Non-Adversarial Video Synthesis with Learned Priors

Abhishek Aich[†,*], Akash Gupta[†,*], Rameswar Panda[‡], Rakib Hyder[†],
Salman Asif[†], Amit K. Roy-Chowdhury[†]
University of California, Riverside[†], IBM Research[‡]
{aaich001@, agupt013@, rpand002, rhyde001@, sasif@ece, amitrc@ece}.ucr.edu

Abstract

Most of the existing works in video synthesis focus on generating videos using adversarial learning. Despite their success, these methods often require input reference frame or fail to generate videos from all modes of training data distribution, with little to no control over the quality of videos that can be generated. Different from these methods, we focus on the problem of generating videos from latent noise vectors, without any reference input frames. To this end, we develop a novel approach that jointly optimizes the input latent space, the weights of a generator and a recurrent neural network without a discriminator through non-adversarial learning. Optimizing for the input latent space along with the network weights allows us to generate videos in a controlled environment, i.e., we can faithfully generate all videos the model has seen during the learning process as well as new unseen videos. Extensive experiments on three challenging and diverse datasets well demonstrate that our approach generates superior quality videos compared to the existing state-of-the-art methods.

1. Introduction

Video synthesis is an open and challenging problem in computer vision. As literature suggests, a deeper understanding of how frame sequences behave spatially and temporally can directly provide insights in choosing priors, future prediction, and feature learning [33, 35]. Much progress has been made in developing a variety of ways to generate videos which can be mainly classified into broadly two categories: video generation frameworks which require random latent vectors without any reference input [33, 26], and class of video generation methods which does require a reference input [35, 10]. Current literature is mostly encompassed by methods from second category which often require some human intervention [35, 10]. In general, Generative adversarial networks (GANs) [8] have shown remarkable success in various kinds of video modality problems [17, 14, 19, 26]. The first category of video generation frameworks, which tackles comparatively more difficult problem, pre-dominantly uses GANs to synthesize videos from latent noise vectors. For example, VGAN [33] and TGAN [26] proposed generative models that synthesize videos from random latent vectors with deep convolutional GAN. Recently, MoCoGAN [32] proposed to decompose a video into content and motion parts using a generator guided by two discriminators. During testing, these frameworks generate videos that is captured in the range of the trained generator, by taking random latent vectors. While all these methods have obtained reasonable performance on commonly used benchmark datasets, they utilize adversarial learning to train their models and hence, inherit the shortcomings of GANs. Specifically, GANs are often very sensitive to multiple factors such as random network initialization, and type of layers employed to build the network [15, 27]. Some infamous drawbacks of GANs are mode-collapse (i.e., able to generate only some parts of the data

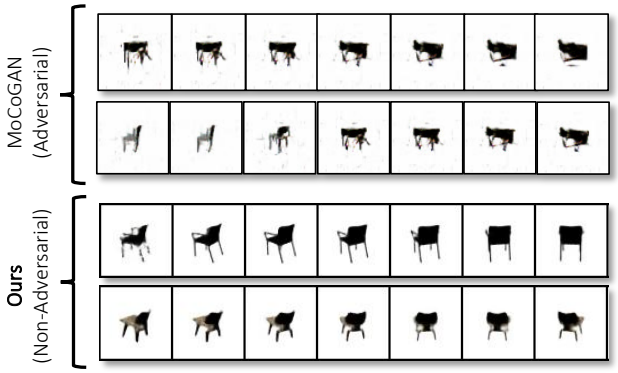


Figure 1: **Comparison of proposed non-adversarial approach to one representative adversarial approach (MoCoGAN [32]) on the Chair-CAD [2] dataset.** *Top*: MoCoGAN often generates blurry frames including similar type of chairs for different videos as the time step increases. *Bottom*: Our approach, on the other hand, generates relatively sharper frames, maintaining consistency with the type of chairs unique to each video in the dataset.

[*]Joint first authors

distribution: see Fig. 1 for an example) and/or vanishing generator gradients due to discriminator being way better in distinguishing fake samples and real samples [1].

Non-adversarial approaches [4, 16, 11] have recently been explored to tackle these challenges. For example, Generative Latent Optimization (GLO) [4] and Generative Latent Nearest Neighbor (GLANN) [11] investigate the importance of inductive bias in convolutional networks by disconnecting the discriminator for a non-adversarial learning protocol of GANs. These works show that without a discriminator, a generator can be learned to map the training images in the given data distribution to a lower dimensional latent space that is learned in conjunction with the weights of the generative network. Such procedure not only avoids the mode-collapse problem of the generators, but also provides the user an optimized low dimensional latent representation (embedding) of the data in contrast with the random latent space as in GANs. Recently Video-VAE [10] proposed to use variational auto-encoder for conditional video synthesis, either by randomly generating or providing the first frame to the model for generating a video. However, the quality of generated videos using Video-VAE often depends on the provided input frame. Non-adversarial video synthesis without any visual inputs still remains as a novel and rarely addressed problem.

In this paper, we propose a novel non-adversarial framework to generate videos in a controllable manner without any reference frame. Specifically, we propose to synthesize videos from two optimized latent spaces, one providing control over the static portion of the video (*static latent space*) and the other over the transient portion of the video (*transient latent space*). We propose to jointly optimize these two spaces while optimizing the network (a generative and a recurrent network) weights with the help of distance-based reconstruction loss, and a triplet loss.

Our approach works as follows. During training, we jointly optimize over network weights and latent spaces (both static and transient) and obtain a common transient latent space, and individual static latent space dictionary for all videos sharing the same class (see Fig. 2). During testing, we randomly choose a static vector from the dictionary, concatenate it with the transient latent vector and generate a video. This enables us to obtain a controlled environment of video generation from learned latent vectors for each video in the given dataset, while maintaining uniform fidelity. In addition, the proposed approach also allows frame interpolation, and generation of unseen videos during training.

The key contributions of our work are as follows.

- We propose a novel framework for generating videos from learned latent vectors without any conditional input reference frame covering all modes of data distribution with almost uniform qualitative fidelity. Our framework obtains a latent space dictionary on both static and transient for the training video dataset, which enables us to generate even unseen videos with almost equal quality by providing combinations of static and transient latent vectors that were not part of training data.
- Our extensive experiments on multiple datasets well demonstrate that the proposed method, without the adversarial training protocol, has better or at par performance with current state-of-the-art methods.

2. Related Works

Our work relates to two major research directions: video synthesis and non-adversarial learning. Here, we focus on some representative methods closely related to our work (see Tab. 1 for a categorization of existing methods).

2.1. Video Synthesis

Video synthesis has been studied from multiple perspectives [33, 26, 32, 10]. VGAN [33] demonstrates that a video can be divided into foreground and background using deep neural networks. TGAN [26] proposes to use a generator to capture temporal dynamics by generating correlated latent codes for each video frame and then using an image generator to map each of this latent code to a single frame for the whole video. MoCoGAN [32] presents a simple approach to separate content and motion latent codes of a video using adversarial learning. The most relevant work to ours is VideoVAE [10] that extends the idea of image generation to video generation using Variational AutoEncoder (VAE) by proposing a structured latent space in conjunction with the VAE architecture for video synthesis. While this method has achieved reasonable performance, it depends on reference input frame to generate a video. In contrast, our method proposes a efficient framework for synthesizing videos from learnable latent vectors without any input frame. This gives a controlled environment for video synthesis that even enables us to generate visually good quality unseen videos through combining static and transient parts.

