Video to Video Generative Adversarial Network for Few-shot Learning Based on Policy Gradient

Yintai Ma [1] **Diego Klabjan** [1] **Jean Utke** [2]

Abstract

The development of sophisticated models for video-to-video synthesis has been facilitated by recent advances in deep reinforcement learning and generative adversarial networks (GANs). In this paper, we propose RL-V2V-GAN, a new deep neural network approach based on reinforcement learning for unsupervised conditional video-to-video synthesis. While preserving the unique style of the source video domain, our approach aims to learn a mapping from a source video domain to a target video domain. We train the model using policy gradient and employ ConvLSTM layers to capture the spatial and temporal information by designing a fine-grained GAN architecture and incorporating spatio-temporal adversarial goals. The adversarial losses aid in content translation while preserving style. Unlike traditional video-to-video synthesis methods requiring paired inputs, our proposed approach is more general because it does not require paired inputs. Thus, when dealing with limited videos in the target domain, i.e., few-shot learning, it is particularly effective. Our experiments show that RL-V2V-GAN can produce temporally coherent video results. These results highlight the potential of our approach for further advances in video-to-video synthesis.

Keywords: Video Generation, Unsupervised Learning, Generative Adversarial Network, Reinforcement Learning

[1]Department of Industrial Enginering and Management Science, Northwestern University, Evanston, Illinois, United States [2]Allstate Insurance Company, Northbrook, Illinois, United States. Correspondence to: Yintai Ma <yintaima2022@u.northwestern.edu>, Diego Klabjan <d-klabjan@northwestern.edu>, Jean Utke <jutke@allstate.com>.

1. Introduction

Video-to-video synthesis is a type of problem where a video is translated into another video while keeping some specific semantic meaning from the first video. Learning to synthesize continuous visual experiences is essential to build intelligent agents. For example, turning a daytime first-person car driving video into a nighttime driving video generates valuable samples for training autonomous driving agents. With such a learned video synthesis model, it would be possible to generate realistic videos without explicitly specifying scene geometry, materials, lighting, and dynamics, which would be cumbersome but necessary when using standard graphics rendering techniques. Learning to synthesize continuous visual videos serves both scientific interests and a wide range of applications including human motion and face translation from one person to another, teaching robots from human demonstration, or converting black-and-white videos to color. The ability to learn and model the temporal dynamics of our visual experience is also essential to building intelligent agents (Bandi et al., 2023).

While the image-based counterpart, the image-to-image synthesis problem, is a well-researched area, the video-to-video synthesis problem is less studied in existing works (Zhuo et al., 2022). The traditional frame-based models (Cao et al., 2014; Thies et al., 2016), in which the models take separate frames as inputs and outputs, ignore the temporal coherence within the videos and result in low visual quality and poor continuity. Most importantly, a frame-based model can not generate a correct sample based on its context or previous frames. Here we show an example in Figure 1, where the goal is to translate an all-black background video to another video with the blue background in its first half and the red background in its second half. The target video has been obtained by the frame-based approach - and it clearly fails. Our proposed model outperforms the frame-based model because our video-to-video synthesis model correctly generates the background color for each output frame while the frame-based model randomly generates the background color for output frames. A more detailed and complex experiment is presented in Section 6.

The image-to-image translation problem has been a popular topic in recent years and it has been widely studied (Isola

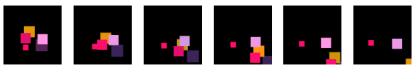
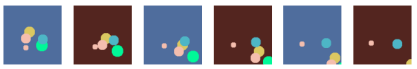
Sequence Description	Frames in the Sequence
Source video sequence	
Target video sequence	

Figure 1. The source video sequence is depicted in the first row and serves as the input to the model. The target video, shown in the second row, is characterized by a blue background in its first half and a red background in its second half.

et al., 2017; Taigman et al., 2017; Francisco et al., 2017; Shrivastava et al., 2017; Zhu et al., 2017a; Liu et al., 2017; Liu & Tuzel, 2016; Huang et al., 2018; Zhu et al., 2017b). However, using these frame-based image-to-image models to translate videos has its inherent problem. As such models take separate images as inputs and outputs, they ignore the temporal coherence within the video and result in poor continuity and low visual quality. Only the models that include a recurrent neural network module or 3DCNN can better handle the video generation. Our proposed models directly model the temporal dynamics within videos with recurrent convolutional layers. We apply ConvLSTM to capture such temporal dynamics, which shows superior performance compared to the traditional way where the LSTM layer is attached after a CNN layer.

In this paper, we propose a reinforcement learning-based deep neural network approach, RL-V2V-GAN, for an unsupervised conditional video-to-video synthesis problem, for which the goal is to learn a mapping from a source video domain to another target video domain while preserving the style native to the target domain. The inputs are videos in the source domain, videos in the target domain, and videos and images in the target domain to serve as style references. The videos in the source domain have the same style as the style videos in the target domain. The goal is to create videos in the target domain that embody the common style by learning from the videos in the source domain.

There are numerous potential applications of RL-V2V-GAN. In e-commerce, it can transform generic product videos (source domain) into personalized recommendation videos (target domain). The common style is how the product is presented. This use case enhances customer engagement and conversion rates for businesses. In urban planning, it can transform abundant daytime cityscape videos (source domain) into mesmerizing nighttime aerial visuals (target domain). The common style is the appearance of buildings and the layout of cities. This aids city planners and promotes tourism by requiring less frequent data collection.

Our proposed video-to-video synthesis approach uses the generative adversarial neural network (GAN) framework. We design a custom GAN architecture, incorporating spatiotemporal adversarial objectives, and apply ConvLSTM layers to capture spatial and temporal information. Adversarial losses are employed to aid in the translation of content while preserving style.

Within our proposed model, RL-V2V-GAN, we leverage the SeqGAN framework (Yu et al., 2016), by integrating Generative Adversarial Networks (GANs) with Reinforcement Learning (RL). Rather than utilizing traditional sequence generation methods, we employ RL, wherein the video generator functions as a stochastic policy. This approach facilitates a nuanced, contextual evaluation of each video frame within its encompassing sequence, ensuring superior temporal coherence. Derived from the GAN discriminator, the reward signals provide crucial feedback for optimization of synthetic video generation. This reliance on RL is pivotal for our model's capability to generate video sequences that closely align with the stylistic attributes of the target domain.

This approach is completely unsupervised since it does not require paired inputs, making it particularly effective for zero or few-shot learning. The experiments demonstrate that our model is capable of generating temporally coherent video results based solely on input visual frames. By leveraging the proven generative adversarial framework of CycleGAN (Wexler et al., 2007), our innovative approach offers unsupervised training of video-to-video synthesis models, enabling the translation of videos to another domain with one domain of training samples and a few in the target domain, resulting in a wide range of visually appealing output videos.

The main contribution of this paper is the first deep reinforcement learning algorithm with a recurrent neural network approach that solves a conditional video-to-video synthesis problem. This novel approach seamlessly integrates reinforcement learning (RL) with generative adversarial networks (GANs), leveraging the strengths of both to achieve temporal coherence and stylistic fidelity in generated video sequences. The existing deep neural network models for conditional video synthesis tasks are frame-based, where each frame in the translated video is generated separately. These types of models suffer when generating a high-quality video over a long period. However, our model natively takes consecutive frames as one single input. It helps to generate videos where the content in each frame is temporally coherent.

This paper is organized as follows. In Section 2, we review the related works and the development of the video retrieval methods. We formally state our model and algorithm in Section 3. We discuss our algorithm in Section 5. We present our numerical experiments results in Sections 6.

2. Related Works

In this section, we discuss advancements in video-to-video synthesis.

Video Synthesis Video synthesis can be broadly categorized into unconditional and conditional approaches, alongside a refined classification based on temporal dimension handling, namely: frame-to-frame, frame-to-video, and video-to-video synthesis.

Unconditional video synthesis operates without explicit input beyond random noise, aiming to generate coherent video content. Pioneering works (Carl et al., 2016; Saito et al., 2017; Tulyakov et al., 2018) extend the traditional GAN framework for unconditional video synthesis, focusing on the conversion of random vectors into video sequences through spatio-temporal convolutional networks and latent image code projection. Saito et al. (2017) propose TGAN to project a latent vector to a set of latent image codes, and then convert these latent image codes to frames with an image generator.

Conversely, conditional video synthesis generates videos based on given inputs or conditions, such as semantic representations or single frames. This approach often leverages temporal models for predicting future poses or generating videos conditioned on specific inputs, as seen in works such as He et al. (2018), Villegas et al. (2017), Walker et al. (2017), and further advancements by Mathieu et al. (2015) and Chen et al. (2017b). Our method falls into this category, employing ConvLSTM layers within a GAN framework to capture spatial and temporal dynamics, ensuring high fidelity in video synthesis.

Diving deeper into the temporal handling categorization, the frame-to-frame video synthesis models take each frame as input for its generator and are mostly CNN-based. They usually include a discriminator in the model to decide whether the generated sequence is a video (Chen et al., 2017a; Gupta et al., 2017; Huang et al., 2017; Ruder et al., 2016).

The second type of frame-to-video synthesis models try to generate a sequence of frames based on one single input frame. Tulyakov et al. (2018)'s MoCoGAN separates the latent space to motion and content subspaces and uses an RNN generator to obtain a sequence of motion embeddings. However, it often struggles with high-resolution or long videos due to the complexity of modeling motion dynamics.

The last video-to-video models are the most advanced ones that directly take temporal patterns as part of the inputs. They usually take the entire video 3-dimension tensor as input. Bansal et al. (2018) propose a video-to-video translation model with ReCycleGAN that includes a cross-domain cyclic loss for temporally coherent frames. Wang et al. (2018) propose a conditional video-to-video synthesis model with a sequential image-based generator. Shen et al. (2023) present MoStGAN-V, which uses temporal motion styles to model diverse and temporally-consistent motions in video generation. This technique enhances the generation of dynamic motions, relevant to our model's objectives. Ma et al. (2024) introduce CVEGAN, a GAN-based model designed for enhancing the quality of compressed video frames. This approach is relevant for applications requiring high-quality video outputs, aligning with our objectives of generating visually coherent videos.

