semantic integrity. Spectrum Random Masking (SRM) [14] avoids altering the image directly by masking certain frequency components in the Fourier domain, retaining the overall structure. Similarly, MixStyle [41] mixies style while keeping the semantic content intact through normalization in feature space. Based on these foundations, other methods introduce adaptive augmentation strategies that dynamically modulate augmentation strength for greater flexibility. Cross-Norm and Self-Norm (CNSN) [19] extends the training distribution using Cross-Norm and narrows the train-test gap with the learnable Self-Norm. Style-Agnostic RL (SAR) [18] generates adversarial examples by adjusting normalization parameters through a generative model. Our method generates diverse domains through Stable Diffusion and adaptively adjusts the extent of domain shift based on changes in episode return while preserving the semantics with saliency maps for more efficient agent learning.

2.2. Diffusion Model for Augmentation

Diffusion models generate images through an iterative denoising process starting from initial Gaussian noise [13]. Recently, several researchers have explored the potential of data augmentation using diffusion models. Trabucco et al. [30] developed a method for creating diverse augmented images in Stable Diffusion through Textual Inversion. Islam et al. [15] applied a Mixup technique that blends half of a diffusion-generated image with its corresponding original. Wang et al. [34] proposed an inter-class augmentation strategy using a diffusion model. Bodur et al. [4] enhanced text-guided localized image editing by introducing prompt augmentation in diffusion models. Aydemir et al. [1] leveraged U-Net features in Stable Diffusion for augmentation to improve saliency prediction. Li et al. [20] developed a method that preserves objects by applying a mask to latent features and diversifying the background. Our method, similar to [20], generates a mask to separate the foreground from the background and provides the agent with varied do-

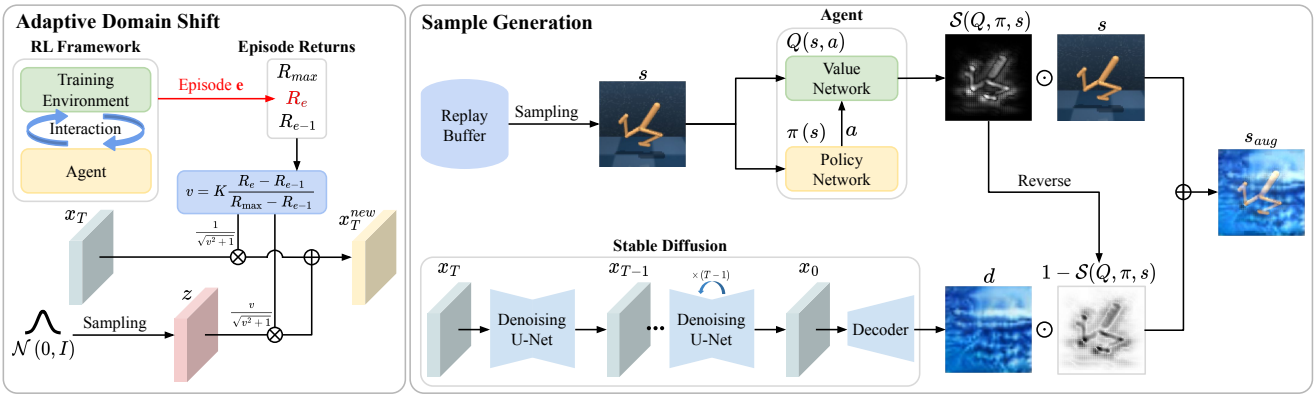


Figure 2. Overview of the DGA2 framework. The Adaptive Domain Shift module dynamically modulates domain variations based on the agent's episode return. The Sample Generation module integrates saliency maps extracted from the agent's value network to preserve task-relevant regions while augmenting background domains with those generated by Stable Diffusion.

mains through diverse backgrounds. Unlike the previous methods, we adjust the domain by modulating the initial noise in the latent space of Stable Diffusion based on the agent's performance feedback.

3. Proposed Method

3.1. Overview

We present DGA2, an augmentation method for VRL that integrates Adaptive Domain Shift using Stable Diffusion and saliency-based augmentation. The overall framework is illustrated in Figure 2. The Adaptive Domain Shift strategy dynamically modulates the extent of domain shifts based on the agent's episode return, ensuring effective adaptation to new domains. Meanwhile, the saliency map, extracted from the agent's value network, is used to retain task-relevant regions while substituting less crucial areas with domain-augmented samples. Algorithm 1 describes the Adaptive Domain Shift mechanism, while Algorithm 2 details the augmented data generation process using the saliency map and Stable Diffusion.

3.2. Adaptive Domain Shift

We compare the current episode return R_e of the agent trained with new domain data against the previous episode return R_{e-1} defined as

$$\Delta R = R_e - R_{e-1}. \tag{1}$$

If $\Delta R > 0$, we adjust the domain based on the improvement rate by modifying the amount of noise added to the current initial noise described as

$$x_T^{\text{new}} = \left(\frac{1}{\sqrt{v^2+1}}\right) x_T + \left(\frac{v}{\sqrt{v^2+1}}\right) z, \tag{2}$$

