

MoCoGAN: Decomposing Motion and Content for Video Generation

Sergey Tulyakov,
University of Trento, Italy
sergey.tulyakov@unitn.it

Ming-Yu Liu, Xiaodong Yang, Jan Kautz
NVIDIA
{mingyul,xiaodongy,jkautz}@nvidia.com

Abstract

Visual information in a natural video can be decomposed into two major components: content and motion. While content encodes the objects present in the video, motion encodes the object dynamics. Based on this prior, we propose the Motion and Content decomposed Generative Adversarial Network (MoCoGAN) framework for video generation. The proposed framework generates a video clip by sequentially mapping random noise vectors to video frames. We divide a random noise vector into content and motion parts. The content part, modeled by a Gaussian, is kept fixed when generating individual frames in a short video clip, since the content in a short clip remains largely the same. On the other hand, the motion part, modeled by a recurrent neural network, aims at representing the dynamics in a video. Despite the lack of supervision signals on the motion-content decomposition in natural videos, we show that the MoCoGAN framework can learn to decompose these two factors through a novel adversarial training scheme. Experimental results on action, facial expression, and on a Tai Chi dataset along with comparison to the state-of-the-art verify the effectiveness of the proposed framework. We further show that, by fixing the content noise while changing the motion noise, MoCoGAN learns to generate videos of different dynamics of the same object, and, by fixing the motion noise while changing the content noise, MoCoGAN learns to generate videos of the same motion from different objects. More information is available in our project page (<https://github.com/sergeytulyakov/mocogan>).

1. Introduction

Deep generative modeling for image generation has recently received an increasing amount of attention, not only because it provides a means to learn deep feature representations in an unsupervised manner that can potentially leverage all the unlabeled images on Internet for training, but also because it can be used to generate novel images useful for various vision applications. As steady progress toward better image generation is made, it is also important

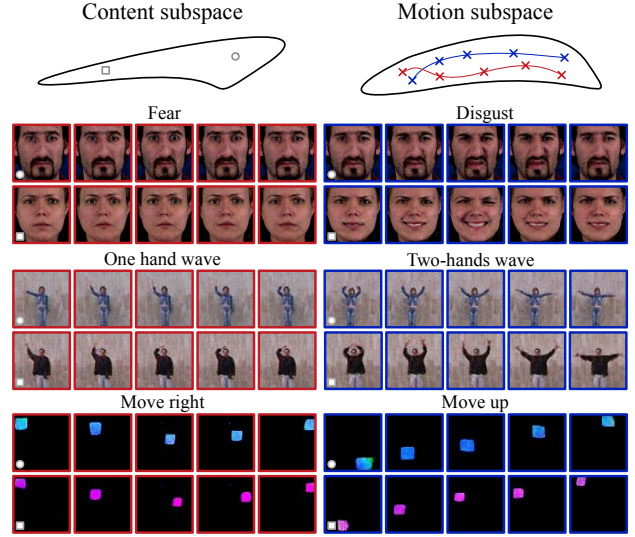


Figure 1: The MoCoGAN framework adopts a motion and content decomposed representation for video generation. It divides the latent space into a content subspace and a motion subspace. By sampling a point in the content subspace and sampling different motion trajectories in the motion subspace, it generates videos of the same object performing different actions. By sampling different points in the content subspace and sampling the same motion trajectory in the motion subspace, it generates videos of different objects performing the same action.

to study the video generation problem. Surprisingly, the extension from generating images to generating videos turns out to be a highly challenging task, although the generated data has only one more dimension, which is time.

The video generation problem is a much harder problem for the following reasons. First, since a video is a spatio-temporal recording of visual information of objects performing various actions, a generative model needs to learn the physical constructs of objects in addition to learning their appearance models. If the learned object constructs are incorrect, the generated video may contain objects performing physically implausible motion. Second, the time dimension brings in a huge amount of variation. Just imag-

ine the amount of speed variations that a person can have as performing a squat movement. Each speed pattern results in a different video, although the appearances of the human in the videos are the same. Third, but not last, as human beings have evolved to be rather sensitive to motion, motion artifacts are particularly pronounced to human eyes. It is more difficult to generate visually convincing videos.