Generally speaking, the videos generated by the above image-to-image models suffer from temporal incoherence. In contrast to the first two categories, we focus on designing a video-to-video translation approach that generates videos with coherent frames. All the above models process videos frame by frame while we generate an entire video in one shot. Unlike traditional methods that heavily rely on paired inputs, our model's unsupervised nature makes it exceptionally resilient and adaptable, proving invaluable especially when faced with limited samples in the target domain.

SeqGAN (Yu et al., 2016) is a sequence generation framework that leverages Generative Adversarial Networks (GAN) to train a generative model. Traditional GANs have limitations when generating sequences of discrete tokens due to the difficulties in passing gradient updates from the discriminative model to the generative model. To overcome this, SeqGAN models the data generator as a stochastic policy in reinforcement learning and performs gradient policy updates directly, bypassing the generator differentiation problem. The RL reward signal is obtained from the GAN discriminator, which is trained to provide positive examples from real sequence data and negative examples from synthetic sequences generated from the generative model. The reward signal is passed back to the intermediate state-action steps using a Monte Carlo search. The policy gradient is calculated based on the expected end reward received from the discriminative model, which represents the likelihood of fooling the discriminator. Extensive experiments on synthetic data and real-world tasks demonstrate the effectiveness of SeqGAN. Based on the advancements of SeqGAN, our work generates the entire video sequence with Policy Gradient. RL-V2V-GAN represents an innovative convergence of deep reinforcement learning with GAN for video synthesis. This strategy not only yields temporally coherent outputs but also comprehends and replicates the intricate stylistic elements inherent to the source domain.

Generative AI and Diffusion Models Generative AI, particularly through advancements in diffusion models, has revolutionized the field of artificial intelligence by enabling the generation of high-quality, realistic data across various domains. Yu et al. (2023) propose Projected Latent Video Diffusion Models (PVDM), which efficiently han-

dle high-resolution video synthesis in a low-dimensional latent space. This method addresses the challenges of high-dimensionality and complex temporal dynamics, which is pertinent to our work. Yang et al. (2023) present a diffusion probabilistic model for video generation that improves perceptual quality and probabilistic frame forecasting. Their approach is relevant for ensuring temporal coherence in generated videos, similar to our objectives. Chen et al. (2023) introduce SEINE, a short-to-long video diffusion model focusing on generative transition and prediction, which can be extended to various tasks such as image-to-video animation. This approach is relevant to enhancing the temporal dynamics in video generation. Esser et al. (2023) present a structure and content-guided video diffusion model that edits videos based on user descriptions, providing fine-grained control over output characteristics and ensuring temporal consistency. This work aligns with our goal of generating high-fidelity, temporally coherent videos. Kim et al. (2023) propose a face video editing framework based on diffusion autoencoders that ensures temporal consistency. This method is pertinent to our goal of maintaining coherence in video synthesis. Zeng et al. (2024) present PixelDance, a novel approach for high-dynamic video generation using diffusion models. This method's success in synthesizing complex scenes and motions is relevant to enhancing the visual dynamics in our video generation.

These approaches primarily utilize diffusion models, whereas our work focuses on GAN-based video synthesis. Generative Adversarial Networks (GANs) and diffusion models each have distinct advantages. GANs excel in generating high-quality, realistic images quickly once trained but can suffer from training instability and mode collapse. Diffusion models offer stable training and diverse outputs but are slower during inference due to their iterative process. GANs are ideal for tasks demanding efficient, high-quality image generation, while diffusion models suit applications requiring stability and diversity. Our work focuses on GAN-based video synthesis and uniquely integrates reinforcement learning to enhance training efficiency and video quality. By leveraging GANs' strengths, our approach ensures that the generated videos are both realistic and stylistically consistent with the target domain.

Video Generation with Reinforcement Learning has seen significant advancements in recent years. Notably, the VIPER framework (Escontrela et al., 2023) leverages video prediction models to generate reward signals for training RL agents. Both VIPER and our approach share common ground in utilizing generative models for video and integrating RL to enhance the learning process. However, VIPER focuses on policy learning by using video model log-likelihoods as reward signals, whereas our RL-V2V-GAN directly addresses the challenge of video-to-video synthesis, generating high-quality, temporally coherent videos. Our unique contribution lies in the novel integration of RL with GAN for unsupervised video synthesis. Unlike VIPER, which aims to train RL agents by mimicking expert trajectories, our approach employs spatio-temporal adversarial objectives and ConvLSTM layers to capture both spatial and temporal information, ensuring the stylistic fidelity and coherence of generated videos. Additionally, our model's ability to perform few-shot learning makes it particularly effective for generating videos in scenarios with limited target domain data. The specific settings that enable our model to solve the problem effectively include using ConvLSTM layers to capture temporal dynamics, applying spatio-temporal adversarial objectives, and employing a robust combination of adversarial, recurrent, recycle, and video losses.

3. Model

This section outlines the mathematical notation and models to accurately specify the elements of RL-V2V-GAN. We start by establishing notation and proceed to explain the model's GAN architecture, including its generators, discriminators, and predictors. We emphasize the model's loss functions. The role of RL in refining video synthesis is then highlighted. Finally, we outline the key neural network components, such as ConvLSTM layers and neural blocks, essential for video generation. This framework facilitates the translation of videos across various domains and styles.

3.1. Notation

We define $\mathcal{X}$ and $\mathcal{Y}$ to be sets of videos $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{y} \in \mathcal{Y}$ in the source and target domain, respectively. The notation T denotes the length of a video, and subscripting $\boldsymbol{x}_i$ selects the i-th frame from video $\boldsymbol{x}$. Notations $\boldsymbol{x}_{:j}$ and $\boldsymbol{y}_{:j}$ represent videos from frames 1 to j. We also use x and y to represent a single frame from the source or target domain. Similarly, we define $\boldsymbol{z} \in \mathcal{Z}$ to represent a style video in the target domain, and $\bar{z} \in \bar{\mathcal{Z}}$ to represent a style image in the target domain. Styles represented by $\mathcal{Z}$ and $\bar{\mathcal{Z}}$ are assumed to be compatible.

3.2. Generative Adversarial Network Model

In this work, the proposed model creates additional videos $\boldsymbol{y} \in \mathcal{Y}$ that incorporate features from both source videos $\boldsymbol{x} \in \mathcal{X}$ and style of $\boldsymbol{z} \in \mathcal{Z}$. Our goal is to overcome the challenge of having a much smaller number of videos $\boldsymbol{y}$ for the desired style and domain. The model takes input videos from $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$, as well as images of desired style $\bar{z} \in \bar{\mathcal{Z}}$. With these inputs, our model generates new videos in the style of $\mathcal{Z}$ and $\bar{\mathcal{Z}}$, addressing the scarcity of videos in $\mathcal{Y}$ in the preferred style.

This GAN model contains sequence generators G_x, G_y, predictors P_x, P_y and discriminators D_x, D_y. Here x is

always from the source domain, y is always from the target domain and z is always a style frame from the target domain. The generators map a video from the source domain to a video in the target domain or from the target domain to the source domain, respectively.

We define $G_y : \boldsymbol{x} \to \boldsymbol{y}$ and $G_x : \boldsymbol{y} \to \boldsymbol{x}$. The predictor uses a video to predict the next frame of this video in the same domain, for the source and target domain, respectively. We define $P_x : \boldsymbol{x}_{:t} \to \boldsymbol{x}_{t+1}$ and $P_y : \boldsymbol{y}_{:t} \to \boldsymbol{y}_{t+1}$. Although the predictor P has the same sequence-to-sequence neural network architecture as the video generator G, we denote $P_x(\cdot)$ and $P_y(\cdot)$ to represent the prediction of the next frame, rather than the entire output sequence. In our model, the generator produces three logits per pixel, corresponding to the RGB channels. These logits are not independent as they are generated using ConvLSTM layers, which capture spatial and temporal dependencies. This approach is similar to an LSTM but operates on the convolutional feature maps, ensuring that the temporal coherence between video frames is maintained. The sequence discriminators have two use cases; they can take either a frame or a sequence of frames as input. When a discriminator takes one frame as input, we define $D_x : x \to [0, 1]$ and $D_y : y \to [0, 1]^2$. In this case, D_y produces two outputs: $D_{y,0}$ predicts whether the sample sequence belongs to the target domain, and $D_{y,1}$ predicts whether the sample has the desired style, based on adversarial learning with samples from $\mathcal{Z}$ or $\bar{\mathcal{Z}}$. Positive samples for $D_{y,1}$ include video data from $z \in \mathcal{Z}$, while generated samples are negative. In the former case, when a video is fed to a source domain discriminator, we define $D_{\boldsymbol{x}} : \boldsymbol{x} \to [0, 1]$ that outputs a probability indicating whether the video sequence belongs to the source domain. When a video is fed to a target domain discriminator, $D_{\boldsymbol{y}}$ outputs two scalar values, denoted as $[D_{\boldsymbol{y},0}, D_{\boldsymbol{y},1}] \in [0, 1]^2$. The probability $D_{\boldsymbol{y},0}$ specifies whether the sample sequence belongs to the target domain, and $D_{\boldsymbol{y},1}$ indicates whether the sample sequence has the desired style. For an example of how the datasets $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{Z}$ are structured, please refer to Table 4, which illustrates data from modern cities in daytime and small-town videos in both daytime and nighttime, along with style references from small-town daytime footage and modern city nighttime images.

Next we list loss components.

3.2.1. Adversarial loss L_g

With video samples $\{\boldsymbol{x}_{:t}\} \in \mathcal{X}$, $\{\boldsymbol{y}_{:t}\} \in \mathcal{Y}$, and $\{\boldsymbol{z}_{:t}\} \in \mathcal{Z}$, as well as frame samples $\{\bar{y}\} \in \bar{\mathcal{Z}}$, $\boldsymbol{y}_t \in \{\boldsymbol{y}_{:t}\}$ and $\boldsymbol{z}_t \in \{\boldsymbol{z}_{:t}\}$, we define the standard adversarial loss to distinguish between true samples and synthetic samples generated by the model. Here, $\bar{y}$ represents an image with the desired style.

