Controlling Neural Style Transfer with Deep Reinforcement Learning

Chengming Feng[1] , **Jing Hu**[1] , **Xin Wang**[2*] , **Shu Hu**[3] , **Bin Zhu**[4] , **Xi Wu**[1*] ,
Hongtu Zhu[5] and **Siwei Lyu**[2]
[1]Chengdu University of Information Technology, China
[2]University at Buffalo, SUNY, USA
[3]Carnegie Mellon University, USA
[4]Microsoft Research Asia, China
[5]University of North Carolina at Chapel Hill, USA
xwang264@buffalo.edu, xi.wu@cuit.edu.cn

Abstract

Controlling the degree of stylization in the Neural Style Transfer (NST) is a little tricky since it usually needs hand-engineering on hyper-parameters. In this paper, we propose the first deep Reinforcement Learning (RL) based architecture that splits one-step style transfer into a step-wise process for the NST task. Our RL-based method tends to preserve more details and structures of the content image in early steps, and synthesize more style patterns in later steps. It is a user-easily-controlled style-transfer method. Additionally, as our RL-based model performs the stylization progressively, it is lightweight and has lower computational complexity than existing one-step Deep Learning (DL) based models. Experimental results demonstrate the effectiveness and robustness of our method.

1 Introduction

Neural style transfer (NST) refers to generation of a pastiche image combining the semantic content of one image (the *content image*) and the visual style of another image (the *style image*) using a deep neural network. NST can be used to create stylized non-photorealistic rendering of digital images with enriched expressiveness and artistic flavors.

Existing NST methods usually generate a stylized image with a one-step approach: a neural network is trained to minimize a loss function of the visual similarity between the content image and the stylized image and the style similarity between the style image and the stylized image [Cheng *et al.*, 2021], and the trained Deep Learning (DL) model is run once to create a stylized image. This one-step approach has an apparent limitation: it is hard to determine a proper level of stylization to fit various flavors of different users since the ultimate metric of style transfer is very subjective. It is observed that generated stylized images by current NST methods tend to be under- or over-stylization [Cheng *et al.*, 2021]. A remedy to under-stylization is to apply the DL model iteratively until a desired level of stylization is reached. However, this solution may suffer from high computational cost due to intrinsic complexity of one-step DL models. Other existing methods, like [Gatys *et al.*, 2015b] and [Huang and Belongie, 2017], play a tradeoff between content and style by adjusting hyper-parameters. These methods are inefficient since there is no guarantee that a user can get the expected output via one-time adjusting.

To address the aforementioned limitations of existing DL-based NST methods, we propose a novel framework in this paper, called *RL-NST*, based on a reinforcement-learning (RL) framework to progressively perform style translation as shown in Fig. 1. Given a content image, we consider the stylized content to be added progressively by the stylizer. More specifically, a stochastic actor in RL-NST first estimates a 2D Gaussian distribution to sample hidden actions, then uses the action to control the stylizer to generate an intermediate stylized image, which is in turn passed to the actor as the input for the next step. Our model also includes a critic to evaluate the latent action. The whole structure is shown in Fig. 2. Furthermore, by using a CNN+RNN architecture [Mirowski *et al.*, 2017] for the actor and stylizer for both frame-wise and step-wise smoothing, our model can perform video NST tasks. To the best of our knowledge, this is the first work that successfully leverages RL for the NST scenario.

Our major contributions can be summarized as follows: **1)** We propose the first reinforcement-learning-based NST method, RL-NST, that facilities step-wise style transfer. It provides more flexible control of the degree of stylization without any hyper-parameter adjustment during generation of stylized images. **2)** Our RL-NST stylizes a content image progressively, with increased stylization along with more iterations (see Fig. 1). It leads to a lightweight NST model compared with existing one-step DL-based methods, making it computationally more efficient. **3)** From our extensive experiments, our RL-NST demonstrates better effectiveness and robustness than existing state-of-the-art methods on both image and video NST tasks.

2 Related Work

Neural Style Transfer. Since the seminal work of Gatys et al. [Gatys *et al.*, 2015b] that uses a neural network to produce striking stylized art images, many methods have been

[*]Corresponding authors.

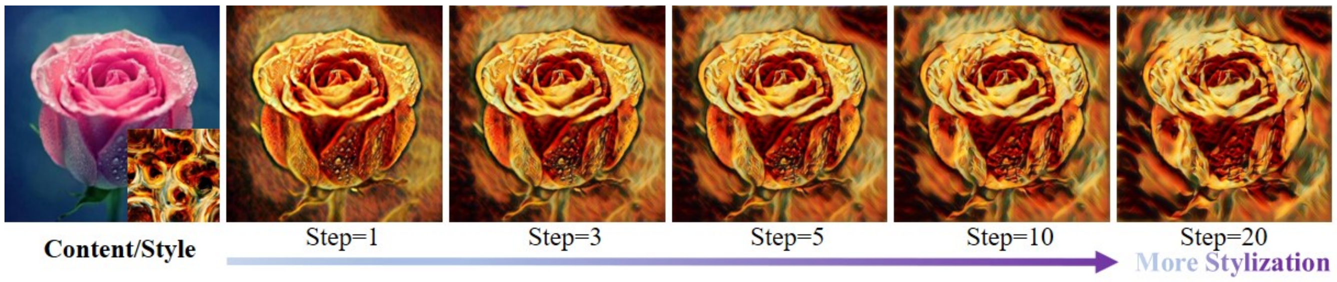

Figure 1: Illustration of our step-wise style transfer process. Content images are stylized smoothly stronger along with prediction steps. Our step-wise method can easily control the degree of stylization: the model tends to preserve more details and structures of the content image in early steps, and synthesize more style patterns in later steps. It is a user-easily-controlled style transfer method.

proposed to improve the quality and/or running efficiency of NST algorithms. Existing NST methods can be divided roughly into two groups: image-optimization-based [Gatys *et al.*, 2015b; Li *et al.*, 2017a; Risser *et al.*, 2017] and model-optimization-based [Johnson *et al.*, 2016; Ulyanov *et al.*, 2016a; Chen *et al.*, 2017; Huang and Belongie, 2017; Sheng *et al.*, 2018; Park and Lee, 2019; An *et al.*, 2021].