Recently, a few attempts to the video generation problem were made through generative adversarial networks (GANs) [11]. In [37], Vondrick *et al.* hypothesized that a video clip is a point in a latent space and proposed learning a mapping from the latent space to video clips. A similar approach was proposed in [28]. We argue that assuming a video clip is a point in the latent space unnecessarily increases the complexity of the problem, because videos of the same action with different execution speed are represented by different points in the latent space. Moreover, this assumption forces every generated video clip to have the same length, while the length of real-world video clips varies. An alternative (and likely more intuitive and efficient) approach would assume a latent space of images and consider that a video clip is generated by traversing the points in the latent space. Video clips of different lengths correspond to paths of different lengths. In addition, as videos are about objects (content) performing actions (motion), the latent space of images should be further decomposed into two subspaces, where the deviation of a point in the first subspace (the content subspace) leads to content changes in a video clip and the deviation in the second subspace (the motion subspace) results in motions. A video clip of a person performing an action will be represented by a point in the content subspace and a trajectory in the motion subspace. Through this modeling, videos of an action with different execution speeds will only result in different traversing speeds of a trajectory in the motion space.

Decomposing motion and content allows a more controlled video generation process. By changing the content representation while fixing the motion trajectory, we have videos of different objects performing the same action. By changing motion trajectories while fixing the content representation, we have videos of the same object performing different actions. Examples of such control from our experiment results are illustrated in Figure 1.

Based on the intuitions discussed above, we propose the Motion and Content decomposed Generative Adversarial Network (MoCoGAN) framework for video generation. The framework is based on the GAN framework. It generates a video clip by sequentially generating video frames. At each time step, a deep image generative network maps a random vector to an image. The random vector consists of two parts where the first part is sampled from a content space and the second part is sampled from a motion space. Since content in a short video clip usually remains

largely the same, we model the sampling from the content space using a Gaussian distribution and use the same realization through generating individual frames in a video clip. On the other hand, sampling from the motion space is modeled through a recurrent neural network where the network parameters are learned during training. In spite of lacking supervision signals about the decomposition of motion and content in natural videos, we show that MoCoGAN can learn to disentangle these two factors through a novel adversarial training scheme. We conduct extensive experimental validation using action, facial expression and Tai Chi datasets. Through qualitative and quantitative experimental validations and comparison to a state-of-the-art video generation method, we verify the effectiveness of the proposed framework. The user study we performed further supports superiority of the proposed approach.

2. Related Work

Video generation is not a new problem in computer vision. Due to limitations in computation, data, and modeling tools, early video generation works focused on generating dynamic texture patterns [31, 38, 8]. In the recent years, with the availability of GPUs, Internet videos, and deep neural networks, we argue that we are now better positioned to tackle this intriguing problem.

Various deep generative models were recently proposed for image generation including (GANs) [11], variational autoencoders (VAEs) [18, 27], moment matching networks [19], PixelCNNs [35], and Plug&Play Generative Networks [23]. In this paper, we propose the MoCoGAN framework for video generation, which is based on GANs.

In the GAN framework, the image generation is learned through solving a 2-player zero-sum game problem where the two players are an image generative network and an image discriminative network. Denton *et al.* [7] proposed a Laplacian pyramid implementation of GANs. Radford *et al.* [25] used a deeper convolution network architecture. Zhang *et al.* [40] stacked two generative networks to progressively render realistic images. CoupledGANs [21] learned to generate corresponding images in different domains, which can be extended to learn to translate an image in one domain to a corresponding one in a different domain in an unsupervised fashion [20]. InfoGAN [5] learned a more interpretable latent representation. Salimans *et al.* [29] proposed several GAN training tricks. Arjovsky *et al.* [3] proposed the Wasserstein GAN framework for a more stable adversarial training. The proposed MoCoGAN framework generates a video clip by sequentially generating images using an image generator. It can easily leverage the advancement in image generation in the GAN framework for improving the quality of the generated videos. As discussed in Section 1, [37, 28] extended the GAN framework to the video generation problem by assuming a latent space

of video clips where all the clips have the same length.

