Supplementary Material for "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

1. Effectiveness on Domain Shift Parameter

We conducted an experiment on the Walker walk for up to 300K steps to evaluate the generalization efficiency with respect to changes K. As illustrated in Table 1, the generalization performance was highest when $K = 12$, indicating that this configuration provided the best balance regarding domain diversity and task relevance. As K increased or decreased significantly from this value, performance tended to decrease, suggesting that too high or too low a value of K impaired the agent's ability to generalize effectively. The overall trend demonstrated that moderate values of K led to robust performance across diverse environments.

	Color Easy	Color Hard	Video Easy	Video Hard	Avg ↑
$K = 1$	586(89)	561(84)	551(91)	472(87)	543
$K = 4$	521(49)	522(43)	516(43)	446(89)	501
$K = 8$	557(59)	538(42)	541(48)	454(55)	523
$K = 10$	527(45)	525(45)	522(45)	455(29)	507
$K = 12$	**622**(52)	**612**(45)	**614**(69)	**507**(96)	**589**
$K = 14$	520(91)	496(82)	496(80)	441(71)	488
$K = 16$	539(30)	525(23)	517(45)	437(59)	505
$K = 18$	608(85)	596(88)	593(78)	506(51)	576
$K = 20$	528(69)	516(66)	509(77)	418(76)	493
$K = 30$	509(43)	497(46)	484(43)	404(73)	474
$K = 50$	547(35)	533(39)	539(41)	472(38)	523

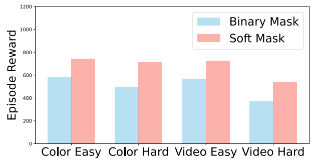
Table 1. Experimental results on domain shift ratio. We provide mean and standard deviation. $(\cdot)$ represents the standard deviation.

2. Analysis on Saliency Mask

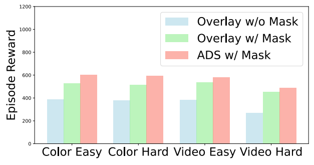
In this experiment, we evaluated the effectiveness of our mask strategy compared with the binary mask proposed in [1]. We trained SAC up to 300K steps in Walker walk and Ball in cup catch. As depicted in Figure 1a, our soft mask approach demonstrated superior performance across all benchmarks compared to the binary mask.

3. Contribution of Saliency Mask

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. As depicted in Figure 1b, even with the mask, Overlay underperformed relative to our approach. These findings indicated that adaptively generating domains using Stable Diffusion enhanced generalization more effectively than relying on a predefined domain manifold.



(a)



(b)

Figure 1. Extensive comparison. (a) Binary mask and soft mask; (b) Overlay using saliency map and our method.

4. Effectiveness on Mask Ratio

We conducted training in the Walker walk for up to 300K steps and then assessed generalization performance with respect to γ. As shown in Table 2, the best performance was achieved at $\gamma = 0.3$. As depicted in Figure 2, where at $\gamma = 0.1$, non-task-relevant background information remained, which limited the agent's exposure to completely new domains. In contrast, starting from $\gamma = 0.3$, we observed that the background in the training environment was almost replaced by domain images generated by our method. However, as γ increased further, only the most critical task-relevant features were emphasized, while the remaining areas were replaced by domain images, leading to a decline in performance. These results implied that identifying an appropriate value of γ was crucial for properly preserving semantic features.

5. Effectiveness on Warm-up stage

We conducted experiments in order to verify the effect of the warm-up stage. Specifically, we categorized the stages

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

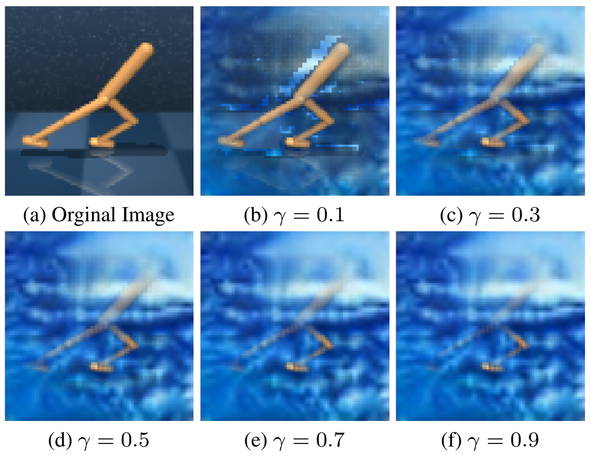


Figure 2. Visualization as the changes of mask ratio γ.