Reinforcement Learning. Reinforcement learning (RL) concerns how an agent takes actions in an environment to maximize its cumulative reward. Standard RL works well on tasks with finite and discrete action spaces. For real-world tasks with high dimensional continuous actions, such as robotic control [Tassa *et al.*, 2018; Gu *et al.*, 2017; Xiang *et al.*, 2022], maximum entropy RL (MERL) and its variants, including Soft Q-learning [Haarnoja *et al.*, 2017; Zhao *et al.*, 2019] and SAC [Haarnoja *et al.*, 2018; Hu *et al.*, 2023], have been proven to have a stable and powerful performance. But they still have limitations to handle high dimensional continuous state and action spaces in image-to-image (I2I) transformation and synthesis. To address this problem, SAEC [Luo *et al.*, 2021] extends the traditional MERL framework [Haarnoja *et al.*, 2018] with an additional executor. It shows promising performance on several I2I tasks. However, its actions are 1D vectors, which do not preserve 2D spatial information of images.

3 Our RL-NST Framework

We formulate NST as a decision-making problem. Instead of directly converting a content image to a stylized image in a single step, we propose to use a lightweight model to perform the translation progressively, with more stylization added to the stylized image as the translation progresses.

Let $(\mathbf{c}, \mathbf{e})$ be a pair of content and style images from image domain $\mathcal{X} \in \mathbb{R}^d$, where d is the dimension. Our model consists of three components, as shown in Fig. 2: an actor π_ϕ parameterized by ϕ, a stylizer η_ψ with model parameters ψ, and a critic Q_θ parameterized by weights θ. The actor generates a latent action according to a stochastic policy, so as to capture the style transfer control. The critic evaluates the generated action. The stylizer leverages the latent action and image details to perform style transfer on the content image and generate a stylized image, referred to as a *moving image*, which updates the environment and is used subsequently as the content image in the next iteration. We use the pre-trained VGG [Simonyan and Zisserman, 2014] as our feature extraction network because it is widely used such as in [Chen *et al.*, 2021; Lin *et al.*, 2021]. Actor π_ϕ and stylizer η_ψ are supervised jointly with content loss, style loss, total variation regularization, and compound temporal regularization (for video only) on the current content image (i.e., the moving image). Furthermore, the actor π_ϕ and critic Q_θ form an actor-critic model.

3.1 RL-NST Settings

In forming our model, we define the infinite-horizon Markov decision process (MDP) as tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is the action space, and $\mathcal{P} : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \rightarrow [0, \infty)$ represents the state transition probability of the next state $\mathbf{s}_{t+1}$ given $\mathbf{s}_t \in \mathcal{S}$ at time t and action $\mathbf{a} \in \mathcal{A}$, $r : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward emitted from each transition, and $\gamma \in [0, 1]$ is the reward discount factor. Specifically, (1) **State**. State $\mathbf{s}_t$ is the moving image, initialized by the content image. (2) **Action**. To extract high-level abstraction $\mathbf{a}_t$ of $\mathbf{s}_t$ from the actor, a stochastic latent action is modeled as $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t|\mathbf{s}_t)$. In practice, it can be obtained by using a reparameterization trick [Kingma and Welling, 2013] $\mathbf{a}_t = f_\phi(\epsilon_t, \mathbf{s}_t)$, where ϵ_t is an input noise vector sampled from a 2D Gaussian distribution. The moving image at time t, i.e., state image $\mathbf{s}_{t+1}$, is created by the stylizer based on $\mathbf{a}_t$ and current state image $\mathbf{s}_t$. (3) **Reward**. It is from the environment $\mathcal{E}$, obtained by measuring the difference between current state $\mathbf{s}_t$ and the style image. The higher the difference is, the smaller the reward is.

3.2 Network Architecture

The **Actor** is a neural network model that consists of three convolutional layers and a residual layer. After each convolutional layer, there is an instance norm layer and a ReLU layer. In the residual layer, we use the residual block designed by He et al. [He *et al.*, 2016]. The actor estimates a 2D Gaussian distribution for sampling our latent actions, which is forwarded to the stylizer to generate the moving image. The dimension of our latent action is 64×64, which is a 2D sample and able to preserve more spatial structure information of images. Notably, the actor in our method is to learn the latent actions, which are more compact than the image presentation. The learned actions are controlled by our framework to guide the

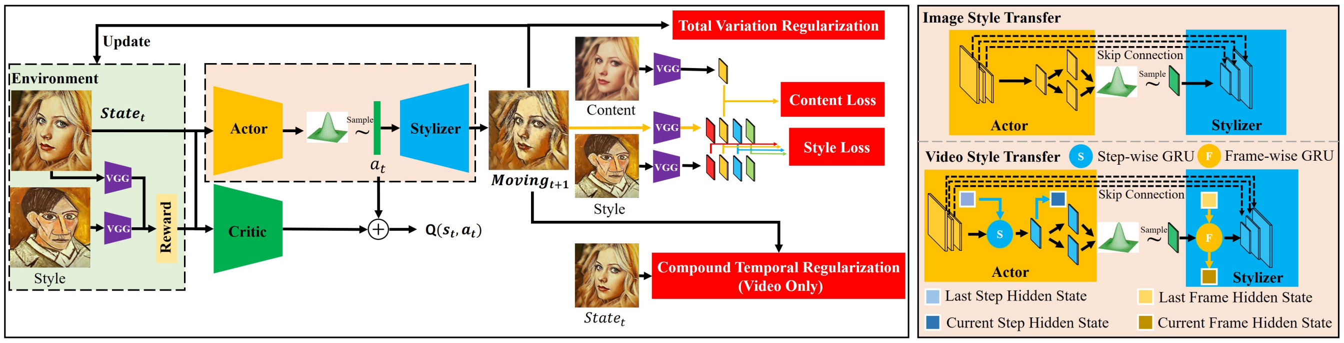

Figure 2: Our RL-NST framework. **Left:** The state is initialized with the content image (or video frame). After the first iteration, we use only the moving image as the state. Latent-action $\mathbf{a}_t$ is sampled from a 2D Gaussian distribution and is concatenated with the critic's output. It is estimated by the policy π_ϕ: $\mathbf{a_t} \sim \pi_\phi(\mathbf{a}_t|\mathbf{s}_t)$. The predicted moving image is generated by stylizer η_ψ. Note that the VGG networks are pre-trained and fixed for the feature extraction during the training process. **Right:** The structure of the actor and stylizer for image and video NST, respectively. More details of the network structure can be found in Appendix.

stylizer to generate stylized images instead of the reconstruction.