2.2. Non-adversarial Learning

Generative adversarial networks, as powerful as they are in pixel space synthesis, are also difficult to train. This is owing to the saddle-point based optimization game between the generator and the discriminator. On top of the challenges discussed in the previous section, GANs require careful user driven configuration tuning which may not guarantee same performance for every run. Some techniques to make the generator agnostic to described problems have been discussed in [27]. The other alternative to the same has given rise to non-adversarial learning of generative networks [4, 11]. Both [4, 11] showed that properties of GANs (convolutional networks) can be mimicked using

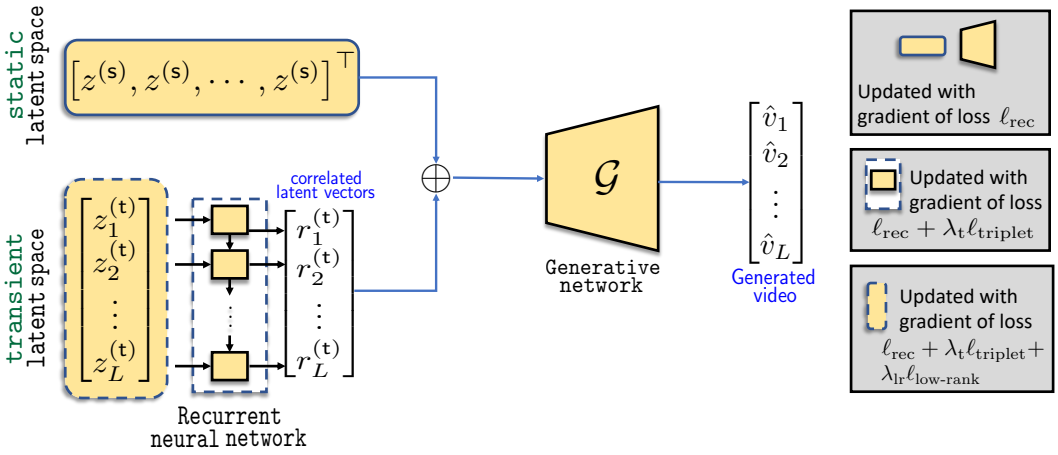


Figure 2: **Overview of the proposed method.** Videos can be broken down into two main parts: static and transient components. To capture this, we map a video (with L frame sequence) into two learnable latent spaces. We jointly learn the static latent space and the transient latent space along with the network weights. We then use these learned latent spaces to generate videos at the inference time. See Sec. 3 for more details.

Methods	Settings		
	Adversarial learning?	Input frame?	Input latent vectors?
VGAN [33]	✓	✗	✓(random)
TGAN [26]	✓	✗	✓(random)
MoCoGAN [32]	✓	✗	✓(random)
Video-VAE [10]	✗	✓	✓(random)
Ours	✗	✗	✓(learned)

Table 1: **Categorization of prior works in video synthesis.** Different from existing methods, our model doesn't require a discriminator, or any reference input frame. However, since we have learned latent vectors, we have control of the kind of videos the model should generate.

simple reconstruction losses while discarding the discriminator.

While there has been some works on image generation from learned latent vectors [4, 11], our work significantly differs from these methods as we do not map all the frames pixel-wise in a given video to the same latent distribution. This is because doing so would require a separate latent space (hence, a separate model for each video) for all the videos in a given dataset, and performing any operation in that space would naturally become video specific. Instead, we divide the latent space of videos sharing the same class into two parts- static and transient. This gives us a dictionary of static latent vectors for all videos and a common transient latent subspace. Hence, any random video of the dataset can now be represented by the combination of one static vector (which remains same for all frames) and the common transient subspace.

3. Formulation

Define a video clip V represented by L frames as $\mathsf{V} = \left[v_1, v_2, \cdots, v_L\right]$. Corresponding to each frame, let there be a point in latent space $\mathcal{Z}_{\mathsf{V}} \in \mathbb{R}^D$ such that

$$\mathcal{Z}_{\mathsf{V}} = \left[z_1, z_2, \cdots, z_L\right] \tag{1}$$

which forms a path of length L. We propose to disentangle a video into two parts: a static constituent, which captures the constant portion of the video common for all frames, and a transient constituent which represents the temporal dynamics between all the frames in the video. Hence, let $\mathcal{Z}_{\mathsf{V}}$ be decomposed as $\mathcal{Z}_{\mathsf{V}} = \mathcal{Z}_{\mathsf{s}} \times \mathcal{Z}_{\mathsf{t}}$ where $\mathcal{Z}_{\mathsf{s}} \in \mathbb{R}^{D_{\mathsf{s}}}$ represents the static subspace and $\mathcal{Z}_{\mathsf{t}} \in \mathbb{R}^{D_{\mathsf{t}}}$ represents the transient subspace with $D = D_{\mathsf{s}} + D_{\mathsf{t}}$. Thus $\{z_i\}_{i=1}^{L}$ in (1) can be expressed as $z_i = \left[z_i^{(\mathsf{s})}, z_i^{(\mathsf{t})}\right]^\top \;\; \forall\, i = 1, 2, \cdots, L$. Next assuming that the video is of short length, we can fix $z_i^{(\mathsf{s})} = z^{(\mathsf{s})}$ for all frames after sampling only once. Therefore, (1) can be expressed as

$$\mathcal{Z}_{\mathsf{V}} = \left[\begin{bmatrix} z^{(\mathsf{s})} \\ z_1^{(\mathsf{t})} \end{bmatrix}, \begin{bmatrix} z^{(\mathsf{s})} \\ z_2^{(\mathsf{t})} \end{bmatrix}, \cdots, \begin{bmatrix} z^{(\mathsf{s})} \\ z_L^{(\mathsf{t})} \end{bmatrix}\right] \tag{2}$$

The transient portion will represent the motion a given video. Intuitively, the latent vectors corresponding to this transient state should be correlated, or in other words, will form a path between $z_1^{(\mathsf{t})}$ and $z_L^{(\mathsf{t})}$. Specifically, the frames in a video are correlated in time and hence a frame v_i at time $i = T$ is a function of all previous frames $\{v\}_{i=1}^{T-1}$. As a result, their corresponding transient representation should

also exhibit such a trajectory. This kind of representation of latent vectors can be obtained by employing a Recurrent Neural Network (RNN) where output of each cell of the network is a function of its previous state or input. Denoting the RNN as $\mathcal{R}$ with weights θ, then the RNN output $\mathcal{R}(z_i) = \{r_i^{(\mathsf{t})}\}\ \forall\ i = 1, 2, \cdots, L$ is a sequence of correlated variables representing the transient state of the video.

