Deep Reinforcement Learning for Image-to-Image Translation

Xin Wang, *Senior Member, IEEE,* Ziwei Luo, Jing Hu, Chengming Feng, Shu Hu, Bin Zhu, Xi Wu, Xin, Li, *Fellow, IEEE*, Siwei Lyu, *Fellow, IEEE*

Abstract—Most existing Image-to-Image Translation (I2IT) methods generate images in a single run of deep learning (DL) models. However, designing a single-step model often requires many parameters and suffers from overfitting. Inspired by the analogy between diffusion models and reinforcement learning, we reformulate I2IT as an iterative decision-making problem via deep reinforcement learning (DRL) and propose a computationally efficient RL-based I2IT (`RL-I2IT`) framework. The key feature in the `RL-I2IT` framework is to decompose a monolithic learning process into small steps with a lightweight model to progressively transform the source image to the target image. Considering the challenge of handling high-dimensional continuous state and action spaces in the conventional RL framework, we introduce meta policy with a new "concept Plan" to the standard Actor-Critic model. This plan is of a lower dimension than the original image, which facilitates the actor to generate a tractable high-dimensional action. In the `RL-I2IT` framework, we also employ a task-specific auxiliary learning strategy to stabilize the training process and improve the performance of the corresponding task. Experiments on several I2IT tasks demonstrate the effectiveness and robustness of the proposed method when facing high-dimensional continuous action space problems. Our implementation of the `RL-I2IT` framework is available at https://github.com/Algolzw/SPAC-Deformable-Registration.

Index Terms—Image to Image Translation, Deep Reinforcement Learning, Meta Policy, Auxiliary Learning.

1 Introduction

Many computer vision problems, such as face inpainting, semantic segmentation, image registration, realistic photo generated from sketch, and neural style transfer, can be unified under the framework of learning image-to-image translation (I2IT) [1]. Existing approaches to I2IT can be categories into either one-step deep-learning (DL) framework (e.g., Variational Autoencoders [2], U-Net [3], and conditional GANs [4]) or iterative diffusion models (e.g., Palette [5], SSDM [6], Plug-and-Play [7]). Directly learning I2IT with one-step DL models typically suffers from two major challenges. One is that to handle high-dimensional I2IT problems, one-step DL models typically have complex structures and many parameters, making them difficult to train and hard to deploy in resource-limited scenarios such as mobile devices. The other is that many of these models do not generalize well [8] due to the abundance of global minima caused by the over-parameterized setting. Although these problems can be potentially alleviated by using multi-scale models or multi-stage pipelines such as diffusion models, we still face the challenge of prohibitive computational complexity.

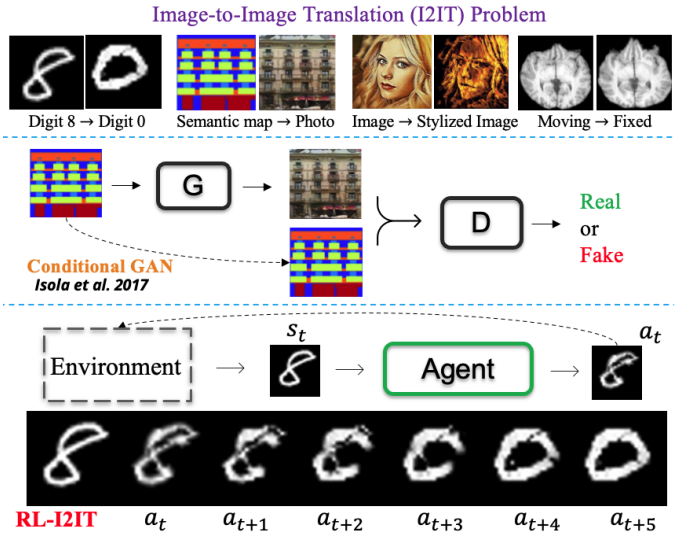

Fig. 1: **Top:** I2I Problem translates an image from a source domain to a target domain. **Mid:** Example of one-step method CGAN [4]. **Bottom:** Our RL-based stepwise I2IT progressively transforms the source image, and the process is demonstrated clearly.

To address these limitations of existing methods, we explore solving I2IT problems by leveraging the recent advances in deep reinforcement learning (DRL). The key idea is to decompose the monolithic learning process into small steps with a lightweight CNN, aiming to improve the quality of predicted results progressively (See Fig. 1). By decomposing a one-step complex task into a series of simpler tasks, our approach can handle the simplified task with a much simpler network rather than using a large, heavily

- *Jing Hu, Chengming Feng, and Xi Wu are with Chengdu University of Information Technology, China. Jing Hu and Xi Wu are the corresponding authors. e-mail: (jing_hu09@163.com, fengxiaoming520@gmail.com, xi.wu@cuit.edu.cn)*
- *Ziwei Luo is with Uppsala University, Sweden. e-mail:(ziwei.luo@it.uu.se)*
- *Shu Hu is with the Department of Computer Information and Graphics Technology, Purdue School of Engineering and Technology at Indiana University-Purdue University Indianapolis, IN, 46202, USA (e-mail: hu968@purdue.edu)*
- *Bin Zhu is with Microsoft Research Asia. e-mail:(binzhu@microsoft.com)*
- *Xin Li is with the Department of Computer Science, University at Albany, State University of New York (SUNY), NY 12222, USA. (e-mail: xli48@albany.edu).*
- *Siwei Lyu is with the Department of Computer Science and Engineering, University at Buffalo, SUNY, USA. e-mail:(siweilyu@buffalo.edu)*
- *Xin Wang is with the Department of Epidemiology and Biostatistics, School of Public Health, University at Albany, State University of New York (SUNY), NY 12222, USA. (e-mail: xwang56@albany.edu).*

parameterized network. Although recent works have successfully applied DRL to solve several visual tasks [9], [10], [11], their action spaces are usually discrete, making them unsuitable for I2IT, which requires continuous action spaces. A promising direction for learning continuous actions is the maximum entropy reinforcement learning (MERL), which improves both exploration and robustness by maximizing a standard RL objective with an entropy term [12]. Soft actor-critic (SAC) [12] is an instance of MERL and has been applied to solve continuous action tasks [13]. However, the main issue hindering the applicability of SAC on I2IT is its inability to handle high-dimensional states and actions effectively. Recently, RAE [14] tried to address this problem by combining SAC with a regularized autoencoder, but it only provides an auxiliary loss for end-to-end RL training and is incapable of the I2IT tasks. Besides, high-dimensional states and actions require training an I2IT-based RL model to make much more exploration and exploitation, which leads to unstable training [14]. One solution to stabilize training is to extract a lower-dimensional visual representation with a separately pre-trained DNN model and learn the value function and corresponding policy in the latent space [15]. However, this approach can not be trained from scratch. Otherwise, it can lead to inconsistent state representations with an optimal policy.

Inspired by the analogy between diffusion models and RL [16], we propose a new DRL framework, named `RL-I2IT`, for I2IT problems to handle high-dimensional continuous state and action spaces. As shown in Fig. 2, the `RL-I2IT` framework comprises three core deep neural networks: a planner, an actor, and a critic. We introduce a new "concept plan" to decompose the decision-making process into two steps, state $\rightarrow$ plan and plan $\rightarrow$ action. We call this process meta policy. The plan is a subspace of appropriate actions based on the current state. It is not applied to the state directly. Instead, it is used to guide the actor to generate a tractable high-dimensional action that interacts with the environment. The plan can be considered as an intermediate transition between state and action. As the input of the actor, the plan has a much lower dimension compared with the state, making it easier for the actor to learn to predict actions. Meanwhile, the plan can be evaluated by the critic efficiently since the Q function is easier to learn in the low-dimensional latent space. Furthermore, compared with training a one-step differentiable DL-based model, it is much harder to learn from such a high-dimensional continuous control problem with traditional RL frameworks. To address it, we also employ a task-specific auxiliary learning strategy to stabilize the training process and improve the performance of the corresponding task. The auxiliary learning part could be any learning technology that is flexible and can readily leverage any other advanced losses or objectives. For example, we use the standard L_2 reconstruction loss as auxiliary learning in many I2IT tasks. Our main contributions can be summarized as follows:

- A new DRL framework `RL-I2IT` is proposed to handle the complex I2IT problem with high-dimensional continuous actions by decomposing the monolithic learning process into small steps.
- To tackle the high-dimensional continuous action learning problem, we propose a stochastic meta policy that divides the decision-making processing into two steps: state $\rightarrow$ low-dimensional plan and plan $\rightarrow$ action. The plan guides the actor to predict a tractable action, and the critic evaluates the plan. The approach makes the whole learning process feasible and computationally efficient.
- Compared to existing DL-based models, our DRL-based model is lightweight, making it simple and computationally efficient. For example, compared to a recent one-step I2IT model pix2pixHD of size 45.9M [17], the size of our model is only 9.7M. The training speed of `RL-I2IT`is estimated to be at least one order of magnitude faster than that of Palette [5].
- Our `RL-I2IT` framework is flexible in incorporating many advanced auxiliary learning methods for various complex I2IT applications. Experimental results on a variety of applications, from face inpainting and neural style transfer to digits transform and deformable image registration, show that our approach achieves state-of-the-art performance.

This paper extends our previous conference papers [18], [19] and [20] substantially in the following aspects: (i) We propose an efficient general RL-based framework for the I2IT problem. In this regard, our previous works [18], [19] and [20] can be considered as special cases of the general framework in this paper. (ii) We provide more technical details for each application of the `RL-I2IT` framework, such as the detailed network architectures. (iii) We provide additional diagnostic experiments for each application to demonstrate the effectiveness of our `RL-I2IT` framework in computer vision and medical image applications. In the neural style transfer task, we add additional experiments to evaluate the necessity of high-dimensional latent space and the user case study. In the medical image registration task, we add more experimental analysis for hyper-parameters, the trade-off between performance and inference time, etc.

The remaining content of this paper is organized as follows. After introducing the background in Section 2, we describe the `RL-I2IT` framework for step-wise I2IT in Section 3. In Section 4 and 5, we demonstrate experimentally the effectiveness and robustness of the `RL-I2IT` framework on computer vision applications (digits transform 4.1, face inpainting 4.2, realistic image translation 4.3, and neural style transfer 4.4) and medical image applications (deformable medical image registration 5.1), respectively. We conclude the paper in Section 6 with discussions of the limitations of the framework and future works.

2 Background

2.1 Image-to-Image Translation

Image-to-image translation (I2IT) aims to translate input images from a source domain to a target domain, such as generating realistic photos from semantic segmentation labels [4], synthesizing completed visual targets from images with missing regions [21], deformable image registration [22], neural style transfer [23], etc. Autoencoder is leveraged in most research works to learn this process by minimizing the reconstruction error between the predicted image and the target. In addition, the generative adversarial network (GAN) is also vigorously studied in I2IT to synthesize realistic images [4]. Subsequent works enhance I2IT performance by using a coarse-to-fine deep learning framework [24] that recursively sets the output of the previous stage as the input of the next stage. In this way, the I2IT task is transformed into a multi-stage, coarse-to-fine solution. Although the recursion can be infinitely applied in practice, it is limited by the increasing model size and training instability. More I2IT-related works can be found in a recent survey paper [1].

More recently, diffusion models have found successful applications in many vision tasks including I2IT [5], [6], [25]. In

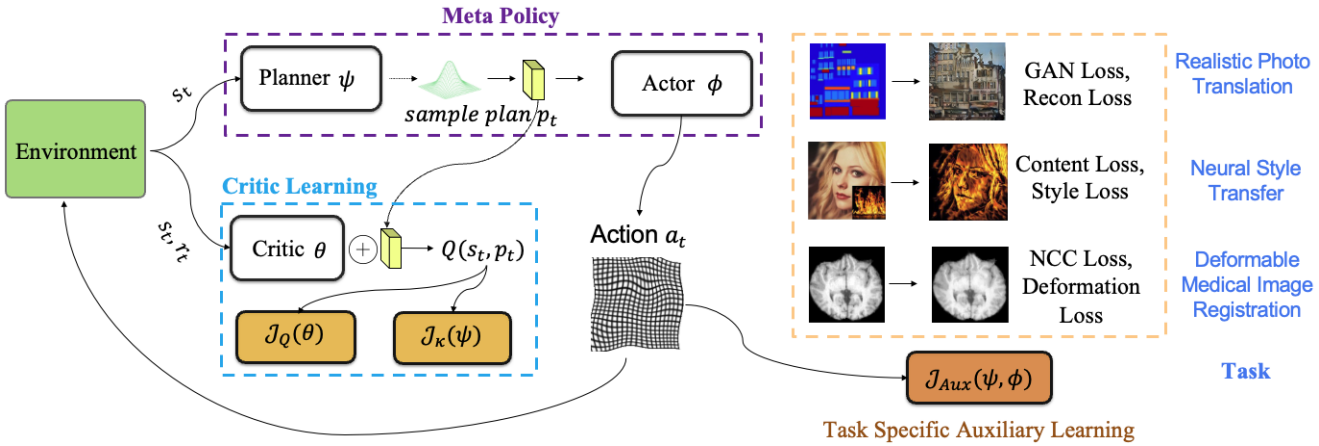

Fig. 2: Our RL-I2IT framework with a Planner-Actor-Critic structure. **Left:** At time step t, the environment receives executable action $\mathbf{a}_t$, and outputs state and reward $(\mathbf{s}_t, r_t)$. In our meta policy, latent plan $\mathbf{p}_t$ is sampled from the planner to guide the actor to generate executable action $\mathbf{a}_t$ that interacts with the environment. The plan is also evaluated by the critic. The nature of $\mathbf{a}_t$ is also task-dependent, for tasks like Deformable Image Registration, $\mathbf{a}_t$ may be a deformation field applied to the current state. For tasks aiming at realistic image generation, such as face inpainting or neural style transfer, $\mathbf{a}_t$ could directly be the target image. **Right:** Task-specific auxiliary learning objectives depend on specific tasks for various purposes, such as stabilizing the training process or improving performance.

Palette [5], diffusion models (DM) outperform strong GAN and regression baselines on four I2IT tasks without task-specific hyper-parameter tuning, architecture customization, or any auxiliary loss. This work has inspired several DM-based approaches to I2IT such as Brownian Bridge Diffusion Model (BBDM) [25] and score-decomposed diffusion models (SSDM) [6]. Inspired by the success of vision-language models, text-driven I2IT based on plug-and-play diffusion features [7] has shown high fidelity to input structure and scene layout, while significantly changing the perceived semantic meaning of objects and their appearance.

2.2 Reinforcement Learning with Continuous Action

RL is described by an infinite-horizon Markov decision process (MDP), defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{U}, r, \gamma)$, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is action, $\mathcal{U} : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, \infty)$ represents the state transition probability density given state $\mathbf{s} \in \mathcal{S}$ and action $\mathbf{a} \in \mathcal{A}$, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward emitted from each transition, and $\gamma \in [0, 1]$ is the reward discount factor. Standard RL learns to maximize the expected sum of rewards from the episodic environments under the trajectory distribution ρ_π.

