Normalization Enhances Generalization in Visual Reinforcement Learning

Lu Li[1*] **Jiafei Lyu**[1*] **Guozheng Ma**[1] **Zilin Wang**[1]
Zhenjie Yang[2] **Xiu Li**[1†] **Zhiheng Li**[1†]

[1]Tsinghua University [2]Shanghai Jiao Tong University

{lilu21, lvjf20}@mails.tsinghua.edu.cn, zhhli@tsinghua.edu.cn

Abstract

Recent advances in visual reinforcement learning (RL) have led to impressive success in handling complex tasks. However, these methods have demonstrated limited generalization capability to visual disturbances, which poses a significant challenge for their real-world application and adaptability. Though normalization techniques have demonstrated huge success in supervised and unsupervised learning, their applications in visual RL are still scarce. In this paper, we explore the potential benefits of integrating normalization into visual RL methods with respect to generalization performance. We find that, perhaps surprisingly, incorporating suitable normalization techniques is sufficient to enhance the generalization capabilities, without any additional special design. We utilize the combination of two normalization techniques, CrossNorm and SelfNorm, for generalizable visual RL. Extensive experiments are conducted on DMControl Generalization Benchmark and CARLA to validate the effectiveness of our method. We show that our method significantly improves generalization capability while only marginally affecting sample efficiency. In particular, when integrated with DrQ-v2, our method enhances the test performance of DrQ-v2 on CARLA across various scenarios, from 14% of the training performance to **97%**. Our project page: https://sites.google.com/view/norm-generalization-vrl/home

1 Introduction

Visual reinforcement learning (RL), which leverages high-dimensional visual observations as inputs, has shown potential in a wide range of tasks, such as playing video games [33, 47] and robotic manipulation [26]. However, generalization remains a major challenge for visual RL methods. Even slight alterations, such as color or background changes, can result in considerable performance degradation in the testing environment, which in turn limits the real-world utility of these algorithms. In light of these challenges, it is essential to develop techniques that can improve the generalization capabilities of visual RL algorithms.

Existing literature mainly enhances the generalization capability of visual RL via data augmentation [16, 54, 12, 48] and domain randomization [45, 36, 35], aiming at learning policies invariant to the changes in the observations. However, recent studies [53, 25] show that certain data augmentation techniques may lead to a decrease in sample efficiency and even cause divergence. Other recent works improve the generalization performance by leveraging pre-trained image encoder [55] or segmenting important pixels from the test environment [2], *etc.* Unfortunately, most of them rely on

[*]Equal contribution.

[†]Corresponding authors.

NeurIPS 2023 Workshop on Generalization in Planning (GenPlan 2023).

knowledge or data from outer sources, *e.g.*, ImageNet [9]. We deem that an ideal method for zero-shot generalization should be able to achieve robust performance without relying on any out-of-domain data or prior knowledge of the target domain, and should be able to adapt effectively to a wide variety of environments and tasks.

Normalization techniques have achieved huge success in computer vision [46, 50, 43] and natural language processing [1, 51, 49]. Numerous normalization-related methods are proposed to improve the generalization capabilities of deep neural networks [40, 41, 13, 20]. Despite their popularity, normalization techniques seem to have not received sufficient attention in deep RL community. Though previous studies have investigated the effectiveness of normalization methods, *e.g.*, layer normalization [19, 34] and spectral normalization [32, 5, 31, 15], in deep RL algorithms, to the best of our knowledge, it is still unclear whether normalization can aid generalization in visual RL. Building upon these insights, we would like to ask the following question:

Can we develop a visual RL agent that employs normalization techniques and does not rely on prior knowledge and out-of-domain data, enabling it to generalize more effectively to unseen scenarios?

This inquiry drives our exploration of CrossNorm and SelfNorm [43], two normalization methods that have been proven to enhance generalization in computer vision tasks under distribution shifts. Since visual RL algorithms always rely on the encoder to output representations for policy learning and action execution, we need to ensure that the learned representation can generalize to unseen scenarios. To fulfill that, we propose to modify the encoder structure of the base visual RL algorithm by incorporating CrossNorm and SelfNorm for the downstream tasks. Our proposed normalization module is plug-and-play, and can be combined with any existing visual RL algorithms.

We evaluate the performance of our method on DeepMind Control Generalization Benchmark [17], a benchmark designed for evaluating generalization capabilities in robotic control tasks, and CARLA [10], a realistic autonomous driving simulator. Extensive experimental results demonstrate that when combined with DrQ [53] and DrQ-v2 [52], our proposed normalization module significantly improves their generalization capabilities without requiring any task-specific modifications or prior knowledge. Furthermore, our proposed module demonstrates compatibility and synergy with other generalization algorithms in visual RL (*e.g.*, SVEA [16]), thereby further enhance their generalization. This indicates the flexibility of our proposed module and its potential to be a valuable addition to the toolset for improving generalization in visual RL tasks. We believe this work offers another chance that allows visual RL algorithms to exhibit greater adaptability and robustness across diverse and dynamic environments. We aspire to propel the field of visual RL forward and broaden the scope of the potential applications of normalization techniques.

2 Related Work

2.1 Generalization in Visual RL

Over the past few years, considerable strides have been made towards narrowing the generalization gap in visual RL. An elementary strategy for improving generalization is to employ regularization techniques, initially developed for supervised learning [28]. These techniques include ℓ_2 regularization [14], entropy regularization [58], and dropout [18]. Unfortunately, these conventional regularization techniques exhibit limited effectiveness in improving generalization of visual RL and, in some cases, they may even have a negative impact on sample efficiency [8, 21]. As a result, recent studies have shifted their focus towards learning robust representations by leveraging bisimulation metrics [56, 23], multi-view information bottleneck (MIB) [11], pre-trained image encoder [55], *etc*. From an orthogonal perspective, data augmentation has demonstrated significant efficacy in enhancing generalization by leveraging prior knowledge as an inductive bias for the agent [25, 16, 54, 30]. However, the effectiveness of data augmentation-based techniques is significantly constrained by their highly task-specific nature and the requirement for substantial expert knowledge [37, 24]. One one hand, applying appropriate data augmentation techniques demands domain-specific knowledge, which limits their applicability to unfamiliar or novel environments. On the other hand, these techniques face challenges in generalizing to new domains due to their reliance on the alignment between augmentations and domain characteristics. In this study, our objective is to explore the utilization of normalization techniques to enhance the generalizability of visual RL, without relying on specific prior knowledge of the shift characteristics between the train and test environments. We note that two

recent studies [24, 30] present a comprehensive analysis of the generalization challenges in RL and the application of data augmentation in visual RL, which can be a nice reference.