The **Stylizer** has three up-sampling layers correspondingly. More importantly, by using 2D Gaussian sampling, our actor-stylizer structure is a fully convolutional network (FCN), which can process images of any input size, instead of only accepting test images of the same size as training images. We also use three skip connections between the actor and the stylizer to stabilize the training process.

To handle video NST tasks, we expand the actor-stylizer to include RNN layers by following the work [Mirowski *et al.*, 2017]. In particular, the ConvGRU layer [Shi *et al.*, 2015] is used for the FCN structure (see the right bottom of Fig. 2). We add Step-wise GRU [Mirowski *et al.*, 2017] to actor and Frame-wise GRU [Donahue *et al.*, 2017] to stylizer. Specifically, for each frame, the hidden state of the Step-wise GRU at each step comes from the output of the Step-wise GRU at the previous step. The role of Step-wise GRU is to make the model maintain better content consistency. Furthermore, the hidden state of the Frame-wise GRU at each step is derived from the output of the Frame-wise GRU at the same step in the previous frame. Frame-wise GRU can make the model maintain better inter-frame consistency.

The **Critic** consists of seven convolutional layers and one fully-connected layer at the end. Since using standard zero-padded convolutions in style transfer leads to serious artifacts on the boundary of a generated image [Ulyanov *et al.*, 2016b], we use reflection padding instead of zero padding for all the networks.

3.3 Model Training

Our RL-NST contains two learning procedures, namely style learning and step-wise learning.

Style Learning

To make the moving image not deviate from the content image, actor π_ϕ (encoder) and stylizer η_ψ (decoder) are trained together to preserve the perceptual and semantic similarity with the content image. More specifically, the actor and the stylizer form a conditional generative process that translates state $\mathbf{s}_t$ to output moving image $\mathbf{m}_t$ via the mapping $\mathbf{m}_t = \eta_\psi(\pi_\phi(\mathbf{s}_t))$ at time t. Note that $\mathbf{s}_t$ is initialized to content image $\mathbf{c}$ and $\mathbf{s}_{t+1}$ is equivalently $\mathbf{m}_t$. Inspired by [Johnson *et al.*, 2016], we apply the content loss $\mathcal{L}^{CO}$, style loss $\mathcal{L}^{ST}$, and total variation regularization $\mathcal{L}^{TV}$ to optimize the model parameters of π_ϕ and η_ψ in the image setting. These losses can better measure perceptual and semantic differences between the moving image and content image $\mathbf{c}$. For the video setting, we add an additional loss named compound temporal regularization $\mathcal{L}^{CT}$, which can force the model to generate temporal consistent results under the compound transformation. More details about these losses are as follows.

Content Loss $\mathcal{L}^{CO}$. Following [Johnson *et al.*, 2016], we use a pre-trained neural network F to extract the high-level feature representatives of $\mathbf{m}_t$ and $\mathbf{c}$. The reason for using this F is to encourage moving image $\mathbf{m}_t$ to be perceptually similar to content image $\mathbf{c}$ but does not force them to match exactly. Denote $F^j(\cdot)$ as the activations of the j-th layer of F. Suppose j-th layer is a convolutional layer, then the output of $F^j(\cdot)$ will be a feature map with size $C^j \times H^j \times W^j$, where C^j, H^j, and W^j represent the number of channels, height, and width in the feature map of layer j, respectively. We apply the Euclidean distance, which is squared and normalized to design the content loss as follows,

$$\mathcal{L}^{CO}(\mathbf{m}_t, \mathbf{c}) = \frac{1}{C^j H^j W^j} \|F^j(\mathbf{m}_t) - F^j(\mathbf{c})\|_2^2.$$

Style Loss $\mathcal{L}^{ST}$. To penalize $\mathbf{m}_t$ when it deviates in content from $\mathbf{c}$ and in style from $\mathbf{e}$, following [Gatys *et al.*, 2015a], we define a Gram matrix $G^j(\mathbf{x}) = \frac{\tilde{F}^j(\mathbf{x})(\tilde{F}^j(\mathbf{x}))^\top}{C^j H^j W^j} \in \mathbb{R}^{C^j \times C^j}$, where $\tilde{F}^j(\cdot)$ is obtained by reshaping $F^j(\cdot)$ into the shape $C^j \times H^j W^j$. The style loss can be defined as a squared Frobenius norm of the difference between the Gram matrices of $\mathbf{m}_t$ and $\mathbf{e}$. To preserve the spatial structure of images, we use a set of layers, J, instead of a single layer j. Thus, we define the style loss to be the sum of losses for each layer $j \in J$ ($J = 4$ in our experiments):

$$\mathcal{L}^{ST}(\mathbf{m}_t, \mathbf{e}) = \sum_{j=1}^{J} \|G^j(\mathbf{m}_t) - G^j(\mathbf{e})\|_F^2.$$

Total Variation Regularization $\mathcal{L}^{TV}$. To ensure spatial smoothness in moving image $\mathbf{m}$, we use a total variation regularizer $\mathcal{L}^{TV}(\mathbf{m}_t)$, which has been widely used in existing works [Mahendran and Vedaldi, 2015; Johnson *et al.*, 2016].