	Color Easy	Color Hard	Video Easy	Video Hard	Avg ↑
$\gamma = 0.1$	448(44)	415(45)	425(40)	269(35)	389
$\gamma = 0.3$	**605**(73)	**595**(73)	**582**(48)	**487**(44)	**567**
$\gamma = 0.5$	512(63)	501(78)	496(64)	411(89)	480
$\gamma = 0.7$	583(24)	557(32)	548(30)	412(35)	525
$\gamma = 0.9$	507(50)	491(42)	482(41)	339(65)	455

Table 2. Experimental results on mask ratio. We provide mean and standard deviation. $(\cdot)$ represents the standard deviation.

Warm-up Steps	Color Easy	Color Hard	Video Easy	Video Hard	Avg ↑
0	537	499	514	441	498
50K	601	549	567	469	547
100K	**732**	**688**	**712**	510	**661**
150K	637	610	577	321	536
200K	703	660	693	581	**659**

Table 3. Experimental results on Warm-up stage.

of applying our method at 0K, 50K, 100K, 150K, and 200K and trained SAC up to 500K in Walker walk. As shown in Table 3, the results indicated that applying our method in the early stages of training led to decreased learning efficiency. Applying the method during the later stages of training, particularly after 100K steps, yielded better overall performance.

6. Training Performance in Procgen

We evaluated the algorithms by comparing their training performances in Procgen. As illustrated in Table 4, PPO performed worse than MixStyle, DrAC, and PPO combined with our method in the test environment, but showed higher performance in the training environment. This indicated that when domain diversity was not ensured, the agent overfitted to the training environment. In contrast, our method integrated with DrAC demonstrated strong performance in both the test and training environments.

7. Quantitative Evaluation of Semantic

We computed the Task Identification (TID) score proposed in [12], which quantifies how well latent variables captured semantic features. The results in Figure 3 demonstrated that DGA2 achieved superior TID scores than the NoAug in both training and test environments in most cases.

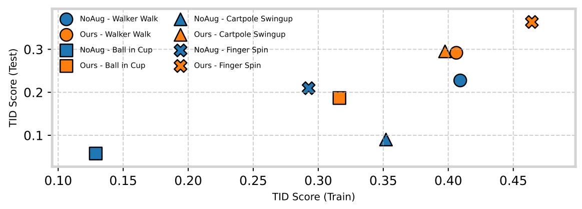


Figure 3. Comparison of TID Scores.

8. Detailed Experimental Results on Comparison with Various Augmentation Methods

We provide detailed comparative results with various augmentation methods in Table 5. We applied Flip, Rotation, Cutout, and Random Conv only to the current state and combined the original samples with the augmented samples to form the training batch. As shown in Table 5, our method generally demonstrated superior performance except for Color Hard. Random Convolution outperformed our method because it introduced color variations similar to those in Color Hard, as illustrated in Figure 5. In contrast, Flip, Rotation, and Cutout disrupted the semantic information of the original image, which prevented the agent from learning effectively.

9. Analysis on Time Complexity

We measured the average training time per episode over 10 episodes for various augmentation methods on DMControl-GB, as shown in Table 6. While DGA2 achieved the highest performance, it also required the most time. This was primarily due to the diffusion-based image augmentation, which added significant computational overhead compared to other methods. In contrast, Overlay, which delivered the second-best performance, exhibited almost the same training time as NoAug, since it only involved mixup using external data. This is a key limitation of our approach, where the time complexity increases due to the more advanced augmentation techniques.