3.1. Learning Network Weights

Define a generative network $\mathcal{G}$ with weights represented by $\boldsymbol{\gamma}$. $\mathcal{G}$ takes latent vectors sampled from $\mathcal{Z}_{\mathsf{V}}$ as input and predicts $\leq L$ frames of the video clip. For a set of N videos, initialize set of D-dimensional vectors $\mathcal{Z}_{\mathsf{V}}$ to form the pair $\Big\{\big(\mathcal{Z}_{\mathsf{V}_1}, \mathsf{V}_1\big),\ \big(\mathcal{Z}_{\mathsf{V}_2}, \mathsf{V}_2\big),\ \cdots,\ \big(\mathcal{Z}_{\mathsf{V}_N}, \mathsf{V}_N\big)\Big\}$. More specifically from (2), defining $\mathbf{z}^{(\mathsf{s})} = \left[z^{(\mathsf{s})}, z^{(\mathsf{s})}, \cdots, z^{(\mathsf{s})}\right] \in \mathbb{R}^{D_{\mathsf{s}} \times L}$, and $\mathbf{z}^{(\mathsf{t})} = \left[z_1^{(\mathsf{t})}, z_2^{(\mathsf{t})}, \cdots, z_L^{(\mathsf{t})}\right] \in \mathbb{R}^{D_{\mathsf{t}} \times L}$, we will have the pairs

$$\left\{\left(\begin{bmatrix}\mathbf{z}^{(\mathsf{s})}\\ \mathbf{z}^{(\mathsf{t})}\end{bmatrix}_1, \mathsf{V}_1\right), \left(\begin{bmatrix}\mathbf{z}^{(\mathsf{s})}\\ \mathbf{z}^{(\mathsf{t})}\end{bmatrix}_2, \mathsf{V}_2\right), \cdots, \left(\begin{bmatrix}\mathbf{z}^{(\mathsf{s})}\\ \mathbf{z}^{(\mathsf{t})}\end{bmatrix}_N, \mathsf{V}_N\right)\right\}.$$

With these pairs, we propose to optimize the weights γ, θ, and input latent vectors $\mathcal{Z}_{\mathsf{V}}$ (sampled once in the beginning of training) in the following manner. Specifically, we jointly optimize for θ, γ, and $\{\mathcal{Z}_{\mathsf{V}_j}\}_{j=1}^{N}$ for every epoch in two stages:

$$\text{Stage 1}: \quad \min_{\gamma}\ \ell\big(\mathsf{V}_j, \mathcal{G}(\mathcal{Z}_{\mathsf{V}_j})|\big(\mathcal{Z}_{\mathsf{V}_j}, \theta\big)\big) \tag{3.1}$$

$$\text{Stage 2}: \quad \min_{\mathcal{Z}_{\mathsf{V}}, \theta}\ \ell\big(\mathsf{V}_j, \mathcal{G}(\mathcal{Z}_{\mathsf{V}_j})|\gamma\big) \tag{3.2}$$

Here, the index j represents a random video out of N videos chosen from the dataset. $\ell(\cdot)$ can be chosen to be any distance based loss. For rest of the paper, we will refer to both (3.1) and (3.2) together as $\min_{\mathcal{Z}_{\mathsf{V}}, \theta, \gamma} \ell_{\text{rec}}$.

Regularized loss function to capture static subspace. The transient subspace, along with the RNN, handles the temporal dynamics of the video clip. To equally capture the static portion of the video, we randomly choose a frame from the video and ask the generator to compare its corresponding generated frame during training. For this, we update the above loss as follows.

$$\min_{\mathcal{Z}_{\mathsf{V}}, \theta, \gamma}\ \left(\ell_{\text{rec}} + \lambda_{\mathsf{s}} \ell_{\text{static}}\right) \tag{4}$$

where $\ell_{\text{static}} = \ell(\hat{v}_k, v_k)$ with $k \in [1, 2, \cdots, L]$ is a randomly chosen index, v_k is the ground truth frame, $\hat{v}_k = \mathcal{G}(\mathbf{z}_k)$, and λ_{s} is the regularization constant. This regularized static loss can also be understood to essentially handle the role of image discriminator in [32, 34] that ensures that the reconstructed frame is close to the ground truth frame.

3.2. Learning Latent Spaces

Non-adversarial learning involves joint optimization of network weights as well as the corresponding input latent space. Apart from the gradients with respect to loss in (4), we propose to further optimize the latent space with gradient of a loss based on the triplet condition as follows.

3.2.1 The Triplet Condition

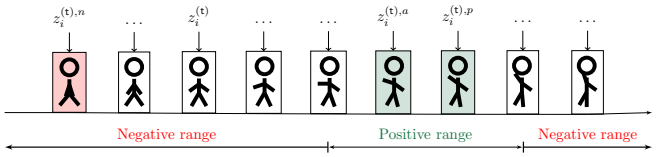


Figure 3: **Triplet Condition in the transient latent space.** Latent code representation of short video clips may lie very near to each other in the transient subspace. Using the proposed triplet condition, our model learns to explain the dynamics of similar looking frames and map them to distinct latent vectors.

Short video clips often have indistinguishable dynamics in consecutive frames which can force the latent code representations to lie very near to each other in the transient subspace. However, an ideal transient space should ensure that the latent vector representation of a frame should only be close to a similar frame than a dissimilar one [28, 29]. To this end, we introduce a triplet loss to (4) that ensures that a pair of co-occuring frames v_i^a (anchor) and v_i^p (positive) are closer but distinct to each other in embedding space than any other frame v_i^n (negative) (see Fig. 3). In this work, positive frames are randomly sampled within a margin range α of the anchor and negatives are chosen outside of this margin range. Defining a triplet set with transient latent code vectors $\{z_i^{(\mathsf{t}),a}, z_i^{(\mathsf{t}),p}, z_i^{(\mathsf{t}),n}\}$, we aim to learn the transient embedding space $\mathbf{z}^{(\mathsf{t})}$ such that

$$\|z_i^{(\mathsf{t}),a} - z_i^{(\mathsf{t}),p}\|_2^2 + \alpha < \|z_i^{(\mathsf{t}),a} - z_i^{(\mathsf{t}),n}\|_2^2$$

$\forall\ \{z_i^{(\mathsf{t}),a}, z_i^{(\mathsf{t}),p}, z_i^{(\mathsf{t}),n}\} \in \Gamma$, where Γ is the set of all possible triplets in $\mathbf{z}^{(\mathsf{t})}$. With the above regularization, the loss in (4) can be written as

$$\min_{\mathcal{Z}_{\mathsf{V}}, \theta, \gamma}\ \left(\ell_{\text{rec}} + \lambda_{\mathsf{s}} \ell_{\text{static}}\right)$$
$$\text{s.t.} \quad \|z_i^{(\mathsf{t}),a} - z_i^{(\mathsf{t}),p}\|_2^2 + \alpha < \|z_i^{(\mathsf{t}),a} - z_i^{(\mathsf{t}),n}\|_2^2 \tag{5}$$

where α is a hyperparameter that controls the margin while selecting positives and negative.