The objective functions are defined using the adversarial losses L_g^y and L_g^x, where the generator minimizes these losses and the discriminator maximizes them, following the standard minimax framework of GANs. Although the log terms in the loss are negative, minimizing the generator's loss is equivalent to maximizing the likelihood of generating realistic data. This setup aligns with common GAN formulations, where both the generator and discriminator operate within the output range of $[0, 1]$.

$$\begin{aligned}&\min_{G_y}\max_{D_y} L_g^y(G_y, D_y) := \\ &\textstyle\sum_s \log D_{y,0}(\bar{y}_s) + \sum_t \log D_{y,1}(\boldsymbol{z}_t) \\ &\textstyle+\sum_t \log D_{y,0}(\boldsymbol{y}_t) + \sum_t \log(1 - D_{y,0}((G_y(\boldsymbol{x}_{:t}))_t))\end{aligned} \tag{1}$$

$$\begin{aligned}&\min_{G_x}\max_{D_x} L_g^x(G_x, D_x) := \\ &\textstyle\sum_t \log D_x(\boldsymbol{x}_t) + \sum_t \log(1 - D_x((G_x(\boldsymbol{y}_{:t}))_t)) \\ &\textstyle+\sum_t \log(1 - D_x((G_x(\boldsymbol{z}_{:t}))_t))\end{aligned} \tag{2}$$

3.2.2. Recurrent Loss L_{rr}

By encouraging the model to generate more temporally coherent frames by a recurrent loss, we can impose more advantage of the temporal ordering. We embed the recurrent temporal predictor P_x and P_y with the recurrent loss L_{rr}^x and L_{rr}^y, respectively.

$$L_{rr}^x(P_x) = \sum_t \|\boldsymbol{x}_{t+1} - P_x(\boldsymbol{x}_{:t})\|^2 \tag{3}$$

$$L_{rr}^y(P_y) = \sum_t \|\boldsymbol{y}_{t+1} - P_y(\boldsymbol{y}_{:t})\|^2 + \sum_t \|\boldsymbol{z}_{t+1} - P_y(\boldsymbol{z}_{:t})\|^2 \tag{4}$$

3.2.3. ReCycle Loss L_{rc}

We introduce the recycle loss L_{rc}, which encourages style and domain consistency when multiple generators and predictors are jointly utilized, crossing domains and time altogether. For example, we start with a video $\boldsymbol{x}_{:t}$, translate it to a video in the target domain denoted by $G_y(\boldsymbol{x}_{:t})$, then increment it on time dimension as $P_y(G_y(\boldsymbol{x}_{:t}))$. We select the last frame, which is the $t+1$-st frame in the source domain with subscript $t+1$ as $G_x((\boldsymbol{y}_{:t}, P_y(G_y(\boldsymbol{x}_{:t}))))_{t+1}$ and compare this frame with the ground truth x_{t+1} in a L-2 loss.

$$\begin{aligned}&L_{rc}^{xy}(G_x, G_y, P_y) \\ = \;&\textstyle\sum_t \|\boldsymbol{x}_{t+1} - G_x((\boldsymbol{y}_{:t}, P_y(G_y(\boldsymbol{x}_{:t}))))_{t+1}\|^2\end{aligned} \tag{5}$$

and

$$\begin{aligned} & L_{rc}^{yx}(G_y, G_x, P_x) \\ = \; & \textstyle\sum_t \|\boldsymbol{y}_{t+1} - G_y((\boldsymbol{x}_{:t}, P_x(G_x(\boldsymbol{y}_{:t})))_{t+1}\|^2 + \\ & \textstyle\sum_t \|\boldsymbol{z}_{t+1} - G_y((\boldsymbol{x}_{:t}, P_x(G_x(\boldsymbol{z}_{:t})))_{t+1}\|^2 \end{aligned} \tag{6}$$

3.2.4. Video Loss L_v

Finally, we leverage the specially designed discriminator capable of processing both individual frames and full videos. We design video loss L_v to supervise the overall style and quality of the generated video for the target domain. The purpose is to ensure the consecutive output frames resemble the temporal dynamics of a real video. The video loss L_v is given by:

$$\begin{aligned} & L_v(G_x, G_y, D_x, D_y) \\ = \; & L_{vx}(G_x, G_y, D_x) + L_{vy}(G_y, G_x, D_y), \end{aligned} \tag{7}$$

where

$$\begin{aligned} & L_{vx}(G_x, G_y, D_x) \\ = \; & \textstyle\sum_t \log D_{\boldsymbol{x}}(\boldsymbol{x}_{:t}) + \\ + \; & \textstyle\sum_t \log(1 - D_{\boldsymbol{x}}(G_x(\boldsymbol{y}_{:t}))) \\ + \; & \textstyle\sum_t \log(1 - D_{\boldsymbol{x}}(G_x(\boldsymbol{z}_{:t}))). \end{aligned} \tag{8}$$

and

$$\begin{aligned} & L_{vy}(G_y, G_x, D_y) \\ = \; & \textstyle\sum_t \log D_{\boldsymbol{y},0}(\boldsymbol{y}_{:t}) \\ + \; & \textstyle\sum_t \log D_{\boldsymbol{y},1}(\boldsymbol{z}_{:t}) \\ + \; & \textstyle\sum_t \log(1 - D_{\boldsymbol{y},0}(G_y(\boldsymbol{x}_{:t}))) \\ + \; & \textstyle\sum_t \log(1 - D_{\boldsymbol{y},1}(G_y(\boldsymbol{x}_{:t}))). \end{aligned} \tag{9}$$

3.2.5. Total Loss of RL-V2V-GAN

We now combine the above losses and rewrite it in a standard min-max formulation to form the total loss used for the proposed RL-V2V-GAN model:

$$\begin{aligned} & \min_{G,P} \max_D L(G, P, D) \\ = \; & L_g^x(G_x, D_x) + L_g^y(G_y, D_y) \\ & + \lambda_{rrx} L_{rr}^x(P_x) + \lambda_{rry} L_{rr}^y(P_y)) \\ & + \lambda_{rcx} L_{rc}^{xy}(G_x, G_y, P_y) + \lambda_{rcy} L_{rc}^{yx}(G_y, G_x, P_x)) \\ & + \lambda_{vx} L_{vx}(G_x, G_y, D_x)) + \lambda_{vy} L_{vy}(G_x, G_y, D_y)) \end{aligned} \tag{10}$$

The gradients of this loss can be separated into two parts. The first part, $\partial(L - L_v)/\partial D$, trains discriminators for non-terminating states. The second part, $\partial L_v/\partial D$, trains discriminators for video input using terminating state rewards. Gradient penalty coefficients $\lambda = \{\lambda_{vx}, \lambda_{vy}, \lambda_{rrx}, \lambda_{rry}, \lambda_{rcx}, \lambda_{rcy}\}$ are added to control the relative importance of each loss components during training.

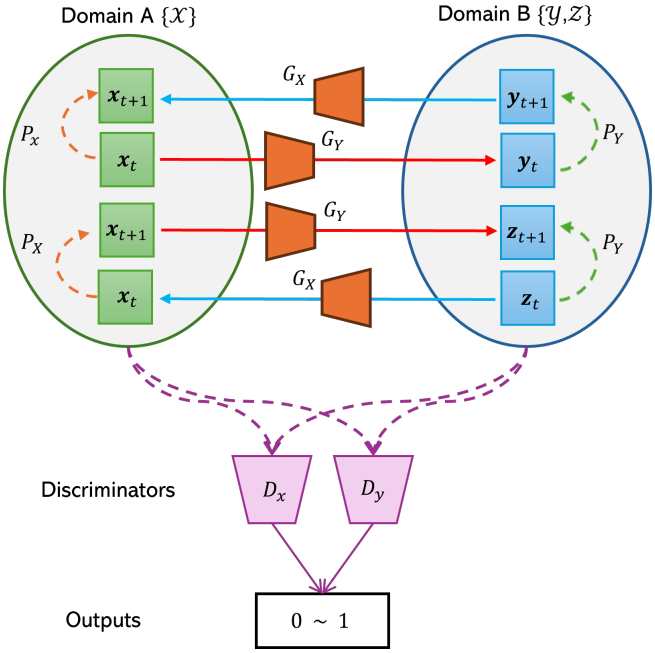


Figure 2. The diagram presents the RL-V2V-GAN model, which integrates sequence generators G_x, G_y, predictors P_x, P_y, and discriminators D_x. D_y for video style transfer. It captures the workflow where G networks transform videos between domains, P networks forecast future frames and D networks assess authenticity and style. The model operates under various losses—adversarial, recurrent, ReCycle, and video—to ensure high-quality, coherent video generation, addressing the challenge of data scarcity in style-specific videos.

3.3. Reinforcement Learning Model

For the reinforcement learning part of the algorithm, we define two Q-networks $Q = \{Q_x(s,a), Q_y(s,a)\}$ and the policy networks $\mu = \{G_x(s), G_y(s), P_x(s), P_y(s)\}$. The state s is defined as a video and action a is a frame that could happen right after such a video. Given a video and a future frame, the Q-networks generate a reward $\hat{r}$ to estimate how likely this frame could happen as the next frame to this video, i.e., $Q(s,a) \to \hat{r}$. We have one Q network for each domain, such that $Q_x(s,a)$ evaluates how good the action is for the source domain, and $Q_y(s,a)$ evaluates the action for the target domain. They are parameterized by θ^Q. Each policy network in μ outputs the action (next frame) for a given state s (video). All four of the generators G_x, G_y, predictors P_x and P_y are policy networks parameterized by $\theta^\mu = \{\theta^{G_x}, \theta^{G_y}, \theta^{P_x}, \theta^{P_x}\}$.