where v represents the improvement rate expressed as

$$v = K\frac{R_e - R_{e-1}}{R_{\max} - R_{e-1}}, \tag{3}$$

where K represents the sensitivity of domain adjustments to changes in episode return and $R_{\max}$ represents the maximum episode return. $x_T \sim \mathcal{N}(0, I)$ is the initial noise in the latent space of Stable Diffusion defining the domain in which the agent is trained. As v increases, the generated domain becomes distinct from the training domain. Conversely, if $\Delta R \leq 0$, the agent continues its learning process within the same domain to bridge the gap between the current and training domains. Once the current episode return R_e reaches the maximum episode return $R_{\max}$, we consider that the agent has sufficiently learned about the current domain and extract new initial noise for learning a new domain while also updating $R_{\max}$ to the new episode return expressed as

$$x_T^{\text{new}} = \begin{cases} z \sim \mathcal{N}(0, I) & \text{if } R_e \geq R_{\max} \\ x_T^{\text{new}} & \text{otherwise,} \end{cases} \tag{4}$$

where

$$R_{\max} = \max(R_{\max}, R_e). \tag{5}$$

3.3. Extracting Saliency Map

State s is sampled from the replay buffer, and the gradient map $\mathcal{G}(Q, s, a)$ is computed by taking the absolute value of the gradient of the value network with respect to the state as

$$\mathcal{G}(Q, s, a) = \left|\frac{\partial Q(s, a)}{\partial s}\right|, \tag{6}$$

where Q represents the value network and a is an action sampled from the policy network π. Each pixel intensity in the gradient map indicates the degree of change in value,

Algorithm 1 Adaptive Domain Shift of DGA2

Input: current episode return R_e, previous episode return R_{e-1}, max episode return R_{max}
Output: Updated initial Gaussian noise x_T^{new}

if $R_e \geq R_{\text{max}}$ **then**
  $R_{\text{max}} \leftarrow R_e$
  $x_T^{\text{new}} \sim \mathcal{N}(0, I)$
else
  $\Delta R \leftarrow R_e - R_{e-1}$
  if $\Delta R > 0$ **then**
    $v \leftarrow K \frac{R_e - R_{e-1}}{R_{\text{max}} - R_{e-1}}$
    $z \sim \mathcal{N}(0, I)$
    $x_T^{\text{new}} \leftarrow \left(\frac{1}{\sqrt{v^2+1}}\right) x_T + \left(\frac{v}{\sqrt{v^2+1}}\right) z$
  end if
end if
$R_{e-1} \leftarrow R_e$

with higher intensity being recognized as important regions related to the task. After that, we normalize each gradient intensity to create a saliency map $\mathcal{S}(Q, s, a)$ where each pixel is a weight of mask described as

$$\mathcal{S}(Q, s, a) = \frac{\mathcal{G}(Q, s, a) - \min(\mathcal{G}(Q, s, a))}{\max(\mathcal{G}(Q, s, a)) - \min(\mathcal{G}(Q, s, a))}. \quad (7)$$

Additionally, we apply an exponential transformation to the saliency map to control the importance of the task-relevant areas. Finally, the saliency map can be defined as

$$\mathcal{S}(Q, s, a) = \mathcal{S}(Q, s, a)^{\gamma}, \quad (8)$$

where $\gamma \in [0, 1]$ is the exponent, with larger values of γ emphasizing task-relevant regions intensely.

3.4. Generating Augmented Samples

The generation process of the domain begins with refining x_T through a series of denoising steps. This iterative process can be described as

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t z, \quad (9)$$

where x_t represents the noisy image at time step t, α_t and $\bar{\alpha}_t$ are noise schedule parameters, ϵ_θ is the learned denoising U-Net, σ_t is the noise scale at step t, and $z \sim \mathcal{N}(0, I)$ is random noise in the latent space of Stable Diffusion sampled from a standard normal distribution. This process continues until we reach x_0. Because Stable Diffusion removes the noise in latent space, the final domain image d is recovered with the decoder $\mathcal{D}$ expressed as $d = \mathcal{D}(x_0)$. We apply the previously created saliency map $\mathcal{S}(Q, s, a)$ as a mask on d and the original data to produce the augmented sample s_{aug} defined as

$$s_{aug} = \mathcal{S}(Q, s, a) \odot s + (1 - \mathcal{S}(Q, s, a)) \odot d, \quad (10)$$

where $\odot$ is the Hadamard product.

Algorithm 2 Sample Generation of DGA2

Input: state s, initial Gaussian noise x_T
Output: augmented sample s_{aug}

$a \sim \pi(s)$
$\mathcal{G}(Q, s, a) \leftarrow \left| \frac{\partial Q(s,a)}{\partial s} \right|$
$\mathcal{S}(Q, s, a) \leftarrow \frac{\mathcal{G}(Q,s,a) - \min(\mathcal{G}(Q,s,a))}{\max(\mathcal{G}(Q,s,a)) - \min(\mathcal{G}(Q,s,a))}$
$\mathcal{S}(Q, s, a) \leftarrow \mathcal{S}(Q, s, a)^{\gamma}$
for $t = T$ **to** 1 **do**
  $z \sim \mathcal{N}(0, I)$
  $x_{t-1} \leftarrow \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t z$
end for
$d = \mathcal{D}(x_0)$
$s_{\text{aug}} \leftarrow (\mathcal{S}(Q, s, a) \odot s + (1 - \mathcal{S}(Q, s, a)) \odot d)$

4. Experimental Results

4.1. Experimental Setup

Simulation. We evaluated our method's generalization capability on three benchmarks: **DMControl-GB**[12] and **Adroit** [39] for off-policy algorithms and **Procgen**[8] for on-policy algorithms. DMControl-GB and Adroit provided continuous control tasks with visual inputs, while Adroit especially focused on the dexterous manipulation tasks. We evaluated the agent's generalization performance across four different distribution shifts: Color Easy, Color Hard, Video Easy, and Video Hard, where Color Benchmark modified environment colors and Video Benchmark replaced backgrounds with random images. Procgen consisted of procedurally generated discrete game tasks similar to the Arcade Learning Environment (ALE) [2], where each game generated levels from different distributions to test generalization to unseen scenarios.