Maximum Entropy RL (MERL) incorporates an entropy term with the policy, and the resulting objective is defined as $\sum_{t=1}^{T} \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \rho_\pi} [r_t(\mathbf{s}_t, \mathbf{a}_t) + \alpha \mathcal{H}(\pi_\phi(\cdot|\mathbf{s}_t))]$, where α is a temperature parameter controlling the balance of the entropy $\mathcal{H}$ and the reward r_t. MERL model has proven stable and powerful in low dimensional continuous action tasks, such as games and robotic controls [12]. However, when facing complex visual problems such as I2IT, where observations and actions are high-dimensional, it remains a challenge for MERL models [26]. Soft actor-critic (SAC) [12] has been shown as a promising framework for learning continuous actions, which is an off-policy actor-critic method that uses the above entropy-based framework to derive the soft policy iteration. The advantage of SAC is that it provides sample efficient learning and stability. It can improve both the exploration and robustness of the learned model. The original SAC paper reports its performance on continuous control tasks with up to 21 dimensions, which is far from enough for handling I2IT tasks. Recent studies [14], [26] have shown that the SAC has limitations when handling high-dimensional states and actions.

More recently, stochastic latent actor-critic (SLAC) [26] improves the SAC by learning representation spaces with a latent variable model which is more stable and efficient for complex continuous control tasks. It can improve both the exploration and robustness of the learned model. However, the capability of SLAC is limited in a continuous action space. The reason is that the latent state representation in SLAC is only used to facilitate the training of the critic, which cannot handle tasks with a high-dimensional action space.

3 Reinforcement Learning for I2IT

3.1 Problem Formulation

In our study, Image-to-Image Translation (I2IT) is reformulated as a multistep decision-making problem, where the transformation from an input image to a target image is not executed in a single step. Instead, we introduce a lightweight Deep Reinforcement Learning (DRL) model that incrementally performs the transformation, allowing the progressive addition of new details. We conceptualize I2IT as a Markov Decision Process (MDP), where translation, denoted as $\mathcal{T}$, moves from the current state $\mathbf{s}$ to the target $\mathbf{y}$ through a defined policy. This approach allows for a more delicate and progressive process of image transformation within the MDP framework, which can be formulated as follows.

$$\mathcal{T}(\mathbf{s}) = \mathcal{T}_t \circ \mathcal{T}_{t-1} \circ \cdots \mathcal{T}_0(\mathbf{s}) = \mathbf{y},$$

where $\circ$ is a composition operator, $\mathcal{T}_t$ is the t-th translation step, which can predict the image from state $\mathbf{s}_t$. Moreover, state $\mathbf{s}$ can be defined according to the specific I2IT task.

3.2 Stochastic Meta Policy of Planning and Acting

Our RL-I2IT framework is designed to handle high-dimensional continuous states and actions in an infinite-horizon Markov Decision Process (MDP). It incorporates a novel component, a planner, specifically for continuous plan space. This approach diverges

from traditional policies that directly map environmental states to actions [27]. Instead, it bifurcates the mapping process into two distinct steps: first state $\rightarrow$ plan, and then plan $\rightarrow$ action. We term this new two-step mapping process as a "meta policy", which allows for more intricate and layered decision-making compared to standard reinforcement learning models. We define the new mapping process as meta policy, and the new MDP for our RL-I2IT can be represented by the tuple $(\mathcal{S}, \mathcal{P}, \mathcal{A}, \mathcal{U}, r, \gamma)$. $\mathcal{S}$ is a set of states, $\mathcal{P}$ is a continuous plan, $\mathcal{A}$ is a continuous action, and $\mathcal{U} : \mathcal{S} \times \mathcal{P} \times \mathcal{S} \times \mathcal{A} \rightarrow [0, \infty)$ represents the state transition probability density of the next state $\mathbf{s}_{t+1}$ given state $\mathbf{s}_t \in \mathcal{S}$, plan $\mathbf{p}_t \in \mathcal{P}$ and action $\mathbf{a}_t \in \mathcal{A}$.

Our RL-I2IT framework is shown in Fig. 2. It comprises three core deep neural networks: the planner, the actor, and the critic with parameters ψ, ϕ, and θ, respectively. The planner aims to generate a high-level plan in low-dimensional latent space to guide the actor. In some sense, the plan can be considered as action clusters or action templates, which are high-level crude actions. Unlike classic policy models, the input of the actor is a stochastic plan instead of the state. That is, the generated plan is forwarded to the actor further to create the high-dimensional action in our meta-policy model. Meanwhile, this plan is evaluated by the critic. By using the meta policy and the stochastic planner-actor-critic structure, RL-I2IT makes the learning process of a complex I2IT task easier.

Formally, suppose a meta policy is defined as (κ, π). The stochastic plan is modeled as a subspace of the deformation field that gives a low-dimensional vector $\mathbf{p}_t$ based on the state $\mathbf{s}_t$, while action $\mathbf{a}_t$ is determined by the plan $\mathbf{p}_t$. Consider a parameterized planner κ_ψ and actor π_ϕ, the stochastic plan is sampled as a representation: $\mathbf{p}_t \sim \kappa_\psi(\mathbf{p}_t|\mathbf{s}_t)$, and the action is generated by decoding the plan vector $\mathbf{p}_t$ into a high-dimensional executable action: $\mathbf{a}_t = \pi_\phi(\mathbf{a}_t|\mathbf{p}_t)$. In practice, we reparameterize the planner and stochastic plan jointly using a neural network approximation $\mathbf{p}_t = f_\psi(\epsilon_t, \mathbf{s}_t)$, known as the reparameterization trick [2], where ϵ_t is an input noise vector sampled from a fixed Gaussian distribution. Moreover, we maximize the entropy of the plan to improve exploration and robustness. The augmented objective function is formulated as follows:

$$\max_{\psi,\phi} \sum_{t=1}^{T} \mathbb{E}_{(\mathbf{s}_t,\mathbf{p}_t,\mathbf{a}_t)\sim\rho_{(\kappa,\pi)}} \left[r_t(\mathbf{s}_t, \mathbf{p}_t, \mathbf{a}_t) + \alpha \mathcal{H}(\kappa_\psi(\cdot|\mathbf{s}_t)) \right], \tag{1}$$

where α is the temperature and $\rho_{(\kappa,\pi)}$ is a trajectory distribution under $\kappa_\psi(\mathbf{p}_t|\mathbf{s}_t)$ and $\pi_\phi(\mathbf{a}_t|\mathbf{p}_t)$.

3.3 Learning Planner and Critic

Unlike conventional RL algorithms, the critic Q_θ in our framework evaluates the plan $\mathbf{p}_t$ instead of the action $\mathbf{a}_t$ since learning a low-dimensional plan in an I2IT problem is easier and more effective. Specifically, the low-dimensional plan is concatenated into the downsampled vector of the critic and outputs the soft Q function $Q_\theta(\mathbf{s}_t, \mathbf{p}_t)$, which is an estimation of the current state plan value, as shown in Fig. 2.

When the critic is used to evaluate the planner, rewards and soft Q values are used to iteratively guide the stochastic meta-policy improvement. In the evaluation step, by following SAC [12], RL-I2IT learns κ_ψ (planner) and fits parametric Q-function $Q_\theta(\mathbf{s}_t, \mathbf{p}_t)$ (critic) using transitions sampled from the replay pool $\mathcal{D}$ by minimizing the soft Bellman residual:

Algorithm 1: Learning Planner-Actor-Critic

Input: I_F, I_M, U_F, U_M, replay pool $\mathcal{D}$
Init: ψ, ϕ, θ, $\bar{\theta}$, $\mathcal{D}$ and environment $\mathcal{E}$
for *each iteration* **do**
  for *each environment step* **do**
    $\mathbf{p}_t \sim \kappa_\psi(\mathbf{p}_t|\mathbf{s}_t)$, $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t|\mathbf{p}_t)$
    $\mathbf{s}_{t+1}, r_t \sim \mathcal{U}(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{p}_t, \mathbf{a}_t)$
    $\mathcal{D} = \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{p}_t, \mathbf{a}_t, r_t, s_{t+1})\}$
  end
  for *each gradient step* **do**
    Sample from $\mathcal{D}$
    Update θ, ψ, ϕ with Eq. (2), Eq. (3), Eq. (7)
  end
end

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

where $V_{\bar{\theta}}(\mathbf{s}_t) = \mathbb{E}_{\mathbf{p}_t\sim\kappa_\psi}[Q_{\bar{\theta}}(\mathbf{s}_t, \mathbf{p}_t) - \alpha \log \kappa_\psi(\mathbf{p}_t|\mathbf{s}_t)]$. We use a target network $Q_{\bar{\theta}}$ to stabilize training, whose parameters $\bar{\theta}$ are obtained by an exponentially moving average of parameters of the critic network [28]: $\bar{\theta} \rightarrow \tau\theta + (1-\tau)\bar{\theta}$. Hyper-parameter $\tau \in [0, 1]$. To optimize $J_Q(\theta)$, we can do the stochastic gradient descent with respect to parameters θ as follows,

$$\begin{aligned} \theta = \theta - \eta_Q \nabla_\theta Q_\theta(\mathbf{s}_t, \mathbf{p}_t) \Big(& Q_\theta(\mathbf{s}_t, \mathbf{p}_t) - r_t \\ & - \gamma \left[Q_{\bar{\theta}}(\mathbf{s}_{t+1}, \mathbf{p}_{t+1}) - \alpha \log \kappa_\psi(\mathbf{p}_{t+1}|\mathbf{s}_{t+1}) \right] \Big). \end{aligned} \tag{2}$$

Since the critic works on the planner, the optimization procedure will also influence the planner's decisions. Following [12], we can use the following objective to minimize the KL divergence between the policy and a Boltzmann distribution induced by the Q-function,

$$\begin{aligned} J_\kappa(\psi) = & \mathbb{E}_{\mathbf{s}_t\sim\mathcal{D}} \left[\mathbb{E}_{\mathbf{p}_t\sim\kappa_\psi} \left[\alpha \log(\kappa_\psi(\mathbf{p}_t|\mathbf{s}_t)) - Q_\theta(\mathbf{s}_t, \mathbf{p}_t) \right] \right] \\ = & \mathbb{E}_{\mathbf{s}_t\sim\mathcal{D}, \epsilon_t\sim\mathcal{N}(\mu,\sigma)} \big[\alpha \log(\kappa_\psi(f_\psi(\epsilon_t, \mathbf{s}_t)|\mathbf{s}_t)) \\ & - Q_\theta(\mathbf{s}_t, f_\psi(\epsilon_t, \mathbf{s}_t)) \big]. \end{aligned}$$

The last equation holds because $\mathbf{p}_t$ can be evaluated by $f_\psi(\epsilon_t, \mathbf{s}_t)$ as we discussed before. It should be mentioned that hyperparameter α can be automatically adjusted by using the method proposed in [12]. Then we can apply the stochastic gradient method to optimize parameters as follows,

$$\begin{aligned} \psi = & \psi - \eta_\psi \Big(\nabla_\psi \alpha \log(\kappa_\psi(\mathbf{p}_t|\mathbf{s}_t)) + \\ & \big(\nabla_{\mathbf{p}_t} \alpha \log(\kappa_\psi(\mathbf{p}_t|\mathbf{s}_t)) - \nabla_{\mathbf{p}_t} Q_\theta(\mathbf{s}_t, \mathbf{p}_t) \big) \nabla_\psi f_\psi(\epsilon_t, \mathbf{s}_t) \Big). \end{aligned} \tag{3}$$

The derivation for the case of the critic evaluating the actor can be found in Appendix A. Our experimental results to be reported in Table 10 will show that the critic evaluates actor's action results in an inferior performance to that the critic evaluates the planner.

3.4 Task Specific Auxiliary Learning

Following our meta policy (κ_ψ, π_ϕ), the framework derives the executable action $\mathbf{a}_t$. To enhance convergence and performance, we adopt auxiliary learning for the planner and actor, tailored to specific tasks. This approach is highly adaptable and capable of integrating various advanced losses and techniques.

For instance, in face inpainting tasks, we focus on reconstructing the predicted faces to match the original ones while also synthesizing more realistic images. This is achieved by employing a discriminator on predicted images with an adversarial loss. The nature of $\mathbf{a}_t$ is also task-dependent: for tasks like Deformable Image Registration, $\mathbf{a}_t$ may be a deformation field applied to the current state. In contrast, for tasks aiming at realistic image generation, such as face inpainting or neural style transfer, $\mathbf{a}_t$ could directly be the target image $\mathbf{y}$. Detailed explanations and examples of these applications are provided in the experimental sections. We elaborate on the auxiliary learning process using face inpainting tasks as an example. Concretely, the empirical objective of the reconstruction part in our framework is:

$$\mathcal{L}_{rec} = \mathbb{E}_{\mathbf{s}_t, \mathbf{y} \sim \mathcal{D}}[\|\mathcal{T}(\mathbf{s}_t) - \mathbf{y}\|_d], \tag{4}$$

where $\mathcal{D}$ is a replay pool, $\|\cdot\|_d$ denotes some distance measure, such as L_1 or L_2. By adding a discriminator D to predicted images, the adversarial loss is defined as:

$$\mathcal{L}_{adv} = \mathbb{E}_{\mathbf{s}_t, \mathbf{y} \sim \mathcal{D}}[\log(D(\mathbf{y})) + \log(1 - D(\mathcal{T}(\mathbf{s}_t)))] \tag{5}$$

In this example, the final auxiliary learning objective can be expressed as

$$\mathcal{J}_{Aux} = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{adv}\mathcal{L}_{adv}, \tag{6}$$

where λ_{rec} and λ_{adv} are the weight terms for reconstruction and adversarial learning. Finally, we can update ψ and ϕ from the planner and the actor by performing the following steps:

$$\psi = \psi - \eta\nabla_\psi \mathcal{J}_{Aux}(\psi, \phi), \quad \phi = \phi - \eta\nabla_\phi \mathcal{J}_{Aux}(\psi, \phi). \tag{7}$$

Note that the additional auxiliary learning may introduce new parameters to learn, such as the discriminator D in the above example. Since our goal is to learn the planner and the actor, which are the only components used in testing, we omit those additional notions for simplicity. More concrete examples of auxiliary learning are introduced in the experiment sections for different I2IT applications. The pseudo-code of optimizing RL-I2IT is described in Algorithm 1. All parameters of RL-I2IT are optimized based on the samples from replay pool $\mathcal{D}$.

3.5 Environment Settings in Practice

In the given RL-I2IT framework, environment designs are tailored for various applications, with detailed guidance in each section. This section outlines general principles for selecting rewards, focusing on the critic's role in evaluating plans rather than actions. The plans, being a subset of potential actions, serve as high-level instructions for the actor to create specific actions. Evaluation measures for structural or global image information, such as SSIM or the DICE score [29], are proposed as rewards for assessing these plans, as the evaluations shown in Table 9. However, the text emphasizes that the choice of reward should remain flexible and be empirically tested in the context of individual applications.