The use of recurrent neural networks for image generation were previously explored in [13, 15]. Specifically, they use recurrent mechanisms to iteratively refine a generated image. Our work is different to [13, 15] in that we use the recurrent mechanism to generate motion embeddings of video frames in a video clip. The image generation is achieved through a convolutional neural network.

The future frame prediction problem studied in [30, 24, 22, 16, 10, 34, 39, 36] is closely related to the video generation problem studied in this paper. In future frame prediction, the goal is to predict future frames in a video given the previous frames in the video. Previous works on future frame prediction can be roughly divided into two categories where one focuses on generating raw pixel values in future frames based on the observed ones [30, 24, 22, 16, 39, 36] while the other focuses on generating transformations for reshuffling the pixels in the previous frames to construct future frames [10, 34]. The availability of previous frames makes future frame prediction a conditional image generation problem, which is different to the video generation problem where the input to the generative network is only a vector drawn from a latent space. We note that [36] used a convolutional LSTM [14] encoder to encode temporal differences between consecutive previous frames for extracting motion information and a convolutional encoder to extract content information from the current image. The concatenation of the motion and content information was then fed to a decoder to predict future frames.

3. Generative Adversarial Networks

A GAN framework [11] consists of two networks: a generative network and a discriminative network. The objective of the generative network is to generate images resembling real images, while the objective of the discriminative network is to distinguish real images from generated ones. Both the generative and discriminative networks are realized as convolutional neural networks (CNNs) in practice.

Let \mathbf{x} be a natural image (a real image) drawn from an image distribution, p_X , and \mathbf{z} be a random vector in $Z_I \equiv \mathbb{R}^d$. Let G_I and D_I be the generative and discriminative networks, respectively. The generative network takes \mathbf{z} as input and outputs an image, $\tilde{\mathbf{x}} = G_I(\mathbf{z})$, that has the same support as \mathbf{x} . We denote the distribution of $G_I(\mathbf{z})$ as p_{G_I} . The discriminative network estimates the probability that an input image is drawn from p_X . Ideally, $D_I(\mathbf{x}) = 1$ if $\mathbf{x} \sim p_X$ and $D_I(\tilde{\mathbf{x}}) = 0$ if $\tilde{\mathbf{x}} \sim p_{G_I}$. The GAN framework corresponds to a 2-player zero-sum game, and the generative and discriminative networks can be trained jointly via solving a minimax problem given by

$$\max_{G_I} \min_{D_I} \mathcal{F}_I(D_I, G_I) \quad (1)$$

where the functional \mathcal{F}_I is given by

$$\mathcal{F}_I(D_I, G_I) = \mathbb{E}_{\mathbf{x} \sim p_X} [-\log D_I(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_{Z_I}} [-\log(1 - D_I(G_I(\mathbf{z})))]. \quad (2)$$

In practice, (1) is solved by alternating the following two gradient update steps:

$$\begin{aligned} \text{Step 1: } \theta_{D_I^{t+1}} &= \theta_{D_I^t} - \lambda^t \nabla_{\theta_{D_I}} \mathcal{F}_I(D_I^t, G_I^t), \\ \text{Step 2: } \theta_{G_I^{t+1}} &= \theta_{G_I^t} + \lambda^t \nabla_{\theta_{G_I}} \mathcal{F}_I(D_I^{t+1}, G_I^t) \end{aligned}$$

where θ_{D_I} and θ_{G_I} are the parameters of D_I and G_I , λ is the learning rate, and t is the iteration number.

Goodfellow *et al.* [11] show that, given enough capacity to D_I and G_I and sufficient training iterations, the distribution, p_{G_I} , converges to p_X by using the above gradient update scheme. As a result, from a random vector input \mathbf{z} , the network G_I can synthesize an image that resembles one drawn from the true distribution, p_X .