10. Analysis on Sample Efficiency

To evaluate the learning efficiency of each augmentation method, we compareed test performance across environment steps, as shown in Figure 4. Although DGA2 is activated only after 100K steps, it rapidly outperforms all

	PPO	RAD	MixStyle	SAR	RADIAL	DrAC	PPO+DGA2	DrAC+DGA2
Bigfish	6.80(2.18)	8.30(3.05)	9.60(2.34)	4.73(2.97)	2.22(0.16)	**13.23**(2.25)	10.97(4.89)	10.83(4.09)
Starpilot	31.27(7.47)	24.63(6.64)	24.30(4.46)	31.63(5.35)	13.49(1.23)	**35.60**(3.20)	31.07(4.87)	35.10(6.73)
Ninja	**8.67**(0.47)	4.33(1.25)	7.00(1.63)	7.67(1.89)	5.27(0.65)	6.33(0.47)	7.33(1.70)	6.67(1.25)
CoinRun	9.67(0.47)	8.33(1.25)	**10.00**(0.00)	8.67(0.47)	7.73(0.94)	8.67(0.94)	8.33(1.25)	9.33(0.47)
Jumper	**9.00**(0.82)	8.00(0.82)	**9.00**(0.82)	7.67(1.70)	7.77(0.24)	7.67(0.47)	8.33(1.25)	**9.00**(0.00)
Climber	9.43(1.14)	3.93(2.29)	8.20(1.84)	7.07(1.21)	4.10(0.41)	9.50(0.73)	9.93(0.26)	**10.00**(1.31)
Dodgeball	3.40(1.56)	4.07(0.77)	4.20(1.30)	2.27(0.94)	2.47(0.43)	6.60(2.04)	5.20(1.02)	**8.20**(3.77)
Maze	**9.67**(0.47)	7.00(0.00)	9.00(1.41)	5.00(2.45)	8.97(0.09)	8.33(1.25)	**9.67**(0.47)	8.67(0.47)
Avg. Rank ↓	3.13	6.25	3.63	5.63	6.88	3.75	3.25	**2.63**

Table 4. Training performance 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.

	NoAug	Flip	Rotation	Cutout	Random Conv	DGA2
Color Easy						
Walker walk	489(34)	533(13)	492(99)	363(113)	635(54)	**694**(59)
Walker stand	775(141)	740(137)	718(128)	421(363)	897(79)	**932**(18)
Cartpole swingup	786(3)	630(46)	384(14)	819(21)	839(14)	**855**(12)
Ball in cup catch	767(66)	567(34)	466(236)	472(292)	**965**(2)	933(17)
Finger spin	766(149)	906(56)	604(430)	910(44)	**979**(8)	965(16)
Avg ↑	717	675	533	597	863	**876**
Color Hard						
Walker walk	474(41)	500(15)	403(107)	316(117)	624(50)	**658**(43)
Walker stand	757(155)	741(135)	605(118)	409(335)	886(87)	**895**(16)
Cartpole swingup	639(29)	464(31)	320(83)	717(18)	**812**(36)	803(25)
Ball in cup catch	637(93)	514(127)	359(177)	388(248)	**964**(2)	922(35)
Finger spin	666(101)	840(104)	466(334)	819(58)	**976**(8)	891(16)
Avg ↑	635	612	431	530	**852**	834
Video Easy						
Walker walk	428(28)	407(41)	253(67)	272(121)	572(53)	**678**(57)
Walker stand	641(78)	668(99)	383(80)	389(334)	771(179)	**918**(26)
Cartpole swingup	477(32)	324(95)	126(13)	669(62)	477(48)	**791**(22)
Ball in cup catch	470(21)	376(96)	334(147)	331(192)	727(158)	**893**(33)
Finger spin	407(95)	583(41)	379(268)	648(73)	715(114)	**942**(18)
Avg ↑	485	472	295	462	652	**844**
Video Hard						
Walker walk	132(25)	100(4)	61(2)	95(3)	225(16)	**484**(71)
Walker stand	237(41)	354(82)	171(19)	234(121)	430(65)	**718**(40)
Cartpole swingup	156(8)	146(23)	92(18)	177(25)	145(17)	**533**(34)
Ball in cup catch	136(27)	117(57)	107(18)	98(27)	243(218)	**613**(68)
Finger spin	17(8)	76(22)	55(72)	112(73)	145(70)	**732**(6)
Avg ↑	136	159	97	143	238	**616**