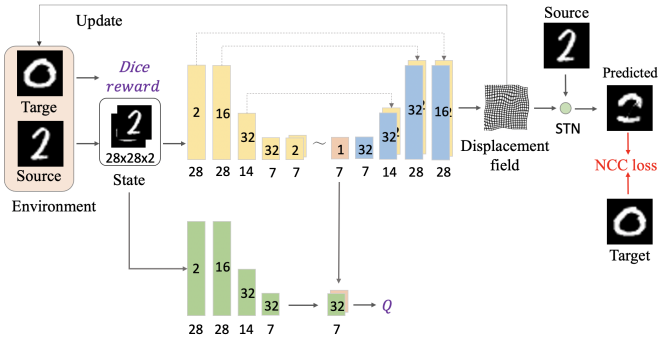

Fig. 3: The network architecture of RL-I2IT for MNIST dataset. Each rectangle represents a 2D image (or feature map), the number of channels is shown inside the rectangle, and the responding resolution is printed underneath. 'STN' stands for spatial transformer network [33], which is used to transform the source image with the predicted displacement field.

4 Applications on Computer Vision

4.1 Digits Transform

We first evaluate our RL-I2IT framework on MNIST [30], which is a dataset of digits and is regarded as a standard sanity check for a proposed method. The goal is to transform between two different images of handwritten digits of 28×28 pixels.

4.1.1 RL-I2IT Setting

For digits transform, the state is a concatenation of the predicted image and the target image. The Dice score [29] is used as the reward, and the NCC loss [31], [32] is leveraged for auxiliary learning. A very simple network structure is used to construct the planer, the actor, and the critic, as shown in Fig. 3. More specifically, the plan is a one-channel 7×7 feature map (49-dimensional plan), and the actor outputs a deformation field, which is used to transform the source image by spatial transformer network (STN) [33]. All convolution operations use a 3×3 kernel with the LeakyReLU activation function. The downsampling operation is performed by max-pooling, and all upsampling operations are performed with the nearest interpolation.

4.1.2 Experiment

The following four types of spatial transforms are used in this experiment: (1) Inner-class transform, which transforms digits within the same class; (2) Cross-class transform, which transforms digits cross different classes; (3) Random transform, which transforms digits cross different classes that are randomly scaled from 0.3 to 1.7 and rotated between 0 and 360 degrees; (4) Continuous and random transform, which randomly selects a set of digits that have been scaled and rotated and then transforms the first digit to the last one in order. In the testing phase, ten digits from 0 to 9 are used as the atlases, and 1000 randomly scaled and rotated digits are used as the moving images, which need to be aligned with the atlas. We use the Dice score as the quantitative measure (the higher, the better).

The left panel of Fig. 4 shows the process of transforming digits using the RL-I2IT framework. The result shows that our method can transform digits step-wise, and it can capture the style and shape accurately. The experiment on random and continuous transform (the bottom left panel of Fig. 4) further shows that our RL-I2IT method is robust to complex transformations. The right panel of Fig. 4 compares our method with VoxelMorph (VM) [32],

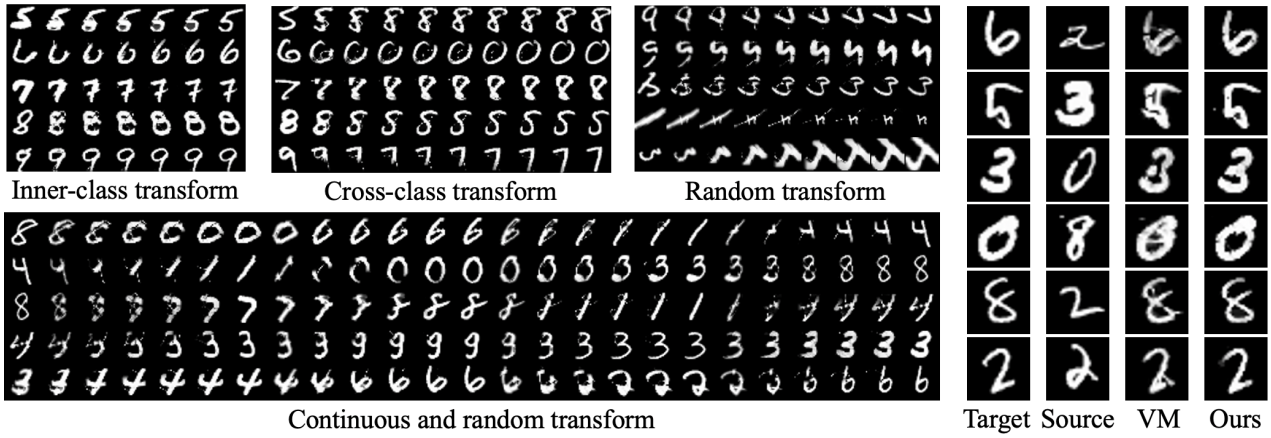

Fig. 4: **Top Left:** examples of using RL-I2IT to gradually transform the source digits (leftmost) to the target digits (rightmost). **Bottom Left:** Given a random set of digits, continuously transform the leftmost digit to the rightmost digit sequentially. **Right:** comparison between our RL-I2IT and VoxelMorph (VM) [32].

which is a state-of-the-art DL-based method for transformation tasks. It can be seen clearly that our method is better at recovering the details and the shapes of the target digit than VM does.

Fig. 5 shows the average Dice scores on the random transform and some transformed results. In the random transform experiment, the moving images are randomly scaled and rotated, which results in a much larger and more complex deformation field. Our method significantly outperforms VM over all digits, both quantitatively and qualitatively, which indicates that the proposed method has better generalizability and can work well on images with large deformations.

4.2 Face Inpainting

In this section, we apply our RL-I2IT framework to the face inpainting task, which aims to fill in a cropped region in the central area of a face with synthesized contents that are both semantically consistent with the original face and visually realistic.

4.2.1 RL-I2IT Setting

For the state, we use the original image with a missing region (center cropped) as the initial state, and the next state is obtained by adding the new predicted image to the missing region. We use the peak signal-to-noise ratio (PSNR) as the reward. We apply the L_1 loss with an adversarial loss for the auxiliary learning, which tries to make the predicted image more realistic and closer to the ground truth image. The λ_{rec} and λ_{adv} in Eq. 6 are set to 1.0 and 0.02, respectively.

The network architecture for face inpainting is shown in Fig. 6. For the planner-actor, we use a similar architecture with context-encoder [21] except for the skip connections and the stochastic sampling operation in the planner. We use the same network structure for all the types of discriminators except minor changes for different GANs. Specifically, for WGAN-GP, the sigmoid function is removed from the final output layer. A spectral normalization is added to each layer of the discriminator of SNGAN [34]. Moreover, the convolution layers of the planner, critic, and discriminator use 4×4 kernels, and the downsampling is performed by convolution with a stride of 2. In this application, the latent action dimension is set to 256.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓
CE [21]	25.764	0.850	0.0955	14.454
CA [24]	24.556	0.840	0.0715	9.950
PIC [36]	26.703	0.870	0.0844	12.470
PEN [35]	23.196	0.634	0.1342	35.422
RN [37]	25.123	0.835	0.0698	7.388
Shift-Net [38]	26.476	0.851	0.0703	7.597
ILO [39]	22.709	0.783	0.0958	13.122
Palette [5]	24.926	0.850	0.0567	4.909
Ours(PSNR)	27.351	0.897	0.0439	**4.697**
Ours(SSIM)	**27.598**	**0.899**	**0.0433**	4.917

TABLE 1: Quantitative results of all methods on Celeba-HQ. We use SNGAN + PSNR and SNGAN + SSIM as rewards respectively.

4.2.2 Experiment

We use the Celeba-HQ dataset in this task, which includes $28,000$ images for training and $2,000$ images for testing. All images have a cropped region of 64×64 pixels in the center. We compare our method with several recent face inpainting methods, including CE [21], CA [24], PEN [35], PIC [36], RN [37], Shift-Net [38], ILO [39], and Palette [5]. Following the previous work [21], [24], [36], we use PSNR and SSIM as the evaluation metrics.

Results and Analysis. The qualitative results produced by our framework and existing state-of-the-art methods are shown in Fig. 7. We can see easily that the RL-I2IT gives obvious visual improvement for synthesizing realistic faces. The RL-I2IT results are very reasonable, and the generated faces are sharper and more natural. This may be attributed to the high-level latent plan $\mathbf{p}_t$, which focuses on learning the global semantic structure and then directs the actor with auxiliary learning to further improve the local details of a generated image. We can also see that the synthesized images of the RL-I2IT can have very different appearances from the ground truth, which indicates that, although our training is based on paired images, the RL-I2IT can successfully explore and exploit data for producing diverse results.

The quantitative comparison is shown in Table 1. We can see that our method achieves the best PSNR and SSIM scores when compared with the existing state-of-the-art methods. As we mentioned before, the reward function in our RL framework is very flexible. Both PSNR- and SSIM-based rewards are suitable

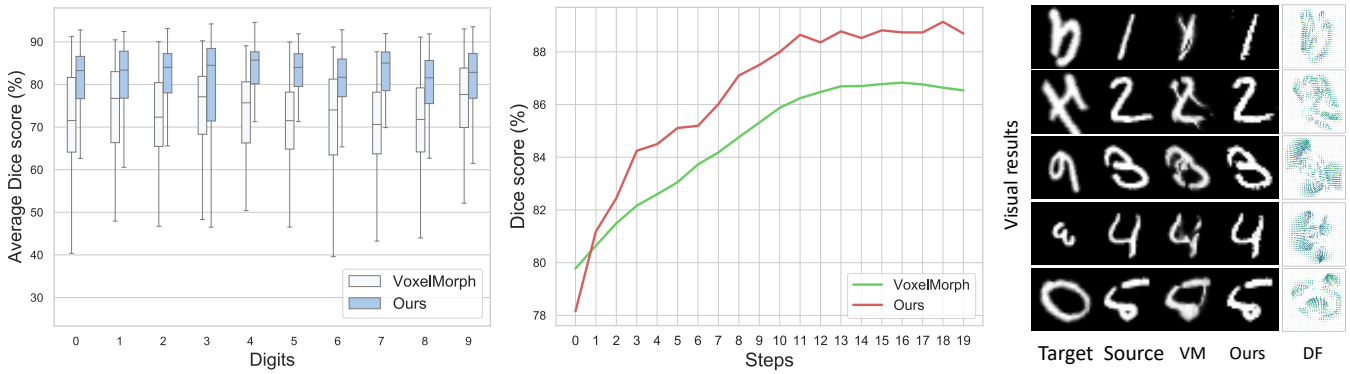

Fig. 5: **Left:** The plot box of Dice scores on 10 digits. **Center:** Step-wise comparison between our RL-I2IT and VoxelMorph (VM) [32]. **Right:** Visual comparison of our method with VM. The scaled and rotated digits are transformed to the fixed (target) digits. The Deformation Filed (DF) column visualizes the estimated deformable fields using RL-I2IT.

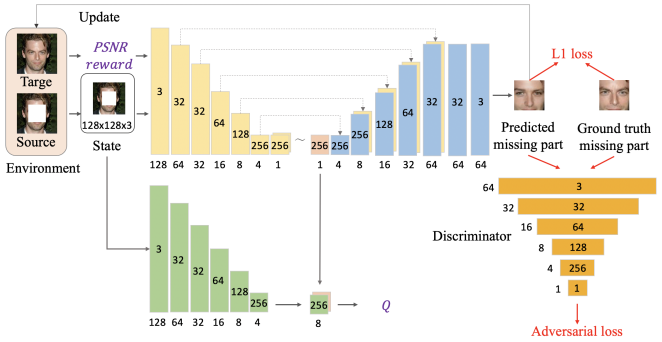

Fig. 6: The network architecture of RL-I2IT for face inpainting. Each rectangle represents a 2D image (or feature map), the number of channels is shown inside the rectangle, and the responding resolution is printed underneath (or on the left for discriminator).

Method	PSNR ↑	SSIM ↑
PA + SNGAN	26.884	0.871
Ours (+ WGAN-GP)	27.091	0.875
Ours (+ RaGAN)	27.080	0.873
Ours (+ SNGAN)	**27.176**	**0.882**

TABLE 2: Ablation study of our RL-I2IT framework on Celeba-HQ testing dataset (all trained with the PSNR reward).

for face inpainting with the RL-I2IT framework.

Ablation Study. To illustrate the stability of training GANs in our framework, we jointly use L_1 and several advanced GAN losses, i.e., WGAN-GP [40], RaGAN [41], and SNGAN [34] for auxiliary learning. We also separately train a planner-actor (PA) model by jointly optimizing the L_1 and the SNGAN loss. The results are shown in Table 2, which indicates that the RL-I2IT framework with different GANs is stable and significantly improves the performance of training the planner-actor with SNGAN alone, further demonstrating the power of the RL-I2IT framework.

4.3 Realistic Photo Translation

In this section, we evaluate our RL-I2IT framework on the general realistic photo translation task.

4.3.1 RL-I2IT Setting

For realistic photo translation, we directly use the source image as the initial state. The next state is obtained by warping the generated image to the source image. We also let the action as the predicted image directly and use the same auxiliary learning settings and network structure as in the face inpainting experiment with the PSNR reward and the SNGAN loss (See Section 4.2 for more details).

4.3.2 Experiment

We use three realistic photo translation tasks to evaluate our framework, (1) segmentation *labels→images* with CMP Facades dataset [42], (2) segmentation *labels→images* and *images→labels* with Cityscapes dataset [43], (3) *edges→shoes* with Edge and shoes dataset [44].

We compare our framework with existing methods, pix2pix [4], PAN [45], and the methods designed for high-quality I2IT task, pix2pixHD [17], DRPAN [46], and CHAN [47]. Moreover, we replace MERL with PPO [48], denoted as Ours-PPO. We use PSNR, SSIM, and LPIPS [49] as the evaluation metrics.

Results and Analysis. The quantitative results are shown in Table 3. With a similar network structure, the proposed method significantly outperforms the pix2pix and PAN models on PSNR, SSIM, and LPIPS over all the datasets and tasks. Our method even achieves a comparable or better performance than the high-quality pix2pixHD and DRPAN models, which have much more complex architectures and training strategies. Moreover, using MERL instead of PPO obviously improves performance on most tasks. These experiments illustrate that the proposed RL-I2IT framework is a robust and effective solution for I2IT.

More importantly, our model is much simpler, with the same architecture as pix2pix. The number of parameters and the computational complexity are shown in Table 4. We can see that the RL-I2IT has much fewer parameters and lower computational complexity. We conclude that our model is lightweight, efficient, and effective.

The qualitative results of our RL-I2IT with other I2IT methods on different tasks are shown in Fig. 8. We can observe that pix2pix and PAN sometimes suffer from mode collapse and yield blurry outputs. The pix2pixHD is unstable on different datasets, especially on Facades and Cityscapes. The DRPAN is more likely to produce blurred artifacts in several parts of the predicted image on Cityscapes. In contrast, the RL-I2IT produces more stable

Fig. 7: Visual comparison of different face inpainting methods. GT means ground truth. Our RL-I2IT uses SNGAN for auxiliary learning. # indicates what reward is used for RL training. Our results have good visual quality even for a large pose face.