Implementation Details. For DMControl-GB, we implemented our method based on [14] and evaluated it on Walker walk, Walker stand, Cartpole swingup, Ball in cup catch, and Finger spin. We integrated our approach with Soft Actor-Critic (SAC)[10], Data-regularized Q (DrQ)[36], and Stabilized Q-Value Estimation under Augmentation (SVEA)[12], comparing it against NoAug, Overlay[11], Spectrum Random Masking (SRM)[14], and CrossNorm and SelfNorm (CNSN)[19]. For Adroit, we followed [33], evaluating our method on Door and Hammer tasks against SVEA, Saliency-Guided Q-Networks

Baseline	SAC					DrQ					SVEA				
Augmentation	NoAug	Overlay	SRM	CNSN	DGA2	NoAug	Overlay	SRM	CNSN	DGA2	NoAug	Overlay	SRM	CNSN	DGA2
Color Easy															
Walker walk	489(34)	467(41)	455(40)	377(26)	**694**(59)	719(44)	818(75)	798(16)	**898**(6)	819(89)	626(420)	895(21)	658(389)	873(23)	**922**(32)
Walker stand	775(141)	820(149)	689(54)	779(125)	**932**(18)	941(19)	958(9)	952(16)	962(11)	**975**(2)	402(279)	968(9)	516(316)	**972**(7)	970(6)
Cartpole swingup	786(3)	830(12)	815(7)	830(25)	**855**(12)	774(16)	827(31)	815(15)	827(5)	**863**(9)	836(24)	837(23)	852(16)	814(19)	**854**(9)
Ball in cup catch	767(66)	930(24)	694(99)	922(31)	**933**(17)	550(321)	**970**(7)	659(396)	924(21)	961(9)	970(2)	**973**(3)	971(1)	945(28)	965(11)
Finger spin	766(149)	937(15)	723(217)	601(425)	**965**(16)	812(133)	960(21)	963(17)	948(37)	**972**(8)	964(13)	907(5)	**983**(1)	976(7)	980(3)
Avg ↑	717	797	675	702	**876**	759	907	837	912	**918**	760	916	796	916	**938**
Color Hard															
Walker walk	474(41)	448(34)	410(17)	369(28)	**658**(43)	543(32)	722(85)	605(25)	**814**(34)	754(74)	622(417)	879(26)	659(376)	873(19)	**915**(31)
Walker stand	757(155)	800(142)	665(52)	748(145)	**895**(16)	852(14)	936(23)	864(10)	**963**(3)	915(33)	439(328)	**964**(6)	487(303)	961(18)	963(15)
Cartpole swingup	639(29)	774(4)	701(15)	696(49)	**803**(25)	643(73)	781(31)	679(27)	707(15)	**807**(17)	827(33)	811(9)	824(29)	795(34)	**838**(11)
Ball in cup catch	637(93)	**926**(15)	701(60)	752(107)	922(35)	541(315)	**957**(2)	518(305)	844(86)	934(15)	968(3)	**973**(4)	969(1)	912(67)	965(11)
Finger spin	666(101)	**909**(30)	669(185)	554(395)	891(16)	651(111)	**931**(33)	859(71)	817(120)	920(37)	939(35)	905(7)	**978**(7)	971(13)	967(19)
Avg ↑	635	771	629	624	**834**	646	865	705	829	**866**	759	906	783	902	**930**
Video Easy															
Walker walk	428(28)	450(44)	396(20)	319(37)	**678**(57)	605(41)	787(88)	778(34)	**829**(8)	812(80)	459(300)	869(22)	589(339)	847(32)	**899**(17)
Walker stand	641(78)	796(146)	626(32)	657(84)	**918**(26)	939(18)	969(5)	963(8)	964(3)	**972**(1)	396(246)	964(9)	464(302)	913(49)	**967**(6)
Cartpole swingup	477(32)	685(6)	535(24)	479(51)	**791**(22)	543(117)	737(69)	602(27)	537(40)	**824**(28)	571(69)	545(71)	811(38)	647(96)	**816**(6)
Ball in cup catch	470(21)	**894**(41)	451(38)	496(71)	893(33)	403(219)	943(13)	390(205)	619(123)	**958**(5)	629(48)	937(16)	593(155)	664(50)	**961**(11)
Finger spin	407(95)	757(24)	511(153)	373(263)	**942**(18)	490(63)	829(29)	670(44)	574(27)	**960**(9)	594(162)	886(18)	878(7)	846(99)	**970**(3)
Avg ↑	485	716	504	465	**844**	596	853	681	705	**905**	530	840	667	783	**923**
Video Hard															
Walker walk	132(25)	292(49)	138(20)	153(11)	**484**(71)	81(22)	394(77)	194(23)	218(45)	**562**(110)	97(41)	440(68)	192(89)	340(14)	**567**(82)
Walker stand	237(41)	610(110)	321(30)	282(41)	**718**(40)	325(9)	**832**(29)	488(72)	424(43)	790(58)	268(118)	764(30)	277(149)	481(59)	**831**(55)
Cartpole swingup	156(8)	295(36)	145(19)	177(11)	**533**(34)	145(22)	439(51)	156(3)	197(23)	**553**(29)	137(23)	203(9)	295(65)	326(92)	**474**(131)
Ball in cup catch	136(27)	345(203)	107(16)	95(47)	**613**(68)	94(30)	459(70)	77(31)	173(34)	**791**(56)	139(68)	481(77)	164(60)	146(92)	**780**(72)
Finger spin	17(8)	212(24)	72(27)	41(32)	**732**(6)	32(22)	353(37)	131(9)	70(18)	**702**(60)	142(18)	660(54)	307(3)	452(84)	**872**(56)
Avg ↑	136	351	157	150	**616**	135	495	209	216	**680**	157	510	247	349	**705**