3.1. Extension to Fixed-length Video Generation

Recently, [37] propose the video generative adversarial network (VGAN) framework to extend the GAN framework to video generation. Let $\mathbf{v}^L = [\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(L)}]$ be a video clip with L frames. The video generation in VGAN is achieved by replacing the vanilla CNN-based image generative and discriminative networks, i.e., G_I and D_I , with the spatio-temporal CNN-based video generative and discriminative networks, i.e., G_{V^L} and D_{V^L} . The video generative network G_{V^L} maps a random vector $\mathbf{z} \in Z_{V^L} \equiv \mathbb{R}^d$ to a video clip, $\tilde{\mathbf{v}}^L = [\tilde{\mathbf{x}}^{(1)}, \dots, \tilde{\mathbf{x}}^{(L)}] = G_{V^L}(\mathbf{z})$ and the video discriminative network D_{V^L} differentiates real video clips from generated ones. Ideally, $D_{V^L}(\mathbf{v}^L) = 1$ if \mathbf{v}^L is sampled from p_{V^L} and $D_{V^L}(\tilde{\mathbf{v}}^L) = 0$ if $\tilde{\mathbf{v}}^L$ is sampled from the video generative network distribution $p_{G_{V^L}}$.

In the VGAN framework, each point in the latent space $\mathbf{z} \in Z_{V^L}$ represents a video clip with a fixed length L . Hence, the VGAN framework can only model videos with L frames. Moreover, for videos of a same action of a person executed in different speeds, VGAN would represent them as different points in Z_{V^L} , which is unintuitive and inefficient. It also does not provide a way to decompose motion and content information in video clips. In order to overcome these limitations, we propose the Motion and Content decomposed GAN (MoCoGAN) framework to extend the GAN framework to video generation.

4. The MoCoGAN Framework

In this section, we will first discuss our latent space representation and then present the MoCoGAN network architecture and the learning algorithm. Finally, we will extend the framework to action-conditioned video generation.

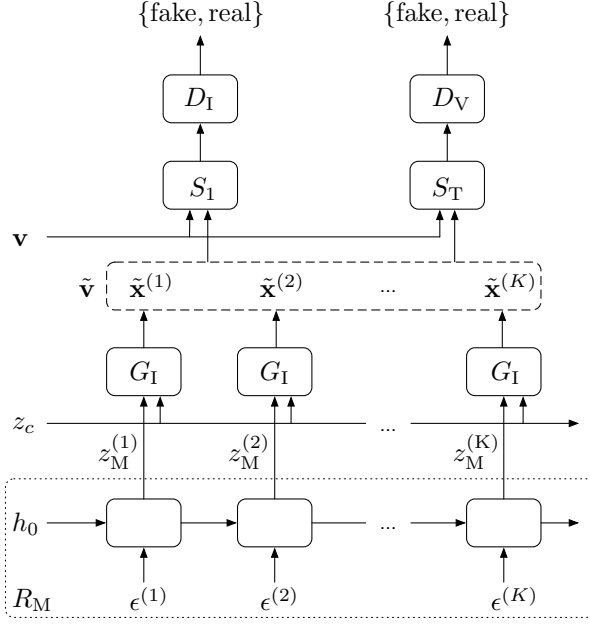


Figure 2: The MoCoGAN framework for video generation. For a video the content vector z_c is sampled once and fixed. Then, a series of random variables $[\epsilon^{(1)}, \dots, \epsilon^{(K)}]$ is sampled and mapped to a series of motion codes $[z_M^{(1)}, \dots, z_M^{(K)}]$ via the recurrent neural network R_M . A generator G_I produces a frame $\tilde{x}^{(k)}$ using the content and the motion vectors $\{z_C, z_M^{(k)}\}$. The discriminators D_I and D_V are trained on real and fake images and videos, respectfully, sampled from the training set \mathbf{v} and the generated set $\tilde{\mathbf{v}}$.

4.1. Latent Space Representation

We assume a latent space of images $Z_I \equiv \mathbb{R}^d$ where each point $\mathbf{z} \in Z_I$ represents an image, and a video of K frames is represented by a path of length K in the latent space, $[\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(K)}]$. By adopting this formulation, videos of different lengths can be generated by paths of different lengths. Moreover, videos of the same action executed with different speeds can be generated by traversing a same path in the latent space with different speeds. More importantly, the formulation allows a simple way to decompose motion and content in video clips.