Table 5. The experimental results on various augmentation methods in DMControl-GB. We provide the mean and standard deviation repeated three times with different random seeds. (·) represents the standard deviation.

Method	SAC	DrQ	SVEA
NoAug	20.2(1.78)	21.82(1.8)	28.06(1.58)
Overlay	19.95(3.46)	19.8(2.66)	27.67(2.91)
SRM	21.82(1.35)	21.43(1.42)	28.85(1.28)
CNSN	32.19(2.12)	36.53(1.62)	46.65(1.98)
DGA2	40.3(2.33)	55.24(1.64)	59.85(2.32)

Table 6. Average runtime (s) for different augmentation methods. We provide mean and standard deviation. (·) represents the standard deviation.

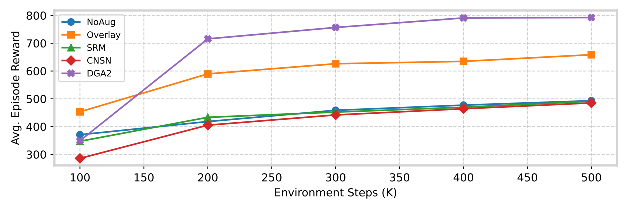


Figure 4. Test performance over environment steps.

baseline methods and achieves the highest performance by 200K steps. This indicates that our method requires significantly fewer environment interactions to reach a given performance level. The result demonstrates that, despite the additional computational cost, DGA2 offers superior sample efficiency.

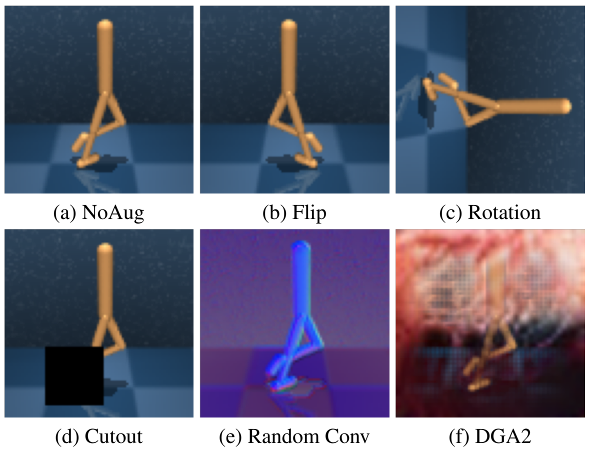


Figure 5. Visualization of various augmentations.

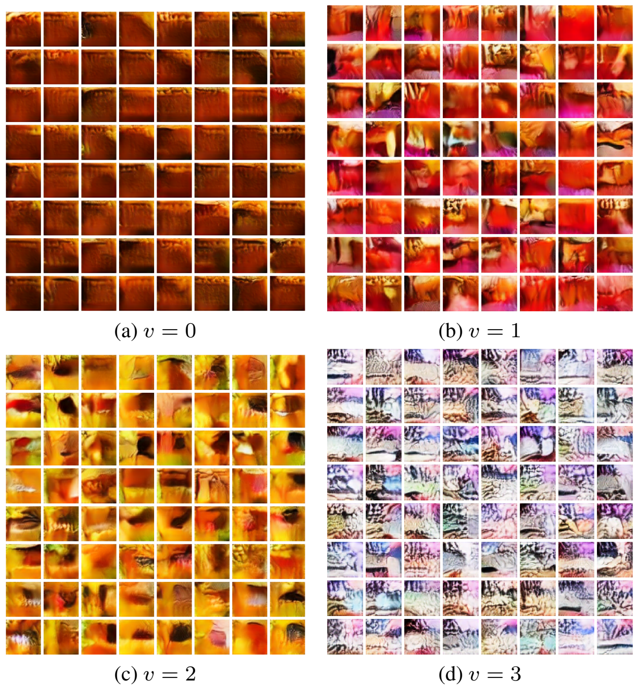


Figure 6. Visualization of domain images as the changes of improvement rate v.