Compound Temporal Regularization $\mathcal{L}^{CT}$. Inspired by [Wang *et al.*, 2020], we add a compound temporal regularization for video style transfer. Specifically, we first generate motions $M(\cdot)$ and then synthesize adjacent frames. With this approach, we do not need to estimate optical flow in the training process and we can guarantee the optical flows are absolutely accurate. Given noise $\triangle$, to maintain temporal consistency, we can minimize the following loss

$$\mathcal{L}^{CT} = \|\eta_\psi(\pi_\phi(M(\mathbf{s}_t) + \triangle)) - M(\mathbf{m}_t)\|_1.$$

Summing up all the components, the final style learning loss is

$$\mathcal{L} = \underbrace{\overbrace{\mathcal{L}^{CO} + \lambda\mathcal{L}^{ST} + \beta\mathcal{L}^{TV}}^{\text{for image}} + \zeta\mathcal{L}^{CT}}_{\text{for video}}, \tag{1}$$

where λ, β, and ζ are hyper-parameters to control the sensitivity of each term. For image style transfer, we use the first three terms. For video style transfer, we use all terms. Then we can update ϕ and ψ from the actor and stylizer by using the gradient descent method with a predefined learning rate η with the following steps:

$$\phi \leftarrow \phi - \eta\nabla_\phi\mathcal{L}, \quad \psi \leftarrow \psi - \eta\nabla_\psi\mathcal{L}. \tag{2}$$

Step-wise Learning

Our step-wise learning is based on the MERL framework [Haarnoja *et al.*, 2018], where rewards and soft Q values are used to iteratively guide the stochastic policy improvement. Moreover, we focus on latent action $\mathbf{a}$, and use it to estimate a soft state-action value to encourage high-level policies. Specifically, we concatenate $\mathbf{a}_t$ to the downsampled vector of the critic and output soft Q function $Q_\theta(\mathbf{s}_t, \mathbf{a}_t)$, which is an estimation of the state value at time t. Because the critic is used to evaluate the actor, rewards r_t and the soft Q values are used to iteratively guide the stochastic policy improvement by minimizing the soft Bellman residual:

$$J_Q(\theta) = \mathbb{E}_{(\mathbf{s}_t,\mathbf{a}_t)\sim\mathcal{D}}\Big[\frac{1}{2}\Big(Q_\theta(\mathbf{s}_t,\mathbf{a}_t) - \big(r_t + \gamma\mathbb{E}_{\mathbf{s}_{t+1}}\left[V_{\bar\theta}(\mathbf{s}_{t+1})\right]\big)\Big)^2\Big],$$

where $\mathcal{D}$ is a replay pool and $V_{\bar\theta}(\mathbf{s}_t) = \mathbb{E}_{\mathbf{a}_t\sim\pi_\phi}[Q_{\bar\theta}(\mathbf{s}_t,\mathbf{a}_t) - \alpha\log\pi_\phi(\mathbf{a}_t|\mathbf{s}_t)]$. Note that we utilize the negative value of $\mathcal{L}^{ST}(\mathbf{s}_t, \mathbf{e})$ for r_t in practice.

The critic network $Q_{\bar\theta}$ is used to stabilize the training, whose parameters $\bar\theta$ are obtained by an exponential moving average of parameters of the critic network [Lillicrap *et al.*, 2015]: $\bar\theta \rightarrow \tau\theta + (1-\tau)\bar\theta$, with hyperparameter $\tau \in [0, 1]$. To optimize $J_Q(\theta)$, we use the gradient descent with respect to parameters θ as follows,

$$\theta \leftarrow \theta - \eta_Q\nabla_\theta Q_\theta(\mathbf{s}_t,\mathbf{a}_t)\Big(Q_\theta(\mathbf{s}_t,\mathbf{a}_t) - r_t - \gamma\left[Q_{\bar\theta}(\mathbf{s}_{t+1},\mathbf{a}_{t+1}) - \alpha\log\pi_\phi(\mathbf{a}_{t+1}|\mathbf{s}_{t+1})\right]\Big),$$

Algorithm 1: RL-NST

```
Input: $\mathbf{c}$, $\mathbf{e}$, and replay pool $\mathcal{D}$
Init: $\phi$, $\psi$, $\theta$, $\bar\theta$, $\mathcal{D} \leftarrow \emptyset$, $\eta$, $\eta_Q$, $\eta_\phi$, and environment $\mathcal{E}$
for each iteration do
    for each environment step do
        $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t|\mathbf{s}_t)$
        $\mathbf{s}_{t+1}, r_t \sim \mathcal{P}(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$
        $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r_t, s_{t+1})\}$
    end
    for each gradient step do
        Sample from $\mathcal{D}$
        Update $\theta$, $\phi$, $\psi$ by using Eq.(2), (3.3), and (3)
    end
end
```

where η_Q is a learning rate. Since the critic works on the actor, it will affect the actor's decisions. Therefore, the following objective can be applied to minimize the KL divergence between the policy and a Boltzmann distribution induced by the Q-function,

$$\begin{aligned}J_\pi(\phi) =& \mathbb{E}_{\mathbf{s}_t\sim\mathcal{D}}\big[\mathbb{E}_{\mathbf{a}_t\sim\pi_\phi}\left[\alpha\log(\pi_\phi(\mathbf{a}_t|\mathbf{s}_t)) - Q_\theta(\mathbf{s}_t,\mathbf{a}_t)\right]\big]\\ =& \mathbb{E}_{\mathbf{s}_t\sim\mathcal{D},\epsilon_t\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\big[\alpha\log(\pi_\phi(f_\phi(\epsilon_t,\mathbf{s}_t)|\mathbf{s}_t))\\ &-Q_\theta(\mathbf{s}_t, f_\psi(\epsilon_t,\mathbf{s}_t))\big].\end{aligned}$$

The last equation holds because $\mathbf{a}_t$ can be evaluated by $f_\phi(\epsilon_t, \mathbf{s}_t)$, as we discussed before.Note that hyperparameter α can be automatically adjusted by using the method proposed in [Haarnoja *et al.*, 2018]. Similarly, we apply the gradient descent method with a learning rate η_ϕ to optimize parameters as follows,

$$\begin{aligned}\phi \leftarrow& \phi - \eta_\phi\Big(\nabla_\phi\alpha\log(\pi_\phi(\mathbf{a}_t|\mathbf{s}_t)) + \big(\nabla_{\mathbf{a}_t}\alpha\log(\pi_\phi(\mathbf{a}_t|\mathbf{s}_t))\\ &-\nabla_{\mathbf{a}_t}Q_\theta(\mathbf{s}_t,\mathbf{a}_t)\big)\nabla_\phi f_\phi(\epsilon_t,\mathbf{s}_t)\Big).\end{aligned} \tag{3}$$

The pseudo-code of optimizing RL-NST is described in Algorithm 1. All parameters are optimized based on the samples from replay pool $\mathcal{D}$.

4 Experiments

We have conducted a series of experiments to evaluate the effectiveness of RL-NST in realizing step-wise style transfer on both image and video NST tasks. Our **code**, a user study, and additional results with more detailed information can be found in the supplementary materials.