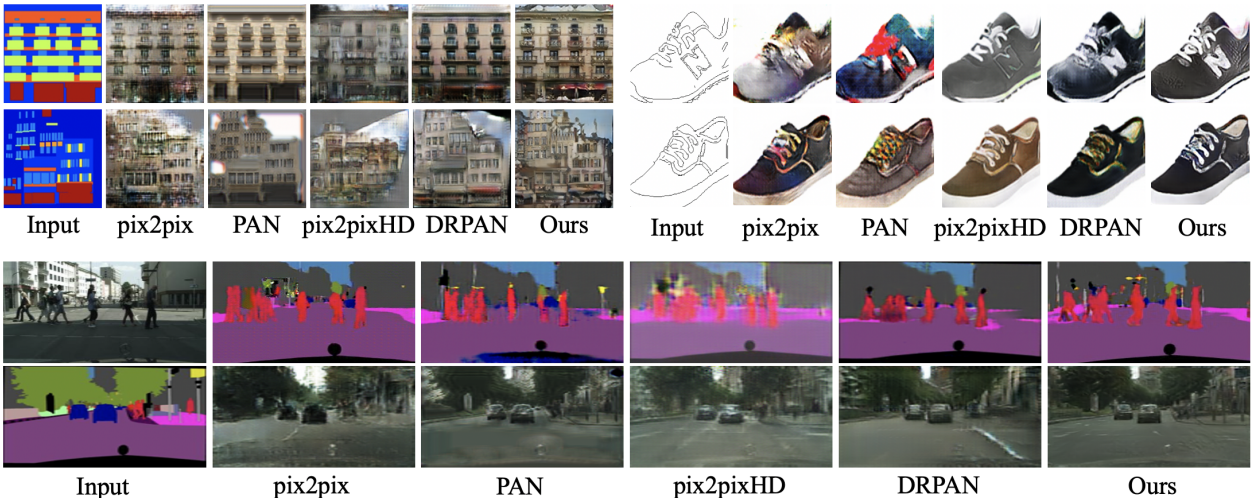

Fig. 8: Visual comparison of our RL-I2IT with pix2pix, PAN, pix2pixHD, and DRPAN over all tasks.

Method	Facades label→image			Cityscapes image→label			Cityscapes label→image			Edges→shoes		
	PSNR	SSIM	LPIPS ↓	PSNR	SSIM	LPIPS ↓	PSNR	SSIM	LPIPS ↓	PSNR	SSIM	LPIPS ↓
pix2pix	12.290	0.225	0.438	15.891	0.457	0.287	15.193	0.279	0.379	15.812	0.625	0.279
PAN	12.779	0.249	0.387	16.317	0.566	0.228	16.408	0.391	0.346	16.097	0.658	0.228
pix2pixHD	12.357	0.162	0.336	17.606	0.581	0.204	15.619	0.361	**0.319**	17.110	0.686	0.220
DRPAN	13.101	0.276	0.354	17.724	0.633	0.214	16.673	0.403	0.343	17.524	**0.713**	0.221
CHAN	13.137	0.231	0.402	17.459	0.641	0.222	16.739	0.401	0.373	18.065	0.692	0.236
Ours-PPO	13.163	**0.308**	0.366	17.168	0.616	0.221	16.685	0.410	0.362	16.914	0.695	0.225
Ours	**13.178**	0.296	**0.324**	**17.969**	**0.659**	**0.203**	**16.848**	**0.412**	0.337	**18.178**	0.698	**0.215**

TABLE 3: Quantitative results of our RL-I2IT and other methods over all datasets. ↓ means lower is better, Ours-PPO means our RL-I2IT using PPO.

Method	pix2pixHD	DRPAN	CHAN	Ours
#Params	45.874M	11.378M	59.971M	**9.730M**
#FLOPs	10.340G	14.208G	19.743G	**3.519G**

TABLE 4: Comparison of the number of parameters and FLOPs (floating point operations, which represent the computational complexity of the model).

and realistic results. Using stochastic meta-policy and MERL helps explore more possible solutions so as to seek out the best generation strategy by trial-and-error in the training steps, leading to a more robust agent for different datasets and tasks.

Evaluation of RL Algorithms. To demonstrate the effectiveness of stochastic meta policy and MERL, we substitute the key components of RL-I2IT with other structures or other state-of-the-art RL algorithms to test their importance. We use DDPG and PPO, respectively. The learning curves of different variants on the four tasks are shown in Fig. 9, which indicates that, by using the stochastic meta policy and the maximum entropy framework, the training process is significantly improved.

4.4 Image Style Transfer

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

The one-step DL approach has an apparent limitation: it is hard to determine a proper level of style for different users since the ultimate metric of style transfer is too subjective. It has been observed that generated stylized images by the current NST

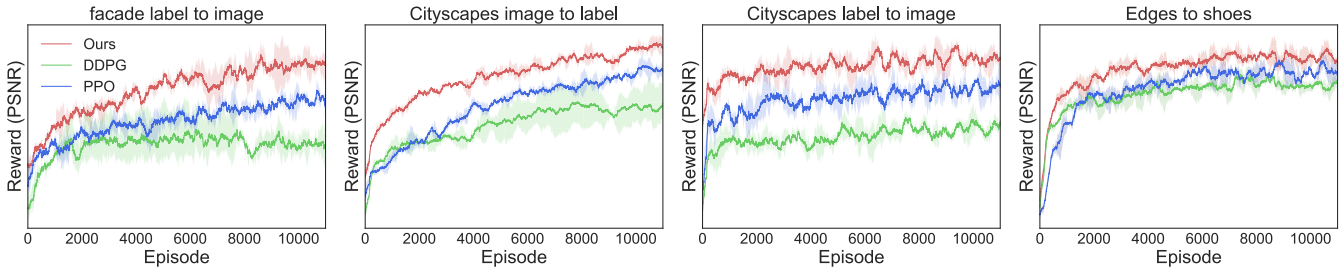

Fig. 9: Learning curves on different I2IT tasks. RL-I2IT performs consistently better than other modified RL algorithms.

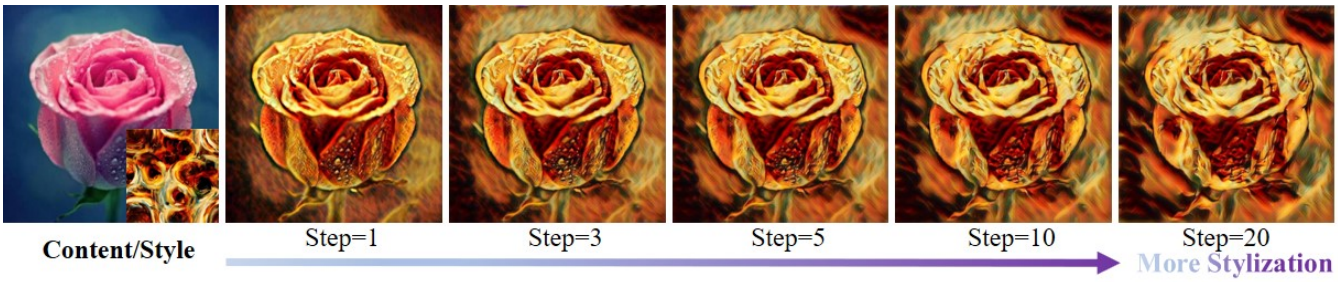

Fig. 10: Illustration of our step-wise style transfer process using the RL-I2IT framework. The content images are stylized stronger with the perdition steps smoothly. The model tends to preserve more details and structures of the content in the early steps and synthesize more style patterns in the later steps. Our step-wise framework allows a user to control the stylization degree easily.

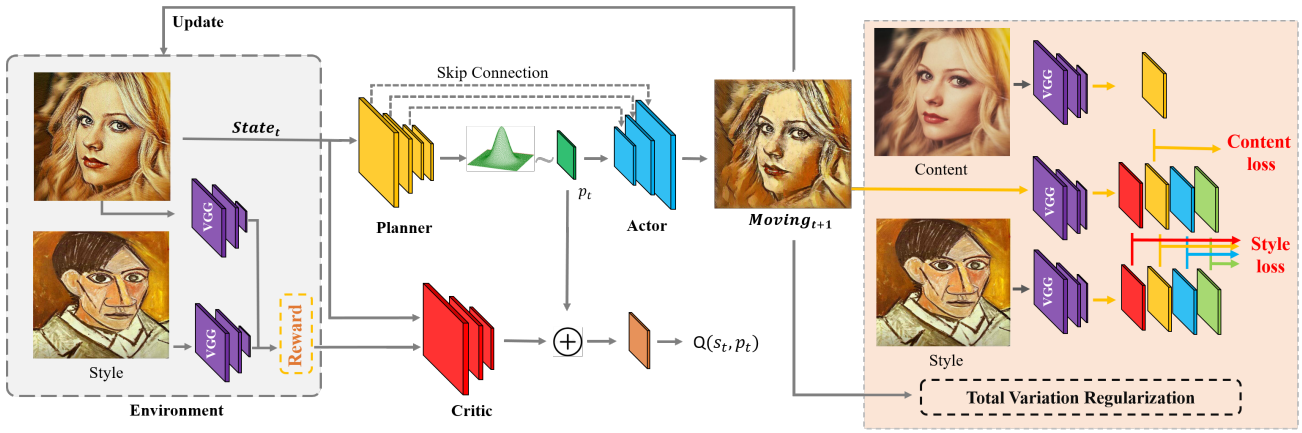

Fig. 11: Details of the RL-I2IT framework for the NST. The state is initialized with the content image. After the first iteration, we use only the moving image as the state. The plan is sampled from a 2D Gaussian distribution and is concatenated with the critic. The predicted moving image is generated by the actor. Note that the VGG networks are pre-trained and fixed for feature extraction during the training process.

methods tend to be under- or over-stylization [50]. A remedy to under-stylization is to use the DL model multiple times, taking the output of the previous round as the input in the current round. However, this may suffer from the high computation cost due to the intrinsic complexity of one-step DL models. Other existing methods, like [23] and [51], play a trade-off between content and style by adjusting hyper-parameters, but this approach is inefficient and hard to control.

Our RL-I2IT framework provides a good solution for NST. It can be used to learn a lightweight NST model that is applied iteratively for NST. To preserve spatial structures of images, the latent plans in our model are sampled from a 2D Gaussian distribution that is estimated by the actor and forwarded to the executor to generate intermediate images. In addition, we develop a Fully Convolutional Network (FCN) based planner-actor structure so that the model can process input images of any size. Fig. 10 shows some examples of our step-wise NST. We can see that our RL-based step-wise method tends to preserve more details and structures of the content image in early steps and synthesize more style patterns in later steps, resulting in a more flexible control of the stylization degree. Furthermore, our model is a lightweight and flexible NST model compared to existing methods, making it more efficient computationally. To the best of our knowledge, this is the first work that successfully leverages RL for the NST scenario.

4.4.1 RL-I2IT Setting

We set the moving image as state $\mathbf{s}_t$, which is initialized by the content image. The moving image at time t, i.e., state image $\mathbf{s}_{t+1}$, is created by the actor and current state image $\mathbf{s}_t$ and plan $\mathbf{p}_t$. The reward is obtained by measuring the difference between the current state $\mathbf{s}_t$ and the style image. The higher the difference is,

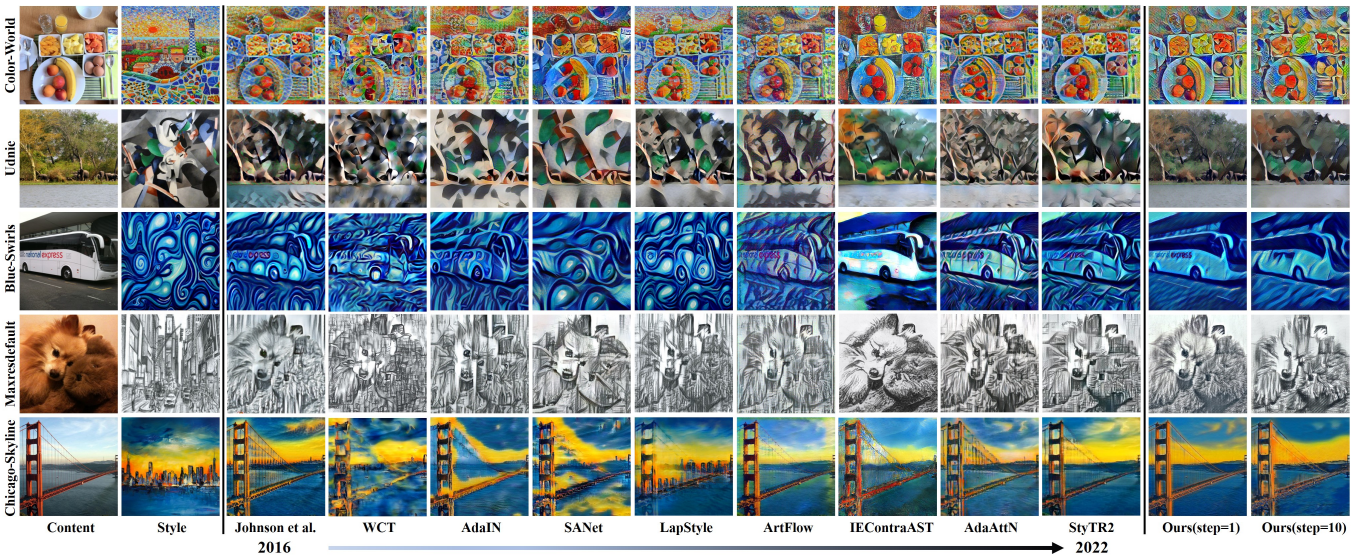

Fig. 12: Qualitative comparison. The first two columns show the content and style images, respectively. The rest of the columns show the stylization results generated with different style transfer methods, including our step-wise results at the rightmost two columns.

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 5: 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. (·×) represents the ratio between current baseline and our method (step=1) under the same metric. The best results are shown in bold.

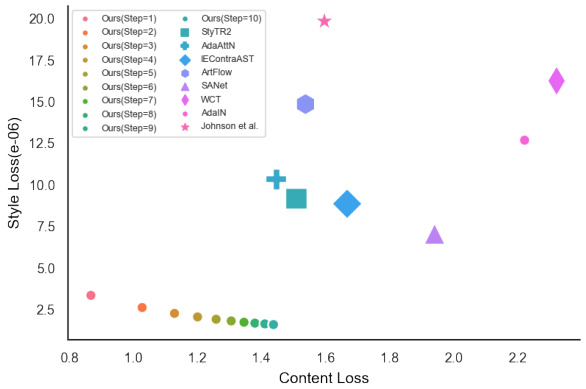

Fig. 13: Comparison of the content loss vs the style loss of different methods on the test dataset (Maxresdefault style), where the loss of ours in 10 steps is plotted by the curve. Closer to (0,0) is better.

the smaller the reward is. We use negative style loss as the reward. The style loss is defined later in this subsection.

The detail of the network architecture is shown in Fig. 11. The planner is a neural network consisting 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. [52]. The planner estimates a 2D Gaussian distribution for sampling our latent plan with the size of 64×64, which is forwarded to the actor to generate the moving image. The actor has three up-sampling layers. We also use three skip connections between the planner and the actor. Our planner-actor is FCN, which can process input images of any size. The critic consists of seven convolutional layers and one fully-connected layer at the end. Since Johnson et al. [53] conclude that using standard zero-padded convolutions in style transfer will lead to serious artifacts on the boundary of the generated image, we use reflection padding instead of zero padding for all the networks.

Style Learning. To make the moving image not deviate from the content image, the model trains the planner and the actor based on collected training data from the agent-environment interaction and changes dynamically in experience replay. More specifically, the planner and actor form a conditional generative process that translates state $\mathbf{s}_t$ to output moving image $\mathbf{m}_t$ 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 [53], 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 planner and actor. These losses can better measure perceptual and semantic differences between the moving image and content image $\mathbf{c}$.