Specifically, we further assume the latent space of images Z_I is decomposed into the content subspace Z_C and the motion subspace Z_M : i.e., $Z_I = Z_C \times Z_M$ where $Z_C = \mathbb{R}^{d_C}$, $Z_M = \mathbb{R}^{d_M}$, and $d = d_C + d_M$. The content subspace models motion-independent appearance in videos, while the motion subspace models motion-dependent appearance in videos. For example, in a video of a person making a smile, content represents the identity of the person, while motion represents the changes of facial muscle configurations of the person. A pair of the person’s identity

and the facial muscle configuration represents a face image of the person. A particular sequence of these pairs represents a video clip of the person making a smile. By swapping the look of the person with the look of another person but using the same facial muscle configurations, a video of a different person making a smile could be represented.

We model the sampling process from the content subspace using a standard multivariate Gaussian distribution: $\mathbf{z}_C \sim p_{Z_C} \equiv \mathcal{N}(\mathbf{z}|0, I_{d_C})$ where I_{d_C} is an identity matrix of size $d_C \times d_C$. Based on the observation that the content remains largely the same in a short video clip, we use the same realization \mathbf{z}_C for generating different frames in a video clip. Given a fixed \mathbf{z}_C , the motion in the video clip is modeled by a path in the motion subspace Z_M . In other words, the sequence is represented by

$$[\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(K)}] = \left[\begin{bmatrix} \mathbf{z}_C \\ \mathbf{z}_M^{(1)} \end{bmatrix}, \dots, \begin{bmatrix} \mathbf{z}_C \\ \mathbf{z}_M^{(K)} \end{bmatrix} \right] \quad (3)$$

where $\mathbf{z}_C \in Z_C$ and $\mathbf{z}_M^{(k)} \in Z_M$ for all k ’s. Since not all paths in Z_M correspond to physically plausible motion, we need to learn to generate valid paths. We model the path generation process using a recurrent neural network.

Let R_M to be a recurrent neural network. At each time step, it takes a vector sampled from a standard multivariate Gaussian distribution as input: $\epsilon^{(k)} \sim p_E \equiv \mathcal{N}(\epsilon|0, I_{d_E})$ where d_E is the dimension of the vector. At each time step, it outputs a vector in Z_M , which is used as the motion representation. Let $R_M(k)$ be the output of the recurrent neural network at time k . Then, $\mathbf{z}_M^{(k)} = R_M(k)$. Intuitively, the function of the recurrent neural network is to map a sequence of independent and identically distributed (i.i.d.) random variables $[\epsilon^{(1)}, \epsilon^{(2)}, \dots, \epsilon^{(K)}]$ to a sequence of correlated random variables $[R_M(1), R_M(2), \dots, R_M(K)]$ representing the dynamics in a video. Various recurrent neural network implementations exist including LSTMs [14] and GRUs [6]. We use GRUs in the MoCoGAN but expect the other architectures would work equally well.

We note that assuming a motion and content decomposed representation in videos does not guarantee that we can learn the true decomposition from data. Moreover, due to the absence of supervised signals, learning the decomposition is a very challenging task. In the following, we present the MoCoGAN network architecture and the learning algorithm for learning the decomposition from data.

4.2. Network Architecture

The MoCoGAN framework is based on GAN, but it consists of 4 networks which are the recurrent neural network R_M , the image generative network G_I , the image discriminative network D_I , and the video discriminative network D_V . The image generative network generates a video clip by sequentially mapping vectors in Z_I to images, from a sequence of vectors $[\begin{bmatrix} \mathbf{z}_C \\ \mathbf{z}_M^{(1)} \end{bmatrix}, \dots, \begin{bmatrix} \mathbf{z}_C \\ \mathbf{z}_M^{(K)} \end{bmatrix}]$ to a sequence of images

$\tilde{\mathbf{v}} = [\tilde{\mathbf{x}}^{(1)}, \dots, \tilde{\mathbf{x}}^{(K)}]$ where $\tilde{\mathbf{x}}^{(k)} = G_I(\begin{bmatrix} \mathbf{z}_C \\ \mathbf{z}_M^{(k)} \end{bmatrix})$ and $\mathbf{z}_M^{(k)}$'s are from the recurrent neural network R_M . We note that the video length K can vary for each video generation.