2.2 Normalization

Normalization techniques play a crucial role in training deep neural networks [29, 38, 50]. They notably enhance optimization by normalizing input features, which is particularly advantageous for first-order optimization algorithms such as Stochastic Gradient Descent (SGD) [6, 60], known to excel in more isotropic landscapes [7]. Batch Normalization [22, 4, 39] (BN) is a method that normalizes intermediate feature maps using statistics computed from mini-batch samples. This technique has been found to significantly aid in the training of deep networks. Drawing inspiration from the success of BN, a variety of normalization techniques have since been introduced to accommodate different learning scenarios, *e.g.*, layer normalization [1, 42, 57], spectral normalization [32, 27], *etc*.

Despite the huge success and wide applications of normalization techniques, they are not commonly employed in deep RL. This is largely attributed to the online learning nature of RL, which leads to a non-independent and identically distributed (non-i.i.d) input data distribution. Such distribution does not align with the requirements of many normalization techniques. [3] shows that direct application of BN and LN proves to be ineffective for RL. Instead, it introduces cross-normalization, which computes mean feature subtraction using both on-policy and off-policy state-action pairs, leading to better sample efficiency. Moreover, spectral normalization has been found to be effective in stabilizing the training process of RL [5, 15].

It is interesting to ask: since normalization techniques have shown benefits for generalization to new tasks in computer vision, then whether normalization techniques have the potential to enhance the generalization ability of the visual RL algorithms. To the best of our knowledge, none of the prior work explores this issue, and our goal in this work is to answer this question.

3 Preliminary

3.1 Visual Reinforcement Learning

We consider learning in a Partially Observable Markov Decision Processes (POMDPs) specified by the tuple $\mathcal{M} : \langle \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{P}, r, \gamma \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{O}$ is the observation space, $\mathcal{A}$ is the action space, $\mathcal{P}(\cdot|s,a) : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the transition probability, $r(s,a) : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the scalar reward function, $\gamma \in [0,1)$ is the discount factor. In the context of generalization setting, we have a set of such POMDPs $M = \{\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_n\}$ while our agent only has access to one fixed POMDP among them, denoted as $\mathcal{M}_0$. We aim to train an RL agent to learn a policy $\pi_\theta(\cdot|s)$ parameterized by the parameter θ in $\mathcal{M}_0$, with the objective of maximizing the expected cumulative return $J(\theta) = \mathbb{E}_{a_t \sim \pi_\theta(\cdot|s_t), s_t \sim \mathcal{P}}[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)]$ across the entire set of POMDPs in a zero-shot manner, where T is the horizon of the POMDP.

3.2 CrossNorm and SelfNorm

CrossNorm and SelfNorm [43] were initially introduced to improve generalization capabilities in the face of distribution shifts within computer vision tasks. To broaden the training distribution, CrossNorm swaps the mean and standard deviation of channel $\mathcal{A}$, denoted as $\mu_\mathcal{A}$ and $\sigma_\mathcal{A}$ respectively, with the mean and standard deviation of channel $\mathcal{B}$, denoted as $\mu_\mathcal{B}$ and $\sigma_\mathcal{B}$ respectively. In other words, it exchanges μ and σ values between channels $\mathcal{A}$ and $\mathcal{B}$, as shown in Equation 1:

$$\mathcal{A}' = \sigma_\mathcal{B} \frac{\mathcal{A} - \mu_\mathcal{A}}{\sigma_\mathcal{A}} + \mu_\mathcal{B}, \qquad \mathcal{B}' = \sigma_\mathcal{A} \frac{\mathcal{B} - \mu_\mathcal{B}}{\sigma_\mathcal{B}} + \mu_\mathcal{A}, \tag{1}$$

While CrossNorm enlarges the training distribution, the motivation of SelfNorm is to bridge the train-test distribution gap. To achieve this, SelfNorm replaces $\mathcal{A}$ and $\mathcal{B}$ with recalibrated mean $\mu'_\mathcal{A} = f(\mu_\mathcal{A}, \sigma_\mathcal{A})\mu_\mathcal{A}$ and standard deviation $\sigma'_\mathcal{A} = g(\mu_\mathcal{A}, \sigma_\mathcal{A})\sigma_\mathcal{A}$, where f and g are the attention functions. The adjusted feature becomes as Equation 2:

$$\hat{\mathcal{A}} = \sigma'_\mathcal{A} \frac{\mathcal{A} - \mu_\mathcal{A}}{\sigma_\mathcal{A}} + \mu'_\mathcal{A}. \tag{2}$$

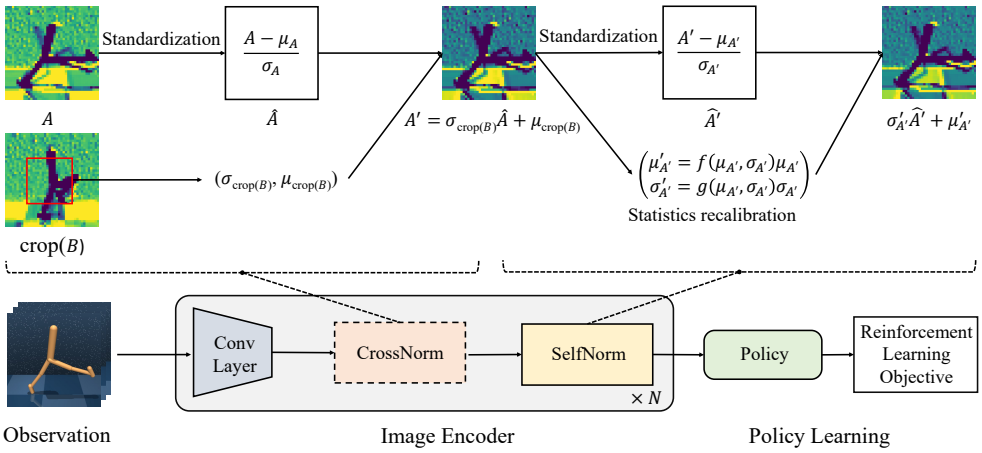


Figure 1: The pipeline of our method. CrossNorm is positioned after the convolutional layer and is followed by SelfNorm. Each CrossNorm layer is randomly activated during training and becomes inactive during testing. Instead, SelfNorm is adopted during training and remains functional during testing. Our method notably does not introduce new learning objective or utilize out-of-domain data.