4.1 Experimental Settings

Datasets. (1) For image style transfer, we select style images from WikiArt [Phillips and Mackintosh, 2011] and use MS-COCO [Lin *et al.*, 2014] as content images in which the training set includes 80K images and the test set includes 40K images. All training images are resized to 256×256. In the inference stage, our method is applicable for content images and style images of any size. (2) For video style transfer, we randomly collect 16 videos of different scenes

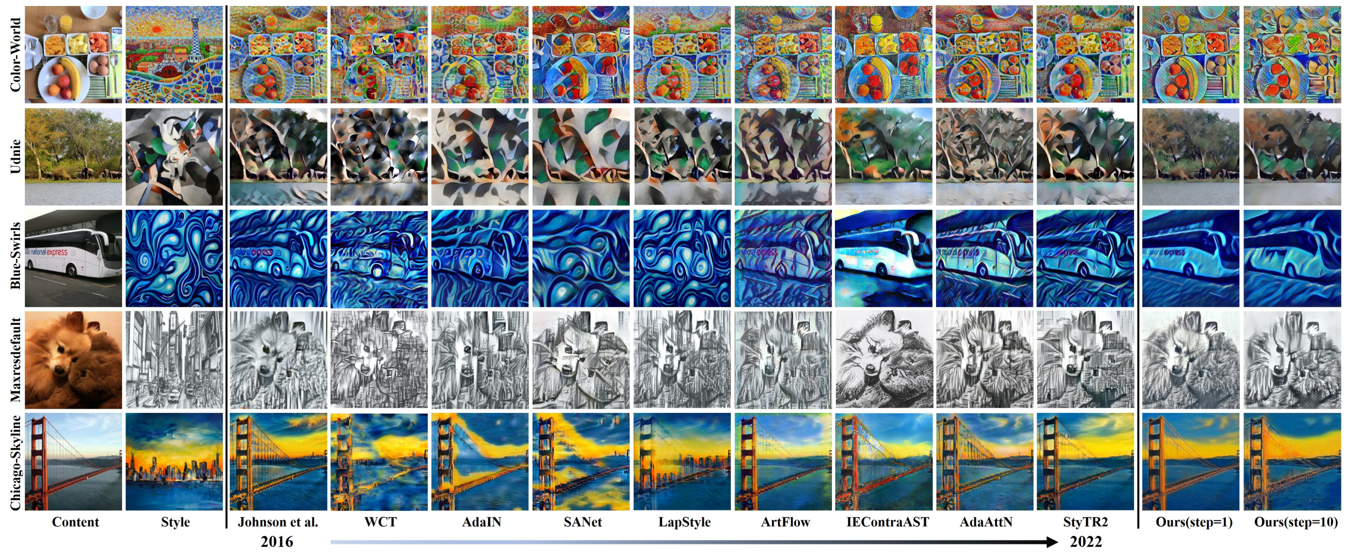

Figure 3: Qualitative comparison. The first two columns show the content and style images, respectively. The rest of the columns show the stylization results generated with different style transfer methods, with the last two columns of our step-wise results at step 1 and 10, respectively.

Methods	Johnson et al.	AdaIN	WCT	SANet	LapStyle	ArtFlow	IEContraAST	AdaAttN	StyTR2	Ours(step=1)	Ours(step=10)
Content loss	1.597	2.222	2.322	1.941	2.292	1.538	1.668	1.447	1.510	**0.868**	**1.387**
Style loss	1.985e-05	1.269e-05	1.626e-05	7.062e-06	2.117e-05	1.486e-05	8.863e-06	1.033e-05	9.178e-06	**3.353e-06**	**1.594e-06**
Time (s)	0.014 (3.5×)	0.140 (35×)	0.690 (172.5×)	0.010 (2.5×)	0.047 (11.75×)	0.127 (31.75×)	0.019 (4.75×)	0.025 (6.25×)	0.058 (14.5×)	**0.004**	0.089
#Params (M)	1.68 (9.33×)	7.01 (38.94×)	34.24 (190.22×)	20.91 (116.17×)	7.79 (43.28×)	6.46 (35.89×)	21.12 (117.33×)	13.63 (75.72×)	35.39 (196.61×)	**0.18**	**0.18**

Table 1: Quantitative comparison of our RL-NST with the baseline methods on the MS-COCO dataset. The speed is obtained with a Pascal Tesla P100 GPU. $(\cdot\times)$ represents the ratio between current baseline and our method (step=1) under the same metric. The best results are shown in bold.

from pexels[pex, 2022]. Then these videos are extracted into video frames and we obtain more than 2.5K frames. We regard these frames as the content images of training set. Note that the style images in the training set are also selected from WikiArt [Phillips and Mackintosh, 2011]. In addition, following [Wang *et al.*, 2020], we use the training set of MPI Sintel dataset [Butler *et al.*, 2012] as the test set, which contains 23 sequences with a total of 1K frames. Similarly, all training frames are resized to 256×256, we use the original frame size in testing.

Baselines and Evaluation Metrics. (1) For image style transfer, we choose the following eight classical and latest state-of-the-art style transfer methods as our baselines: Johnson et al. [Johnson *et al.*, 2016], WCT [Li *et al.*, 2017b], AdaIN [Huang and Belongie, 2017], SANet [Park and Lee, 2019], LapStyle [Lin *et al.*, 2021], ArtFlow [An *et al.*, 2021], IEContraAST [Chen *et al.*, 2021], AdaAttN [Liu *et al.*, 2021], and StyTR2 [Deng *et al.*, 2021b]. Following StyTR2 [Deng *et al.*, 2021b], we evaluate all the algorithms in terms of stylization effect, computing time, content loss, and style loss. (2) For video style transfer, we compare our method with the following four popular methods: Linear [Li *et al.*, 2019], MCCNet [Deng *et al.*, 2021a], ReReVST [Wang *et al.*, 2020], and AdaAttN [Liu *et al.*, 2021]. Following [Liu *et al.*, 2021], we use temporal loss as the evaluation metric to compare the stability of stylized results. All these methods are performed using their public codes with the default settings.