Table 1. Comparison with other methods in DMControl-GB after training on 500K environment steps. We provide the mean and standard deviation of episode return repeated three times with different random seeds. (·) represents the standard deviation.

(SGQN)[3], and Pre-trained Image Encoder for Generalizable visual reinforcement learning (PIE-G)[38], integrating our method with PIE-G. The success rate was used as the metric [39]. For Procgen, we followed [18], selecting eight environments and implementing our method accordingly. In DMControl-GB and Adroit, each method was trained for 500K steps, while Procgen used 25M steps [26]. All experiments were repeated three times with different random seeds. Agents were trained on 200 initial levels, with generalization assessed in the *easy* mode using average rank [18]. Complete return and rank results for DMControl-GB and Procgen are reported in Appendix. Our method was combined with PPO [29] and DrAC [26], and compared against PPO, Reinforcement Learning with Augmented Data (RAD)[16], MixStyle[41], Style-based Active Representation (SAR)[18], Robust ADversarIAl Loss (RADIAL)[25], and DrAC. We set γ to 0.3 and K to 12 for all algorithms, except for Data-regularized Actor-Critic (DrAC), where we set K to 10. To ensure stability in saliency map generation, we employed a warm-up phase before applying our method, at 100K steps for DMControl-GB and Adroit, and 1M steps for Procgen. During training, we incorporated both the augmented and original states within the same batches. For the Stable Diffusion setting, we employed the distilled Stable Diffusion v1.4 and TinyAutoEncoder models. We set the denoising steps to 3.

4.2. Evaluation on DMControl-GB

As shown in Table 1, DGA2 consistently achieved robust performance across all benchmarks, demonstrating adaptability and effectiveness across varied environments. In the Color Benchmark, our approach reliably enhanced performance across tasks featuring color variations. This advantage was further enhanced in the Video Benchmark. Specifically, on the Video Hard, our method improved baseline performance by 353% with SAC. For DrQ, we observed a remarkable 404% improvement, and with SVEA, our method led to a 349% increase. Applying our method to SAC without any additional augmentations, we observed superior performance compared to all baselines. Moreover, when comparing all baselines combined with other augmentation methods, SAC integrated with our approach achieved the highest performance on Video Hard. These results emphasized that our approach, even when applied independently, significantly improved the agent's generalization capabilities. While SRM and CNSN contributed to performance improvements in DrQ and SVEA, their effectiveness was limited in the Video Benchmark and sometimes even reduced SAC's performance due to inherent constraints. SRM's reliance on masking the spectral distribution of training data restricted its capacity to generate novel domain variations. Similarly, CNSN's cross-norm operations within the same domain limited its ability to create

	PPO	RAD	MixStyle	SAR	RADIAL	DrAC	PPO+DGA2	DrAC+DGA2
Bigfish	3.00 (1.71)	5.17 (4.06)	5.47 (2.41)	1.37 (0.45)	1.22 (0.56)	5.60 (3.83)	8.17 (3.31)	**12.80** (1.56)
Starpilot	27.53 (1.02)	30.03 (1.47)	28.07 (2.49)	31.70 (11.39)	12.43 (0.69)	32.07 (4.98)	26.10 (5.68)	**37.93** (4.08)
Ninja	5.33 (1.25)	4.00 (0.82)	5.67 (0.94)	5.00 (0.00)	3.67 (0.53)	4.67 (1.25)	**6.67** (0.94)	6.00 (1.63)
CoinRun	9.00 (0.00)	6.67 (1.70)	8.33 (0.47)	8.33 (1.25)	7.43 (0.26)	8.67 (0.47)	8.33 (0.47)	**9.33** (0.47)
Jumper	5.67 (1.25)	**7.00** (0.00)	5.67 (0.47)	4.00 (1.63)	5.60 (0.65)	5.00 (1.63)	5.67 (1.70)	5.67 (1.70)
Climber	7.00 (1.61)	3.60 (0.91)	6.03 (2.03)	4.00 (0.36)	2.81 (0.07)	6.33 (1.31)	7.53 (0.48)	**8.10** (1.06)
Dodgeball	1.20 (0.59)	3.20 (0.82)	1.73 (0.98)	1.27 (0.34)	1.43 (0.12)	**5.87** (2.56)	1.80 (0.59)	5.73 (2.50)
Maze	4.67 (1.25)	4.33 (0.47)	**7.67** (0.47)	5.67 (2.05)	5.23 (0.26)	5.67 (1.25)	5.00 (0.82)	6.00 (1.41)
Avg. Rank ↓	4.75	5.38	3.63	5.38	7.00	3.63	3.50	**1.50**

Table 2. Comparison with other methods in Procgen after training on 25M environment steps. We provide the mean and standard deviation of episode return trained with three different random seeds. (·) represents the standard deviation. Avg. Rank is the average of the rankings assigned for each task.