Both the image discriminative network D_I and the video discriminative network D_V play the role of judge, providing critical feedbacks to the image generative network G_I and the recurrent neural network R_M for generating realistic videos during adversarial training. Although D_I and D_V have a similar function, they have different specialties. The image discriminative network is specialized in criticizing G_I based on each generated image. It is trained to output 1 for a video frame sampled from a real video clip \mathbf{v} and to output 0 for a video frame sampled from $\tilde{\mathbf{v}}$. On the other hand, the video discriminative network D_V provides feedbacks to G_I based on the generated video clip. D_V takes a fixed length video clip, say T frames, as the input. It is trained to output 1 for a video clip sampled from a real video and to output 0 for a video clip sampled from $\tilde{\mathbf{v}}$. Different to D_I which is based on vanilla CNN architecture, D_V is based on spatio-temporal CNN architecture. We note that the clip length T is a hyperparameter, which is set to 16 throughout our experiments. We also note that T can be smaller than the generated video length K . A video of length K can be divided into $K - T + 1$ clips in a sliding-window fashion, and each of the clips can be fed into D_V .

The video discriminative network D_V criticizes the generated video clips. In addition to providing feedbacks based on the appearance in each image (which D_I also does), it also evaluates the generated motion. Since the image generative network G_I has no concept of motion¹, the criticisms on the motion part are redirected to the recurrent neural network R_M . For generating a video with realistic dynamics for fooling D_V , R_M has to learn to generate a sequence of motion codes $[z_M^{(1)}, \dots, z_M^{(K)}]$ from a sequence of i.i.d. noise inputs $[\epsilon^{(1)}, \dots, \epsilon^{(K)}]$ in a way such that G_I can sequentially map $z^{(k)} = [z_C, z_M^{(k)}]$ to consecutive frames in a video.

Ideally, D_V alone should be sufficient for training G_I and R_M , because D_V provides feedback on both static image appearance and video dynamics. However, we empirically found that using D_I significantly improves the convergence of the adversarial training. We suspect this is due to the difficulty in training D_V . As D_V has to learn to encode both spatial and temporal information, training D_V is more difficult than training D_I , which only needs to focus on appearance. Once D_I is well-trained, G_I can learn to generate realistic images based on D_I 's feedback. In this stage, the function of D_V becomes much simpler. It only needs to be good at providing feedback about the generated video dynamics for training R_M . We note that, in a recent work [9], a GAN framework for image generation with mul-

tiple discriminative networks was proposed. In the work, all the discriminative networks had the same function, and they competed with each other for catching a synthesized image. The proposed work is different to [9] in that the two discriminative networks D_I and D_V have different functions. The carefully staged collaboration between D_I and D_V enables the MoCoGAN framework to generate realistic videos.

4.3. Learning

Let p_V be the distribution of video clips of various lengths. Let κ be a discrete random variable denoting the length of a video clip sampled from p_V . In practice, we can estimate the distribution of κ , termed p_K , by computing a histogram of video clip length from training data. The distribution of the generated videos using the MoCoGAN framework can be described by

$$p_{\tilde{\mathbf{v}}}(\tilde{\mathbf{v}}) = \prod_{k=1}^{\kappa} p_E(\epsilon) \cdot p_K(\kappa) \cdot p_{Z_C}(\mathbf{z}_C). \quad (4)$$

Basically, we can generate a video by first sample a content vector \mathbf{z}_C and a length κ . We then run R_M for κ steps and, at each time step, R_M takes a random variable ϵ as the input. A generated video by MoCoGAN is then given by

$$\tilde{\mathbf{v}} = \left[G_I\left(\begin{bmatrix} \mathbf{z}_C \\ R_M(1) \end{bmatrix}\right), \dots, G_I\left(\begin{bmatrix} \mathbf{z}_C \\ R_M(\kappa) \end{bmatrix}\right) \right]. \quad (5)$$