11. Qualitative Evaluation on Domain Images

We visualized domain images generated from the initial Gaussian noises, which changed according to the improvement rate v. We generated 64 images from the updated initial Gaussian noise for each of the values $v = 0, 1, 2,$ and 3. As shown in Figure 6, we observed that consistent domain images were generated from the same initial Gaussian noise. Additionally, as depicted in Figure 7, it could be seen that as v increased, the domain progressively diverged from the domain (a). Specifically, when $v = 4$, the generated domain noticeably deviated from the initial domain, exhibiting more distinct features and a wider range of variations.

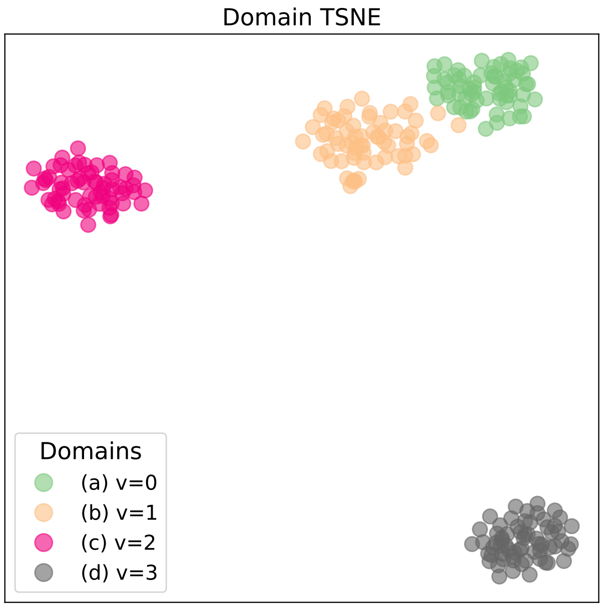


Figure 7. t-SNE of domain images as the changes of v.

12. Visualization of Augmented Samples

In this section, we qualitatively evaluated our method by visualizing the saliency maps and the augmented samples in DMControl-GB and Procgen. we set $\gamma = 0.5$ for visualization. As shown in Figure 8, our method effectively preserved task-relevant regions while generating diverse domain images. For Procgen, we found that the saliency map covered a relatively wider area compared to DMControl-GB as depicted in Figure 9. This was because, unlike DMControl-GB which primarily focused on moving objects, Procgen required consideration of multiple elements. The saliency maps were formed over obstacles that needed to be accounted for during gameplay.

13. Extended Metric Reporting

We provided both return and rank metrics for main results in DMControl-GB and Procgen as illustrated in Table 7 and 8.

14. Selection Criteria for Comparison Targets

Our method was integrated with Soft Actor-Critic (SAC)[2], Data-regularized Q (DrQ)[14], and Stabilized Q-Value Estimation under Augmentation (SVEA)[4] as baselines. When assessing the performance of data augmentation, we determined that comparing our method to algorithms like DrQ and SVEA, which efficiently leverage augmented data, was more suitable. Additionally,