In the reinforcement learning training process, we characterize each transition by $(s, a, r, s', a^{\text{true}}, p)$, where a^{true} represents the actual subsequent frame following the sequence s. The positional variable p indicates whether the action frame has reached the maximum number of frames in a video, denoted as T. The length of each video is fixed

at T frames. The reward r is computed based on whether the state has reached the end of a video. When this terminal state is reached (i.e., $p = T - 1$), the reward is obtained from the video loss: $r \leftarrow -\lambda_v L_v$. Otherwise, the reward is calculated from the adversarial, recurrent, and recycle losses: $r_t \leftarrow -(L_g + \lambda_{rr} L_{rr} + \lambda_{rc} L_{rc})$. This ensures that the reward captures both the quality of the generated frame and its temporal coherence with previous frames.

To enhance stability and facilitate effective learning, transitions are continuously collected in a replay buffer $\mathcal{B}$ during training. This buffer is crucial for implementing delayed policy updates. New transitions are added until the buffer reaches its capacity limit, at which point older experiences are replaced by newer ones. This ensures a diverse set of experiences for sampling mini-batches, which are then used to update the policy networks.

To improve training stability and reduce fluctuations, our model employs target networks, Q' and μ', as stable versions of Q-networks and policy networks, respectively. These target networks update their weights gradually, mirroring the primary networks at specific intervals. This strategy, rooted in deep Q-learning techniques, ensures smoother training by offering a consistent benchmark for policy evaluation and reward estimation, thereby minimizing prediction variability and enhancing learning reliability. We also refer to the target networks with its parameter as $\theta^{Q'}$ and $\theta^{\mu'}$.

3.4. Neural Network Components

In our RL-V2V-GAN model, both the generators G_x, G_y and predictors P_x and P_y are sequence to sequence auto-encoders. They share the same architecture, constructed by three different core neural network blocks: block R, RPB, and URB. Illustrated in Figures 4a, 4b, and 4c, each block plays a pivotal role in processing the video sequences. The ConvLSTM layers, integrated within these blocks, are enhanced with residual connections, pooling layers, and batch normalization layers to capture both spatial and temporal dynamics effectively. Among these, the R block (Residual), featuring a LeakyReLU activation function, stands as the foundation, ensuring efficient data flow through the model. The RPB block (Residual-Pooling-BatchNormalization) serves a critical function in compressing the input dimensions, facilitating a more manageable representation of the data. Conversely, the URB block (Upsampling-Residual-BatchNormalization) is tasked with expanding these compressed inputs back to their higher dimensional form, thus preserving the integrity of the video's original structure and detail.

With the definitions of block R, RPB, and URB in place, we can now describe the architecture of our generators and discriminators in Figure 5. Our discriminator contains a sequence of RPB blocks, followed by a 3D convolutional layer that maps the video input to a single binary decision. On the other hand, the generator consists of an encoding part composed of RPB blocks and a decoding part made up of URB blocks. However, it is worth noting that the embedding vector obtained from the encoding part is not a standalone representation of the video, as there are residual connections established between each corresponding pair of RPB-URB layers.

4. Model Configuration

The RL-V2V-GAN model employs a unified architecture across its network components for streamlined video synthesis. This design ensures that the generated videos maintain temporal coherence and high visual quality. The configuration of the model is detailed in Table 1. All four networks share the same encoder architecture of 3 RPB layers. Both generators G and predictors P use the same decoder structure of 3 URB block layers to generate video as output. Similarly, both discriminators D and Q-networks Q have the same decoder structure of 2 RPB block layers and 2 fully connected (FC) layers to generate scalar outputs for effective video content analysis and reward estimation. The Q-network uses ReLU in the final fully connected layer to allow for a wide range of output values, while the discriminator uses Sigmoid to output probabilities.

Table 1. The encoder and decoder blocks in all networks

NETWORK	ENCODER	DECODER	FINAL FUNC.
GENERATORS (G)	3 RPB	3 URB	CONVLSTM
PREDICTORS (P)	3 RPB	3 URB	CONVLSTM
DISCRIMINATORS (D)	3 RPB	2 RPB-FC-FC	SIGMOID
Q-NETWORKS (Q)	3 RPB	2 RPB-FC-FC	RELU

The RL-V2V-GAN model trains four policy networks (G_x, G_y, P_x, P_y), two discriminators (D_x, D_y), and two Q-networks (Q_x, Q_y). These networks are initialized with pre-trained weights, denoted as θ^{μ}, θ^{D}, and θ^{Q} for the policy networks, discriminators, and Q-networks, respectively.

The training process uses several hyper-parameters, including the learning rate α, gradient penalty coefficients $\lambda = \{\lambda_{vx}, \lambda_{vy}, \lambda_{rrx}, \lambda_{rry}, \lambda_{rcx}, \lambda_{rcy}\}$, and mini-batch size m. Additionally, the function $\mathbf{I}(\cdot)$ is defined such that $\mathbf{I}(True) := 1$ and $\mathbf{I}(False) := 0$ to aid in the computation of various loss functions and policy updates.

5. Training Scheme

The RL-V2V-GAN training algorithm integrates reinforcement learning and generative adversarial networks to achieve high-quality video-to-video synthesis. The training algorithm is shown as Algorithm 1 and visually depicted in Figure 3. The training process is designed to optimize

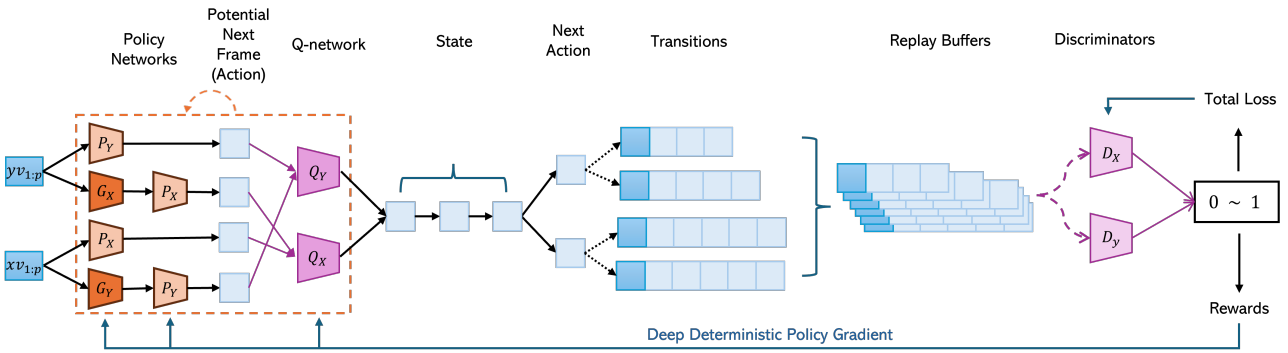
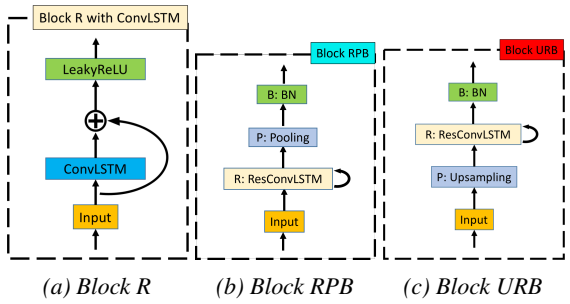
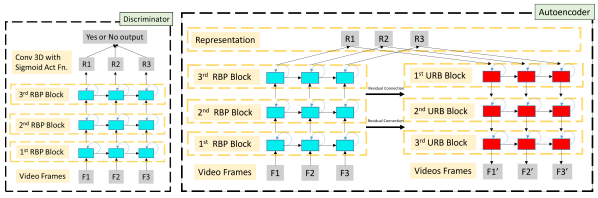


Figure 3. This figure showcases the reinforcement learning mechanism of RL-V2V-GAN, involving Q-networks (Q_x, Q_y) and policy networks (μ). States (s) are videos, and actions (a) are potential frames. Q-networks assess the flow of frame sequences, guiding the model to produce coherent and stylistically accurate videos. The system collects transitions in a replay buffer, optimizing for actions that yield realistic sequences and updates policy and Q-network with policy gradient.



(a) Block R *(b) Block RPB* *(c) Block URB*

Figure 4. (a) shows the structure of block R. (b) shows the structure of block RPB. (c) shows the structure of block URB.



(a) Discriminator *(b) Generator*

Figure 5. (a) shows the structure of a discriminator. This instance contains three layers of RPB blocks and a 3D convolutional layer with a sigmoid activation function. (b) shows the structure of a generator. This instance contains three layers of RPB blocks in the encoder and three layers of URB blocks in the decoder, respectively.

the policy networks and discriminators through a combination of policy gradient and adversarial training. At a high level, the algorithm begins by initializing the replay buffer and network parameters, including the policy networks, Q-networks, and discriminators. During each iteration of the training loop, the algorithm processes each video in the dataset to construct transition mini-batches. This involves selecting sequences of frames as states, generating actions using the policy networks, and calculating rewards based on the adversarial, recurrent, recycle, and video losses. The transitions are stored in the replay buffer. The deep deterministic policy gradient (DDPG) method is then used to update the Q-networks and policy networks. Simultaneously, random batch of samples from the original dataset are used to update the discriminators based on the total losses of the GAN, ensuring both temporal coherence and stylistic consistency in the generated videos.

The DDPG approach relies on the expected finite horizon undiscounted return, denoted by J, which is represented as the cumulative reward expected from following the policy over a finite time horizon in this algorithm. The algorithm optimizes J by computing the policy gradient, ensuring that the policy networks generate actions that maximize this expected return. This leads to effective updates of the networks, enabling the generation of high-quality video sequences.

Let $P \in \{P_x, P_y\}$ and $G, G' \in \{G_x, G_y\}$. The notation $G'(P(G(s)))$ requires special attention. Specifically, G' indicates that the generator used in the second step must be different from the generator used in the first step. This is because the initial state s, which is a sequence of frames, must belong to either the source or target domain. The process involves first using one generator G to transfer the state s to the other domain, then using the corresponding predictor P in the new domain to increment on the time

axis, and finally using a different generator G' to convert the state back to the original domain. This ensures that the transitions maintain domain consistency while capturing temporal dynamics.