3.3. Full Objective Function

For any choice of differentiable generator $\mathcal{G}$, the objective (4) will be differentiable with respect to $\mathcal{Z}_{\mathsf{V}}$, and (γ, θ)

[5]. We initialize $\mathcal{Z}_{\mathsf{V}}$ by sampling them from two different Gaussian distributions for both static and transient latent vectors. We also ensure that the latent vectors $\mathcal{Z}_{\mathsf{V}}$ lay on the unit ℓ_2 sphere, and hence, we project $\mathcal{Z}_{\mathsf{V}}$ after each update by dividing its value by $\max(1, \|\mathcal{Z}_{\mathsf{V}}\|)$ [4]. Finally, the complete objective function can be written as follows.

$$\min_{\mathcal{Z}_{\mathsf{V}},\theta,\gamma} \; (\ell_{\text{rec}} + \lambda_{\text{s}}\ell_{\text{static}} + \lambda_{\text{t}}\ell_{\text{triplet}}) \tag{6}$$

where $\ell_{\text{static}} = \ell(\hat{v}_k, v_k)$, λ_{t} is a regularization constant for the triplet loss, and $\ell_{\text{triplet}} = \max\Big(\|z_i^{(\text{t}),a} - z_i^{(\text{t}),p}\|_2^2 + \alpha - \|z_i^{(\text{t}),a} - z_i^{(\text{t}),n}\|_2^2, 0\Big)$. The weights of the generator γ and static latent vector $z^{(\text{s})}$ are updated by gradients of the losses ℓ_{rec} and ℓ_{static}. The weights of the RNN θ, and transient latent vectors $\mathbf{z}^{(\text{t})}$ are updated by gradients of the losses ℓ_{rec}, ℓ_{static} and ℓ_{triplet}.

3.3.1 Low Rank Representation for Interpolation

The objective of video frame interpolation is to synthesize non-existent frames in-between the reference frames. While the triplet condition ensures that similar frames have their transient latent vectors nearby, it doesn't ensure that they lie in a good manifold. A good manifold would enable a simple linear interpolation to yield latent vectors that remain in the low dimensional space as their neighbors, in turn generating frames that form plausible motion with their subsequent as well as preceding frames [12, 4]. This means that the transient latent subspace can be represented in a much lower dimensional space compared to its larger ambient space. So, to enforce such a property, we propose to project the latent vectors into a low dimensional space while learning them along with the network weights. Mathematically, the loss in (5) can be written as

$$\begin{aligned} &\min_{\mathcal{Z}_{\mathsf{V}},\theta,\gamma} \; (\ell_{\text{rec}} + \lambda_{\text{s}}\ell_{\text{static}} + \lambda_{\text{t}}\ell_{\text{triplet}}) \\ &\text{s.t.} \qquad \mathsf{rank}\Big(\mathbf{z}^{(\text{t})}\Big) = \rho \end{aligned} \tag{7}$$

where $\mathsf{rank}(\cdot)$ indicates rank of the matrix and, ρ is a hyperparameter that decides what manifold $\mathbf{z}^{(\text{t})}$ is to be projected on. We achieve this by reconstructing $\mathbf{z}^{(\text{t})}$ matrix from its top ρ singular vectors in each iteration [7]. Note that, we only employ this condition for optimizing the latent space for the frame interpolation experiments in Sec. 4.3.3.

4. Experiments

In this section, we present extensive experiments to demonstrate the effectiveness of our proposed approach in generating videos through learned latent spaces.

4.1. Datasets

We evaluate the performance of our approach using three publicly available datasets which have been used in many prior works [10, 33, 32].

Chair-CAD [2]. This dataset consists of total 1393 chair-CAD models, out of which we randomly choose 820 chairs for our experiments with the first 16 frames, similar to [10]. The rendered frame in each video for all the models are center-cropped and then resized to $64 \times 64 \times 3$ pixels. We obtain the transient latent vectors for all the chair models with one static latent vectors for the training set.

Weizmann Human Action [9]. This dataset provides 10 different actions performed by 9 people, amounting to 90 videos. Similar to Chair-CAD, we center-cropped each frame, and then resize to $64 \times 64 \times 3$ pixels. For this dataset, we train our model to obtain nine static latent vectors (for nine different identities) and ten transient latent vectors (for ten different actions) for videos with 16 frames each.

Golf scene dataset [33]. Golf scene dataset [33] contains 20,268 golf videos with $128 \times 128 \times 3$ pixels which further has 583,508 short video clips in total. We randomly chose 500 videos with 16 frames each and resized the frames to $64 \times 64 \times 3$ pixels. Same as the Chair-CAD dataset, we obtained the transient latent vectors for all the golf scenes and one static latent vectors for the training set.

4.2. Experimental Settings

We implement our framework in PyTorch [22] and will make our source codes publicly available. See the supplementary material for details on the different hyperparameters used in all our experiments.

Network Architecture. We choose DCGAN [23] as the generator architecture for the Chair-CAD and Golf scene dataset, and conditional generator architecture from [21] for the Weizmann Human Action dataset for our experiments. For the RNN, we employ a one-layer gated recurrent unit network with 500 hidden units [6].

Choice of Loss Function for ℓ_{rec} and ℓ_{static}. One straight forward loss function that can be used is the mean squared loss, but it has been shown in literature that it leads to generation of blurry pixels [36]. Moreover, it has been shown empirically that generative functions in adversarial learning focus on edges [4]. Motivated by this, the loss function for ℓ_{rec} and ℓ_{static} is chosen to be the Laplacian pyramid loss $\mathcal{L}_{\text{Laplacian}}$ [18] defined as

$$\mathcal{L}_{\text{Laplacian}}(v, \hat{v}) = \sum_l 2^{2^l} |\mathsf{L}^l(v) - \mathsf{L}^l(\hat{v})|_1$$

where $\mathsf{L}^l(\cdot)$ is the l-th level of the Laplacian pyramid representation of the input.

Baselines. We compare our proposed method with two adversarial methods. For Chair-CAD and Weizmann Human Action, we use MoCoGAN [32] as the baseline, and for

Golf scene dataset, we use VGAN [33] as the baseline. We use the publicly available code for MoCoGAN and VGAN, and set the hyper-parameters as recommended in the published work. We also compare two different versions of the proposed method by ablating the proposed loss functions. Note that, we couldn't compare our results with Video-VAE [10] using our performance measures (described below) as the implementation has not been made available by the authors, and to the best of our efforts we couldn't reproduce the results provided by them.

Performance measures. Past video generation works have been evaluated quantitatively on Inception score (IS) [10]. But, it has been shown that IS is not a good evaluation metric for pixel domain generation, as the maximal IS score can obtained by synthesizing a video from every class or mode in the given data distribution [20, 3, 31]. Moreover, a high IS does not guarantee any confidence on the quality of generation, but only on the diversity of generation. Since a generative model trained using our proposed method can generate all videos using the learned latent dictionary, and for a fair comparison with baselines, we use the following two measures, similar to measures provided in [32]. We also provide relevant bounds computed on real videos for reference.

(1) **Relative Motion Consistency Score (MCS: lower the better)**: Difference between consecutive frames captures the moving components, and hence motion in a video. So, firstly each frame in the generated video, as well as the ground-truth data, is represented as a feature vector computed using a VGG16 network [30] pre-trained on ImageNet [25] at the `relu3_3` layer. Secondly, the averaged frame feature difference vector for both set of videos is computed, denoted by $\hat{f}$ and f respectively. Finally, the relative MCS is then given by $\log_{10}\left(\|f - \hat{f}\|_2^2\right)$.

(2) **Frame Consistency Score (FCS: higher the better)**: This score measures the consistency of the static portion of the generated video frames. We keep the first frame of the generated video as reference and compute the averaged structural similarity measure for all frames. The FCS is then given by the average of this measure over all videos.