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. Return ↑	717	797	675	702	**876**	759	907	837	912	**918**	760	916	796	916	**938**
Avg. Rank ↓	3.6	2.2	4.4	3.6	**1.0**	5	2.4	3.6	2.4	**1.4**	4.2	2.8	2.6	3.4	**2.0**
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. Return ↑	635	771	629	624	**834**	646	865	705	829	**866**	759	906	783	902	**930**
Avg. Rank ↓	3.8	1.8	3.8	4.2	**1.4**	4.8	**1.8**	4.0	2.4	2.0	3.8	2.6	2.8	3.6	**2.2**
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. Return ↑	485	716	504	465	**844**	596	853	681	705	**905**	530	840	667	783	**923**
Avg. Rank ↓	4.0	1.8	4.0	4.0	**1.2**	4.6	2.2	3.8	3.2	**1.2**	4.6	2.6	3.6	3.2	**1.0**
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. Return ↑	136	351	157	150	**616**	135	495	209	216	**680**	157	510	247	349	**705**
Avg. Rank ↓	4.4	2.0	3.8	3.8	**1.0**	4.8	1.8	3.8	3.4	**1.2**	5.0	2.4	3.6	3.0	**1.0**

Table 7. 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 and average rank.

	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. Return ↑	7.93	8.00	8.58	7.67	4.98	9.36	8.66	**11.45**
Avg. Rank ↓	4.75	5.38	3.63	5.38	7.00	3.63	3.50	**1.50**

Table 8. 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.

we chose SAC as a baseline to evaluate the effectiveness of our method in isolation. We compared our approach against NoAug, Overlay [3], Spectrum Random Masking (SRM)[5], and CrossNorm and SelfNorm (CNSN)[8]. We selected Overlay to compare the effectiveness of domain diversity using our method with that created from a predefined manifold. Also, we chose SRM and CNSN to assess which approach better preserves semantic integrity while enhancing domain diversity. For Procgen, we combined our method with Proximal Policy Optimization (PPO)[11] and Data-regularized Actor-Critic (DrAC)[10]. We combined the proposed method with PPO to demonstrate its effectiveness in isolation on an on-policy algorithm and selected DrAC for its ability to effectively leverage augmented data, similar to DrQ and SVEA in DMControl-GB. We compared our method with PPO, RAD [6], MixStyle [16], SAR [7], Robust ADversarIAl Loss (RADIAL) [9], and DrAC. We selected RAD to compare whether our method preserves

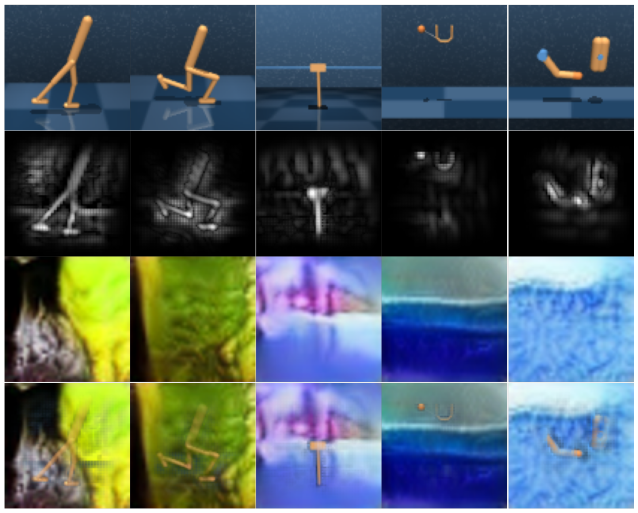


Figure 8. Visualization of augmented images in DMControl-GB. **1st row**: original image, **2nd row**: saliency map, **3rd row**: domain image, **4th row**: augmented image.

semantic integrity while enhancing diversity better than image transformation-based augmentations. MixStyle and SAR were chosen to assess which approach, between our method and existing normalization-based semantic preservation methods, is more effective across the different tasks. Finally, we selected RADIAL to evaluate the effectiveness of our method compared to an adversarial perturbation-based approach for training more robust agents under unseen domains.