Content Loss $\mathcal{L}^{CO}$. Following [53], 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

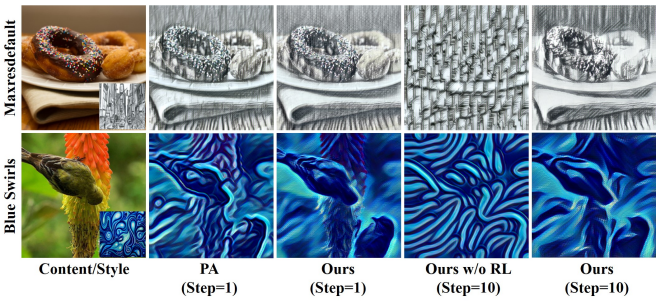

Fig. 14: Comparison of our RL-I2IT framework with the Planner-Actor (PA) model at step 1 and ours without the RL model runs 10 steps.

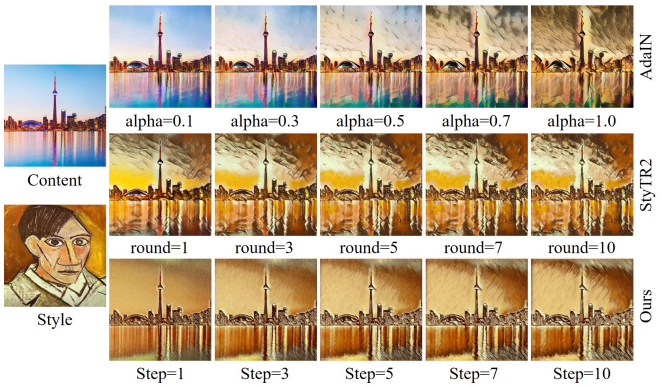

Fig. 15: Comparison of Ours with AdaIN and StyTR2 in various hyperparameter settings.

Style	Step	Method	Content Loss	Style Loss
Maxresdefault	1	Planner-Actor (PA)	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	Planner-Actor (PA)	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 6: Content loss, and style loss of several variants of our proposed method.

$C^j \times H^j \times W^j$, 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 $\mathbf{e}$, we define by following [54] 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 [53], [55].

Putting all components together, the final style loss is

$$\mathcal{L} = \mathcal{L}^{CO} + \lambda \mathcal{L}^{ST} + \beta \mathcal{L}^{TV}, \tag{8}$$

where λ and β are hyperparameters to control the sensitivity of each term.

4.4.2 Experiment

Dataset. We select style images from WikiArt [56] and use MS-COCO [57] as content images. For the latter, the training set includes 80K images, and the test set includes 40K images. All the training images are resized to 256 × 256 pixels. We note that our method at the inference stage is applicable for content images and style images of any size. Following StyTR2 [58], we use content loss, style loss and computing time as the evaluation metrics.

Baselines. We choose several state-of-the-art style transfer methods as our baselines, including Johnson et al. [53], WCT [59], AdaIN [51], SANet [60], LapStyle [61], ArtFlow [62], IEContraAST [63], AdaAttN [64], and StyTR2 [58]. All these methods are performed with their public codes with default settings.

Implementation Details. In the experiment, we set $\lambda = 1e5$, $\beta = 1e-7$ in Eq (8). These settings yield nearly the best performance in our experiments.

Qualitative Comparison. Fig. 12 shows some stylized results of our model and the baseline methods. For content images with fine structures, such as the forest image (Undie style), all the baseline methods, including ArtFlow which is proposed to solve the content leak issue, produce messy stylized images with content structures completely lost. SANet has repeated texture patterns for all the cases, and most of its 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, resulting in smoothed stylization results. More importantly, as we mentioned before, despite the fact that stylization is quite subjective, our step-wise method provides flexible control of different degrees of stylization to fit the needs of different users.

Quantitative Results. To be consistent with all compared methods shown in Fig. 12, we compare our method with all baselines without caring which type (single or multiple styles) they are. The quantitative results are shown in Table 5. 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. To better visualize step-wise results, we also compare the two losses of our model in steps 1-10 with the baseline methods in Fig. 13. It is clear that our model can provide lower style and content losses. In addition, our model boasts significantly fewer parameters and operates at 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. 14, compared with the method that only uses Planner-Actor (PA), our method can generate more stable and clear stylized images at step 1. At step 10, PA loses the content information completely without the help of RL (ours

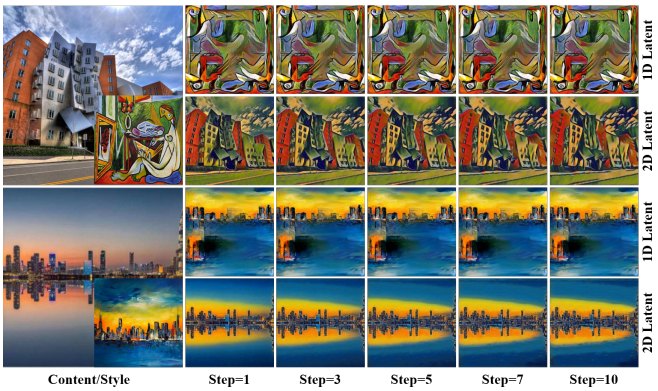

Fig. 16: The necessity of high-dimensional latent space.

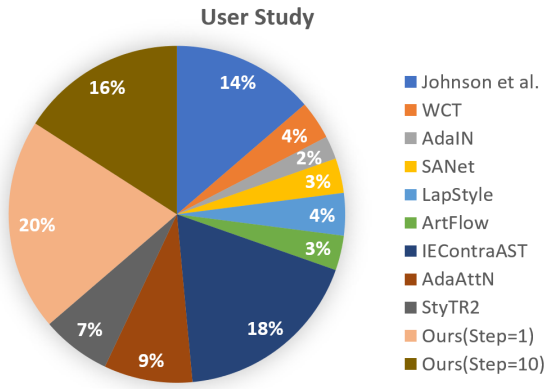

Fig. 17: User preference results of nine competitive methods.

w/o RL), while our method can still produce amazing results. We also show the corresponding quantitative comparison in Table 6. We can easily see that at both steps 1 and 10, our model achieves the best performance consistently. This indicates RL can improve the performance of DL-based NST models. (2) Since 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. 15. 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. (3) We evaluate the necessity of the high-dimension for style transfer, showing in Fig. 16. From the result, it can be seen that 1D latent vector can only generate a single style pattern and cannot retain the semantic information of the content. In contrast, high-dimensional latent tensor preserves the structure and semantic information of content, allowing for simultaneous stylization and content reconstruction.

User Study. We conduct a user survey to collect users' preferences for the results of our method and nine competitive methods. Specifically, we use 5 stylized images and 10 content images in this study and randomly select 5 combinations of content and style. For each combination, we display the stylized images side by side to the subjects, and ask participants to choose the favorite. To reduce the burden on the subjects, our method only shows the results of step=1 and step=10. Finally, we collect 270 votes from 54 users and show the percentage of votes for each method in Fig. 17. In general, the results of our algorithm are favored by the most subjects.

4.5 Video Style Transfer

Similar to image style transfer, video style transfer focuses on converting the visual style within a video sequence, allowing it to exhibit different artistic styles, colors, and appearances while maintaining the fundamental content and structure of objects and scenes in the video. Video style transfer is a complex task as it necessitates considering the temporal continuity and stability of the video, ensuring a smooth transition between frames during the style conversion process to avoid flickering or disjointed effects.

The recent works in the field of video style transfer [64], [64], [65], [66], [67] have been successful. However, these methods are facing the challenge of limited diversity. They can only generate results with a singular degree of stylization, without considering the creation of stylized videos tailored to different audiences with varying degrees of stylization.

To achieve a diverse range of stylization levels in video style transfer, we employ the `RL-I2IT` framework. Specifically, we have made adjustments to the neural network for image style transfer. By using a CNN+RNN architecture [68] for the planner and actor for both frame-wise and step-wise smoothing, our model can perform video NST tasks. Fig. 18 illustrates some examples of our step-wise video Neural Style Transfer. We observe that our `RL-I2IT` framework approach not only retains the advantages seen in image style transfer, namely the tendency to preserve more details and structure of the frame in the early steps while synthesizing more style patterns in the later steps but also generates stable outputs across different degrees of stylization.

4.5.1 *`RL-I2IT` Setting*

In video style transfer, we maintain most of the settings used in image style transfer. The difference lies in initializing the moving images using frames.

Building upon the image style transfer network, we made slight modifications. The network's details are illustrated in Fig. 19. We introduced two GRUs: the step-wise GRU and the frame-wise GRU. The step-wise GRU retains information between steps, ensuring smoother step-wise stylization. Meanwhile, the frame-wise GRU preserves information between frames, enforcing consistency in style patterns among adjacent frames.

Compound Temporal Regularization $\mathcal{L}^{CT}$. Inspired by [67], 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

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

The remaining loss functions remain consistent with those used in image style transfer. Summing up all the components, the final style learning loss is

$$\mathcal{L} = \mathcal{L}^{CO} + \lambda\mathcal{L}^{ST} + \beta\mathcal{L}^{TV} + \zeta\mathcal{L}^{CT}, \tag{9}$$

where λ, β, and ζ are hyper-parameters to control the sensitivity of each term.

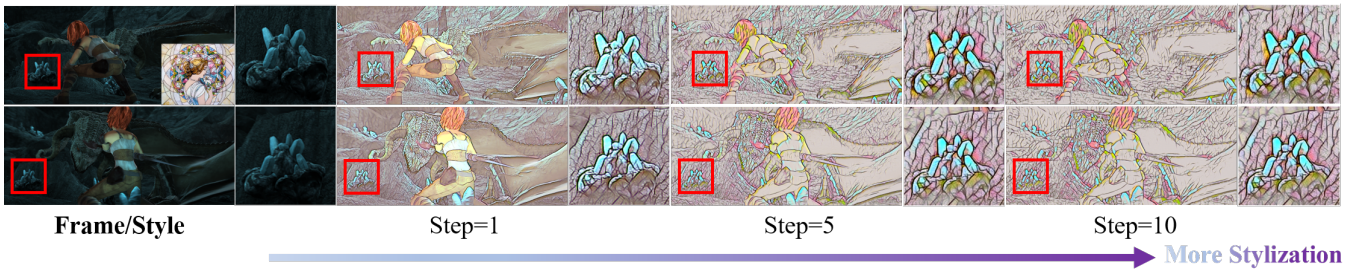

Fig. 18: Illustration of our step-wise video style transfer process using the RL-I2IT framework. The frames are stylized stronger with the perdition steps smoothly. The model tends to preserve more details and structures of the content in the early steps and synthesize more style patterns in the later steps. Our model is capable of generating stable stylized results across different degrees of stylization.

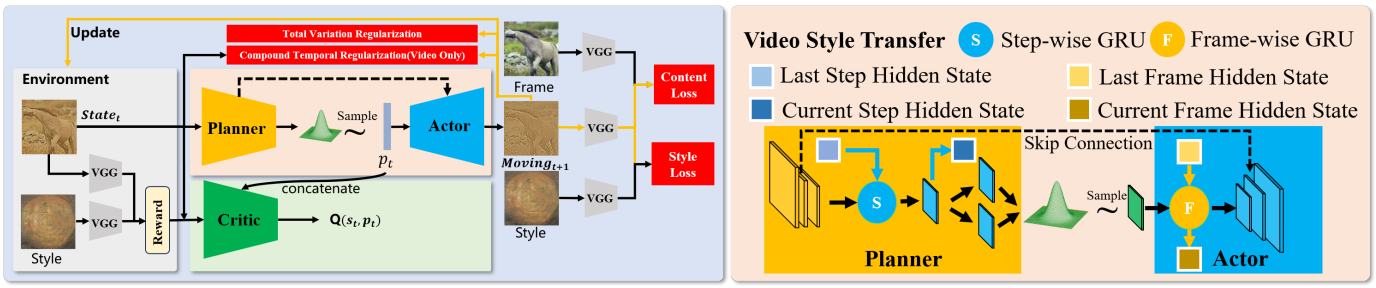

Fig. 19: Details of the RL-I2IT framework for the video NST. The state is initialized with the frame. After the first iteration, we use only the moving image as the state. Note that the VGG networks are pre-trained and fixed for feature extraction during the training process.

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 7: 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.

4.5.2 Experiment

Dataset. For video style transfer, we randomly collect 16 videos of different scenes from pexels [69]. 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 [56]. In addition, following [67], we use the training set of MPI Sintel dataset [70] 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. For video style transfer, we compare our method with the following four popular methods: Linear [65], MCCNet [66], ReReVST [67], and AdaAttN [64]. Following [64], 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 $\lambda = 1e5$, $\beta = 1e-7$, and $\zeta = 1e2$ in Eq. (9). These settings yield nearly the best performance in our experiments. Following [67], 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 $\sigma \sim \mathcal{U}(0.001, 0.002)$.

Qualitative Comparison. We show the visualization results of our method compared with the four latest video style transfer methods in Fig. 20, 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 coherence in some regions that are close to the edge of objects such as the head and shoulder.

Quantitative Results. As shown in Table 7, we choose 23 different sequences from the MPI Sintel dataset [70] and eight different style images to calculate the average of temporal losses for

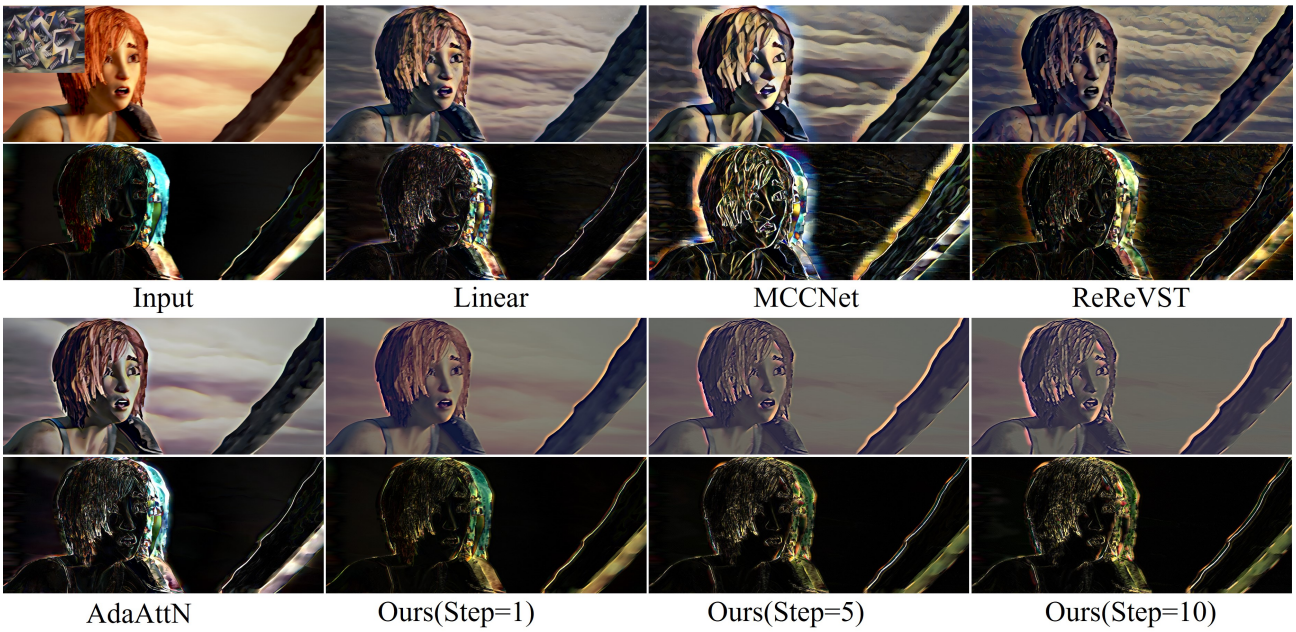

Fig. 20: Comparison of video style transfer between our method and the compared methods. For each method, the top portion shows the video frame stylized results. The bottom portion shows the heatmap of the differences between two adjacent video frames.