4.3. Qualitative Results

Fig. 5 shows some examples with randomly selected frames of generated videos for the proposed method and the adversarial approaches MoCoGAN [32] and VGAN [33]. For Chair-CAD [2] and Weizmann Human Action [9] dataset, it can be seen that the proposed method is able to generate visually good quality videos with a non-adversarial training protocol, whereas MoCoGAN produces blurry and inconsistent frames. Since we use optimized latent vectors unlike MoCoGAN (which uses random latent vectors for video generation), our method produces visually more appealing videos. Fig. 5 particularly presents two important points. As visualized for the Chair-CAD videos, the adversarial approach of MoCoGAN produces not only blurred chair images in the generated video, but also they are non-uniform in quality. Further, it can be seen that as the time step increases, MoCoGAN tends to generate the same chair for different videos. This shows a major drawback of the adversarial approaches, where they fail to learn all modes of the data distribution. Our approach overcomes this by producing a optimized dictionary of latent vectors which can be used for generating any video in the data distribution easily. To further validate our method for qualitative results, we present the following experiments.

4.3.1 Qualitative ablation study

Fig. 4 qualitatively shows the contribution of the specific parts of the proposed method on Chair-CAD [2]. First, we investigate the impact of input latent vector optimization. For a fair comparison, we optimize the model for same number of epochs. It can be observed that the model benefits from the joint optimization of input latent space to produce better visual results. Next, we validate the contribution of ℓ_{static} and ℓ_{triplet} on a difficult video example whose chair color matches with the background. Our method combined with ℓ_{static} and ℓ_{triplet}, is able to distinguish between the white background and the white body of the chair model.

Actions	Identites								
	P1	P2	P3	P4	P5	P6	P7	P8	P9
run	•	•	•		•				
walk	•	•			•		•		
jump		•		•	•	•	•	•	•
skip	•	•	•			•	•	•	•

Table 3: **Generating videos by exchanging unseen actions by identities.** Each cell in this table indicates a video in the dataset. Only cells containing the symbol • indicate that the video was part of the training set. We randomly generated videos corresponding to rest of the cells indicated by symbols •, •, •, and •, visualized in Fig. 6.

4.3.2 Action exchange

Our non-adversarial approach can effectively separate the static and transient portion of a video, and generate videos unseen during the training protocol. To validate these points, we choose a simple *matrix* completion for the combination of identities and actions in the Weizmann Human action [9] dataset. For training our model, we created a set of videos (without any cropping to present the complete scale of the frame) represented by the cells marked with • in Tab. 3. Hence, the unseen videos correspond to the cells

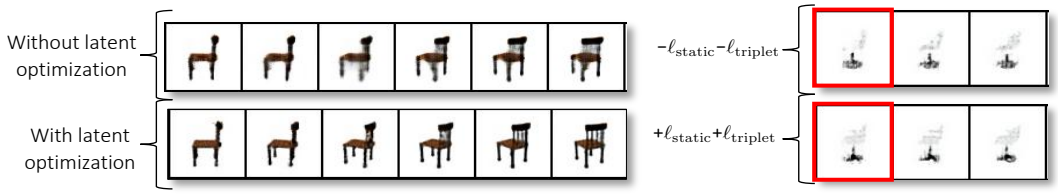


Figure 4: **Qualitative ablation study on Chair-CAD [2].** *Left*: It can be seen that the model is not able to generate good quality frames properly resulting in poor videos when the input latent space is not optimized, whereas with latent optimization, the generated frames are sharper. *Right*: The impact of ℓ_{static} and ℓ_{triplet} is indicated by the red bound boxes. Our method with ℓ_{static} and ℓ_{triplet} captures the difference between the white background and white chair, whereas without these two loss functions, the chair images are not distinguishable from their background. + and − indicate presence and absence of the terms, respectively.

(a) Chair-CAD [2] (b) Weizmann Human Action [9] (c) Golf [33]

Figure 5: **Qualitative results comparison with state-of-the-art methods.** We show two generated video sequences for MoCoGAN [32] (for (a) Chair-CAD [2], (b) Weizmann Human action [9]), VGAN [33] (for (c) Golf scene [33]) (*top*), and the proposed method (**Ours**, *bottom*). The proposed method produces visually sharper, and consistently better using the non-adversarial training protocol. More examples have been provided in the supplementary material.

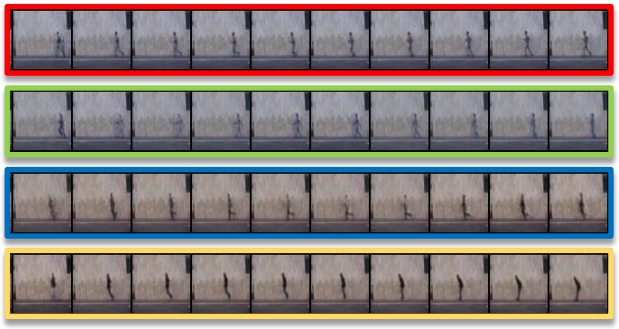


Figure 6: **Examples of action exchange to generate unseen videos.** This figure demonstrates the effectiveness of our method in disentangling static and transient portion of a video, as well as the efficacy in generating videos unseen during training. The colored bounding boxes indicate the unseen video generated referred in Tab. 3.

not marked with •. During testing, we randomly generated these unseen videos (marked with •, •, and • in Tab. 3), and the visual results are shown in Fig. 6. This experiment clearly validates the our claim that firstly, our method is able to disentangle the static (identity and background) and transient (activity) part of a video, and generate good quality unseen videos by using combinations of action and identities not part of training set.

4.3.3 Frame interpolation

To show our methodology can be employed for frame interpolation, we trained our model using the loss (7) for $\rho = 2$ and $\rho = 10$. During testing, we generated intermediate frames by interpolating learned latent variables of two distinct frames. For this, we computed the difference $\Delta z^{(\mathsf{t})}$ between the learned latent vectors of second ($z_2^{(\mathsf{t})}$) and fifth ($z_5^{(\mathsf{t})}$) frame, and generated three unseen frames using $\{z_2^{(\mathsf{t})} + {}^{n}/_{2}\Delta z^{(\mathsf{t})}\}_{n=1}^{3}$, after concatenating with $z^{(\mathsf{s})}$. Fig. 7 shows the results of interpolation between the second and the fifth frames for two randomly chosen videos. Thus, our method is able to produce dynamically consistent frames with respect to the reference frames without any pixel clues.

4.4. Quantitative Results

Quantitative result comparisons with respect to baselines have been provided in Tab. 1. Compared to videos generated by the adversarial method MoCoGAN [32], we report

	MCS ↓	FCS ↑
Bound	0.0	0.91
MoCoGAN [32]	4.11	0.85
Ours ($-\ell_{\text{triplet}}-\ell_{\text{static}}$)	3.83	0.77
Ours ($+\ell_{\text{triplet}}+\ell_{\text{static}}$)	**3.32**	**0.89**

(a) Chair-CAD [2]

	MCS ↓	FCS ↑
Bound	0.0	0.95
MoCoGAN [32]	3.41	0.85
Ours ($-\ell_{\text{triplet}}-\ell_{\text{static}}$)	3.87	0.79
Ours ($+\ell_{\text{triplet}}+\ell_{\text{static}}$)	**2.63**	**0.90**

(b) Weizmann Human Action [9]

	MCS ↓	FCS ↑
Bound	0.0	0.97
VGAN [33]	3.61	**0.88**
Ours ($-\ell_{\text{triplet}}-\ell_{\text{static}}$)	3.78	0.84
Ours ($+\ell_{\text{triplet}}+\ell_{\text{static}}$)	**2.71**	0.84

(c) Golf [33]

Table 2: **Quantitative results comparison with state-of-the-art methods.** We obtained better scores on the proposed method on both Chair-CAD [2], Weizmann Human Action [9], and Golf [33] datasets, compared to the adversarial approaches (MoCoGAN, and VGAN). Best scores have been highlighted in bold.