15. Reproductivity Description

15.1. Detailed Implementation Guideline

For Stable Diffusion, we generated 40×40 domain images and upscaled them to the original resolution. No augmentations were applied to SAC. For DrQ, a random shift was applied to both the current and next states when sampling from the replay buffer. In contrast, SVEA applied a random shift and random convolution only to the current state as [8]. We set γ to 0.3 and K to 12. In Overlay, augmentation was applied to the current state by mixing data from the Places dataset [15] with the original data at a 50% ratio, following [4]. For SRM, we followed the guidelines of [5], applying augmentations to both the current and next states with a 50% probability. CNSN randomly activated 5 out of 11 CrossNorm layers following [8], but we did not use Overlay for SVEA. Finally, we used a batch size of 128 for all methods. For both our method and the Overlay approach, we initially sampled a batch of 64, then generated an augmented batch of size 64, and finally concatenated these to construct a 128-sized batch. We set γ to 0.3 and K to 12 for PPO, while for DrAC, we set K to 10. We implemented RAD on top of PPO and applied random crop for all transitions in the batch. Inspired by SAR, MixStyle was implemented by inserting a MixStyle layer between the second and third layers. For PPO, SAR, RADIAL, and DrAC, we set the hyperparameters as the default setting of the codebase. In Procgen, all transitions from one episode are divided into mini-batches. We set the number of mini-batches to eight. However, for our method in PPO, which includes augmented transitions in batch, we increase the number of mini-batches to 16 to ensure consistent batch sizes. For DrAC, it was designed to separate the original sample and augmented sample for joint learning, so we applied our method to the augmented sample portion, with the mini-batch size set to 8. We provide the value of the hyperparameters of DMControl-GB in Table 9. For Procgen, we provide the value of the hyperparameters in Table 10. For Adroit, we followed the default setting of [13] and provide the value of the hyperparameters in Table 11.

Hyperparameter	Value
Input size	$9\times84 \times 84$
Stacked images	3
Random shift	±4
Action repeat	2 (finger)
	8 (cartpole)
	4 (otherwise)
Discount factor	0.99
Episode length	1000
Number of environment steps	500,000
Replay buffer size	500,000
Optimizer	Adam
Batch size	64 (DGA2, Overlay) 128 (otherwise)
SVEA coefficients	$\alpha = 0.5, \beta = 0.5$
Actor update frequency	2
Critic update frequency	2
ψ update frequency	2
ψ momentum coefficient	0.05 (encoder)
	0.01 (critic)
Mask Ratio	0.3
Domain Shift Ratio	12

Table 9. Table of Hyperparameters in DMControl-GB.

Hyperparameter	Value
Input size	$3\times64 \times 64$
Stacked images	No
Generalized advantage estimates	0.95
Discount factor	0.999
Number of environment steps	25M (24,985,600)
Optimizer	Adam
Learning rate	5×10^{-4}
Number of processes	64
Timesteps per rollout	256
Epoches per rollout	3
Number of Mini-batches	16 (PPO+DGA2), 8 (Others)
Entropy bonus	0.01
PPO gradient clip range (ϵ)	0.2
Reward normalization	Yes
Mask Ratio	0.3
Domain Shift Ratio	12 (PPO), 10 (DrAC)

Table 10. Table of Hyperparameters in Procgen.