As f and g learn to scale $\mu_{\mathcal{A}}$ and $\sigma_{\mathcal{A}}$ based on their values, the method adapts to the specific characteristics of the data. While CrossNorm expands the data distribution, SelfNorm aims to emphasize the discriminative styles shared by both training and test distributions while de-emphasizing the insignificant styles.

4 Method

4.1 Enhancing Generalization in Visual RL via Normalization

The primary challenge in visual RL generalization stems from distribution shifts in observations. This issue is particularly prominent due to the diverse and dynamic nature of environments in RL tasks. Recognizing the proven effectiveness of CrossNorm and SelfNorm in bolstering generalization under distribution shift in computer vision tasks, we explore the possibilities of these normalization techniques in visual RL. By integrating CrossNorm and SelfNorm, we aim to enhance the generalization capability of visual RL, fostering the learning of more robust and generalizable representations.

Although computer vision tasks and visual RL tasks both involve the representation learning of visual input, their respective data distributions can be quite different. While CrossNorm is inspired by the observation that computer vision datasets are typically rich and diverse, stemming from a variety of sources, visual RL generally involves training the agent within a single task and environment. This situation results in a notably limited data distribution. In other words, the difference between the mean and standard deviation of channel $\mathcal{A}$ and channel $\mathcal{B}$ tends to be small, thus diminishing the effect of the CrossNorm. Hence, it becomes crucial to further diversify and expand the data distribution. To achieve this, we utilize random cropping during the computation of the channel's mean μ and standard deviation σ, as illustrated in Equation 3. This technique can result in a wider distribution of the mean and the standard deviation values, further contributing to its ability to adapt to various data distributions.

$$\mathcal{A}' = \sigma_{\text{crop}(\mathcal{B})}\frac{\mathcal{A}-\mu_{\mathcal{A}}}{\sigma_{\mathcal{A}}} + \mu_{\text{crop}(\mathcal{B})}, \qquad \mathcal{B}' = \sigma_{\text{crop}(\mathcal{A})}\frac{\mathcal{B}-\mu_{\mathcal{B}}}{\sigma_{\mathcal{B}}} + \mu_{\text{crop}(\mathcal{A})}, \tag{3}$$

We present the pipeline of our proposed method in Figure 1, where our core contribution is the proposal of a plug-and-play module that is equipped with cropped CrossNorm and SelfNorm. Notably, we arrange CrossNorm immediately after the convolution layer, followed by SelfNorm. This sequence is designed to optimally leverage the effects of these two operations, with CrossNorm augmenting the feature diversity before SelfNorm performs intra-instance normalization. Taking into account their characteristics, CrossNorm is activated solely during the training phase, whereas SelfNorm is utilized during the training phase and remains functional during the testing phase.

During each forward pass in the training process, a predetermined number of CrossNorm layers are randomly activated. For these activated layers, each instance in the mini-batch has its μ and σ values for every channel swapped with those of the same channels of another randomly chosen instance. The remaining CrossNorm layers stay inactive during this process. Generally, how many CrossNorm layers can be activated strongly depends on how many hidden layers the encoder of the base algorithm has. We allow a dynamic utilization of the CrossNorm layers because unlike supervised learning, where the model usually has a strong supervised signal and various methods can be applied to learn task-relevant representations, visual RL is lack of sufficient supervised signals. It is thus difficult for it to effectively capture important knowledge from the pixels. As a result, the training process in visual RL is often more fragile and susceptible to disruptions. By selecting an appropriate number of active CrossNorm layers during the training process, we can effectively manage the learning difficulty, ensuring more stable training dynamics in the learning process.

The role of CrossNorm can be seen as a form of data augmentation. However, unlike traditional data augmentation methods that have been used in visual RL, CrossNorm operates directly on the feature maps rather than the raw observations. This distinction allows CrossNorm to facilitate more diverse alterations. On the other hand, similar to traditional data augmentation methods, CrossNorm improves generalization at the cost of sample efficiency, while SelfNorm aims to offset this trade-off, thereby ensuring a more stable learning process.

Importantly, our method does not introduce new learning objectives or require any out-of-domain data or prior knowledge. This makes it a self-contained and flexible approach to generalization. Moreover, our method is not only compatible with standard RL algorithms but can also be seamlessly integrated with other techniques aimed at enhancing the generalization of visual RL, and can further improve the robustness of these methods. This versatility further underscores the generality of our approach.

5 Experiments

Our experiments are aimed to investigate the following questions: (1) Does our method enhance the generalization capabilities of vanilla visual RL methods and to what extent does it impact the training performance? (2) Is our proposed method general enough to be integrated with existing generalization methods in visual RL to further enhance their capability?

5.1 Generalization on CARLA Autonomous Driving Tasks

5.1.1 Experimental setup

To assess our method in realistic scenarios and better gauge its effectiveness and generalization capabilities, We evaluate the performance of our method in the CARLA autonomous driving simulator, which offers realistic observations and complex driving scenarios.

We build our method upon DrQ-v2 [52] and compare the generalization ability of DrQ-v2+CNSN with state-of-the-art methods and strong baselines: **DrQ-v2** [52]: our base visual RL algorithm, which is the prior state-of-the-art model-free visual RL algorithm in terms of sample efficiency. It demonstrates superior performance on a variety of tasks while maintaining high sample efficiency, making it a suitable foundation for our research in developing more generalizable visual RL methods. **SVEA** [16]: the previous state-of-the-art data augmentation based method for generalization, which achieves improved performance by reducing Q-variance through the use of an auxiliary loss.

Figure 2: In CARLA autonomous driving simulator, agents are trained under one fixed weather condition. These agents are then expected to generalize to unseen weather conditions in a zero-shot manner. These weather conditions vary in aspects such as lighting, humidity, and other factors, leading to differences not only in visual observation but also in the dynamics of the environment.

Our experimental setting in CARLA is adapted from [56]. Due to the fact that the encoder in DrQ-v2 has four hidden layers, the maximum number of activated CrossNorm modules in DrQ-v2+CNSN is four. In CARLA experiments, all four CrossNorm layers are activated during the training phase. All agents are trained under one fixed weather condition for 200,000 environment steps. Their performance is then assessed across various other weather conditions within the same map and task, as shown in Figure 2. Moreover, it's worth noting that not only the visual observations change with different weather conditions, but also the dynamics of POMDP might vary due to factors like rain.