Algorithm	SVEA	SGQN	PIE-G	DGA2
	Color Easy			
Door	49.0(5.7)	34.3(24.7)	85.0(0.8)	**88.0**(7.6)
Hammer	36.0(13.7)	**64.7**(4.5)	56.0(8.6)	62.3(11.4)
	Color Hard			
Door	14.7(3.3)	14.3(9.8)	56.3(7.4)	**61.0**(2.8)
Hammer	16.7(4.0)	24.7(9.0)	**49.3**(9.8)	36.0(15.5)
	Video Easy			
Door	51.3(5.7)	33.0(22.6)	**85.0**(5.4)	82.0(4.2)
Hammer	11.3(6.9)	54.3(6.9)	57.0(8.0)	**64.0**(10.7)
	Video Hard			
Door	9.7(3.3)	7.3(6.1)	41.3(6.8)	**53.0**(4.5)
Hammer	6.3(3.9)	18.3(5.8)	**45.7**(10.6)	22.0(8.8)

Table 3. Comparison with other methods in Adroit after training on 500K environment steps. We provide the mean and standard deviation of success rate trained with three different random seeds. (·) represents the standard deviation.

diverse scenarios. By contrast, our method leveraged the rich knowledge within Stable Diffusion to generate diverse domain images, enhancing the agent's robustness in unseen environments. Overlay performed well overall, indicating that introducing external datasets could effectively improve the agent's robustness. However, a significant performance gap emerged in Video Hard, where all regions except task-relevant ones were replaced with unseen backgrounds. Overlay altered the domain independently of the agent's progress, resulting in suboptimal performance. Additionally, Overlay, which merely mixed original images with external datasets, struggled to preserve these task-relevant regions. In contrast, our approach adapted the domain based on the agent's feedback, enabling more effective learning outcomes while maintaining essential task-related information through saliency.

4.3. Evaluation on Procgen

As shown in Table 2, algorithms combined with our method achieved consistently better performance than other approaches. By incorporating our method, both PPO and DrAC demonstrated significant improvements over their baselines. In particular, PPO with our method achieved superior performance over the baseline in 5 of the 8 games, and DrAC with the proposed method outperformed the baseline algorithm in 7 of the 8 games. Furthermore, our method outperformed MixStyle and SAR, which used normalization to preserve semantic information during augmentation. SAR presented relatively lower performance than MixStyle and PPO due to SAR's reliance on training a generative model, which required additional training steps to produce effective augmentation samples. In contrast, our method adaptively augmented samples based on agent feedback, eliminating the need for extra training. RAD and RADIAL also demonstrated lower performance than the baseline PPO. For RAD, this performance drop could be attributed to semantic distortion caused by pixel transformations. While RADIAL aimed to train agents robust to adversarial noise, it lacked the capability to expose agents to diverse domains. Consequently, our method integrated effectively with on-policy algorithms and proved robust and effective across different tasks.

Method	Color Easy	Color Hard	Video Easy	Video Hard	Avg ↑
NoAug	717	635	485	136	493
Flip	675	612	472	159	480
Rotation	533	431	295	97	339
Cutout	597	530	462	143	433
Random Conv	863	**852**	652	238	651
DGA2	**876**	834	**844**	**616**	**793**

Table 4. Comparison with various augmentation methods.

4.4. Evaluation on Adroit

Table 3 presented the quantitative evaluation of DGA2 on the Adroit benchmark. Overall, DGA2 exhibited strong generalization performance compared to SVEA, SGQN, and PIE-G across various domain shift scenarios. In the Door task, it achieved the highest success rates in all settings, demonstrating robust adaptability to both color and video perturbations. In the Color Hard and Video Hard settings, DGA2 attained success rates of 61.0% and 53.0%, respectively, outperforming PIE-G. In the Hammer task,

Method	Color Easy	Color Hard	Video Easy	Video Hard	Avg ↑
ID	520	481	519	429	487
DS(S)	480	458	472	403	453
DS(E)	573	546	552	459	533
CD	507	493	508	447	489
ADS	**605**	**595**	**582**	**487**	**567**

Table 5. Experimental results on different domain shift strategies.

while DGA2 performed well in the Color Easy and Video Easy settings, its performance in the Color Hard and Video Hard settings was lower than that of PIE-G.

4.5. Comparison with Various Augmentations

We conducted a comparison between our method and various augmentation methods proposed in [16], specifically Flip, Rotation, Cutout, and Random Conv. SAC was used as the baseline algorithm. As shown in Table 4, our method demonstrated overall better performance. In contrast, Flip, Rotation, and Cutout resulted in performance degradation compared to the baseline, likely due to their modifications that disrupted task-relevant information. Random Conv achieved the second-best performance by introducing diverse domains, though its transformations remained limited to local feature perturbations without adaptive control. Unlike these static transformations, our method dynamically adjusts domain shifts with the agent's learning progress. This adaptive mechanism leads to improved generalization across diverse domains.