The parameters σ_1 and σ_2 control the balance between exploration and exploitation. We set both σ_1 and σ_2 to constant values of 0.8. However, it is also a common practice to adjust these probabilities dynamically, allowing them to converge towards 1 or 0 as the training progresses. This convergence helps the model focus more on exploitation as it becomes more confident in its learned policies. We use the notation Bernoulli(σ), where $\sigma \in \{\sigma_1, \sigma_2\}$. Sampling from Bernoulli(σ) yields 1 with probability σ and 0 with probability $1 - \sigma$. In the algorithm, σ_1 determines whether to load a transition from the replay buffer or to generate a new transition, while σ_2 decides whether to select the next action based on the maximum Q-value or choose a random action. The discount factor γ plays a critical role in balancing short-term and long-term rewards. A higher γ encourages the model to prioritize future gains, whereas a lower γ results in a strategy that favors immediate rewards. This balance allows the agent to plan ahead while still valuing the immediate outcomes of its actions.

Overall, the RL-V2V-GAN algorithm is designed to leverage the strengths of both reinforcement learning and GANs, ensuring that the generated videos are both temporally coherent and stylistically consistent with the target domain. The detailed steps and considerations discussed here are critical for achieving the high performance demonstrated by the model.

6. Numerical Experiments

In this section, we present our experimental results, which demonstrates the effectiveness of our proposed method in terms of video generation quality compared to the state-of-the-art methods.

6.1. Datasets

We performed experiments on four distinct datasets to assess the performance of our proposed method for unsupervised video-to-video translation. These datasets provide comprehensive benchmarks for evaluating the efficacy of our approach.

The first dataset consists of synthetic videos. The $\mathcal{X}$ set contains 600 frames of colorful rectangles moving on a black background, while $\mathcal{Y}$ and $\mathcal{Z}$ have 100 and 100 frames of circles moving on color-changing and black backgrounds, respectively. The 100 supplemental style images $\bar{\mathcal{Z}}$ consist of random circles on either blue or red backgrounds. Samples are shown in Table 2.

The second dataset, a flower dataset, contains 550 frames of red flowers blooming in black backgrounds as $\mathcal{X}$, 100 frames of yellow flowers blooming in natural backgrounds as $\mathcal{Y}$, 900 images of yellow flowers blooming in black backgrounds as $\mathcal{Z}$, and 2,000 images of random color flowers on natural backgrounds as $\bar{\mathcal{Z}}$. The $\mathcal{X}$ and $\mathcal{Z}$ videos are from the StyleGAN flower dataset, while the $\mathcal{Y}$ and $\bar{\mathcal{Z}}$ sets are from the YouTube-8M dataset. Samples are shown in Table 3.

The third dataset, a city aerial dataset, contains 2,000 frames of daytime modern city aerial videos as $\mathcal{X}$, 500 frames of night time small-town aerials as yv^T, 1,500 images of nighttime city aerial photos as $\bar{\mathcal{Z}}$, and 1,000 frames of small-town aerial videos during daytime as $\mathcal{Z}$. All of the videos are from the YouTube-8M dataset. Samples are shown in Table 4.

The fourth, proprietary dataset, encompasses both synthetic and real-world images of 2 different types. It consists of 2,000 synthetic images labeled $\mathcal{X}$, alongside 500 real-world images of type A, tagged as $\mathcal{Y}$. Additionally, there are 1,000 real-world images of type B, categorized under $\mathcal{Z}$, and 4,000 real-world images depicting type A, identified as $\bar{\mathcal{Z}}$. For an in-depth overview of the dataset's composition, see Tables 5 and 6.

6.2. Implementation

We implemented the model using the Tensorflow 1.15 framework and trained it on four NVIDIA 3070 GPUs or equivalents. We used the softmax loss function for the auto-encoder and applied L2 regularization of penalty ratio of 0.001 to the model's trainable parameters. These parameters were initialized using Xavier initialization.

For the seq2seq auto-encoder, sequence-wise normalization was applied across multiple video sequences within each mini-batch. We computed the mean and variance statistics across all timesteps within this mini-batch for each output channel. Activation functions are ReLU. The optimization

Algorithm 1 RL-V2V-GAN Training Algorithm

Input The discriminators $D_x(s, a|\theta^{D_x}), D_y(s, a|\theta^{D_y})$, Q-networks $Q_x(s, a|\theta^{Q_x}), Q_y(s, a|\theta^{Q_y})$ and policy networks $G_x(s|\theta^{G_x}), G_y(s|\theta^{G_y}), P_x(s|\theta^{P_x})$ and $P_y(s|\theta^{P_y})$, learning rate α, gradient penalty coefficients $\lambda = \{\lambda_v, \lambda_{rr}, \lambda_{rc}\}$, mini-batch size m, discount factor γ.

Output Trained policy networks $\mu(\cdot|\theta^{\mu})$ and discriminators $D(\cdot|\theta^{D})$

Initialize replay buffer $\mathcal{B}$.
Initialize D, G, and μ with pre-trained weights.
Initialize Q' and μ' with $\theta^{Q'} = \theta^{Q}$ and $\theta^{\mu'} = \theta^{\mu}$.
Loop max epochs times:
    for each video v in training set $\mathcal{X} \cup \mathcal{Y} \cup \mathcal{Z}$ **do**
        Set mini-batch $\mathbf{B} = \emptyset$
        Repeat m times:
            Sample $u_1 \sim$ Bernoulli(σ_1) to decide how to get the next transition
            if $u_1 = 1$ and $\mathcal{B} \neq \emptyset$ **then**
                Load random tuple $(s, a, r, s', a^{\text{true}}, p)$ from reply buffer $\mathcal{B}$ and add to mini-batch $\mathbf{B}$
            else
                Randomly pick $p \in \{1, 2, \ldots, T-1\}$ and select p consecutive frames from video v as state s
                Use policy network to generate two candidate actions: $a_1 = P(s)$ and $a_2 = G^{(1)}(P(G^{(2)}(s)))$ where $P \in \{P_x, P_y\}, G^{(1)}, G^{(2)} \in \{G_x, G_y\}$, and the choice depends on the membership of v.
                Sample $u_2 \sim$ Bernoulli(σ_2) to decide how to choose the next action
                if $u_2 = 1$ **then**
                    Select next action $a \in \arg\max\{Q(s, a_1), Q(s, a_2)\}$
                else
                    Select a random frame a from $\boldsymbol{x} \cup \boldsymbol{y} \cup \boldsymbol{z} \cup \{\bar{y}\}$ as next action a
                if $p = T - 1$ **then**
                    Obtain reward from video loss $r \leftarrow -\lambda_v L_v$
                else
                    Obtain reward from adversarial, recurrent and recycle loss $r \leftarrow -(L_g + \lambda_{rr} L_{rr} + \lambda_{rc} L_{rc})$
                Get next state s' by concatenating current state s with the action a.
                Get ground truth action a^{true} based on video v
                Add tuple $(s, a, r, s', a^{\text{true}}, p+1)$ to buffer $\mathcal{B}$ and to mini-batch $\mathbf{B}$
        for each transition $x = (s, a, r, s', a^{\text{true}}, p) \in \mathbf{B}$ **do**
            Compute target: $\hat{r}_x \leftarrow r + \gamma Q_x(s, \mu(s|\theta^{\mu'})|\theta^{Q'_x}) \cdot \mathbf{I}(a^{\text{true}} \in \{\boldsymbol{x}\}) + \gamma Q_y(s, \mu(s|\theta^{\mu'})|\theta^{Q'_y}) \cdot \mathbf{I}(a^{\text{true}} \in \{\boldsymbol{y}\})$
        Update θ^{Q} by minimizing the loss $\mathcal{L}^{Q} = \frac{1}{m} \sum_{x=(\bar{s},\bar{a},\cdot) \in \mathbf{B}} (\hat{r}_x - Q(\bar{s}, \bar{a}|\theta^{Q}))^2$
        Update θ^{μ} with deep deterministic policy gradient:
            $\nabla_{\theta^{\mu}} J \approx -\frac{1}{m} \sum_{(\bar{s},\cdot) \in \mathbf{B}} \nabla_a Q(s, a|\theta^{Q})|_{s=\bar{s}, a=\mu(\bar{s})} \circ \nabla_{\theta^{\mu}} \mu(s|\theta^{\mu})|_{s=\bar{s}}$
        Update target networks $\theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'}$ and $\theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'}$
        Construct batch $\hat{\mathbf{B}}$ of size m from $\boldsymbol{x}$ or $\boldsymbol{y}$ or $\boldsymbol{z}$, mimicking the membership of v in $\mathcal{X} \cup \mathcal{Y} \cup \mathcal{Z}$. Videos in $\hat{\mathbf{B}}$ are of type $\boldsymbol{x}_{:t}$ or $\boldsymbol{y}_{:t}$ or $\boldsymbol{z}_{:t}$ for randomly selected values of t
        Let $\hat{\mathbf{B}}_1$ be the set of those videos in $\hat{\mathbf{B}}$ with $t = T$, $\hat{\mathbf{B}}_2 = \hat{\mathbf{B}} \setminus \hat{\mathbf{B}}_1$
        Update θ^{D} by gradient $\dfrac{\partial L}{\partial \theta^{D}}$ with respect to $\hat{\mathbf{B}}_1$
        Update θ^{D} by gradient $\dfrac{\partial (L - L_v)}{\partial \theta^{D}}$ with respect to $\hat{\mathbf{B}}_2$

Table 2. Data examples of the artificial dataset

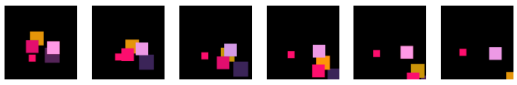
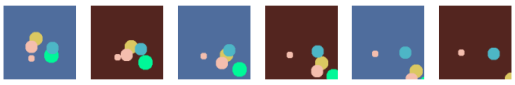
DESCRIPTION	SOURCE DOMAIN RECTANGLES MOVING IN BLACK BACKGROUND	TARGET DOMAIN CIRCLES MOVING IN CHANGING BACKGROUNDS
VIDEO DATA	 $\mathcal{X}$, 600 FRAMES OF COLORFUL RECTANGLES	 $\mathcal{Y}$, 100 FRAMES, COLORFUL CIRCLES  $\mathcal{Z}$, 100 FRAMES, GRAY CIRCLES
IMAGE DATA	NONE	 $\bar{\mathcal{Z}}$, 100 IMAGES, COLOR CIRCLES