Implementation Details. In the experiment, we set $\lambda = 1e5$, $\beta = 1e-7$, $\zeta = 1e2$ in Eq. (1), and $\eta = 1e-4$ in Eq. (2). These settings yield nearly the best performance in our experiments. Following [Wang *et al.*, 2020], in $\mathcal{L}^{CT}$, $M(\cdot)$ is implemented by warping with a random optical flow. Specifically, for a frame of size $H \times W$, we first generate a Gaussian map (wavy twists) M_{wt} of shape $H/100 \times W/100 \times 2$, mean 0, and standard deviation 0.001. Second, M_{wt} is resized to $H \times W$ and blurred by a Gaussian filter of kernel size 100. Finally, we add two random values (translation motion) M_{tm} of range [-10,10] to M_{wt}, and obtain M. In addition, random noise $\triangle \sim \mathcal{N}(0, \sigma^2 I)$, where $\sigma \sim \mathcal{U}(0.001, 0.002)$.

4.2 Evaluations on Image RL-NST

Qualitative Comparison. Fig. 3 shows some stylized results of our RL-NST and the baseline methods. For content images with fine structures such as the forest image (Udnie style), all the baseline methods proposed to address content leakage, including ArtFlow, produce messy stylized images with a complete loss of content structure. Moreover, SANet has repeated texture patterns for all the cases, and most of its

Style	Step	Method	Content Loss	Style Loss
Maxresdefault	1	Actor-Stylizer (AS)	0.787	5.686e-06
		Ours	**0.557**	**1.883e-06**
	10	Ours w/o RL	2.093	3.433e-05
		Ours	**0.945**	**1.093e-06**
Blue Swirls	1	Actor-Stylizer (AS)	2.265	3.275e-05
		Ours	**1.016**	**5.280e-06**
	10	Ours w/o RL	3.374	7.747e-05
		Ours	**1.654**	**2.178e-06**

Table 2: Content loss, and style loss of several variants of our proposed method.

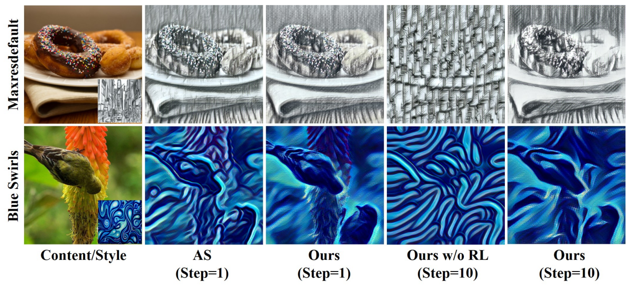

Figure 4: Comparison of Ours with the Actor-Stylizer (AS) model at step 1 and Ours without the RL model at step 10.

results are hard to generate sharp edges.

In contrast, our method can produce stable and diversified stylized results with good content structures. This may be attributed to our step-wise solution. More specifically, the content image is stylized progressively and hence smoothed stylization results are obtained. More importantly, as we mentioned before, despite that stylization is quite subjective, our step-wise method provides flexible control of different degrees of stylization to fit the need of different users.

Quantitative Results. To be consistent with all compared methods shown in Fig. 3, we compare our method with all baselines without caring which type (single or multiple styles) they are. The quantitative results are shown in Table 1. Our RL-NST (step=1) achieves better performance than the baseline methods in all evaluation metrics. Our method still has low content and style losses even if the step is equivalent to 10, which means our method is friendly to the user for choosing the results from specific steps accordingly. In addition, it is clear that our model has much fewer parameters and a faster speed. For example, the time cost and the parameter size of our method are $2/7$ and $1/9$ of Johnson et al., and $4/47$ and $1/43$ of LapStyle, respectively.

Ablation Study. (1) We study the effect of the RL model in our framework. As shown in Fig. 4, compared with the method that uses only Actor-Stylizer (AS), our method can generate more stable and clear stylized images at step 1. At step 10, AS loses the content information completely without the help of RL (Ours w/o RL), while our method can still produce amazing results. We also show the corresponding numerical comparison in Table 2. We can easily see that our method achieves the best performance consistently in both steps 1 and 10. This study indicates that RL can indeed improve the performance of DL-based NST models. (2) Since

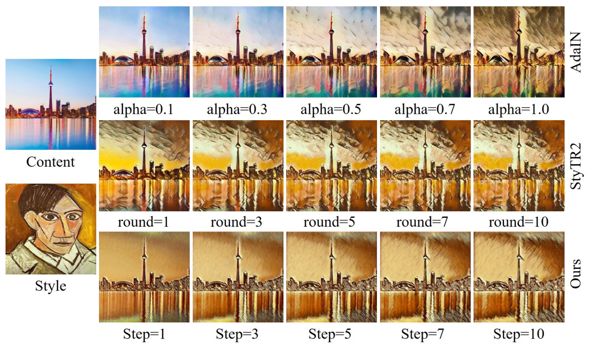

Figure 5: Comparison of Ours with AdaIN and StyTR2 in various hyperparameter settings.

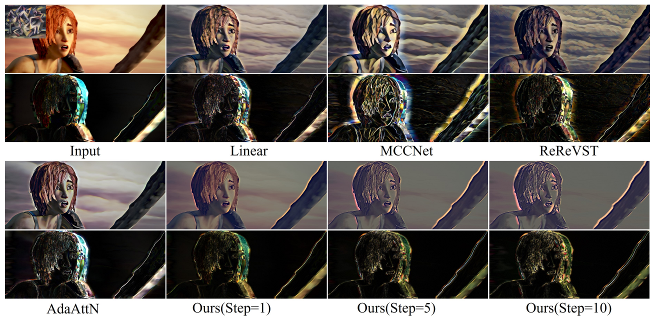

Figure 6: Comparison of video style transfer between our method and the compared methods. For each method, the top portion shows the video frame stylized results. The bottom portion shows the heatmap of the differences between two adjacent video frames.

AdaIN and StyTR2 methods can adjust the hyperparameter 'alpha'$\in [0, 1]$ and the round of repetitive stylization to control the degree of stylization in the final results, respectively, we compare our method with them accordingly in Fig. 5. From the visualization results, we can see that the results of AdaIN are in the under-stylized state even if the style control hyperparameter is changed. Moreover, StyTR2 gets the results with small style changes and low quality after multiple rounds. However, our method not only ensures the gradual change in style, but also produces very smooth results.