Since we employ DrQ-v2 as our base visual RL algorithm and baseline method, we also adapt and reimplement SVEA using the DrQ-v2 structure to ensure a fair comparison. We then train the two variations of SVEA on CARLA, one applying random convolution as data augmentation and the other employing random overlay with images from Places365 dataset [59], respectively.

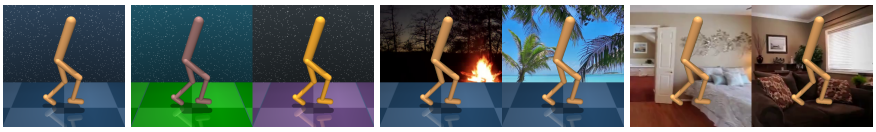
5.1.2 Generalization performance

The generalization performance results are shown in Table 1. The results indicate that DrQ-v2 cannot adapt to new weather with different lighting, humidity, *etc*. However, by combing it with CNSN, DrQ-v2+CNSN is enough to generalize well on most of the unseen complicated scenes without a performance drop. Notably, DrQ-v2+CNSN significantly improves the test average performance from DrQ-v2's 14% of the training performance to **97%** of the training performance.

Moreover, it can be seen that both variants of SVEA, using random convolution and random overlay respectively, exhibit a significant performance drop in unseen weather conditions. For example, SVEA(conv) trained under HardRainNoon achieves an average return of 53 when tested under WetNoon, while DrQ-v2+CNSN attains an average performance of **173**, despite the fact that DrQ-v2+CNSN has lower training performance than SVEA(conv). The primary reason for the significant performance drop of SVEA is that the two data augmentation techniques it employs do not align well with the test environments. Consequently, these augmentations do not provide sufficient generalization capability for unseen weather conditions, which ultimately limits SVEA's robustness in these scenarios. This finding underscores the necessity for more adaptable and versatile visual RL techniques that can effectively cope with the dynamic and intricate nature of real-world environments. Instead, our method does not rely on any task-specific data augmentation or prior knowledge, and can lead to more robust performance in a wide range of real-world scenarios.

Table 1: **CARLA generalization results.** Training and testing performance (episode return) of methods trained in one fixed weather and evaluated on other 6 unseen weather conditions. We separately conduct training under two distinct weather conditions: WetCloudySunset (WCS) and HardRainNoon (HRN). SVEA(conv) refers to the variant of SVEA that utilizes random convolution for data augmentation, while SVEA(overlay) denotes the variant that employs random overlay for data augmentation. For a fair comparison, we have reimplemented these two versions of SVEA using DrQ-v2. The results presented are final performance averaged over 5 random seeds, with each seed corresponding to 50 evaluation episodes for each weather condition.

Method	DrQ-v2		DrQ-v2+CNSN		SVEA(conv)		SVEA(overlay)	
Training Weather	WCS	HRN	WCS	HRN	WCS	HRN	WCS	HRN
Training	$249_{\pm 23}$	$249_{\pm 34}$	$225_{\pm 11}$	$225_{\pm 14}$	$221_{\pm 25}$	$243_{\pm 28}$	$173_{\pm 87}$	$204_{\pm 11}$
WetCloudySunset	$249_{\pm 23}$	$118_{\pm 43}$	$225_{\pm 11}$	$\mathbf{211_{\pm 9}}$	$221_{\pm 25}$	$184_{\pm 18}$	$173_{\pm 87}$	$30_{\pm 21}$
MidRainSunset	$184_{\pm 18}$	$-2_{\pm 11}$	$\mathbf{233_{\pm 32}}$	$\mathbf{208_{\pm 11}}$	$184_{\pm 44}$	$59_{\pm 91}$	$160_{\pm 24}$	$68_{\pm 22}$
HardRainSunset	$36_{\pm 26}$	$-3_{\pm 10}$	$\mathbf{230_{\pm 21}}$	$\mathbf{221_{\pm 16}}$	$169_{\pm 41}$	$79_{\pm 93}$	$148_{\pm 31}$	$87_{\pm 18}$
WetNoon	$2_{\pm 6}$	$5_{\pm 4}$	$\mathbf{210_{\pm 9}}$	$\mathbf{173_{\pm 43}}$	$82_{\pm 85}$	$51_{\pm 53}$	$1_{\pm 6}$	$-1_{\pm 2}$
SoftRainNoon	$-2_{\pm 7}$	$-6_{\pm 8}$	$\mathbf{232_{\pm 40}}$	$\mathbf{205_{\pm 19}}$	$101_{\pm 90}$	$59_{\pm 69}$	$57_{\pm 50}$	$14_{\pm 26}$
MidRainyNoon	$89_{\pm 38}$	$-3_{\pm 8}$	$\mathbf{237_{\pm 27}}$	$\mathbf{215_{\pm 17}}$	$190_{\pm 38}$	$69_{\pm 95}$	$143_{\pm 29}$	$166_{\pm 36}$
HardRainNoon	$145_{\pm 20}$	$249_{\pm 34}$	$\mathbf{237_{\pm 25}}$	$225_{\pm 14}$	$190_{\pm 36}$	$243_{\pm 28}$	$146_{\pm 25}$	$204_{\pm 11}$
Average test return	$54_{\pm 56}$	$18_{\pm 49}$	$\mathbf{230_{\pm 29}}$	$\mathbf{206_{\pm 27}}$	$153_{\pm 75}$	$81_{\pm 88}$	$109_{\pm 67}$	$61_{\pm 61}$



(a) Training (b) "*Color hard*" (c) "*Video easy*" (d) "*Video hard*"

Figure 3: Examples of training and testing environments in DMC-GB.

5.2 Generalization on DMControl Generalization Benchmark

5.2.1 Experimental setup

We also assess our method on the DeepMind Control Generalization Benchmark (DMC-GB) [17], a well-established benchmark for evaluating the generalization capabilities of visual RL algorithms, based on DeepMind Control Suite [44]. In DMC-GB, agents are trained in standard DeepMind Control environments and subsequently evaluated in visually disturbed environments. These disturbances include changes in color (*color hard*) and the replacement of backgrounds with moving videos (*video easy*, *video hard*), as shown in Figure 3.