Table 3. Data examples of the flower dataset

DESCRIPTION	SOURCE DOMAIN RED FLOWERS BLACK BACKGROUND	TARGET DOMAIN YELLOW FLOWER IN NATURE
VIDEO DATA	 $\mathcal{X}$, 550 FRAMES	 $\mathcal{Y}$, 100 FRAMES, IN NATURE  $\mathcal{Z}$, 900 FRAMES, IN BLACK
IMAGE DATA	NONE	 $\bar{\mathcal{Z}}$, 2000 IMAGES, IN NATURE

Table 4. Data examples of the city aerial dataset

DESCRIPTION	SOURCE DOMAIN MODERN CITIES IN DAYTIME	TARGET DOMAIN SMALL-TOWN IN NIGHTTIME
VIDEO DATA	 $\mathcal{X}$, 2000 FRAMES	 $\mathcal{Y}$, 500 FRAMES, SMALL-TOWN NIGHTTIME  $\mathcal{Z}$, 1000 FRAMES, SMALL-TOWN DAY-TIME
IMAGE DATA	NONE	 $\bar{\mathcal{Z}}$, 1500 IMAGES, MODERN CITY NIGHTTIME

Table 5. Description of the proprietary dataset source domain

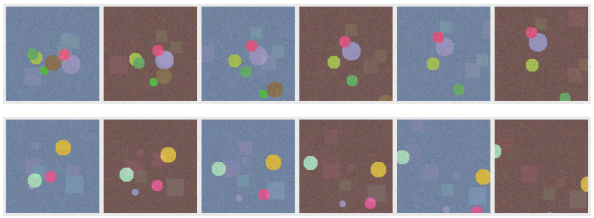
DESCRIPTION	SOURCE DOMAIN SYNTHETIC
VIDEO DATA	$\mathcal{X}$, TYPE A, 2,000 FRAMES
IMAGE DATA	NONE

Table 6. Description of the proprietary dataset target domain

DESCRIPTION	TARGET DOMAIN REAL WORLD
VIDEO DATA	$\mathcal{Y}$, TYPE A, 500 FRAMES $\mathcal{Z}$, TYPE B, 1,000 FRAMES
IMAGE DATA	$\bar{\mathcal{Z}}$, TYPE A, 4,000 IMAGES

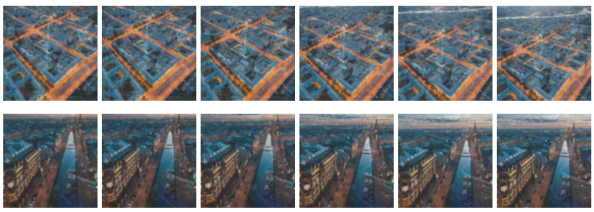
Table 7. Generated Samples from RL-V2V-GAN

ARTIFICIAL DATASET



FLOWER DATASET

CITY AERIAL DATASET



of our proposed model was performed using the stochastic gradient descent optimizer with a batch size of 64. Training was executed over 100 epochs, with an early stopping strategy that halts training if the validation set FID score (Karras et al., 2019) does not decrease for 5 consecutive epochs to prevent overfitting. We initially set the learning rate at 0.0001, applying a decaying schedule that reduces it by a factor of 10 every 10 epochs to refine parameter adjustments. The optimizer configuration included a weight decay of $3 \cdot 10^{-3}$ and a momentum of 0.97, optimizing the convergence process. We use gradient clipping. To update the target networks, we use soft updates with τ to control the degree of the update. Specifically, τ determines the interpolation between the current parameters of the target networks and the learned parameters. In our experiments, we chose $\tau = 0.005$.

The encoder contained 3 RPB layers and a dense layer. Each RPB block took an input tensor of size 256 x 256 x 3. The ResConvLSTM layer employed a 3 x 3 kernel, a stride of 1, and had 64 hidden states. The subsequent ResConvLSTM used the same kernel size, padding, and stride but contained 32 hidden states. The output tensor after the residual connection was of size 256 x 256 x 16, which was then reduced to 128 x 128 x 16 after pooling. The output tensor was further reduced to 32 x 32 x 16 after the third RPB block. This tensor was mapped to a 4,000-entry embedding vector by a dense layer.

For the decoders, the URB block's embedding vector was first transformed by a dense layer to 32 x 32 x 16 before entering the first URB block. Each URB block's ResConvLSTM utilized a 3 x 3 kernel, a stride of 1, and had 32 hidden states. The final ConvLSTM layer in the decoder had 3 filters, producing an output tensor of size 256 x 256 x 3. The discriminators and Q-networks have two last, fully-connected layers of 1,000 and 200 neurons.

In our experiments, the values of the lambda parameters λ_{rrx}, λ_{rry}, λ_{rcx}, λ_{rcy}, λ_{vx}, and λ_{vy} were selected based on a combination of empirical tuning and theoretical considerations. For the artificial and proprietary datasets, we set λ_{rrx} and λ_{rry} to 1 to ensure strong temporal coherence in the predicted frames. The values of λ_{rcx} and λ_{rcy} were set to 0.5 to balance the ReCycle losses, promoting consistency in style and domain transitions. For the video loss terms, λ_{vx} and λ_{vy} were set to 1, giving a moderate weight to overall video quality while allowing adversarial and recurrent losses to dominate. For the flower dataset, we used λ_{rrx} and λ_{rry} values of 1, λ_{rcx} and λ_{rcy} values of 0.125, and λ_{vx} and λ_{vy} values of 0.5. For the city aerial dataset, we used λ_{rrx} and λ_{rry} values of 1, λ_{rcx} and λ_{rcy} values of 0.5, and λ_{vx} and λ_{vy} values of 0.5. These values were determined through grid search to optimize the validation set FID scores. In our numerical experiments, the discount factor γ is set

to 0.99 to prioritize long-term rewards and account for the sequential nature of video frames.

The raw videos varied in length and resolution. We normalized them to 30 frames per second, cropped to the center 256 x 256 squares, and standardized the values on each channel to have be mean zero and unit variance across all videos. This standardized the data in terms of frame size, rate, and value ranges to effectively train deep learning models on the dataset. For the purpose of our numerical experiments, we set the video length to $T = 6$, as this allows the model to focus on key transitions and avoid overfitting to long, potentially redundant sequences, while still capturing essential temporal dynamics efficiently. Longer sequences can be supported by the algorithm as needed. All hyper parameters were either set based on experience or were partially ad-hoc optimized. In our numerical experiments, the capacity of reply buffer $\mathcal{B}$ is $10,000$.

The training process involves two key phases: pre-training and reinforcement learning fine-tuning. In the pre-training phase, the autoencoder behind generators G_x, G_y, and predictors P_x and P_y are trained from scratch using adversarial loss, recurrent loss, and recycle loss. The autoencoder is trained for up to 50 epochs, with early stopping based on the FID score from the validation set. After pre-training convergence, we introduce the proposed video loss function and continue training the model parameters using reinforcement learning. The pre-trained model is fine-tuned for 100 additional epochs or until early stopping criteria are met.

6.3. Benchmark and Evaluation

In this study, we evaluated the performance of our proposed model against two state-of-the-art algorithms, RecycleGAN and MoCoGAN, on various benchmark datasets. RecycleGAN is an unsupervised video retargeting that seamlessly integrates spatial and temporal information with adversarial and recycle losses. MoCoGAN, on the other hand, is a motion and content decomposed generative adversarial network framework for video generation. It maps a sequence of random vectors to a sequence of video frames, with each vector consisting of a content and motion part, and is capable of generating videos with interchangeable content and motion.

In our evaluation, we use the Fréchet Inception Distance (FID) to assess our model's performance, as it comprehensively evaluates the quality of generated videos by comparing feature vectors from generated and real videos using a pre-trained Inception v3 network. FID's ability to measure differences in distribution means and covariances has made it the preferred metric, overcoming limitations of other measures. The lower the FID, the better the model's performance.

Table 8. FID Score for generated samples from all datasets.

DATA SET	RECYCLEGAN	MOCOGAN	RL-V2V-GAN
ARTIFICIAL	9.9	9.7	6.2
PROPRIETARY	8.3	8.0	7.3
FLOWER	6.9	8.8	6.9
CITY AERIAL	7.8	8.3	7.5

6.4. Results

Our results, as shown in Table 8, indicate that RL-V2V-GAN performs the best among the three models on the artificial, proprietary, and aerial datasets, with FID scores of 6.2, 7.3, and 7.5 respectively. Meanwhile, on the flower dataset, both RL-V2V-GAN and ReCycleGAN perform similarly with an FID score of 6.9. In general, the results demonstrate the superiority of our proposed RL-V2V-GAN model, which consistently outperforms the benchmarks and has a lower FID score, indicating higher-quality results. These findings validate the effectiveness of our proposed method in transforming videos from one domain to another in an unsupervised manner. For inference, the dataset is split 70/15/15 into the train, validation, and test sets. The test set is evaluated with the FID score.

Table 9. Computational time for each model

DATA SET	MODEL	RUNTIME PER EPOCH (MIN)	EPOCHS TO CONVERGE	RUNTIME TO CONVERGE (MIN)
ARTIFICIAL	MOCOGAN	2	20	40
	RECYCLEGAN	3	27	81
	RL-V2V-GAN	4	6	24
PROPRIETARY	MOCOGAN	24	25	600
	RECYCLEGAN	35	31	1085
	RL-V2V-GAN	47	16	752
FLOWER	MOCOGAN	13	35	455
	RECYCLEGAN	20	44	880
	RL-V2V-GAN	25	23	575
CITY AERIAL	MOCOGAN	18	32	576
	RECYCLEGAN	27	40	1080
	RL-V2V-GAN	35	18	630

Table 9 presents the computational time for each model across four different datasets. The results indicate that while the RL-V2V-GAN model generally requires more time per epoch compared to MoCoGAN and ReCycleGAN, it significantly reduces the total runtime to convergence due to requiring fewer epochs. For instance, in the artificial dataset, RL-V2V-GAN converges in only 24 minutes, compared to 40 minutes for MoCoGAN and 81 minutes for ReCycleGAN. Similarly, for the proprietary dataset, RL-V2V-GAN converges in 752 minutes, whereas MoCoGAN and ReCycleGAN take 600 and 1085 minutes, respectively. This trend is consistent across all datasets, demonstrating that RL-V2V-GAN's efficient learning process results in quicker convergence. The proposed model's ability to achieve high-quality results in fewer epochs highlights its superior performance

and efficiency in video-to-video synthesis tasks.