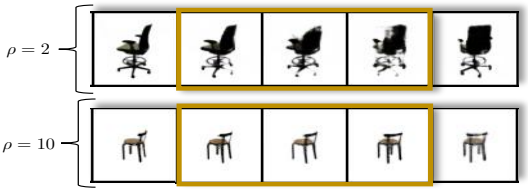


Figure 7: **Examples of frame interpolation.** An important advantage of our method is translation of interpolation in the learned latent space to video space using (7). It can be observed that as ρ increases, the interpolation (bounded by color) is better. Note that the adjacent frames are also generated frames, and not ground truth frames.

a relative decrease of 19.22% in terms of MCS, and 4.70% relative increase in terms of FCS for chair-CAD dataset [2]. For the Weizmann Human Action [9] dataset, the proposed method is reported to have a relative decrease of 22.87% in terms of in terms of MCS, and 4.61% relative increase in terms of FCS. Similarly for Golf scene dataset [33], we perform competitively with VGAN [33] with a observed relative decrease of 24.90% in terms of in terms of MCS. A important conclusion from these results is that our proposed method, being non-adversarial in nature, learns all modes of distribution, and is able to perform at par with adversarial approaches. It should be noted that a better loss function for ℓ_{rec} would produce stronger result. We leave this for future works.

4.4.1 Quantitative Ablation Study

In this section, we demonstrate the contribution of different components in our proposed methodology on the Chair-CAD [2] dataset. For all the experiments, we randomly generate 500 videos using our model by using the learned latent vector dictionary. We divide the ablation study into two parts. Firstly, we present the results for impact of the learned latent vectors on the network modules. For this, we simply generate videos once with the learned latent vectors (+Z), and once with randomly sampled latent vectors from a different distribution (−Z). The inter-dependency of our model weights and the learned latent vectors can be interpreted Tab. 4a. We see that there is a relative decrease of 16.16% in MCS from 3.96 to 3.32, and 18.66% of relative increase in FCS. This shows that naturally due to the protocol of model optimization, the generation of videos is highly dependent on these optimized latent vectors.

	MCS ↓	FCS ↑
Bound	0	0.91
−Z	3.96	0.75
+Z	3.32	0.89

(a) With respect to latent space optimization.

	MCS ↓	FCS ↑
Bound	0	0.91
$-\ell_{\text{triplet}}-\ell_{\text{static}}$	3.83	0.77
$-\ell_{\text{triplet}}+\ell_{\text{static}}$	3.82	0.85
$+\ell_{\text{triplet}}-\ell_{\text{static}}$	3.36	0.81
$+\ell_{\text{triplet}}+\ell_{\text{static}}$	3.32	0.89

(b) With respect to loss functions

Table 4: **Ablation study of proposed method on Chair-CAD** [2]. In (a), we evaluate contributions of latent space optimization (Z). In (b), we evaluate contributions of ℓ_{triplet} and ℓ_{static} in four combinations. + and − indicate presence and absence of the terms, respectively.

Secondly, we investigate the impact of the proposed losses on the proposed method. Specifically, we look into four possible combinations of ℓ_{triplet} and ℓ_{static}. The results are presented in Tab. 4b. It can observed that the combination of triplet loss ℓ_{triplet} and static loss ℓ_{static} provides the best result when employed together, indicated by the relative decrease of 14.26% in MCS from 3.83 to 3.32.

5. Conclusion

We present a non-adversarial approach for synthesizing videos by jointly optimizing both network weights and input latent space. Specifically, our model consists of a global static latent variable for content features, a frame specific transient latent variable, a deep convolutional generator, and a recurrent neural network which are trained using a distance based reconstruction loss, including a triplet based loss. Our approach allows us to generate videos from any mode of data distribution, perform frame interpolation, and generate videos unseen during training. Experiments on three standard datasets show the efficacy of our proposed approach over state-of-the-methods.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223, 2017. 2

[2] Mathieu Aubry, Daniel Maturana, Alexei A Efros, Bryan C Russell, and Josef Sivic. Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3762–3769, 2014. 1, 5, 6, 7, 8, 11, 12, 13, 14

[3] Shane Barratt and Rishi Sharma. A note on the inception score. *arXiv preprint arXiv:1801.01973*, 2018. 6

[4] Piotr Bojanowski, Armand Joulin, David Lopez-Pas, and Arthur Szlam. Optimizing the latent space of generative networks. In *International Conference on Machine Learning*, pages 599–608, 2018. 2, 3, 5

[5] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. Compressed sensing using generative models. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 537–546. JMLR. org, 2017. 5

[6] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 5

[7] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001. 5

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 1

[9] Lena Gorelick, Moshe Blank, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. *IEEE transactions on pattern analysis and machine intelligence*, 29(12):2247–2253, 2007. 5, 6, 7, 8, 11, 12, 13, 14

[10] Jiawei He, Andreas Lehrmann, Joseph Marino, Greg Mori, and Leonid Sigal. Probabilistic video generation using holistic attribute control. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 452–467, 2018. 1, 2, 3, 5, 6

[11] Yedid Hoshen, Ke Li, and Jitendra Malik. Non-adversarial image synthesis with generative latent nearest neighbors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5811–5819, 2019. 2, 3

[12] Rakib Hyder and M. Salman Asif. Generative models for low-dimensional video representation and reconstruction. *IEEE Transactions on Signal Processing*, pages 1–1, 2020. 5

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 12

[14] Yong-Hoon Kwon and Min-Gyu Park. Predicting future frames using retrospective cycle gan. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1811–1820, 2019. 1

[15] Jerry Li, Aleksander Madry, John Peebles, and Ludwig Schmidt. Towards understanding the dynamics of generative adversarial networks. *arXiv preprint arXiv:1706.09884*, 2017. 1

[16] Ke Li and Jitendra Malik. Implicit maximum likelihood estimation. *arXiv preprint arXiv:1809.09087*, 2018. 2

[17] Xiaodan Liang, Lisa Lee, Wei Dai, and Eric P Xing. Dual motion gan for future-flow embedded video prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1744–1752, 2017. 1

[18] Haibin Ling and Kazunori Okada. Diffusion distance for histogram comparison. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 246–253. IEEE, 2006. 5

[19] William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104*, 2016. 1

[20] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study. In *Advances in neural information processing systems*, pages 700–709, 2018. 6

[21] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 5, 11

[22] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017. 5, 12

[23] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 5

[24] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016. 12

[25] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet: Large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 6