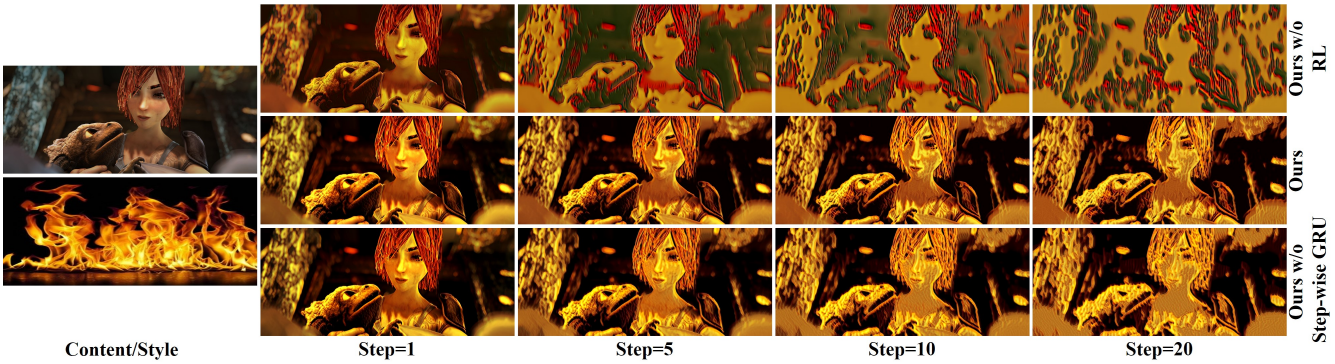

Fig. 21: Comparison of our method, our method without using RL (PA method), and our method without using Step-wise GRU. RL makes the results from our model more stable and Step-wise GRU makes the output has higher quality.

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.

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 8: 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.

(1) As shown in Fig. 21 (first and second rows), our method generates more stable and clearer stylized results than the method using only Planner-Actor without RL. After step 5, PA 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. 21 (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 8 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 Applications on Medical Image

5.1 Deformable Image Registration

In this section, we apply the `RL-I2IT` to deformable image registration (DIR), which is an ill-posed problem formalized as

the optimization of a function balancing the similarity between images and the plausibility of the deformation [22], [32], [71]. Given a pair of images (I_F, I_M), both from the image domain $\mathcal{X} \rightarrow \mathbb{R}^d$, where d is the dimension, where I_F is the fixed image and I_M is the moving image. Denote Ω_w as a registration model parameterized by w, the output of which is a deformation field. Then, the process of aligning the moving image to the fixed image can be written as $I_M \circ \Omega_w(I_F, I_M)$. Then, the pairwise registration is formulated as a minimization problem based on the following energy function:

$$\min_w E(w) := G(I_F, I_M \circ \Omega_w(I_F, I_M)) + \lambda R(\Omega_w(I_F, I_M)) \tag{10}$$

where G represents a distance metric measuring the similarity between the fixed image and the warped image, R represents a regularization constraining the deformation field, λ is a regularization parameter. G can be any distance metric, such as the sum of squared differences (SSD), the normalized mutual information (NMI), or the negative normalized cross-correlation (NCC) [31], [32].

5.1.1 `RL-I2IT` Setting

Instead of predicting the deformation field in one shot as traditional DL-based DIR methods, our framework decomposes the registration task into T steps. Suppose action $\mathbf{a}_t$ is the current deformation field at time step t, which is generated by the planner κ_ψ and actor π_ϕ based on the fixed image I_F and the intermediate moving image I_{M_t}. Let $\Omega^t_{\psi,\phi}$ represent the accumulated deformation field composed by $\mathbf{a}_t$ and the previous deformation field $\Omega^{t-1}_{\psi,\phi}$. We can compute $\Omega^t_{\psi,\phi}$ with a recursive composition function:

$$\Omega^t_{\psi,\phi} = \begin{cases} 0 & \text{if } t = 0, \\ \mathcal{C}(\mathbf{a_t}, \Omega^{t-1}_{\psi,\phi}) & \text{otherwise,} \end{cases} \tag{11}$$

where

$$\mathcal{C}(\mathbf{a_t}, \Omega^{t-1}_{\psi,\phi}) = \Omega^{t-1}_{\psi,\phi} + (\mathbf{a}_t \circ \Omega^{t-1}_{\psi,\phi}). \tag{12}$$

To generate the intermediate moving image $I_{M_{t+1}}$ at time step $t+1$, we warp the initial moving image I_M with the accumulated deformation field $\Omega^t_{\psi,\phi}$, so as to eliminate the warping bias in the multi-step recursive registration process [80]. In this way, the registration result can be progressively improved by predicting the deformation field from coarse to refined. Using the step-wise notion, our `RL-I2IT` framework reformulates the DIR optimization problem (Eq.(10)) as

$$\min_{\psi,\phi} E(\psi, \phi) := \frac{1}{T} \sum_{t=1}^{T} G(I_F, I_{M_t} \circ \Omega^t_{\psi,\phi}) + \lambda R(\Omega^t_{\psi,\phi}), \tag{13}$$

where we use our `RL-I2IT` framework to learn a tuple (ψ, ϕ) instead of the parameter w in Eq.(10).

An overview of the environment of our `RL-I2IT` framework is shown in Fig. 22. In the beginning, the environment contains only an image pair (I_F, I_M), then K-means [82] with three clustering labels is performed for a voxel-wise segmentation in an unsupervised manner. The obtained segmentation maps (U_F, U_M) assign each voxel to a virtual anatomical structure label. At time step t, state $\mathbf{s}_t$ comprises the fixed image I_F and the moving image I_{M_t}, $\mathbf{s}_t = (I_F, I_{M_t})$. The next state s_{t+1} is obtained by warping I_M with the composed deformable field $\Omega^t_{\psi,\phi}$: $s_{t+1} = (I_F, I_M \circ \Omega^t_{\psi,\phi})$, where the warping operator is the popular spatial transformer network (STN) [33]. The reward r_t is defined based on the Dice score [29]:

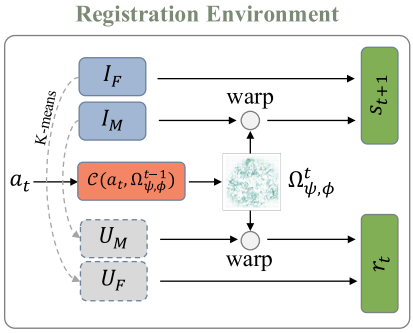

Fig. 22: Illustration of the deformable registration environment of `RL-I2IT`. When the environment receives an action $\mathbf{a}_t$, it outputs the next state s_{t+1} and the reward r_t. Specifically, the environment comprises a pair of images (I_F, I_M) and generates corresponding segmentation maps (U_F, U_M) with the K-means clustering. $\mathcal{C}(a_t, \Omega^{t-1}_{\psi,\phi})$ is the function that applies action a_t to the accumulated deformation field $\Omega^t_{\psi,\phi}$. The next state s_{t+1} is obtained by concatenating I_F and the warped moving image $I_M \circ \Omega^t_{\psi,\phi}$. The reward r_t is obtained by Eq. (14) to evaluate the improvement of the deformation field.

$$r_t = \text{Dice}(U_F, U_M \circ \Omega^t_{\psi,\phi}) - \text{Dice}(U_F, U_M \circ \Omega^{t-1}_{\psi,\phi}), \tag{14}$$

where $\text{Dice}(U_1, U_2) = 2 \cdot \frac{|U_1 \cap U_2|}{|U_1| + |U_2|}$. This reward function explicitly evaluates the improvement of the predicted deformation field $\Omega^t_{\psi,\phi}$.

Auxiliary Learning. After getting action $\mathbf{a}_t$ and following meta policy (κ_ψ, π_ϕ), we can obtain $\Omega^t_{\psi,\phi}$ based on Eq. (11). We use the local normalized cross-correlation (NCC) [32] to measure the similarity between the fixed image and the warped moving image: $G(I_F, I_{M_t}) = NCC(I_F, I_M \circ \Omega^t_{\psi,\phi})$, where a higher NCC indicates a better alignment.

Moreover, in order to generate a realistic warped moving image, we smooth the deformation field by using total variation regularizer [83]: $R(\Omega^t_{\psi,\phi}) = \|\nabla \Omega^t_{\psi,\phi}\|_2^2$. The final registration loss for the auxiliary learning J_{Aux} is then defined as

$$J_{Aux}(\psi, \phi) = \mathbb{E}_{s_t \sim \mathcal{D}}[-NCC(I_F, I_M \circ \Omega^t_{\psi,\phi}) + \lambda \|\nabla \Omega^t_{\psi,\phi}(\mathbf{s}_t)\|_2^2].$$

5.1.2 Experiment

In this section, we evaluate our `RL-I2IT` framework on the 2D and 3D medical image registration tasks.

Datasets. For the 2D registration, we use 2,302 pre-processed 2D scans from ADNI [84], ABIDE [85], and ADHD [86] for training and apply K-Means to obtain corresponding voxel-wise segmentation maps. 40 pre-processed slices from LONI Probabilistic Brain Atlas (LPBA) [87] are used for the evaluation, each of which contains the ground truth of a segmentation map with 56 manually delineated anatomical structures. All images are resampled to 128×128 pixels. The first slice of LPBA is used

Method	2D Registration LPBA	Time(s)	#Params	3D Registration SLIVER	LSPIG	Time(s)	#Params
SyN [72]	55.47±3.96	4.57	-	89.57±3.34	81.83±8.30	269	-
Elastix [73]	53.64±3.97	2.20	-	90.23±2.39	81.19±7.47	87.0	-
LDDMM [74]	52.18±3.48	3.27	-	83.94±3.44	82.33±7.14	41.4	-
VM [75]	55.36±3.94	0.02	105K	86.37±4.15	81.13±7.28	0.13	356K
VM-diff [76]	55.88±3.78	0.02	118K	87.24±3.26	81.38±7.21	0.16	396K
R2N2 [77]	51.84±3.30	0.46	3,591K	-	-	-	-
GMFlow [78]	52.52±1.90	0.05	468k	-	-	-	-
COTR [79]	52.53±1.89	2312.29	1838k	-	-	-	-
RCN [80]	-	-	-	89.59±3.18	82.87±5.69	2.44	21,291K
SYMNet [81]	-	-	-	86.97±3.82	82.78±7.20	0.18	1,124K
RL-I2IT (t=20, SSIM reward)	56.43±3.76	0.16	107K	90.27±3.85	83.69±6.74	1.05	458K
RL-I2IT (t=1, Dice reward)	55.21±3.55	0.02	107K	84.81±4.42	80.61±7.94	0.07	458K
RL-I2IT (t=10, Dice reward)	56.12±3.68	0.08	107K	90.01±3.79	**84.67±6.05**	0.55	458K
RL-I2IT (t=20, Dice reward)	**56.57±3.71**	0.16	107K	**90.28±3.66**	84.40±6.24	1.05	458K

TABLE 9: The Dice score (%) results of our RL-I2IT (t indicates the t-th step) and the baseline methods. The execution time for the 3D registration is tested on the SLIVER dataset. Note that R2N2, GMFlow, and COTR work only for the 2D registration, and RCN and SYMNet are only for the 3D registration.

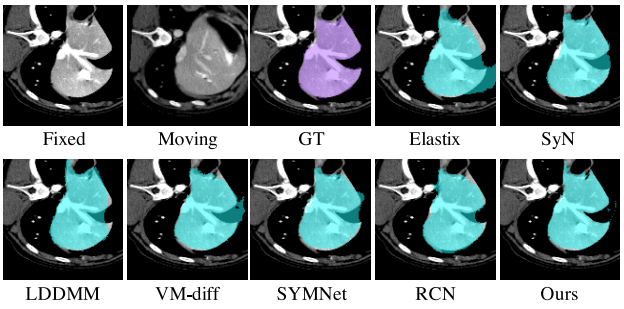

Fig. 23: Visual results of our RL-I2IT and the baseline methods on the LSPIG 3D liver dataset. The warped moving image obtained by our RL-I2IT is more similar to the ground truth.

as the atlas, and the remaining images are used as the moving images.

For the 3D registration, we use Liver Tumor Segmentation (LiTS) [88] challenge data for training, which contains 131 CT scans with the annotated segmentation ground truth. The SLIVER [89] dataset has 20 scans with liver segmentation ground truth, half of which are used as the testing data. To further evaluate the generalization capability of our model, we conduct a cross-subjects experiment: the model is trained with the human liver dataset but tested with the pig liver data set. Concretely, we use the same model trained on the LiTS dataset and test the model on the challenging Liver Segmentation of Pigs (LSPIG) [80] dataset, which contains 17 paired CT scans from pigs and the corresponding liver segmentation ground truth. All 3D volumes are resampled to $128 \times 128 \times 128$ pixels and pre-affined as a standard pre-processing step in the DIR task.

Baselines We compare our method with seven state-of-the-art DL-based DIR methods: VoxelMorph (VM) [75], VM-diff [76], SYMNet [81], R2N2 [77], RCN [80], GMFlow [78], COTR [79], and two top-performing conventional registration algorithms, SyN [72] and Elastix [73] with B-Spline [90]. VM uses a U-Net structure with NCC loss to learn deformable registration, and VM-diff is its probabilistic diffeomorphic variant. SYMNet is a one-shot 3D registration method. R2N2 and RCN are multi-step methods for the 2D and 3D registration, respectively. Both GMflow and COTR can only deal with 2D image registration. GMFlow is an optical flow estimation method. COTR is an image-matching method using a coordinate query. For a fair comparison, we use the same network structure as VM in our RL-I2IT framework. The Dice score is used as the reward function and evaluation metric. To evaluate the robustness of our RL-I2IT framework, we also provide the registration results of using SSIM as the reward function.

Results. Table 9 summarizes the performance of our method and the baseline methods. We can see that our RL-I2IT outperforms the baseline methods over all the cases. The experimental results on the demanding LSPIG can better reflect the strength of our method. The LSPIG dataset has large deformation fields and is quite different from the training dataset (LiTS) in terms of structure and appearance. The good performance of our method on LSPIG shows that the RL-I2IT framework can handle large deformation and has better generalizability than conventional DL-based methods. Note that RL-I2IT performs registration step by step. It is slower than most one-step methods, such as VM and SYMNet but is still faster than other multi-step methods, such as R2N2 and RCN. Compared with the RL-I2IT using the Dice reward, the RL-I2ITusing the SSIM reward achieves comparable results on SLIVER but slightly worse on the LSPIG dataset. Fig. 23 visualizes a registration result on the LSPIG dataset by overlaying the warped moving segmentation map on the fixed image. This result shows that our model successfully learns 3D registration even when encountering a large deformation field and facing a large discrepancy between the training and the testing. Moreover, the RL-I2IT outperforms the two step-wise methods, RCN and R2N2, which demonstrates the effectiveness of our framework.