4.6. Comparison with Domain Shift Strategies

We evaluated the effectiveness of the Adaptive Domain Shift (ADS) strategy by comparing four methods: training within an identical domain (ID), domain switching at fixed steps (DS(S)), domain switching upon episode termination (DS(E)), and progressively changing domains at the end of each episode (CD). We trained SAC up to 300K steps in Walker walk. As shown in Table 5, DS(S) resulted in lower performance compared to ID. Although DS(S) introduced greater domain diversity than the other approaches, it restricted learning efficiency. When comparing our approach to DS(E) and CD, the proposed method produced superior results. This suggested that adaptively adjusting the domain based on episode return was more effective than merely exposing the agent to a variety of domains. We also applied a mask to Overlay and compared its performance with our approach. This enabled us to assess the effectiveness of the domains generated by our method in comparison to those from a predefined manifold while maintaining semantic consistency. In Sec. 3 of Appendix, we provide more details and additional experimental results.

4.7. Analysis on Adaptive Domain Shift

We analyzed how the learning domain gradually shifted according to the Adaptive Domain Shift (ADS) strategy as the episode return evolved. Specifically, we examined whether the domain transition occurred progressively in response to changes in the episode return. To achieve this, we generated new domains based on the variation in v, which quantified the rate of change in the episode return. We then measured the Fréchet Inception Distance (FID) [37] between each generated domain and the initial domain to quantify the extent of the domain shift. Table 6 presented the FID values for different values of v, revealing a consistent trend where the generated domains progressively deviated from the initial domain as v increased. This trend confirms that ADS successfully adjusts the learning domain based on the agent's performance, promoting the progressive domain shift.

v	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0
FID	181	230	262	253	275	264	290	281	286	295

Table 6. Domain distance as the change of improvement rate v.

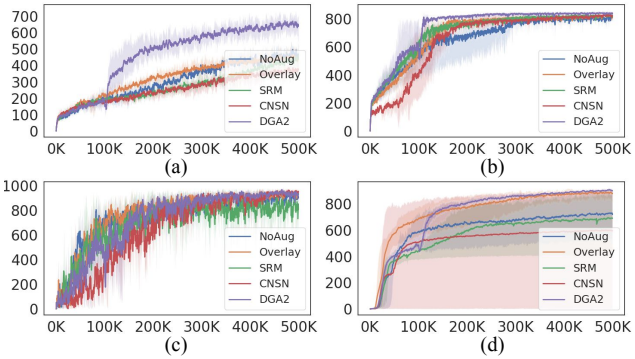


Figure 3. Training performance of each augmentation method. The x-axis represents the number of environment steps and the y-axis represents the episode return. (a) Walker walk, (b) Cartpole swingup, (c) Ball in cup catch, (d) Finger spin.

4.8. Analysis on Training Performance

We assessed the training performance of augmentation methods in SAC across four tasks: Walker walk, Cartpole swingup, Ball in cup catch, and Finger spin. As shown in Figure 3, early training showed no significant differences among methods. However, from 100K steps onward, our method led to a notable increase in episode rewards. The most significant gain was in Walker walk, where our method created a clear gap. In Cartpole swingup, the agent rapidly converged after applying our method. In Finger spin, our method initially lagged behind but later showed sharp improvement, ultimately surpassing others. In Ball in cup catch, training performance remained similar across methods, but during testing, particularly in the Video Hard environment, our method outperformed others.

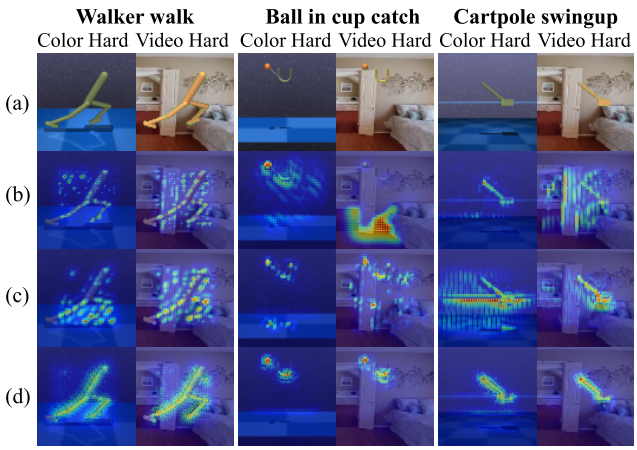
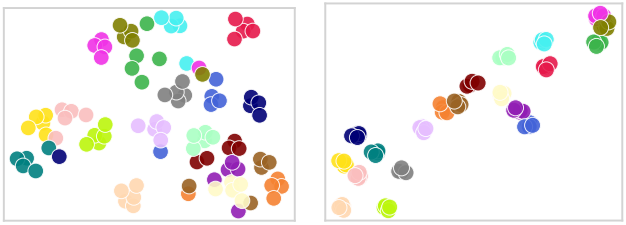


Figure 4. Visualization of saliency map of Walker walk and Ball in cup catch in Color Hard and Video Hard. (a) Original Image, Saliency Map of (b) NoAug, (c) Overlay, and (d) DGA2.



(a) NoAug (b) DGA2

Figure 5. Visualization of latent vectors. Each point represents a domain, and points of the same color share semantic information.

4.9. Qualitative Evaluation through Saliency Map

We qualitatively assessed generalization using saliency maps in unseen domains. Experiments on Walker walk, Ball in cup catch, and Cartpole swingup in Color Hard and Video Hard compared our method to NoAug and Overlay. As shown in Figure 4, NoAug often attended to irrelevant backgrounds, failing to identify task-relevant areas in Ball in cup catch (Video Hard). Overlay improved attention but exhibited similar tendencies with NoAug, especially in Walker walk. In contrast, our method consistently focused on task-relevant areas across both Color Hard and Video Hard.