For the easy tasks in DeepMind Control Suites, we utilize DrQ as our base visual reinforcement learning algorithm. For medium tasks that DrQ struggles to solve, we employ DrQ-v2 due to its capability to address complex locomotion tasks using pixel observations, providing a more effective solution for these more challenging tasks. To ensure a fair comparison, we have re-implemented SVEA using DrQ-v2 as its base algorithm for medium tasks, considering that the original SVEA was implemented based on DrQ. Our experimental setting mainly follows that of SVEA [16]. For the easy tasks, all agents were trained for 500,000 steps in the vanilla training environments without visual alteration. Meanwhile, for the medium tasks, the training process is extended to 1,500,000 steps for all methods. Note that DrQ contains 11 hidden layers in its encoder while DrQ-v2 only has 4. Across our experiments, we randomly activate 5 out of 11 CrossNorm layers in DrQ-CNSN during the training phase and activate all 4 CrossNorm layers for DrQ-v2. Furthermore, we recognize PIE-G [55] as a state-of-the-art baseline, particularly effective in addressing the challenging *video hard* scenarios. We utilize the ResNet+CNSN pre-trained model, deactivating all CrossNorm and SelfNorm during the RL agent's training. For PIE-G+CNSN, a ResNet50+CNSN pre-trained model from [43] is employed. Meanwhile, for PIE-G, we use a ResNet50 pre-trained model from the *torchvision* package.

5.2.2 Generalization performance

To further assess the effectiveness and flexibility of the CrossNorm and SelfNorm in aiding the generalization ability of the visual RL policies, we build CrossNorm and SelfNorm on top of four visual RL algorithms, DrQ, DrQ-v2, SVEA, and PIE-G. We activate 5 out of 11 CrossNorm layers for SVEA on easy tasks and all 4 CrossNorm layers for SVEA (like DrQ-v2) on medium tasks. We assess the testing performance of DrQ+CNSN, DrQ-v2+CNSN, SVEA+CNSN, and PIE-G+CNSN across the following settings: *color hard*, *video easy*, and *video hard*, where *color hard* tasks have randomly jittered color, *video easy* and *video hard* tasks replace the background with the unseen moving videos. Notably, the most challenging one is *video hard*, where the reference plane of the ground is also removed. We adopt SVEA with random overlay for all these settings and baselines, since it performs better than SVEA(conv) on *video easy* and *video hard* environments. This enables us to investigate whether our module (CrossNorm and SelfNorm) can further enhance generalization when integrated with strong data augmentation-based approaches. As illustrated in Table 2, incorporating CrossNorm and SelfNorm significantly improves test performance in most of the testing environments compared to the original methods, while maintaining comparable performance in the remaining situations. In particular, when applied to DrQ and DrQ-v2, our method achieves substantial improvements in *video easy* and *video hard* environments, with average performance improvement of **155%** and **80%**, respectively. Additionally, when combined with SVEA, our method yields notable improvements across most environments. Similarly, in combination with PIE-G, our approach registers significant advancements in *video hard* scenarios. These results further substantiate the efficacy and adaptability of our proposed method.

Table 2: **DMC-GB generalization results.** Performance on *video easy* and *video hard* testing environments. SVEA refers to the implementation of SVEA that utilizes random overlay as data augmentation method. All the results are averaged over 5 random seeds. *color hard* results can be found in **Appendix A**.

Easy tasks-*video easy*	DrQ	+CNSN	SVEA	+CNSN	PIE-G	+CNSN
Walker Walk	682±89	**792±67**	819±71	**842±58**	**917±15**	**923±8**
Walker Stand	873±83	**957±12**	961±8	**967±6**	**961±7**	**956±9**
Cartpole Swingup	485±105	**498±26**	**782±27**	752±26	**421±76**	353±40
Ball in cup Catch	318±157	**584±83**	871±106	**913±45**	854±54	**892±43**
Medium tasks-*video easy*	DrQ-v2	+CNSN	SVEA	+CNSN	PIE-G	+CNSN
Cheetah Run	42±19	**274±35**	**408±78**	**404±29**	327±54	**347±34**
Walker Run	124±31	**452±22**	**611±20**	**609±18**	520±16	**541±17**

Easy tasks-*video hard*	DrQ	+CNSN	SVEA	+CNSN	PIE-G	+CNSN
Walker Walk	104±22	**166±28**	377±93	**480±46**	633±59	**669±42**
Walker Stand	289±49	**492±62**	834±46	**871±23**	**902±38**	856±38
Cartpole Swingup	138±9	**171±13**	393±45	**417±31**	285±45	**309±19**
Ball in cup Catch	92±23	**199±138**	403±174	**691±72**	**741±108**	721±7
Medium tasks-*video hard*	DrQ-v2	+CNSN	SVEA	+CNSN	PIE-G	+CNSN
Cheetah Run	21±5	**49±4**	68±9	**88±9**	153±40	**162±23**
Walker Run	24±2	**43±2**	120±8	**148±8**	252±7	**281±5**

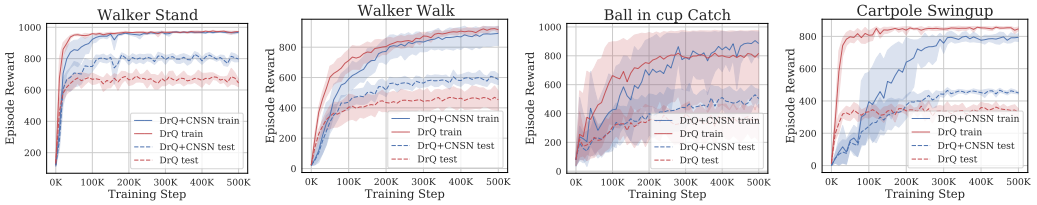


Figure 4: Training and testing performance of DrQ+CNSN against DrQ. The red line is DrQ and blue one corresponds to DrQ+CNSN. The test performance is calculated as the average across the three test settings of DMC-GB, *i.e.*, *color hard*, *video easy*, *video hard*.

5.2.3 Sample efficiency and generalization gap

We present the learning curves of DrQ and DrQ+CNSN on four tasks in Figure 4. One can find that the generalization gap is significantly reduced by incorporating CrossNorm and SelfNorm. It is worth noting that adopting normalization techniques harms the sample efficiency in the training environments. However, such sacrifice is tolerable since the difference in the training curves on most of the tasks are marginal, while the generalization capability of the agent is largely boosted.

5.3 Ablation Study

To validate the essentiality of the design choices incorporated into our method, we perform a series of ablation studies to delve deeper into the understanding of our proposed approach.