5.1.3 Analysis

Step-wise registration. The key idea of our method is to decompose the monolithic registration process into small steps using a lightweight CNN model and progressively improve the transformed results. Fig. 24 shows an example of the step-wise registration process. The first row visualizes deformation fields that are predicted by our method in a step-wise and coarse-to-fine manner. In Fig. 25, we compare our method with PPO [48] and DL-based methods, such as VM-dff, RCN, and R2N2, using step-wise registration. As the step increases, the performance of DL-based methods becomes worse, while the RL-based methods are quite stable. In addition, the Dice score of RL-I2IT increases all the time on both LPBA and SLIVER datasets.

Compare with other RL methods. To demonstrate the effectiveness of our method on the reinforcement learning side,

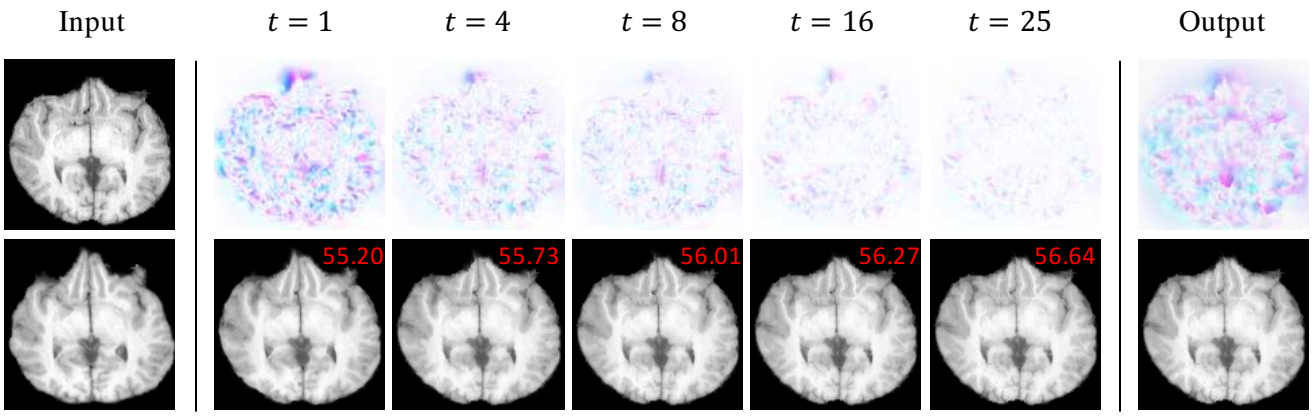

Fig. 24: A step-wise registration example of RL-I2IT on the LPBA dataset. The first row is the visualized displacement field, where deep color represents a large deformation. The red number at the top right corner is the Dice score.

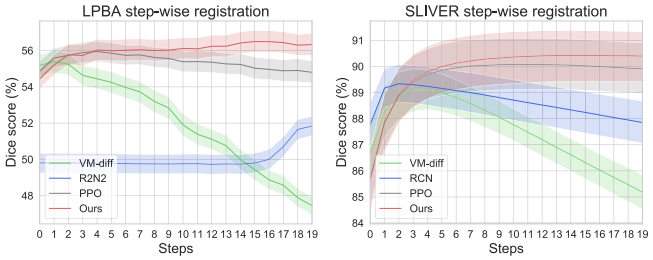

Fig. 25: The step-wise registration results. The RL-based methods (our RL-I2IT and PPO) perform more stably than the DL-based methods, and our RL-I2IT achieves the best performance.

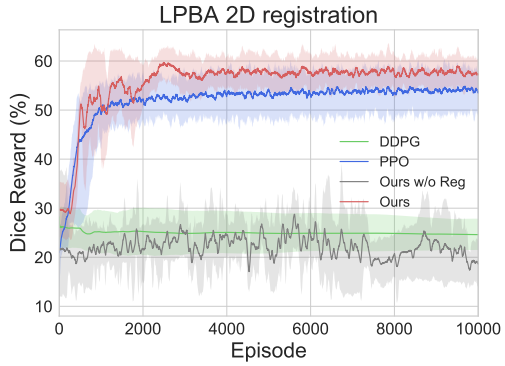

Fig. 26: Learning curves of several RL-based methods on the LPBA dataset.

we modify our framework with other popular RL models such as PPO [48] and DDPG [28] in the Planner-Critic learning process. We also compare with our variant that discards the DL-based unsupervised registration loss (RL-I2IT w/o Reg). The quantitative results are shown in Table 10, and their training curves are shown in Fig. 26. We can see that RL-I2IT w/o Reg and DDPG, which use deterministic policy, fail to converge. This indicates that the RL agent can hardly deal with the DIR problem without the unsupervised registration loss. In addition, our RL-I2IT achieves better performance than PPO.

Ablation Experiments. Some important components in our framework, such as reinforcement learning, unsupervised registration learning, and the evaluating plan with the critic, are analyzed. Note that when the registration loss is discarded, the deformation field is practically the only action, and both the planner and actor

	LPBA	SLIVER	LSPIG
PPO-modified	55.82±3.49	89.30±3.63	83.55±6.24
RL-I2IT-action	55.58±3.70	88.75±3.69	81.80±7.51
RL-I2IT w/o RL	54.89±3.80	85.43±4.14	80.72±7.34
RL-I2IT w/o Reg	44.67±3.74	79.34±4.02	72.45±6.25
RL-I2IT	**56.57±3.71**	**90.28±3.66**	**84.40±6.24**

TABLE 10: The Dice scores (%) of several variants of RL-I2IT. 'RL-I2IT-action' indicates that the critic evaluates the actor's action instead of the planner in RL-I2IT.

Method	SLIVER		LSPIG	
	mean($\lvert J_\phi \rvert$)	std($\lvert J_\phi \rvert$)	mean($\lvert J_\phi \rvert$)	std($\lvert J_\phi \rvert$)
VM	0.9263	0.0106	0.9204	0.0112
RCN	0.8066	0.0906	0.7183	0.1126
RL-I2IT (t=1)	0.9545	0.0084	0.9637	0.0110
RL-I2IT (t=10)	0.9176	0.0160	0.9306	0.0202
RL-I2IT (t=20)	0.8631	0.0376	0.8951	0.0334

TABLE 11: Quantitative results of the Jacobian determinants.

are trained with the RL objective. As summarized in Table 10, the result is unsatisfactory if we train RL-I2IT without reinforcement learning, and it becomes worse if the training discards unsupervised registration loss. When the critic evaluates the actor's action (RL-I2IT-action), it results in an inferior performance as compared with the RL-I2IT when the critic evaluates the planner.

We also use the Jacobian determinant to assess the regularity of the predicted displacement field. The results are shown in Table 11. A small standard deviation of the Jacobian determinant indicates a smooth displacement. We can see from the table that our deformation fields are plausible and smooth. Furthermore, we are the first to use SSIM as a reward function to perform registration. The comparison between using SSIM and the Dice score as the reward is shown in Table 12. Despite using SSIM, which can perform well compared with other methods, its overall performance of all steps is still inferior to the Dice reward, as we can see from Table 12. The tradeoff between the Dice score and the inference time is shown in Fig. 27. We can see that the proposed RL-I2IT achieves a better tradeoff between registration performance and computational efficiency.

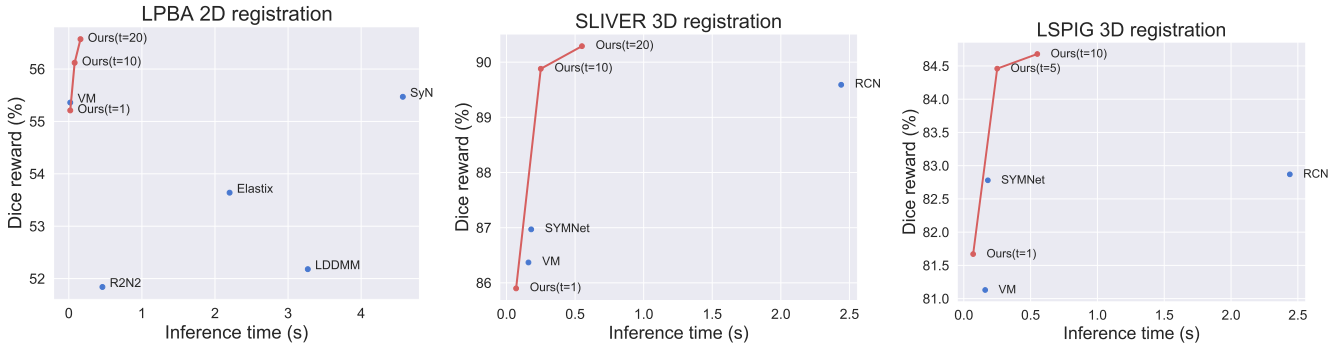

Fig. 27: Trade-off between Dice score and inference time over all datasets.

Reward	SLIVER			LSPIG		
	t=1	t=10	t=20	t=1	t=10	t=20
SSIM	83.69	89.71	90.27	79.14	83.80	83.69
DICE	**84.81**	**90.01**	**90.28**	**80.61**	**84.67**	**84.40**

TABLE 12: Comparison of SSIM and segmentation rewards.

6 Conclusion

In this paper, we propose a reinforcement learning-based framework, RL-I2IT, to handle the I2IT problem. Our RL-I2IT framework is an off-policy planner-actor-critic model. It can efficiently learn good policies in spaces with high-dimensional continuous states and actions. The core component in RL-I2IT is the proposed meta policy with a new component 'plan', which is defined in latent subspace and can guide the actor to generate high-dimensional executable actions. To the best of our knowledge, we are the first to propose an RL framework for the I2IT problem. Experiments based on diverse applications demonstrate that this architecture achieves significant gains over existing state-of-the-art methods.

There are several potential limitations in our proposed framework. One potential limitation is that our framework can perform only single-style NST tasks. Arbitrary style transfer methods usually use a pre-training model to extract depth features, while our current RL-based framework directly interacts with the current state instead of using pre-extracted features as input. This difference limits our current framework, which cannot perform arbitrary style transfer. However, the main goal of the NST task in this paper 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. Using a single-style NST model is sufficient to serve the purpose. That said, we can extend the current framework to support arbitrary style transfer by observing the depth features of the state. Another potential limitation is that the number of steps of the testing process is a predefined hyper-parameter, which can be improved by learning from the model automatically.

In the future, we will try to address the aforementioned limitations of our proposed framework. We expect that the proposed architecture can potentially be extended to all I2IT tasks.

Appendix A
Learning with Critic on Actor

When the critic is used to evaluate the actor, the rewards and the soft Q values are used to guide the stochastic policy improvement iteratively, where the $\mathbf{a}_t$ is concatenated on the state $\mathbf{s}_t$ as the input of the critic. In evaluation step, follow SAC [12], RL-I2IT learns the actor π_ϕ and fits the parametric Q-function $Q_\theta(\mathbf{s}_t, \mathbf{a}_t)$ (critic) using transitions sampled from the replay pool $\mathcal{D}$ by minimizing the soft Bellman residual,

$$J_Q(\theta) = \mathbb{E}_{\mathcal{D}}\left[\frac{1}{2}\Big(Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - \big(r_t + \gamma \mathbb{E}\left[V_{\bar{\theta}}(\mathbf{s}_{t+1})\right]\big)\Big)^2\right], \tag{15}$$

where $V_{\bar{\theta}}(\mathbf{s}_t) = \mathbb{E}_{\mathbf{a}_t \sim \pi_\phi}[Q_{\bar{\theta}}(\mathbf{s}_t, \mathbf{a}_t) - \alpha \log \pi_\phi(\mathbf{a}_t|\mathbf{p}_t)]$. γ is the discount factor. We use a target network $Q_{\bar{\theta}}$ to stabilize training, whose parameters $\bar{\theta}$ are obtained by an exponentially moving average of parameters of the critic network [28]: $\bar{\theta} \rightarrow \tau\theta + (1-\tau)\bar{\theta}$. The hyper-parameter $\tau \in [0, 1]$. To optimize the $J_Q(\theta)$, we can do the stochastic gradient descent [12] with respect to the parameters θ as follows,

$$\begin{aligned}\theta = \theta - \eta_Q \nabla_\theta Q_\theta(\mathbf{s}_t, \mathbf{a}_t)\Big(&Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - r_t \\ &- \gamma\left[Q_{\bar{\theta}}(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}) - \alpha \log \pi_\phi(\mathbf{a}_{t+1}|\mathbf{p}_{t+1})\right]\Big).\end{aligned} \tag{16}$$

Since the critic works on the actor, the optimization procedure will also influence the planner's decisions. Therefore, the improvement step attempts to optimize the actor and the planner parameters ϕ, ψ. Following [12], we can use the following objective to minimize the KL divergence between the policy and a Boltzmann distribution induced by the Q-function,

$$\begin{aligned}J_{\kappa,\pi}(\psi, \phi) =& \mathbb{E}_{\mathcal{D}}\left[\alpha \log(\pi_\phi(\mathbf{a}_t|\mathbf{p}_t)) - Q_\theta(\mathbf{s}_t, \mathbf{a}_t)\right] \\ =& \mathbb{E}_{\mathcal{D}}\left[\alpha \log(\pi_\phi(\mathbf{a}_t|f_\psi(\epsilon_t, \mathbf{s}_t))) - Q_\theta(\mathbf{s}_t, \mathbf{a}_t)\right].\end{aligned} \tag{17}$$

The last equation holds because $\mathbf{p}_t$ can be replaced by $f_\psi(\epsilon_t, \mathbf{s}_t)$ as we discussed before. It should be mentioned that the hyperparameter α can be automatically adjusted by using one proposed method from [12]. Then we can apply the stochastic gradient method to optimize parameters as follows,

$$\psi = \psi - \eta_\psi \frac{\alpha \nabla_{\mathbf{p}_t} \pi_\phi(\mathbf{a}_t|\mathbf{p}_t) \cdot \nabla_\psi f_\psi(\epsilon_t, \mathbf{s}_t)}{\pi_\phi(\mathbf{a}_t|\mathbf{p}_t)}, \tag{18}$$

$$\phi = \phi - \eta_\phi \frac{\alpha \nabla_{\mathbf{a}_t} \pi_\phi(\mathbf{a}_t|\mathbf{p}_t)}{\pi_\phi(\mathbf{a}_t|\mathbf{p}_t)}. \tag{19}$$

Appendix B
Meta Policy with Skip Connections

Like in a MERL model, the stochastic meta policy and maximum entropy in our framework improve the exploration for more diverse generation possibilities, which helps to prevent agents from

producing a single type of plausible output during training (known as mode-collapse).