4.3 Evaluations on Video RL-NST

Qualitative Comparison. We show the visualization results of our method compared with the four latest video style transfer methods in Fig. 6, wherein, for each method, the top portion shows the specific stylized results and the bottom portion is the heatmap of the differences in the adjacent frames of the input and stylized videos. Note that the adjacent frame indexes are the same for all methods. We can find that our method produces refined stylized results and our results are closest to the input frames. In particular, our method can highly promote the stability of video style transfer. The differences in our results are closest to the difference from input frames without reducing the effect of stylization. It is clear that MCCNet and ReReVST fail to keep the coherence of videos. In addition, Linear and AdaAttN also fail to keep the

Methods \ Styles	La_muse	Sketch	En_campo_gris	Brushstrokes	Picasso	Trial	Asheville	Contrast	Average
LinearStyleTransfer	2.602	1.792	1.795	2.321	2.947	1.451	5.043	4.524	2.809
ReReVST	1.450	8.155	7.050	7.026	10.772	7.888	19.493	12.886	9.340
MCCNet	4.493	2.050	2.759	2.591	2.854	2.486	6.750	4.820	3.600
AdaAttN	3.442	1.976	2.660	2.561	2.941	1.698	5.775	3.587	3.080
Ours(Step=1)	**0.885**	**1.196**	**0.453**	**0.883**	**1.447**	**0.527**	**1.735**	**1.045**	**1.021**
Ours(Step=5)	**1.436**	**1.509**	**0.855**	**1.499**	**1.980**	**0.704**	**2.327**	**1.550**	**1.483**
Ours(Step=10)	1.867	**1.695**	**1.141**	**1.807**	**2.394**	**0.852**	**2.854**	**1.842**	**1.807**

Table 3: Comparison of the average temporal losses ($\times 10^{-2}$) from 23 different sequences of our method with other baseline methods on different styles. The last column shows the average scores among all styles in each method.

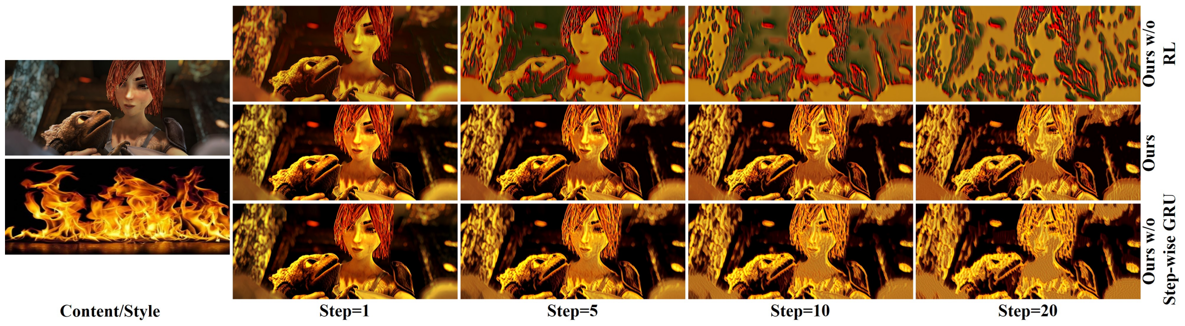

Figure 7: Comparison of our method, our method without using RL (AS method), and our method without using Step-wise GRU. RL makes the results from our model more stable and Step-wise GRU makes the output has higher quality.

Methods \ Styles		La_muse	Brushstrokes
Step=1	Ours w/o FWG	1.8939	1.0933
	Ours	**1.1351**	**0.8679**
Step=5	Ours w/o FWG	2.3991	1.4731
	Ours	**1.8883**	**1.4331**
Step=10	Ours w/o FWG	3.1329	1.8000
	Ours	**2.3836**	**1.7053**

Table 4: Comparison of our method with and without using Frame-wise GRU (FWG). The average of temporal losses ($\times 10^{-2}$) from eight sequences are reported on two styles.

coherence in some regions that are close to the edge of objects such as the head and shoulder.

Quantitative Results. As shown in Table 3, we choose 23 different sequences from the MPI Sintel dataset [Butler *et al.*, 2012] and eight different style images to calculate the average of temporal losses for comparison. It is clear that our method (step=1 and 5) outperforms the compared methods in all style settings. Our method still has a low temporal error even if step=10.

Ablation Study. We investigate the effect of the individual parts of the network structure group on the results, including the RL, Step-wise GRU, and Frame-wise GRU.

(1) As shown in Fig. 7 (first and second rows), our method generates more stable and clearer stylized results than the method using only Actor-Stylizer without RL. After step 5, AS no longer has the ability to keep the content information and style information, while our method with RL can still produce good results. (2) Similarly, we have compared our method with the results produced when Step-wise GRU is removed, and the results are shown in Fig. 7 (second and third rows). We can clearly see that most of the face details of the protagonist and dragon have been lost at step 10 when using our method without the Step-wise GRU. Also, the external details of the protagonist and dragon are completely lost in step 20. Our method with using Step-wise GRU, on the other hand, obtains very fine results even at step 20. (3) Table 4 shows the comparison of the temporal loss of our method with Frame-wise GRU (FWG) and without FWG. We find that the temporal loss is very low if we use FWG, which means the obtained final results are more consistent from frame to frame. The above experiments have shown that RL, Step-wise , and Frame-wise GRU all greatly improve the performance of the model.

5 Conclusion

In this paper, we propose a new RL-based framework called RL-NST for both image and video NST tasks. It achieves style transfer step by step for a flexible control of the level of stylization. Despite using a lightweight neural network model, RL-NST can handle an extremely high-dimensional latent action space (up to 64×64) and is thus capable of generating visually more satisfying artistic images than existing NST methods. Experimental results have demonstrated the effectiveness of the proposed method. The main goal of this work is to show the effectiveness of stylization-level controlling with our RL-based method and the superiority of our method in achieving the best NST quality. Therefore we use a single-style NST model. In the future, we would like to extend our model to multiple-style transfer tasks.

Acknowledgements

This work was supported in part by Sichuan province Key Technology Research and Development project under Grant No. 2023YFG0305, 2023YFG0124, CUIT Science and Technology Innovation Capacity Enhancement Program Project under Grant KYTD202206.