5.3.1 Ablation of CrossNorm and SelfNorm

Our proposed module is a combination of (cropped) CrossNorm (CN) and SelfNorm (SN). To investigate the individual contributions of CN and SN to generalization capability, we evaluate DrQ+CN and DrQ+SN on several tasks from the DMC-GB and CARLA. This analysis will help us understand the impact of each component on the overall performance of our proposed method. The results are shown in Table 3.

Table 3: **Ablation study results.** This table presents the impact of various components on the performance of our method. w/o Crop refers to DrQ+CNSN without using random cropping in CrossNorm. The results of the CARLA benchmark were obtained by training in the WetCloudySunset weather condition and testing in 6 other different weather conditions.

Tasks	Setting	Method				
		DrQ	+CN	+SN	+CNSN	w/o Crop
Walker Walk	*color hard*	520±91	**823±21**	188±34	815±65	634±124
	video easy	682±89	829±60	207±39	**842±58**	664±121
	video hard	104±22	**196±41**	89±24	166±28	130±35
Walker Stand	*color hard*	770±71	**951±27**	525±66	942±19	841±50
	video easy	873±83	945±33	445±113	**957±12**	857±129
	video hard	289±49	461±81	223±22	**492±62**	322±46
Cartpole Swingup	*color hard*	586±52	**695±38**	187±34	679±35	560±134
	video easy	485±105	**515±29**	135±9	498±26	410±89
	video hard	138±9	**183±4**	111±22	171±13	155±20
Ball in cup Catch	*color hard*	365±210	885±73	174±6	**894±78**	463±89
	video easy	318±157	**599±29**	161±33	584±83	391±116
	video hard	92±23	146±54	75±44	**199±138**	104±35
CARLA	unseen weather	54±56	183±91	71±70	**230±29**	185±94

As previously mentioned, while computer vision datasets often originate from diverse sources, the training of visual RL agents typically occurs within a single task and environment, leading to a relatively narrow data distribution compared to that of computer vision data. Therefore, it's understandable that using SelfNorm alone aids computer vision tasks but could reduce the robustness of visual RL. The μ and σ of the feature maps tend to be relatively stable, causing SelfNorm to overfit, which ultimately leads to a decrease in generalization performance.

It seems that using CrossNorm alone upon DrQ sometimes results in comparable test performance against DrQ+CNSN. However, in more complex autonomous driving scenarios, we observe that relying solely on CrossNorm does not yield performance as good as using both CrossNorm and SelfNorm. The results suggest that SelfNorm may only be effective in visual RL tasks with the existence of CrossNorm. Furthermore, the empirical results in CARLA scenarios also validate that. It is interesting to note here that it seems that for complex real-world applications, it is beneficial to combine the above two normalization techniques.

5.3.2 Ablation on the random cropping of CrossNorm

We also investigate how random cropping of CrossNorm (Equation 3) helps the generalization in DMC-GB tasks, as shown in Table 3. The results show that the inclusion of random cropping when calculating μ and σ in CrossNorm significantly improves generalization performance compared to cases without cropping.

6 Conclusion

In this paper, we explore the potential benefits of normalization techniques on the generalization capabilities of visual RL and propose a novel normalization module containing CrossNorm and SelfNorm for generalizable RL. By conducting extensive experiments upon different base algorithms across diverse tasks in two generalization benchmarks, DMC-GB and CARLA autonomous driving simulator, we demonstrate that our method is able to enhance generalization capability without the help of out-of-domain data and prior knowledge. These characteristics establish our approach as a self-contained method for achieving generalizable visual RL. Our method can be integrated with any visual RL algorithm, making it a valuable approach for tackling unpredictable environments.

Limitation: The limitations of our method lie in one needs to predetermine the number of activated CrossNorm layers, and may need some experimentation to obtain optimal results for different environments.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] David Bertoin, Adil Zouitine, Mehdi Zouitine, and Emmanuel Rachelson. Look where you look! saliency-guided q-networks for generalization in visual reinforcement learning. In *Neural Information Processing Systems*, 2022.

[3] Aditya Bhatt, Max Argus, Artemij Amiranashvili, and Thomas Brox. Crossnorm: Normalization for off-policy td reinforcement learning, 2019.

[4] Johan Bjorck, Carla P. Gomes, and Bart Selman. Understanding batch normalization. In *Neural Information Processing Systems*, 2018.

[5] Nils Bjorck, Carla P Gomes, and Kilian Q Weinberger. Towards deeper deep reinforcement learning with spectral normalization. *Advances in Neural Information Processing Systems*, 34:8242–8255, 2021.

[6] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *International Conference on Computational Statistics*, 2010.

[7] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[8] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*. PMLR, 2019.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[11] Jiameng Fan and Wenchao Li. Dribo: Robust deep reinforcement learning via multi-view information bottleneck. In *International Conference on Machine Learning*, pages 6074–6102. PMLR, 2022.

[12] 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 Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 3088–3099. PMLR, 18–24 Jul 2021.

[13] Xinjie Fan, Qifei Wang, Junjie Ke, Feng Yang, Boqing Gong, and Mingyuan Zhou. Adversarially adaptive normalization for single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8208–8217, 2021.

[14] Jesse Farebrother, Marlos C Machado, and Michael Bowling. Generalization and regularization in dqn. *arXiv preprint arXiv:1810.00123*, 2018.

[15] Florin Gogianu, Tudor Berariu, Mihaela C Rosca, Claudia Clopath, Lucian Busoniu, and Razvan Pascanu. Spectral normalisation for deep reinforcement learning: an optimisation perspective. In *International Conference on Machine Learning*, pages 3734–3744. PMLR, 2021.

[16] 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, 2021.

[17] Nicklas Hansen and Xiaolong Wang. Generalization in reinforcement learning by soft data augmentation. In *International Conference on Robotics and Automation*, 2021.

[18] Matthew Hausknecht and Nolan Wagener. Consistent dropout for policy gradient reinforcement learning. *arXiv preprint arXiv:2202.11818*, 2022.

[19] Takuya Hiraoka, Takahisa Imagawa, Taisei Hashimoto, Takashi Onishi, and Yoshimasa Tsuruoka. Dropout q-functions for doubly efficient reinforcement learning. *ArXiv*, abs/2110.02034, 2021.