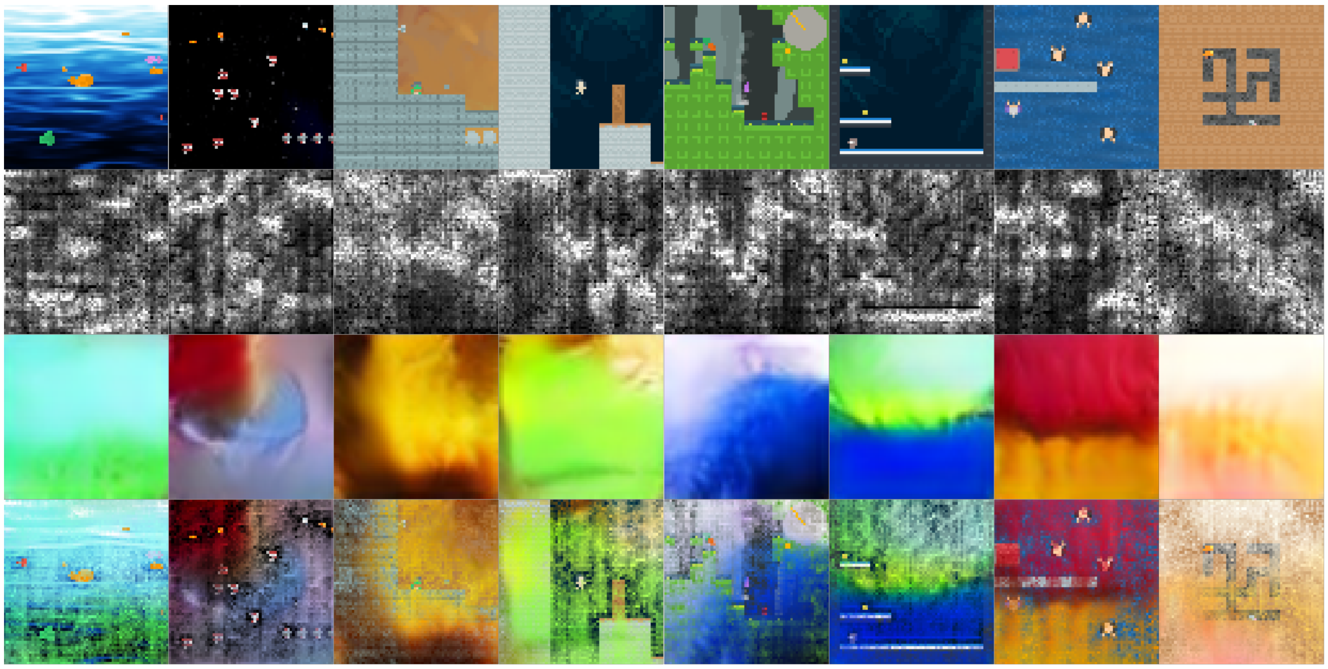


Figure 9. Visualization of augmented images in Procgen. **1st row**: original image, **2nd row**: saliency map, **3rd row**: domain image, **4th row**: augmented image.

Hyperparameter	Value
Input size	$9 \times 84 \times 84$
Discount factor γ	0.99
Action repeat	2
Stacked images	3
Learning rate	$1e^{-4}$
Random shift	± 4
Episode length	200
Evaluation episodes	100
Batch size	256
Optimizer	Adam

Table 11. Table of Hyperparameters in Adroit.

15.2. Hardware Setting

We conducted our experiments in a distributed manner across multiple server environments due to the limitation of resources. Specially, for the DMControl-GB experiments, we utilized servers equipped with RTX 3090 GPUs paired with Intel(R) Core(TM) i9-10900X CPUs, RTX 2080 Ti GPUs with Intel(R) Core(TM) i9-10900X CPUs, and RTX A4000 GPUs with Intel(R) Xeon(R) Gold 6426Y CPUs. For the Procgen experiments, we employed servers with RTX A5000, RTX 3090 GPUs paired with Intel(R) Xeon(R) E-2234 CPUs, as well as RTX 3090 GPUs with Intel(R) Xeon(R) E-2334 CPUs.

15.3. Conda Environment Setting

Conda environments were configured appropriately to match the specific requirements of each server setup.

16. Limitation and Future Work

DGA2 has two limitations: (1) One of the limitation of our approach is the time cost associated with generating augmented images based on the diffusion model. We aim to explore the integration of our method with faster diffusion models as part of our future work. (2) While our method adjusts the learning domain based on agent feedback, it does not guarantee that the domain will always be optimal for learning. We will refine the adaptive mechanism in subsequent studies to provide more suitable domains for agent training.

References

[1] 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. 1

[2] 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

[3] 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. 5

[4] 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. 4, 6

[5] 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. 5, 6

[6] 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. 5

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

[8] 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. 5, 6

[9] 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

[10] 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

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

[12] Wonil Song, Hyesong Choi, Kwanghoon Sohn, and Dongbo Min. A simple framework for generalization in visual rl under dynamic scene perturbations. *Advances in Neural Information Processing Systems*, 37:121790–121826, 2024. 2

[13] 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. 6

[14] 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

[15] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017. 6

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