[26] Masaki Saito, Eiichi Matsumoto, and Shunta Saito. Temporal generative adversarial nets with singular value clipping. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2830–2839, 2017. 1, 2, 3, 12

[27] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016. 1, 2

[28] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015. 4, 12

[29] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1134–1141. IEEE, 2018. 4, 12

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

[31] L Theis, A van den Oord, and M Bethge. A note on the evaluation of generative models. In *International Conference on Learning Representations (ICLR 2016)*, pages 1–10, 2016. 6

[32] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1526–1535, 2018. 1, 2, 3, 4, 5, 6, 7, 8, 12, 13

[33] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, pages 613–621, 2016. 1, 2, 3, 5, 6, 7, 8, 11, 12, 14

[34] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. In *Advances in Neural Information Processing Systems*, pages 1144–1156, 2018. 4, 12

[35] Tsun-Hsuan Wang, Yen-Chi Cheng, Chieh Hubert Lin, Hwann-Tzong Chen, and Min Sun. Point-to-point video generation. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 1

[36] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss functions for neural networks for image processing. *arXiv preprint arXiv:1511.08861*, 2015. 5

Non-Adversarial Video Synthesis with Learned Priors (Supplementary Material)

Page #	Content
2	**Dataset Descriptions**
2	**Implementation Details** • Hyper-parameters • Other details
3	**More Qualitative Examples** • Qualitative examples on Chair-CAD [2] • Qualitative examples on Weizmann Human Action [9] • Qualitative examples on Golf scene [33] • Interpolation examples on Chair-CAD [2] and Weizmann Human Action [9]

Table 5: Supplementary Material Overview.

A. Dataset Descriptions

Figure 8: **Sample videos from datasets used in the paper.** Two unprocessed video examples from Chair-CAD [2], Weizmann Human Action [21], and Golf Scene [33] datasets have been presented here. As seen from the examples, datasets are diverse in nature, different in categories and present unique challenges in learning the transient and static portions of the videos. Best viewed in color.

Chair-CAD [2]**.** This dataset provides 1393 chair-CAD models. Each model frame sequence is produced using two elevation angles in addition to thirty one azimuth angles. All the chair models have been designed to be at a fixed distance with respect to the camera. The authors provide four video sequences per CAD model. We choose the first 16 frames of each video for our paper, and consider the complete dataset as one class.

Weizmann Human Action [9]**.** This dataset is a collection of 90 video sequences showing nine different identities performing 10 different actions, namely, `run`, `walk`, `skip`, `jumping-jack` (or 'jack'), `jump-forward-on-two-legs`

(or 'jump'), `jump-in-place-on-two-legs` (or 'pjump'), `gallopsideways` (or 'side'), `wave-two-hands` (or 'wave2'), `waveone-hand` (or 'wave1'), and `bend`. We randomly choose 16 consecutive frames for every video in each iteration during training.

Golf Scene [33]. [33] released a dataset containing 35 million clips (32 frames each) stabilized by SIFT+RANSAC. It contains several categories filtered by a pre-trained Place-CNN model, one of them being the Golf scenes. The Golf scene dataset contains 20,268 golf videos. Due to many non-golf videos being part of the golf category (due to inaccurate labels), this dataset presents a particularly challenging data distribution for our proposed method. Note that for a fair comparison, we further selected our training set videos from this provided dataset pertaining to golf action as close as possible. We then trained the VGAN [33] model on this selected videos for a fair comparison.

B. Implementation Details

We used Pytorch [22] for our implementation. The Adam optimizer [13], with $\epsilon = 10^{-8}$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$, was used to update the model weights and SGD optimizer [24], with momentum $= 0.9$, was used to update the latent spaces. The corresponding learning rate for the generator $\tau_{\mathcal{G}}$, the RNN $\tau_{\mathcal{R}}$, and the latent spaces $\tau_{\mathcal{Z}_\mathrm{v}}$ were set as values indicated in Tab. 6.

Hyper-parameters. Our method and [32, 33, 26] that generate videos from latent priors have no dataset split as the task is to synthesize high quality videos from the data distribution, and then evaluate the model performance. All hyperparameters (except D_s, D_t) are set as described in [29, 28, 32, 34] (e.g. α from [29]). For D_s and D_t, we follow the strategy used in Sec. 4.3 of [32] and observe that our model generates videos with good visual quality (FCS) and plausible motion (MCS) for Chair-CAD when (D_s, D_t) = (206, 50). Same strategy is used for all datasets. The hyper-parameters employed with respect to each dataset used in this paper is given in Tab. 6. $\mathcal{G}$ and $\mathcal{R}$ refer to the generator, with weights γ, and RNN, with weight θ, respectively. $\tau_{(\cdot)}$ represents the learning rate. $\mu_{(\cdot)}$ represents the number of epochs. D_s and D_t refer to the static and transient latent dimensions, respectively. λ_s and λ_t refer to the static loss, and triplet loss regularization constants, respectively. α is the margin for triplet loss. l refers to the level of the Laplacian pyramid representation used in ℓ_{rec}.

Datasets	Hyper-parameters										
	D_s	D_t	λ_s	λ_t	α	$\tau_{\mathcal{G}}$	$\tau_{\mathcal{R}}$	$\tau_{\mathcal{Z}_\mathrm{v}}$	μ_γ	$\mu_{\mathcal{Z}_\mathrm{v},\theta}$	l
Chair-CAD [2]	206	50	0.01	0.01	2	6.25×10^{-5}	6.25×10^{-5}	12.5	5	300	4
Weizmann Human Action [9]	56	200	0.01	0.1	2	6.25×10^{-5}	6.25×10^{-3}	12.5	5	700	3
Golf Scene [33]	56	200	0.01	0.01	2	0.1	0.1	12.5	10	1000	4

Table 6: Hyper-parameters used in all experiments for all datasets.

Other details. We performed all our experiments on a system with 48 core Intel(R) Xeon(R) Gold 6126 processor with 256GB RAM. We used NVIDIA GeForce RTX 2080 Ti for all GPU computations during training. Further, NVIDIA Tesla K40 GPUs were used for computation of all evaluation metrics in our experiments. All our implementations are based on non-optimized PyTorch based codes. Our runtime analysis revealed that the it took on average one to two days to train the model and obtain learned latent codes.

C. More Qualitative Examples

In this section, we provide more qualitative results of generated videos synthesized using our proposed approach on each dataset (Fig. 9 for Chair-CAD [2] dataset, Fig. 10 for Weizmann Human Action [9] dataset, and Fig. 11 for Golf scene [33] dataset). We also provide more examples interpolation experiment in Fig. 12.