[20] Lei Huang, Jie Qin, Yi Zhou, Fan Zhu, Li Liu, and Ling Shao. Normalization techniques in training dnns: Methodology, analysis and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[21] Maximilian Igl, Kamil Ciosek, Yingzhen Li, Sebastian Tschiatschek, Cheng Zhang, Sam Devlin, and Katja Hofmann. Generalization in reinforcement learning with selective noise injection and information bottleneck. *Advances in neural information processing systems*, 32, 2019.

[22] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[23] Mete Kemertas and Tristan Aumentado-Armstrong. Towards robust bisimulation metric learning. *Advances in Neural Information Processing Systems*, 34:4764–4777, 2021.

[24] Robert Kirk, Amy Zhang, Edward Grefenstette, and Tim Rocktäschel. A survey of generalisation in deep reinforcement learning. *arXiv preprint arXiv:2111.09794*, 2021.

[25] 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.

[26] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[27] Zinan Lin, Vyas Sekar, and Giulia C. Fanti. Why spectral normalization stabilizes gans: Analysis and improvements. In *Neural Information Processing Systems*, 2020.

[28] Zhuang Liu, Xuanlin Li, Bingyi Kang, and Trevor Darrell. Regularization matters in policy optimization-an empirical study on continuous control. In *International Conference on Learning Representations*, 2020.

[29] Ekdeep Singh Lubana, Robert P. Dick, and Hidenori Tanaka. Beyond batchnorm: Towards a unified understanding of normalization in deep learning. In *Neural Information Processing Systems*, 2021.

[30] 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.

[31] K. Mehta, Anuj Mahajan, and Priyesh Kumar. Effects of spectral normalization in multi-agent reinforcement learning. *ArXiv*, abs/2212.05331, 2022.

[32] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[33] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[34] Emilio Parisotto, H. Francis Song, Jack W. Rae, Razvan Pascanu, Çaglar Gülçehre, Siddhant M. Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, Matthew M. Botvinick, Nicolas Manfred Otto Heess, and Raia Hadsell. Stabilizing transformers for reinforcement learning. In *International Conference on Machine Learning*, 2019.

[35] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810. IEEE, 2018.

[36] Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. Asymmetric actor critic for image-based robot learning, 2017.

[37] 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.

[38] Tim Salimans and Diederik P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Neural Information Processing Systems*, 2016.

[39] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *Neural Information Processing Systems*, 2018.

[40] Amartya Sanyal, Philip H. S. Torr, and Puneet Kumar Dokania. Stable rank normalization for improved generalization in neural networks and gans. *ArXiv*, abs/1906.04659, 2019.

[41] Seonguk Seo, Yumin Suh, Dongwan Kim, Jongwoo Han, and Bohyung Han. Learning to optimize domain specific normalization for domain generalization. In *European Conference on Computer Vision*, 2019.

[42] Jiacheng Sun, Xiangyong Cao, Hanwen Liang, Weiran Huang, Zewei Chen, and Zhenguo Li. New interpretations of normalization methods in deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[43] Zhiqiang Tang, Yunhe Gao, Yi Zhu, Zhi Zhang, Mu Li, and Dimitris Metaxas. Crossnorm and selfnorm for generalization under distribution shifts. In *ICCV 2021*, 2021.

[44] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

[45] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[46] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization, 2017.

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

[48] 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.

[49] Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. Multi-passage bert: A globally normalized bert model for open-domain question answering. In *Conference on Empirical Methods in Natural Language Processing*, 2019.

[50] Yuxin Wu and Kaiming He. Group normalization. *International Journal of Computer Vision*, 128:742–755, 2018.

[51] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, 2020.

[52] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021.

[53] 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.

[54] Zhecheng Yuan, Guozheng Ma, Yao Mu, Bo Xia, Bo Yuan, Xueqian Wang, Ping Luo, and Huazhe Xu. Don't touch what matters: Task-aware lipschitz data augmentationfor visual reinforcement learning. *arXiv preprint arXiv:2202.09982*, 2022.

[55] Zhecheng Yuan, Zhengrong Xue, Bo Yuan, Xueqian Wang, Yi Wu, Yang Gao, and Huazhe Xu. Pre-trained image encoder for generalizable visual reinforcement learning. In *Neural Information Processing Systems*, 2022.

[56] Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. *arXiv preprint arXiv:2006.10742*, 2020.

[57] Biao Zhang and Rico Sennrich. Root mean square layer normalization. *ArXiv*, abs/1910.07467, 2019.

[58] Chiyuan Zhang, Oriol Vinyals, Remi Munos, and Samy Bengio. A study on overfitting in deep reinforcement learning. *arXiv preprint arXiv:1804.06893*, 2018.

[59] 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*, 2017.

[60] Martin A. Zinkevich, Markus Weimer, Alex Smola, and Lihong Li. Parallelized stochastic gradient descent. In *Neural Information Processing Systems*, 2010.

A Additional Results

In this section, we provide additional results to further illustrate the findings of our study.

A.1 PIE-G experiments

A.2 Additional DMC-GB generalization results

As illustrated in Table 4, the results from the *color hard* environments further confirm that the combination of CrossNorm and SelfNorm significantly enhances the generalization performance under *color hard* evaluation setting.

Table 4: **DMC-GB *color hard* generalization results.** Performance comparison of DrQ and DrQ+CNSN on *color hard* testing environment. All the results are averaged over 5 random seeds.

Easy tasks-*color hard*	DrQ	+CNSN
Walker Walk	$520_{\pm 91}$	$\mathbf{815}_{\pm \mathbf{65}}$ (**+56.7%**)
Walker Stand	$770_{\pm 71}$	$\mathbf{942}_{\pm \mathbf{19}}$ (**+22.3%**)
Cartpole Swingup	$586_{\pm 52}$	$\mathbf{679}_{\pm \mathbf{35}}$ (**+15.9%**)
Ball in cup Catch	$365_{\pm 210}$	$\mathbf{894}_{\pm \mathbf{78}}$ (**+144.9%**)
Medium tasks-*color hard*	DrQ-v2	+CNSN
Cheetah Run	$144_{\pm 29}$	$\mathbf{345}_{\pm \mathbf{57}}$ (**+139.6%**)
Walker Run	$90_{\pm 21}$	$\mathbf{429}_{\pm \mathbf{16}}$ (**+376.7%**)

A.3 Generalization Performance Comparison against other Normalization Techniques

In addition to CrossNorm and SelfNorm, we also investigate two other types of normalization techniques prevalent in deep learning: batch normalization (BN) and spectral normalization (SpecN). We integrate each of these into the image encoder of DrQ separately to assess their potential to enhance the generalization performance. BN layers are positioned after every convolution layer in the image encoder. When utilizing SpecN, we follow the conclusion proposed by [15] that using too many SpecN layers can decrease the capacity of networks and be detrimental to learning. Therefore, in our setting, SpecN layers are only placed after the second, third, and fourth convolution layers of the image encoder. We train these agents on two DMC-GB tasks and evaluate their generalization performance in three settings.