4.10. Visualization of Latent Vector

We visualized the latent vectors from the encoder through t-SNE [31] to evaluate the agent's ability to capture task-relevant information across the domains. We compared our method with the NoAug. A set of images from Walker walk was collected with every benchmark applied. Each image across domains conveyed distinct semantic information. If the agent demonstrates good generalization, ten clusters are formed, each consisting of five elements within the feature space. As depicted in Figure 5, our method successfully grouped latent vectors that shared the same semantic information across different domains. In contrast, the baseline model, while able to group similar semantic information, did not achieve the same level of clarity and separation as our approach. These results implied that the agent trained with our method could consistently extract task-relevant information from unseen domains.

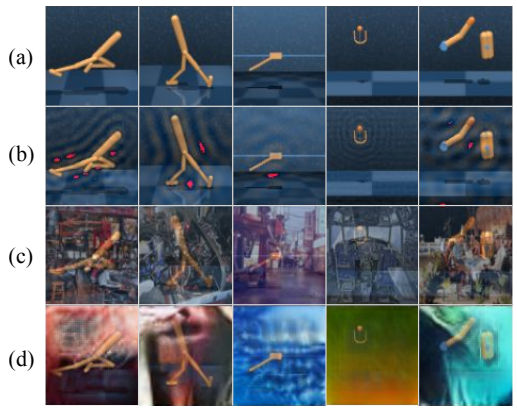


Figure 6. Visualization of augmented images. (a) Original images, and augmented images from (b) SRM, (c) Overlay, and (d) DGA2.

4.11. Visualization of Augmented Samples

We visualized the augmented images and selected SRM and Overlay as comparisons to observe pixel-level changes alongside our method. As depicted in Figure 6, augmented images produced by SRM consistently exhibited a recurring pattern because SRM masked specific regions of the frequency component. However, SRM limited its ability to generate entirely distinct domain manifolds. In contrast, Overlay demonstrated a degree of variability by blending external datasets. Nevertheless, since Overlay lacked selectivity in emphasizing task-relevant regions, those features were diluted or overshadowed. Our method, similar to Overlay, was capable of generating a diverse of domains. Additionally, by integrating saliency, our approach highlighted and preserved task-relevant regions.

5. Conclusion

In this work, we propose Diffusion Guided Adaptive Augmentation (DGA2), a novel augmentation method for improving generalization in Visual Reinforcement Learning (VRL). DGA2 leverages Stable Diffusion to enhance domain diversity while introducing an Adaptive Domain Shift strategy that dynamically adjusts domain variations based on the agent's learning progress. This approach ensures a progressive adaptation to new domains, improving training stability and generalization. Experimental results demonstrate that DGA2 significantly enhances robustness to unseen domains.

6. Acknowledgements

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) under Grant No. RS-2025-00564137, and in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) grant funded by the Korean government (MSIT) under Grant RS-2022-00155911, Artificial Intelligence Convergence Innovation Human Resources Development (Kyung Hee University), and in part by Convergence security core talent training business support program under Grant IITP-2023-RS-2023-00266615).

References

[1] Bahar Aydemir, Deblina Bhattacharjee, Tong Zhang, Mathieu Salzmann, and Sabine Süsstrunk. Data augmentation via latent diffusion for saliency prediction. In *European Conference on Computer Vision*, pages 360–377. Springer, 2025. 2

[2] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013. 4

[3] David Bertoin, Adil Zouitine, Mehdi Zouitine, and Emmanuel Rachelson. Look where you look! saliency-guided q-networks for generalization in visual reinforcement learning. *Advances in Neural Information Processing Systems*, 35:30693–30706, 2022. 2, 5

[4] Rumeysa Bodur, Binod Bhattarai, and Tae-Kyun Kim. Prompt augmentation for self-supervised text-guided image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8829–8838, 2024. 2

[5] Wei Chen and Nicolas Rojas. Trakdis: A transformer-based knowledge distillation approach for visual reinforcement learning with application to cloth manipulation. *IEEE Robotics and Automation Letters*, 9(3):2455–2462, 2024. 1

[6] Hyesong Choi, Hunsang Lee, Seongwon Jeong, and Dongbo Min. Environment agnostic representation for visual reinforcement learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 263–273, 2023. 1

[7] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International conference on machine learning*, pages 1282–1289. PMLR, 2019. 1

[8] Karl Cobbe, Chris Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pages 2048–2056. PMLR, 2020. 2, 4

[9] Linxi Fan, Guanzhi Wang, De-An Huang, Zhiding Yu, Li Fei-Fei, Yuke Zhu, and Animashree Anandkumar. Secant: Self-expert cloning for zero-shot generalization of visual policies. In *Proceedings of the 38th International Conference on Machine Learning*, pages 3088–3099. PMLR, 2021. 1

[10] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018. 4

[11] Nicklas Hansen and Xiaolong Wang. Generalization in reinforcement learning by soft data augmentation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13611–13617. IEEE, 2021. 1, 4

[12] Nicklas Hansen, Hao Su, and Xiaolong Wang. Stabilizing deep q-learning with convnets and vision transformers under data augmentation. *Advances in neural information processing systems*, 34:3680–3693, 2021. 2, 4

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2

[14] Yangru Huang, Peixi Peng, Yifan Zhao, Guangyao Chen, and Yonghong Tian. Spectrum random masking for generalization in image-based reinforcement learning. *Advances in Neural Information Processing Systems*, 35:20393–20406, 2022. 1, 2, 4