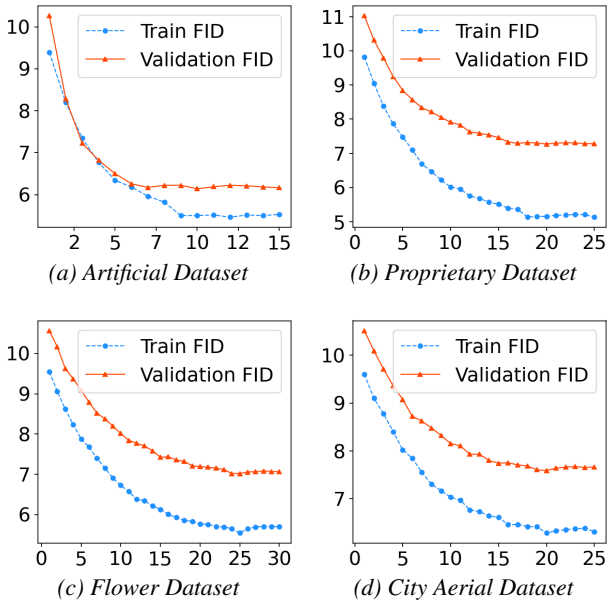


Figure 6. Train and validation FID scores over reinforcement learning epochs for the Artificial, Proprietary, Flower, and City Aerial datasets. Each subplot shows the training (blue) and validation (orange) FID scores, highlighting the model's learning dynamics and generalization capabilities.

Figure 6 shows the training and validation FID scores as a function of reinforcement learning epochs for the Artificial, Proprietary, Flower, and City Aerial datasets. Across all datasets, the training FID scores (blue curves) decrease rapidly initially, indicating efficient learning. The validation FID scores (orange curves) generally follow a similar trend, suggesting good generalization. The consistent gap between training and validation FID scores in each dataset indicates that the model avoids overfitting. Particularly, the Artificial and Flower datasets show significant early improvements, while the Proprietary and City Aerial datasets demonstrate steady progress. These results highlight the RL-V2V-GAN model's ability to learn and maintain performance across diverse datasets, producing high-quality video synthesis.

6.4.1. Reinforcement Learning Training Strategies

In Figure 7, we compare the performance of three distinct reinforcement learning training schemes: standard Q-learning, twin Q-network, and delayed policy update methods. This study is conducted on the city aerial dataset. The standard Q-learning approach updates the value of an action in a state by using the weighted average of the current value and the newly acquired information. Although this method showed promising results in the early epochs, it led to the lowest Q values upon convergence. The twin Q-network approach

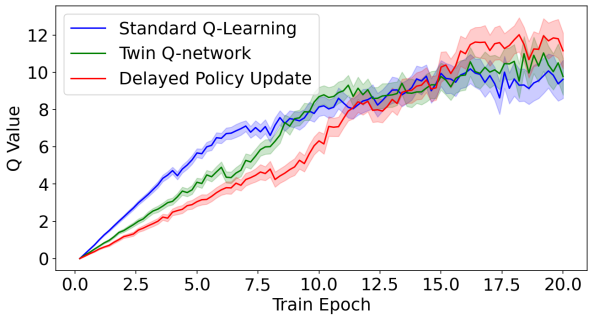


Figure 7. Reward progression for standard Q-learning, twin Q-network, and delayed policy update. Shaded regions indicate the variability (standard deviation) across multiple (n=10) independent runs.

employs two separate Q-value estimators to provide unbiased Q-value estimates, achieving moderate Q values at both the initial and final epochs. Finally, the delayed policy update method integrates techniques from clipped double-Q learning and target policy smoothing. Initially, it displayed the lowest Q values, but it ultimately reached the highest Q values at convergence, illustrating its potential to enhance stability in traditional reinforcement learning training while decreasing the time required for convergence. We use the delayed policy update method in our main computation results due to its superior performance in achieving higher Q values and enhancing training stability.

6.5. Ablation Studies

We set up an ablation study centered on evaluating the efficacy of disparate training methodologies applied to our proposed model. The results in Table 10 show the comparative impacts of pre-training across all four datasets. The first column in the table shows the results of training the model from scratch without any pre-training. The second column shows the results of using pre-trained models. The results show that pre-training the model can significantly improve its performance, as demonstrated by the lower FID scores in the second column compared to the first.

Table 10. FID score for comparing training from scratch vs using pre-trained weights.

Data set	No Pre-train	Pre-trained
Artificial	11.2	6.2
Proprietary	9.5	7.3
Flower	10.6	6.9
Aerial	11.9	7.5

This ablation study demonstrates that pre-training is an effective technique for improving the performance of our proposed model. By providing better initial parameters, pre-training speeds up convergence and improves performance, as shown by the lower FID scores.

We further conducted an ablation study to investigate the impact of different loss components on the performance of our RL-V2V-GAN model. Specifically, we analyzed the effect of setting each type of loss — adversarial, recurrent, recycle, and video — to zero. The FID scores for three datasets (Artificial, City Aerial, and Flower) under all four scenarios are shown in Table 11.

Table 11. FID scores for ablation study on the effect of removing different types of losses.

	Dataset		
Loss Type	Artificial	City Aerial	Flower
All Losses	6.2	7.5	6.9
No Adversarial Loss	8.4	8.9	8.1
No Recurrent Loss	9.3	10.2	9.1
No Recycle Loss	8.7	10.1	8.9
No Video Loss	7.5	9.4	8.5

The results indicate that the removal of any loss component leads to a deterioration in performance, as evidenced by higher FID scores across all datasets. The first row in the table represents the performance of the model when all four loss components — adversarial, recurrent, recycle , and video — are included, serving as the baseline for comparison. For the Artificial dataset, the FID scores increased from 6.2 to 8.4, 9.3, 8.7, and 7.5 when the adversarial, recurrent, recycle, and video losses were removed, respectively, demonstrating the importance of each loss component. Similarly, for the City Aerial dataset, FID scores rose from 7.5 to 8.9, 10.2, 10.1, and 9.4, and for the Flower dataset, from 6.9 to 8.1, 9.1, 8.9, and 8.5 for the removal of the same losses.

Overall, this ablation study confirms the necessity of each loss component in our RL-V2V-GAN model to ensure the generation of temporally coherent, stylistically consistent, and high-quality videos. However, the video loss component appears to have a slightly lesser impact on quality compared to the recurrent and recycle losses.

7. Limitation and Future Directions

The RL-V2V-GAN architecture, while demonstrating significant potential in the domain of video retargeting, is confronted with certain constraints that merit attention. A notable challenge lies in the model's current constraint of generating singular style outputs from a given input image. This limitation suggests an avenue for enhancement through the integration of a one-to-many translation model within the framework. Such an advancement would not only elevate the output's diversity but also its creative dimension, paving the way for a richer variety of video retargeting possibilities.

Moreover, the capability of the model to process a limited quantity of video frames emerges as a constraint, rooted in the extensive computational requirements necessitated by the high-dimensional nature of video data within the neural network architecture. Given the inherently multimodal character of videos, which frequently encompasses both visual and audio streams, this limitation underscores the complexity of video processing. A promising direction for future research involves the development of a multimodal model adept at assimilating various data inputs, including audio. This approach holds the potential to refine the model's output, fostering a more comprehensive and nuanced interpretation of the data, thereby enriching the resultant video retargeting outcomes.

8. Conclusion

This work introduces RL-V2V-GAN, a model in the realm of unsupervised video-to-video synthesis, leveraging the strength of reinforcement learning and GANs for the generation of temporally coherent video sequences in a target style from limited source material. Our approach effectively mitigates the challenge of sparse data in the target domain, demonstrating a significant advancement over existing methods through its ability to learn and replicate complex video styles without requiring paired input videos.

The experimental outcomes underscore the model's superior performance across diverse datasets, as evidenced by favorable FID scores when benchmarked against state-of-the-art models like ReCycleGAN and MoCoGAN. This success is attributed to our model's innovative integration of reinforcement learning, which enables nuanced temporal dynamics modeling, and a carefully designed GAN architecture that ensures style consistency and content fidelity.

RL-V2V-GAN's efficiency in synthesizing high-quality video content opens new avenues for applications in areas demanding robust video manipulation capabilities, such as content creation, multimedia editing, and virtual reality environments. By achieving a delicate balance between generative flexibility and output coherence, this work opens up new opportunities in the field of video generation and expands the current capabilities of generative models, paving the way for future research to explore even more sophisticated models within this promising domain.

References

Bandi, A., Adapa, P. V. S. R., and Kuchi, Y. E. V. P. K. The Power of Generative AI: A Review of Requirements, Models, Input–Output Formats, Evaluation Metrics, and Challenges. *Future Internet*, 15(8):260, 2023.

Bansal, A., Ma, S., Ramanan, D., and Sheikh, Y. Recycle-GAN: Unsupervised Video Retargeting. In *Proceedings of the European Conference on Computer Vision*, pp. 119–

135, 2018.

Cao, C., Hou, Q., and Zhou, K. Displaced Dynamic Expression Regression For Real-time Facial Tracking And Animation. *ACM Transactions on Graphics*, 33(4):1–10, 2014.

Carl, V., Torralba, A., and Hamed, P. Anticipating Visual Representations from Unlabeled Video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 98–106, 2016.

Chen, D., Liao, J., Yuan, L., Yu, N., and Hua, G. Coherent Online Video Style Transfer. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2017-October, pp. 1114–1123, 2017a.