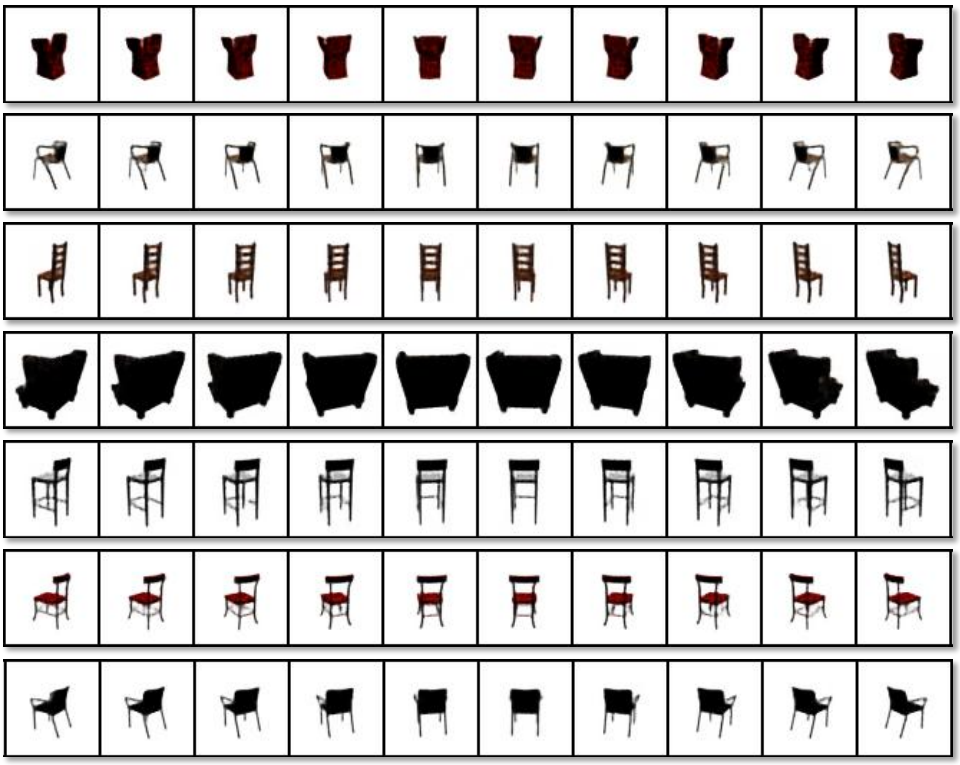


Figure 9: **Qualitative results on Chair-CAD** [2]. On this large scale dataset, our model is able to capture the intrinsic rotation and color of videos unique to each chair model. This shows the efficacy of our approach, compared to adversarial approaches such as MoCoGAN [32] which produce the same chair for all videos, with blurry frames (See Fig. 1 of main mainuscript).

Figure 10: **Qualitative results on Weizmann Human Action** [9]. The videos show that our model produces sharp visual results with the combination of trained generator, RNN along with 9 identities, and 10 different action latent vectors.

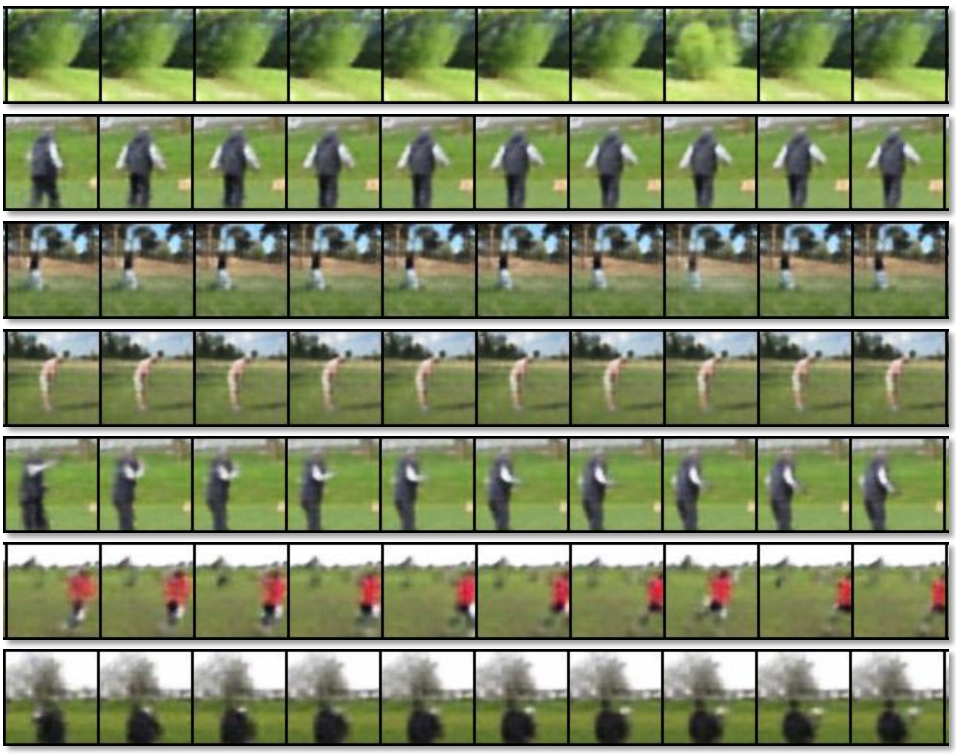
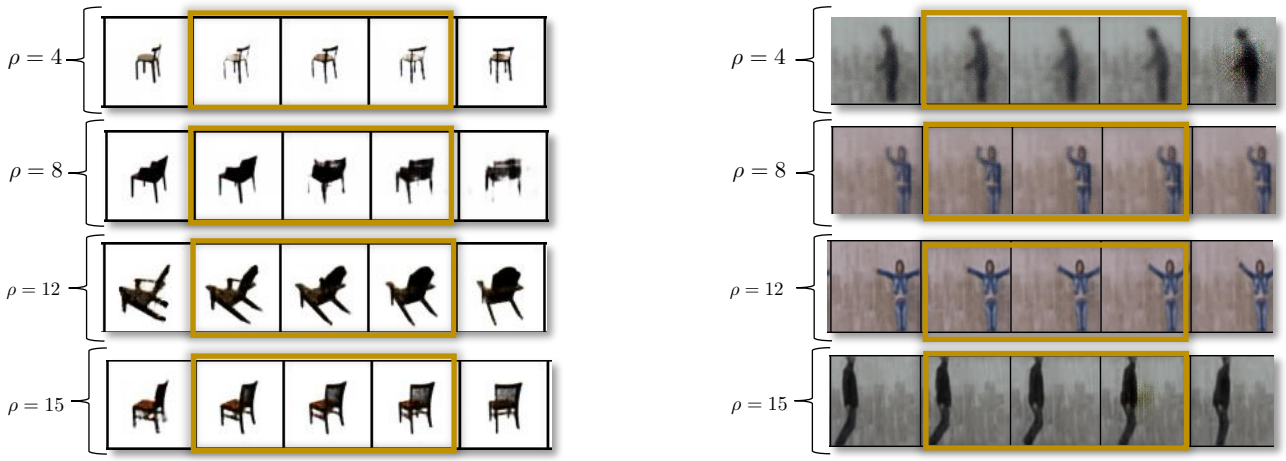


Figure 11: **Qualitative results on Golf Scene** [33]. Our proposed approach produces visually good results on this particularly challenging dataset. Due to incorrect labels on the videos, this dataset has many non-golf videos. Our model is still able to capture the static and transient portion of the videos, although better filtering can still improve our results.



(a) Chair-CAD [2]

(b) Weizmann Human Action [9]

Figure 12: **More interpolation results**. In this figure, ρ represents the rank of transient latent vectors $\mathbf{z}_t$. We present the interpolation results on (a) Chair-CAD dataset, and (b) Weizmann Human Action, for different values of ρ. It can be observed that as ρ increases, the interpolation becomes clearer.