[15] Khawar Islam, Muhammad Zaigham Zaheer, Arif Mahmood, and Karthik Nandakumar. Diffusemix: Label-preserving data augmentation with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27621–27630, 2024. 2

[16] Misha Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. *Advances in neural information processing systems*, 33:19884–19895, 2020. 1, 2, 5, 7

[17] Ngan Le, Vidhiwar Singh Rathour, Kashu Yamazaki, Khoa Luu, and Marios Savvides. Deep reinforcement learning in computer vision: a comprehensive survey. *Artificial Intelligence Review*, pages 1–87, 2022. 1

[18] Juyong Lee, Seokjun Ahn, and Jaesik Park. Style-agnostic reinforcement learning. In *European Conference on Computer Vision*, pages 604–620. Springer, 2022. 1, 2, 5

[19] Lu Li, Jiafei Lyu, Guozheng Ma, Zilin Wang, Zhenjie Yang, Xiu Li, and Zhiheng Li. Normalization enhances generalization in visual reinforcement learning. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, pages 1137–1146, 2024. 1, 2, 4

[20] Yuhang Li, Xin Dong, Chen Chen, Weiming Zhuang, and Lingjuan Lyu. A simple background augmentation method for object detection with diffusion model. In *European Conference on Computer Vision*, pages 462–479. Springer, 2024. 2

[21] Zhixuan Liang, Yao Mu, Hengbo Ma, Masayoshi Tomizuka, Mingyu Ding, and Ping Luo. Skilldiffuser: Interpretable hierarchical planning via skill abstractions in diffusion-based task execution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16467–16476, 2024. 1

[22] Guozheng Ma, Zhen Wang, Zhecheng Yuan, Xueqian Wang, Bo Yuan, and Dacheng Tao. A comprehensive survey of data augmentation in visual reinforcement learning. *arXiv preprint arXiv:2210.04561*, 2022. 1

[23] Guozheng Ma, Linrui Zhang, Haoyu Wang, Lu Li, Zilin Wang, Zhen Wang, Li Shen, Xueqian Wang, and Dacheng Tao. Learning better with less: Effective augmentation for sample-efficient visual reinforcement learning. *Advances in Neural Information Processing Systems*, 36:59832–59859, 2023. 2

[24] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *Advances in neural information processing systems*, 2013. 1

[25] Tuomas Oikarinen, Wang Zhang, Alexandre Megretski, Luca Daniel, and Tsui-Wei Weng. Robust deep reinforcement learning through adversarial loss. *Advances in Neural Information Processing Systems*, 34:26156–26167, 2021. 5

[26] Roberta Raileanu, Maxwell Goldstein, Denis Yarats, Ilya Kostrikov, and Rob Fergus. Automatic data augmentation for generalization in reinforcement learning. *Advances in Neural Information Processing Systems*, 34:5402–5415, 2021. 5

[27] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 2

[28] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021. 2

[29] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 5

[30] Brandon Trabucco, Kyle Doherty, Max A Gurinas, and Ruslan Salakhutdinov. Effective data augmentation with diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024. 2

[31] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9 (11), 2008. 8

[32] Kaixin Wang, Bingyi Kang, Jie Shao, and Jiashi Feng. Improving generalization in reinforcement learning with mixture regularization. *Advances in Neural Information Processing Systems*, 33:7968–7978, 2020. 1

[33] Ziyu Wang, Yanjie Ze, Yifei Sun, Zhecheng Yuan, and Huazhe Xu. Generalizable visual reinforcement learning with segment anything model. *arXiv preprint arXiv:2312.17116*, 2023. 4

[34] Zhicai Wang, Longhui Wei, Tan Wang, Heyu Chen, Yanbin Hao, Xiang Wang, Xiangnan He, and Qi Tian. Enhance image classification via inter-class image mixup with diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17223–17233, 2024. 2

[35] Rui Yang, Jie Wang, Qijie Peng, Ruibo Guo, Guoping Wu, and Bin Li. Learning robust representations with long-term information for generalization in visual reinforcement learning. In *The Thirteenth International Conference on Learning Representations*. 1

[36] Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International conference on learning representations*, 2021. 4

[37] Yu Yu, Weibin Zhang, and Yun Deng. Frechet inception distance (fid) for evaluating gans. *China University of Mining Technology Beijing Graduate School*, 3(11), 2021. 7

[38] Zhecheng Yuan, Zhengrong Xue, Bo Yuan, Xueqian Wang, Yi Wu, Yang Gao, and Huazhe Xu. Pre-trained image encoder for generalizable visual reinforcement learning. *Advances in Neural Information Processing Systems*, 35:13022–13037, 2022. 5

[39] Zhecheng Yuan, Sizhe Yang, Pu Hua, Can Chang, Kaizhe Hu, and Huazhe Xu. Rl-vigen: A reinforcement learning benchmark for visual generalization. *Advances in Neural Information Processing Systems*, 36:6720–6747, 2023. 2, 4, 5

[40] Di Zhang, Bowen Lv, Hai Zhang, Feifan Yang, Junqiao Zhao, Hang Yu, Chang Huang, Hongtu Zhou, Chen Ye, et al. Focus on what matters: Separated models for visual-based rl generalization. *Advances in Neural Information Processing Systems*, 37:116960–116986, 2024. 1

[41] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. In *International Conference on Learning Representations*, 2021. 1, 2, 5