Chen, L.-C., Papandreou, G., Schroff, F., and Adam, H. Rethinking Atrous Convolution for Semantic Image Segmentation. In *Computing Research Repository*, 2017b.

Chen, X., Wang, Y., Zhang, L., Zhuang, S., Ma, X., Yu, J., Wang, Y., Lin, D., Qiao, Y., and Liu, Z. SEINE: Short-to-Long Video Diffusion Model for Generative Transition and Prediction. In *The 20th International Conference on Learning Representations*, 2023.

Escontrela, A., Adeniji, A., Yan, W., Jain, A., Peng, X. B., Goldberg, K., Lee, Y., Hafner, D., and Abbeel, P. Video Prediction Models as Rewards for Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 68760–68783, 2023.

Esser, P., Chiu, J., Atighehchian, P., Granskog, J., and Germanidis, A. Structure and Content-Guided Video Synthesis with Diffusion Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7346–7356, 2023.

Francisco, S., Silberman, N., and Dohan, D. Unsupervised Pixel Level Domain Adaptation with Generative Adversarial Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Gupta, A., Johnson, J., Alahi, A., and Fei-Fei, L. Characterizing and Improving Stability in Neural Style Transfer. In *Proceedings of the IEEE International Conference on Computer Vision*, volume October, pp. 4087–4096, 2017.

He, J., Lehrmann, A., Marino, J., Mori, G., and Sigal, L. Probabilistic Video Generation using Holistic Attribute Control. In *Proceedings of the European Conference on Computer Vision*, pp. 452–467, 2018.

Huang, H., Wang, H., Luo, W., and Ma, L. Real-Time Neural Style Transfer for Videos. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 783–791, 2017.

Huang, X., Liu, M.-Y., Belongie, S., and Kautz, J. Multimodal Unsupervised Image-to-Image Translation. In *In Proceedings of the European Conference on Computer Vision*, pp. 172–189, 2018.

Isola, P., Zhu, J. Y., Zhou, T., and Efros, A. A. Image-to-image Translation With Conditional Adversarial Networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume January, pp. 5967–5976, 2017.

Karras, T., Laine, S., and Aila, T. A Style-Based Generator Architecture for Generative Adversarial Networks. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume June, pp. 4396–4405, 2019.

Kim, G., Shim, H., Kim, H., Choi, Y., Kim, J., and Yang, E. Diffusion Video Autoencoders: Toward Temporally Consistent Face Video Editing via Disentangled Video Encoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6091–6100, 2023.

Liu, M.-Y. and Tuzel, O. Coupled Generative Adversarial Networks. In *Advances in Neural Information Processing Systems*, 2016.

Liu, M.-Y., Breuel, T., and Kautz, J. Unsupervised Image-to-Image Translation Networks. In *Advances in Neural Information Processing Systems*, 2017.

Ma, D., Zhang, F., and Bull, D. R. CVEGAN: A perceptually-inspired GAN for Compressed Video Enhancement. *Signal Processing: Image Communication*, 127:117127, September 2024.

Mathieu, M., Couprie, C., and LeCun, Y. Deep Multi-scale Video Prediction Beyond Mean Square Error. In *Proceedings of International Conference on Learning Representations*, pp. 1386–1425, 2015.

Ruder, M., Dosovitskiy, A., and Brox, T. Artistic Style Transfer For Videos. In *Proceedings of Pattern Recognition: 38th German Conference*, pp. 26–36, 2016.

Saito, M., Matsumoto, E., and Saito, S. Temporal Generative Adversarial Nets with Singular Value Clipping. In *Proceedings of the IEEE International Conference on Computer Vision*, volume October, pp. 2849–2858, 2017.

Shen, X., Li, X., and Elhoseiny, M. MoStGAN-V: Video Generation With Temporal Motion Styles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5652–5661, 2023.

Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., and Webb, R. Learning From Simulated and Unsupervised Images Through Adversarial Training. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Taigman, Y., Polyak, A., and Wolf, L. Unsupervised Cross-domain Image Generation. In *International Conference on Learning Representations*, 2017.

Thies, J., Zollhofer, M., Stamminger, M., Theobalt, C., and NieBner, M. Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2387–2395, 2016.

Tulyakov, S., Liu, M.-Y., Yang, X., and Kautz, J. MoCoGAN: Decomposing Motion and Content for Video Generation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Villegas, R., Yang, J., Hong, S., Lin, X., and Lee, H. Decomposing Motion and Content for Natural Video Sequence Prediction. In *International Conference on Learning Representations*, 2017.

Walker, J., Marino, K., Gupta, A., and Hebert, M. The Pose Knows: Video Forecasting by Generating Pose Futures. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3352–3361, 2017.

Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Liu, G., Tao, A., Kautz, J., and Catanzaro, B. Video-to-Video Synthesis. In *Proceedings of International Conference on Neural Information Processing Systems*, pp. 1152–1164, 2018.

Wexler, Y., Shechtman, E., and Irani, M. Space-Time Completion of Video. *Pattern Analysis and Machine Intelligence, IEEE Transactions*, 29(3):463–476, 2007.

Yang, R., Srivastava, P., and Mandt, S. Diffusion Probabilistic Modeling for Video Generation. *Entropy*, 25(10): 1469, October 2023.

Yu, L., Zhang, W., Wang, J., and Yu, Y. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, pp. 2852–2858, 2016.

Yu, S., Sohn, K., Kim, S., and Shin, J. Video Probabilistic Diffusion Models in Projected Latent Space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18456–18466, 2023.

Zeng, Y., Wei, G., Zheng, J., Zou, J., Wei, Y., Zhang, Y., and Li, H. Make pixels dance: High-dynamic video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8850–8860, 2024.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. In *IEEE International Conference on Computer Vision*, 2017a.

Zhu, J.-Y., Zhang, R., Pathak, D., Darrell, T., Efros, A. A., Wang, O., Research, A., and Shechtman, E. Toward Multimodal Image-to-Image Translation. In *Advances in Neural Information Processing Systems*, pp. 465–476, 2017b.

Zhuo, L., Wang, G., Li, S., Wu, W., and Liu, Z. Fast-Vid2Vid: Spatial-Temporal Compression for Video-to-Video Synthesis. In *Proceedings of European Conference on Computer Vision 2022*, pp. 289–305, 2022.

Appendix: Notation Table

VARIABLE	DESCRIPTION	DIMENSION
N	NUMBER OF VIDEOS IN A DOMAIN	(1)
T	NUMBER OF FRAMES IN A FULL VIDEO	(1)
W	WIDTH OF A FRAME	(1)
H	HEIGHT OF A FRAME	(1)
$\mathcal{X}$	SOURCE DOMAIN VIDEOS	(N, T, W, H)
$\mathcal{Y}$	TARGET DOMAIN VIDEOS	(N, T, W, H)
$\mathcal{Z}$	TARGET DOMAIN STYLE VIDEOS	(N, T, W, H)
$\bar{\mathcal{Z}}$	TARGET DOMAIN STYLE IMAGES	(N, W, H)
$\boldsymbol{x}_{:t} \in \mathcal{X}$	VIDEO SEQUENCE FROM SOURCE DOMAIN	(T, W, H)
$\boldsymbol{y}_{:t} \in \mathcal{Y}$	VIDEO SEQUENCE FROM TARGET DOMAIN	(T, W, H)
$\boldsymbol{z}_{:t} \in \mathcal{Z}$	STYLE VIDEO SEQUENCE FROM TARGET DOMAIN	(T, W, H)
$\boldsymbol{x}_t \in \mathcal{X}$	FRAME FROM A VIDEO IN THE SOURCE DOMAIN	(W, H)
$\boldsymbol{y}_t \in \mathcal{Y}$	FRAME FROM A VIDEO IN THE TARGET DOMAIN	(W, H)
$\bar{z} \in \bar{\mathcal{Z}}$	STYLE IMAGE FROM TARGET DOMAIN	(W, H)
s	STATE, SEQUENCE OF FRAMES	(T, W, H)
a	ACTION, NEXT FRAME OF STATE s	(W, H)
r	REWARD FOR THE ACTION TAKEN	(1)
s'	NEXT STATE AFTER TAKING ACTION a	(T+1, W, H)
μ	POLICY NETWORKS $\mu = \{G_x, G_y, P_x, P_y\}$	(T, W, H) $\rightarrow$ (W, H)
G_x	GENERATOR FROM TARGET TO SOURCE DOMAIN: $\boldsymbol{y} \rightarrow \boldsymbol{x}$	(T, W, H) $\rightarrow$ (T, W, H)
G_y	GENERATOR FROM SOURCE TO TARGET DOMAIN: $\boldsymbol{x} \rightarrow \boldsymbol{y}$	(T, W, H) $\rightarrow$ (T, W, H)
P_x	PREDICTOR IN SOURCE DOMAIN: $\boldsymbol{x}_{:t} \rightarrow \boldsymbol{x}_{t+1}$	(T, W, H) $\rightarrow$ (W, H)
P_y	PREDICTOR IN TARGET DOMAIN: $\boldsymbol{y}_{:t} \rightarrow \boldsymbol{y}_{t+1}$	(T, W, H) $\rightarrow$ (W, H)
$Q(s, a)$	Q-NETWORK	(T, W, H) $\rightarrow$ (1)
$D(s, a)$	DISCRIMINATOR	(T, W, H) $\rightarrow$ (1)
D_x	DISCRIMINATOR FOR SOURCE DOMAIN FRAMES: $x \rightarrow [0, 1]$	(W, H) $\rightarrow$ (1)
$D_{\boldsymbol{x}}$	DISCRIMINATOR FOR SOURCE DOMAIN VIDEOS: $\boldsymbol{x} \rightarrow [0, 1]$	(T, W, H) $\rightarrow$ (1)
D_y	DISCRIMINATOR FOR TARGET DOMAIN FRAMES: $y \rightarrow [0, 1]^2$	(W, H) $\rightarrow$ (2)
$D_{\boldsymbol{y}}$	DISCRIMINATOR FOR TARGET DOMAIN VIDEOS: $\boldsymbol{y} \rightarrow [0, 1]^2$	(T, W, H) $\rightarrow$ (2)