Table 5: Comparison of generalization performance with different normalization techniques on DMC-GB. The results demonstrate that BN and SpecN do not lead to improvements in generalization performance under distribution shift.

Tasks	Setting	Method			
		DrQ	+BN	+SpecN	+CNSN
Walker Walk	*color hard*	$520_{\pm 91}$	$257_{\pm 89}$	$525_{\pm 64}$	$\mathbf{815}_{\pm \mathbf{65}}$
	video easy	$682_{\pm 89}$	$479_{\pm 109}$	$739_{\pm 19}$	$\mathbf{842}_{\pm \mathbf{58}}$
	video hard	$104_{\pm 22}$	$57_{\pm 12}$	$145_{\pm 20}$	$\mathbf{166}_{\pm \mathbf{28}}$
Cartpole Swingup	*color hard*	$586_{\pm 52}$	$164_{\pm 48}$	$512_{\pm 107}$	$\mathbf{679}_{\pm \mathbf{35}}$
	video easy	$485_{\pm 105}$	$182_{\pm 66}$	$375_{\pm 14}$	$\mathbf{498}_{\pm \mathbf{26}}$
	video hard	$138_{\pm 9}$	$113_{\pm 13}$	$130_{\pm 2}$	$\mathbf{171}_{\pm \mathbf{13}}$

As shown in Table 5, the results show that both BN and SpecN do not improve the generalization performance. Furthermore, BN leads to a significant decrease in generalization capabilities. This can be attributed to the fact that BN assumes the test data distribution is the same as the training data distribution, which can result in performance degradation when facing distribution shift. Previous

literature suggests that SpecN is effective in maintaining a stable learning process for RL, particularly for very deep neural networks. Based on our results, It appears that SpecN does not significantly affect the generalization performance when faced with visual disturbances.

A.4 Parameter Study

The number of active CrossNorm layers is a crucial hyper-parameter in our method. In this section, we discuss how the performance of our approach is affected by varying the number of active CrossNorm layers. Our investigation reveals that activating too many CrossNorm layers when combining them with DrQ during training can lead to divergence in the learning process. For example, when applied to the Cartpole Swingup task, activating all 11 CrossNorm layers resulted in divergence across all 5 random seeds, leading to an average reward of only 158 in the training environments. As a result, to ensure generalizability and consistency in our experiments, we choose to activate only 5 out of the 11 CrossNorm layers for the DrQ+CNSN configuration. On the other hand, the difference between DrQ and DrQ-v2 lies in their encoder architecture and algorithm. We found that activating all 4 CrossNorm layers for DrQ-v2+CNSN did not result in divergence in our experiments. This allows us to fully leverage the benefits of CrossNorm in enhancing the generalization performance of DrQ-v2+CNSN. This observation highlights the importance of activating the appropriate number of CrossNorm layers for each algorithm to ensure stable training. The optimal number of active CrossNorm layers may indeed vary depending on the specific encoder architecture and algorithm employed.

B Training Details

In this section, we provide our detailed settings in experiments. All experiments were conducted using a single GeForce GTX 3090 GPU and Intel Xeon Silver 4210 CPUs. All code assets used for this project came with MIT licenses. **Our code** for the CARLA and DMC-GB experiments can be found in the supplementary material.

B.1 CARLA experiments

While the majority of the hyper-parameters remain the same as in the original implementation of DrQ-v2, several were modified to better adapt to the CARLA environments. The complete hyper-parameter settings are presented in Table 6. All agents are trained under a fixed weather condition for 200,000 environment steps, and their performance is then evaluated in six unseen weather conditions within the same map and task.

B.2 DMC-GB experiments

For the DrQ algorithm, we adopt the exact same hyper-parameters as those used in the implementation of DrQ in the DMC-GB. For DrQ-v2, we adhere to the same hyper-parameters as the original implementation of DrQ-V2 for DMC tasks. The complete hyper-parameter settings are presented in Table 7. For the easy tasks in the DeepMind Control (DMC), we utilize the DrQ algorithm as the base RL algorithm and train the agents for 500,000 environment steps. For the medium tasks, we adopt the DrQ-v2 algorithm as the base RL algorithm for all methods, and the training steps are extended to 1,500,000 steps.

Table 6: A default set of hyper-parameters used in CARLA experiments.

DrQ-v2 Hyper-parameters	
Frame rendering	$84 \times 252 \times 3$
Stacked frames	3
Replay buffer capacity	100,000
Action repeat	4
Exploration steps	100
n-step returns	3
Batch size	512
Discount γ	0.99
Optimizer	Adam
Learning rate	1e-4
Agent update frequency	2
Critic Q-function soft-update rate τ	0.01
Exploration stddev. clip	0.3
Exploration stddev. schedule	$\mathrm{linear}(1.0, 0.1, 100000)$
CrossNorm Hyper-parameters	
active CrossNorm	4 out of 4
SVEA Hyper-parameters	
SVEA coefficients	$\alpha = 0.5, \beta = 0.5$

Table 7: A default set of hyper-parameters used in DMC-GB experiments.

DrQ Hyper-parameters	
Action repeat	2
Discount γ	0.99
Replay buffer size	500,000
Optimizer	Adam
Learning rate (θ)	1e-3
Learning rate (α of SAC)	1e-4
Batch size	128
DrQ-v2 Hyper-parameters	
Replay buffer size	1,000,000
Action repeat	2
Exploration steps	2000
n-step returns	3
Batch size	256
Discount γ	0.99
Optimizer	Adam
Learning rate	1e-4
Agent update frequency	2
Critic Q-function soft-update rate τ	0.01
Exploration stddev. clip	0.3
Exploration stddev. schedule	$\mathrm{linear}(1.0, 0.1, 500000)$
CrossNorm Hyper-parameters	
active CrossNorm (w/ DrQ)	5 out of 11
active CrossNorm (w/ DrQ-v2)	4 out of 4
SVEA Hyper-parameters	
SVEA coefficients	$\alpha = 0.5, \beta = 0.5$