Contribution Statement

The contributions of Chengming Feng and Jing Hu to this paper were equal.

References

[An *et al.*, 2021] Jie An, Siyu Huang, Yibing Song, Dejing Dou, Wei Liu, and Jiebo Luo. Artflow: Unbiased image style transfer via reversible neural flows. In *CVPR*, pages 862–871, 2021.

[Butler *et al.*, 2012] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, pages 611–625. Springer, 2012.

[Chen *et al.*, 2017] Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, and Gang Hua. Stylebank: An explicit representation for neural image style transfer. In *CVPR*, 2017.

[Chen *et al.*, 2021] Haibo Chen, Zhizhong Wang, Huiming Zhang, Zhiwen Zuo, Ailin Li, Wei Xing, Dongming Lu, et al. Artistic style transfer with internal-external learning and contrastive learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[Cheng *et al.*, 2021] Jiaxin Cheng, Ayush Jaiswal, Yue Wu, Pradeep Natarajan, and Prem Natarajan. Style-aware normalized loss for improving arbitrary style transfer. In *CVPR*, 2021.

[Deng *et al.*, 2021a] Yingying Deng, Fan Tang, Weiming Dong, Haibin Huang, Chongyang Ma, and Changsheng Xu. Arbitrary video style transfer via multi-channel correlation. *AAAI*, 2021.

[Deng *et al.*, 2021b] Yingying Deng, Fan Tang, Xingjia Pan, Weiming Dong, Chongyang Ma, and Changsheng Xu. Stytrˆ 2: Unbiased image style transfer with transformers. *arXiv preprint arXiv:2105.14576*, 2021.

[Donahue *et al.*, 2017] Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):677–691, 2017.

[Gatys *et al.*, 2015a] Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. *NeurIPS*, 28:262–270, 2015.

[Gatys *et al.*, 2015b] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[Gu *et al.*, 2017] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE ICRA*, pages 3389–3396. IEEE, 2017.

[Haarnoja *et al.*, 2017] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *ICML*, pages 1352–1361. PMLR, 2017.

[Haarnoja *et al.*, 2018] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[Hu *et al.*, 2023] Jing Hu, Zhikun Shuai, Xin Wang, Shu Hu, Shanhui Sun, Siwei Lyu, and Xi Wu. Attention guided policy optimization for 3d medical image registration. *IEEE Access*, 2023.

[Huang and Belongie, 2017] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE ICCV*, pages 1501–1510, 2017.

[Johnson *et al.*, 2016] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*. Springer, 2016.

[Kingma and Welling, 2013] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[Li *et al.*, 2017a] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. Demystifying neural style transfer. *arXiv preprint arXiv:1701.01036*, 2017.

[Li *et al.*, 2017b] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. *Advances in neural information processing systems*, 30, 2017.

[Li *et al.*, 2019] Xueting Li, Sifei Liu, Jan Kautz, and Ming-Hsuan Yang. Learning linear transformations for fast image and video style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3809–3817, 2019.

[Lillicrap *et al.*, 2015] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[Lin *et al.*, 2014] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014.

[Lin *et al.*, 2021] Tianwei Lin, Zhuoqi Ma, Fu Li, Dongliang He, Xin Li, Errui Ding, Nannan Wang, Jie Li, and Xinbo

Gao. Drafting and revision: Laplacian pyramid network for fast high-quality artistic style transfer. In *CVPR*, pages 5141–5150, 2021.

[Liu *et al.*, 2021] Songhua Liu, Tianwei Lin, Dongliang He, Fu Li, Meiling Wang, Xin Li, Zhengxing Sun, Qian Li, and Errui Ding. Adaattn: Revisit attention mechanism in arbitrary neural style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6649–6658, 2021.

[Luo *et al.*, 2021] Ziwei Luo, Jing Hu, Xin Wang, Siwei Lyu, Bin Kong, Youbing Yin, Qi Song, and Xi Wu. Stochastic actor-executor-critic for image-to-image translation. *IJCAI*, 2021.

[Mahendran and Vedaldi, 2015] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *CVPR*, pages 5188–5196, 2015.

[Mirowski *et al.*, 2017] Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andrew J Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, et al. Learning to navigate in complex environments. *ICLR*, 2017.

[Park and Lee, 2019] Dae Young Park and Kwang Hee Lee. Arbitrary style transfer with style-attentional networks. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5880–5888, 2019.

[pex, 2022] Pexels. https://www.pexels.com/, 2022. Accessed: 2022-03-12.

[Phillips and Mackintosh, 2011] Fred Phillips and Brandy Mackintosh. Wiki art gallery, inc.: A case for critical thinking. *Issues in Accounting Education*, 26(3):593–608, 2011.

[Risser *et al.*, 2017] Eric Risser, Pierre Wilmot, and Connelly Barnes. Stable and controllable neural texture synthesis and style transfer using histogram losses. *arXiv preprint arXiv:1701.08893*, 2017.

[Sheng *et al.*, 2018] Lu Sheng, Ziyi Lin, Jing Shao, and Xiaogang Wang. Avatar-net: Multi-scale zero-shot style transfer by feature decoration. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8242–8250, 2018.

[Shi *et al.*, 2015] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015.

[Simonyan and Zisserman, 2014] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[Tassa *et al.*, 2018] 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.

[Ulyanov *et al.*, 2016a] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *ICML*, volume 1, page 4, 2016.

[Ulyanov *et al.*, 2016b] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[Wang *et al.*, 2020] Wenjing Wang, Shuai Yang, Jizheng Xu, and Jiaying Liu. Consistent video style transfer via relaxation and regularization. *IEEE Transactions on Image Processing*, 29:9125–9139, 2020.

[Xiang *et al.*, 2022] Yanfei Xiang, Xin Wang, Shu Hu, Bin Zhu, Xiaomeng Huang, Xi Wu, and Siwei Lyu. Rmbench: Benchmarking deep reinforcement learning for robotic manipulator control. *arXiv preprint arXiv:2210.11262*, 2022.

[Zhao *et al.*, 2019] Xujiang Zhao, Shu Hu, Jin-Hee Cho, and Feng Chen. Uncertainty-based decision making using deep reinforcement learning. In *2019 22th International Conference on Information Fusion (FUSION)*, pages 1–8. IEEE, 2019.