One specific characteristic in our framework is that we also add skip-connections from each down-sampling layer of the planner to the corresponding up-sampling layer of the actor, as shown in Fig. 2. In this way, a natural-looking image is more likely to be reconstructed since the details of state $\mathbf{s}_t$ can be passed to the actor by skip-connections. Besides, since both $\mathbf{p}_t$ and $\mathbf{s}_t$ can be used by the actor to generate the executable action $\mathbf{a}_t$, over-exploration of the action space can be avoided in our `RL-I2IT` framework, where the variance is limited by the passed detail information.

Furthermore, the skip-connections also facilitate back-propagation of the gradients of the auxiliary learning part to the actor. It is also a key point to accelerate and stabilize training and avoid over-exploration since it helps the actor to focus on the refined details to bypass the coarse information from input to target.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grants 61602065 and 42130608, Sichuan province Key Technology Research and Development project under Grant 2021YFG0038.

Xin Wang is supported by University at Albany, SUNY Start-up Grant.

REFERENCES

[1] Y. Pang, J. Lin, T. Qin, and Z. Chen, "Image-to-image translation: Methods and applications," *arXiv preprint arXiv:2101.08629*, 2021.

[2] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *ICLR*, 2014.

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[4] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[5] C. Saharia, W. Chan, H. Chang, C. Lee, J. Ho, T. Salimans, D. Fleet, and M. Norouzi, "Palette: Image-to-image diffusion models," in *ACM SIGGRAPH 2022 Conference Proceedings*, 2022, pp. 1–10.

[6] S. Sun, L. Wei, J. Xing, J. Jia, and Q. Tian, "Sddm: score-decomposed diffusion models on manifolds for unpaired image-to-image translation," in *International Conference on Machine Learning*. PMLR, 2023, pp. 33 115–33 134.

[7] N. Tumanyan, M. Geyer, S. Bagon, and T. Dekel, "Plug-and-play diffusion features for text-driven image-to-image translation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1921–1930.

[8] B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro, "Exploring generalization in deep learning," *arXiv preprint arXiv:1706.08947*, 2017.

[9] J. C. Caicedo and S. Lazebnik, "Active object localization with deep reinforcement learning," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2488–2496.

[10] J. Hu, Z. Luo, X. Wang, S. Sun, Y. Yin, K. Cao, Q. Song, S. Lyu, and X. Wu, "End-to-end multimodal image registration via reinforcement learning," *Medical Image Analysis*, vol. 68, p. 101878, 2021.

[11] Z. Luo, X. Wang, X. Wu, Y. Yin, K. Cao, Q. Song, and J. Hu, "A spatiotemporal agent for robust multimodal registration," *IEEE Access*, vol. 8, pp. 75 347–75 358, 2020.

[12] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[13] Y. Xiang, X. Wang, S. Hu, B. Zhu, X. Huang, X. Wu, and S. Lyu, "Rmbench: Benchmarking deep reinforcement learning for robotic manipulator control," *IEEE/RSJ International Conference on Intelligent Robots (IROS)*, 2023.

[14] D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus, "Improving sample efficiency in model-free reinforcement learning from images," *arXiv preprint arXiv:1910.01741*, 2019.

[15] A. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine, "Visual reinforcement learning with imagined goals," *arXiv preprint arXiv:1807.04742*, 2018.

[16] K. Black, M. Janner, Y. Du, I. Kostrikov, and S. Levine, "Training diffusion models with reinforcement learning," *arXiv preprint arXiv:2305.13301*, 2023.

[17] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8798–8807.

[18] Z. Luo, J. Hu, X. Wang, S. Lyu, B. Kong, Y. Yin, Q. Song, and X. Wu, "Stochastic actor-executor-critic for image-to-image translation," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 2021, pp. 2775–2781.

[19] C. Feng, J. Hu, X. Wang, S. Hu, B. Zhu, X. Wu, H. Zhu, and S. Lyu, "Controlling neural style transfer with deep reinforcement learning," *International Joint Conference on Artificial Intelligence (IJCAI)*, 2023.

[20] Z. Luo, J. Hu, X. Wang, S. Hu, B. Kong, Y. Yin, Q. Song, X. Wu, and S. Lyu, "Stochastic planner-actor-critic for unsupervised deformable image registration," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 2, 2022, pp. 1917–1925.

[21] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2536–2544.

[22] B. D. de Vos, F. F. Berendsen, M. A. Viergever, H. Sokooti, M. Staring, and I. Išgum, "A deep learning framework for unsupervised affine and deformable image registration," *Medical image analysis*, vol. 52, pp. 128–143, 2019.

[23] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," *arXiv preprint arXiv:1508.06576*, 2015.

[24] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5505–5514.

[25] B. Li, K. Xue, B. Liu, and Y.-K. Lai, "Bbdm: Image-to-image translation with brownian bridge diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1952–1961.

[26] A. X. Lee, A. Nagabandi, P. Abbeel, and S. Levine, "Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model," in *Neural Information Processing Systems (NeurIPS)*, 2020.

[27] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[28] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[29] L. R. Dice, "Measures of the amount of ecologic association between species," *Ecology*, vol. 26, no. 3, pp. 297–302, 1945.

[30] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[31] G. Haskins, U. Kruger, and P. Yan, "Deep learning in medical image registration: a survey," *Machine Vision and Applications*, vol. 31, no. 1, pp. 1–18, 2020.

[32] G. Balakrishnan, A. Zhao, M. R. Sabuncu, J. Guttag, and A. V. Dalca, "An unsupervised learning model for deformable medical image registration," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9252–9260.

[33] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," *arXiv preprint arXiv:1506.02025*, 2015.

[34] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *arXiv preprint arXiv:1802.05957*, 2018.

[35] Y. Zeng, J. Fu, H. Chao, and B. Guo, "Learning pyramid-context encoder network for high-quality image inpainting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1486–1494.

[36] C. Zheng, T.-J. Cham, and J. Cai, "Pluralistic image completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1438–1447.

[37] T. Yu, Z. Guo, X. Jin, S. Wu, Z. Chen, W. Li, Z. Zhang, and S. Liu, "Region normalization for image inpainting." in *AAAI*, 2020, pp. 12 733–12 740.

[38] Z. Yan, X. Li, M. Li, W. Zuo, and S. Shan, "Shift-net: Image inpainting via deep feature rearrangement," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 1–17.

[39] G. Daras, J. Dean, A. Jalal, and A. G. Dimakis, "Intermediate layer optimization for inverse problems using deep generative models," *arXiv preprint arXiv:2102.07364*, 2021.

[40] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein gans," *Advances in neural information processing systems*, 2017.

[41] A. Jolicoeur-Martineau, "The relativistic discriminator: a key element missing from standard gan," *arXiv preprint arXiv:1807.00734*, 2018.

[42] R. Tyleček and R. Šára, "Spatial pattern templates for recognition of objects with regular structure," in *German Conference on Pattern Recognition*. Springer, 2013, pp. 364–374.

[43] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[44] A. Yu and K. Grauman, "Fine-grained visual comparisons with local learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 192–199.

[45] C. Wang, C. Xu, C. Wang, and D. Tao, "Perceptual adversarial networks for image-to-image transformation," *IEEE Transactions on Image Processing*, vol. 27, no. 8, pp. 4066–4079, 2018.

[46] C. Wang, W. Niu, Y. Jiang, H. Zheng, Z. Yu, Z. Gu, and B. Zheng, "Discriminative region proposal adversarial network for high-quality image-to-image translation," *International Journal of Computer Vision*, 2019.

[47] F. Gao, X. Xu, J. Yu, M. Shang, X. Li, and D. Tao, "Complementary, heterogeneous and adversarial networks for image-to-image translation," *IEEE Transactions on Image Processing*, vol. 30, pp. 3487–3498, 2021.

[48] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[49] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595.

[50] J. Cheng, A. Jaiswal, Y. Wu, P. Natarajan, and P. Natarajan, "Style-aware normalized loss for improving arbitrary style transfer," in *CVPR*, 2021.

[51] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *CVPR*, 2017, pp. 1501–1510.

[52] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[53] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *ECCV*. Springer, 2016.

[54] L. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," *NeurIPS*, vol. 28, pp. 262–270, 2015.

[55] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *CVPR*, 2015, pp. 5188–5196.

[56] F. Phillips and B. Mackintosh, "Wiki art gallery, inc.: A case for critical thinking," *Issues in Accounting Education*, vol. 26, no. 3, pp. 593–608, 2011.

[57] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*. Springer, 2014, pp. 740–755.

[58] Y. Deng, F. Tang, X. Pan, W. Dong, C. Ma, and C. Xu, "Stytr^ 2: Unbiased image style transfer with transformers," *arXiv preprint arXiv:2105.14576*, 2021.

[59] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang, "Universal style transfer via feature transforms," *Advances in neural information processing systems*, vol. 30, 2017.

[60] D. Y. Park and K. H. Lee, "Arbitrary style transfer with style-attentional networks," in *CVPR*, 2019, pp. 5880–5888.

[61] T. Lin, Z. Ma, F. Li, D. He, X. Li, E. Ding, N. Wang, J. Li, and X. Gao, "Drafting and revision: Laplacian pyramid network for fast high-quality artistic style transfer," in *CVPR*, 2021, pp. 5141–5150.

[62] J. An, S. Huang, Y. Song, D. Dou, W. Liu, and J. Luo, "Artflow: Unbiased image style transfer via reversible neural flows," in *CVPR*, 2021, pp. 862–871.

[63] H. Chen, Z. Wang, H. Zhang, Z. Zuo, A. Li, W. Xing, D. Lu *et al.*, "Artistic style transfer with internal-external learning and contrastive learning," *NeurIPS*, vol. 34, 2021.

[64] S. Liu, T. Lin, D. He, F. Li, M. Wang, X. Li, Z. Sun, Q. Li, and E. Ding, "Adaattn: Revisit attention mechanism in arbitrary neural style transfer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6649–6658.

[65] X. Li, S. Liu, J. Kautz, and M.-H. Yang, "Learning linear transformations for fast image and video style transfer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3809–3817.

[66] Y. Deng, F. Tang, W. Dong, H. Huang, C. Ma, and C. Xu, "Arbitrary video style transfer via multi-channel correlation," *AAAI*, 2021.

[67] W. Wang, S. Yang, J. Xu, and J. Liu, "Consistent video style transfer via relaxation and regularization," *IEEE Transactions on Image Processing*, vol. 29, pp. 9125–9139, 2020.

[68] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. J. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu *et al.*, "Learning to navigate in complex environments," *ICLR*, 2017.

[69] "Pexels," https://www.pexels.com/, 2022, accessed: 2022-03-12.

[70] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *ECCV*. Springer, 2012, pp. 611–625.

[71] X. Yang, R. Kwitt, and M. Niethammer, "Fast predictive image registration," in *Deep Learning and Data Labeling for Medical Applications*. Springer, 2016, pp. 48–57.

[72] B. B. Avants, C. L. Epstein, M. Grossman, and J. C. Gee, "Symmetric diffeomorphic image registration with cross-correlation: evaluating automated labeling of elderly and neurodegenerative brain," *Medical image analysis*, vol. 12, no. 1, pp. 26–41, 2008.

[73] S. Klein, M. Staring, K. Murphy, M. A. Viergever, and J. P. Pluim, "Elastix: a toolbox for intensity-based medical image registration," *IEEE transactions on medical imaging*, vol. 29, no. 1, pp. 196–205, 2009.

[74] M. F. Beg, M. I. Miller, A. Trouvé, and L. Younes, "Computing large deformation metric mappings via geodesic flows of diffeomorphisms," *International journal of computer vision*, vol. 61, no. 2, pp. 139–157, 2005.

[75] G. Balakrishnan, A. Zhao, M. R. Sabuncu, J. Guttag, and A. V. Dalca, "Voxelmorph: a learning framework for deformable medical image registration," *IEEE transactions on medical imaging*, vol. 38, no. 8, pp. 1788–1800, 2019.

[76] A. V. Dalca, G. Balakrishnan, J. Guttag, and M. R. Sabuncu, "Unsupervised learning of probabilistic diffeomorphic registration for images and surfaces," *Medical image analysis*, vol. 57, pp. 226–236, 2019.

[77] R. Sandkühler, S. Andermatt, G. Bauman, S. Nyilas, C. Jud, and P. C. Cattin, "Recurrent registration neural networks for deformable image registration," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019.

[78] H. Xu, J. Zhang, J. Cai, H. Rezatofighi, and D. Tao, "Gmflow: Learning optical flow via global matching," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8121–8130.

[79] W. Jiang, E. Trulls, J. Hosang, A. Tagliasacchi, and K. M. Yi, "Cotr: Correspondence transformer for matching across images," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6207–6217.

[80] S. Zhao, Y. Dong, E. I. Chang, Y. Xu *et al.*, "Recursive cascaded networks for unsupervised medical image registration," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 10 600–10 610.

[81] T. C. Mok and A. Chung, "Fast symmetric diffeomorphic image registration with convolutional neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 4644–4653.

[82] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1. Oakland, CA, USA, 1967, pp. 281–297.

[83] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: nonlinear phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.

[84] S. G. Mueller, M. W. Weiner, L. J. Thal, R. C. Petersen, C. R. Jack, W. Jagust, J. Q. Trojanowski, A. W. Toga, and L. Beckett, "Ways toward an early diagnosis in alzheimer's disease: the alzheimer's disease neuroimaging initiative (adni)," *Alzheimer's & Dementia*, vol. 1, no. 1, pp. 55–66, 2005.

[85] A. Di Martino, C.-G. Yan, Q. Li, E. Denio, F. X. Castellanos, K. Alaerts, J. S. Anderson, M. Assaf, S. Y. Bookheimer, M. Dapretto *et al.*, "The autism brain imaging data exchange: towards a large-scale evaluation of the intrinsic brain architecture in autism," *Molecular psychiatry*, vol. 19, no. 6, pp. 659–667, 2014.

[86] P. Bellec, C. Chu, F. Chouinard-Decorte, Y. Benhajali, D. S. Margulies, and R. C. Craddock, "The neuro bureau adhd-200 preprocessed repository," *Neuroimage*, vol. 144, pp. 275–286, 2017.
[87] D. W. Shattuck, M. Mirza, V. Adisetiyo, C. Hojatkashani, G. Salamon, K. L. Narr, R. A. Poldrack, R. M. Bilder, and A. W. Toga, "Construction of a 3d probabilistic atlas of human cortical structures," *Neuroimage*, vol. 39, no. 3, pp. 1064–1080, 2008.
[88] P. Bilic, P. F. Christ, E. Vorontsov, G. Chlebus, H. Chen, Q. Dou, C.-W. Fu, X. Han, P.-A. Heng, J. Hesser *et al.*, "The liver tumor segmentation benchmark (lits)," *arXiv preprint arXiv:1901.04056*, 2019.
[89] T. Heimann, B. Van Ginneken, M. A. Styner, Y. Arzhaeva, V. Aurich, C. Bauer, A. Beck, C. Becker, R. Beichel, G. Bekes *et al.*, "Comparison and evaluation of methods for liver segmentation from ct datasets," *IEEE transactions on medical imaging*, vol. 28, no. 8, pp. 1251–1265, 2009.
[90] D. Rueckert, L. I. Sonoda, C. Hayes, D. L. Hill, M. O. Leach, and D. J. Hawkes, "Nonrigid registration using free-form deformations: application to breast mr images," *IEEE transactions on medical imaging*, vol. 18, no. 8, pp. 712–721, 1999.