Recall that our D_I and D_V take one frame and T consecutive frames in a video as input, respectively. In order to represent these mechanisms, we need to introduce two random access functions S_1 and S_T . The function S_1 takes a video clip (either $\mathbf{v} \sim p_V$ or $\tilde{\mathbf{v}} \sim p_{\tilde{\mathbf{v}}}$) and outputs a random frame in the clip. On the other hand, the function S_T takes a video clip and randomly returns T consecutive frames in the clip. With these notations, we are now ready to present the MoCoGAN learning problem, which is given by

$$\max_{G_I, R_M} \min_{D_I, D_V} \mathcal{F}_V(D_I, D_V, G_I, R_M) \quad (6)$$

where the objective function $\mathcal{F}_V(D_I, D_V, G_I, R_M)$ is

$$\begin{aligned} & \mathbb{E}_{\mathbf{v} \sim p_V} [-\log D_I(S_1(\mathbf{v}))] + \\ & \mathbb{E}_{\tilde{\mathbf{v}} \sim p_{\tilde{\mathbf{v}}}} [-\log(1 - D_I(S_1(\tilde{\mathbf{v}})))] + \\ & \mathbb{E}_{\mathbf{v} \sim p_V} [-\log D_V(S_T(\mathbf{v}))] + \\ & \mathbb{E}_{\tilde{\mathbf{v}} \sim p_{\tilde{\mathbf{v}}}} [-\log(1 - D_V(S_T(\tilde{\mathbf{v}})))] \end{aligned} \quad (7)$$

We note that understanding (7) is straightforward. From the point of view of a discriminative network, the first and second terms encourage D_I to output 1 for a video frame sampled from a real video clip \mathbf{v} and 0 for a video frame from a generated one $\tilde{\mathbf{v}}$. Similarly, the third and forth terms encourage D_V to output 1 for T consecutive frames in a real

¹This is the case because G_I simply maps a vector to a still image. It does not know what a video is.

video clip \mathbf{v} and 0 for T consecutive frames in a generated one $\tilde{\mathbf{v}}$. From the point of view of the generative network and the recurrent neural network, the second and forth term encourage them to generate a realistic video that both a random frame or a random T -consecutive frames in the video is so realistic that no discriminators can tell apart.

To train MoCoGAN, we can extend the alternating gradient update scheme for GAN training. Although 4 networks exist in a MoCoGAN, we only need to alternate between two gradient update steps given by

$$\begin{aligned} \text{Step 1: } \theta_{D_I^{t+1}} &= \theta_{D_I^t} - \lambda^t \nabla_{\theta_{D_I}} \mathcal{F}_V(D_I^t, D_V^t, G_I^t, R_M^t), \\ \theta_{D_V^{t+1}} &= \theta_{D_V^t} - \lambda^t \nabla_{\theta_{D_V}} \mathcal{F}_V(D_I^t, D_V^t, G_I^t, R_M^t), \\ \text{Step 2: } \theta_{G_I^{t+1}} &= \theta_{G_I^t} + \lambda^t \nabla_{\theta_{G_I}} \mathcal{F}_V(D_I^{t+1}, D_V^{t+1}, G_I^t, R_M^t) \\ \theta_{R_M^{t+1}} &= \theta_{R_M^t} + \lambda^t \nabla_{\theta_{R_M}} \mathcal{F}_V(D_I^{t+1}, D_V^{t+1}, G_I^t, R_M^t) \end{aligned}$$

because D_I and D_V are conditionally independent and G_I and R_M work together for generating videos. In the first step, we update D_I and D_V , which are in charge of criticizing the generated video clips. In the second step, we update G_I and R_M , which are in charge of the generation process.

4.4. Action Conditioned Video Generation

When action category labels are available in training data, we can utilize the labels to improve video generation results. We can also use the action labels to learn to perform action-conditioned video generation by using a condition GAN model akin to [26]. For example, we can generate face videos conditioning on the facial expression label such as smile, anger, or disgust. We note that in addition to motion in a video, an action label may also impact static appearances in a video. For example, as generating a sport video conditioning on an action label, appearances of athletics in the video also depend on the action label. For example, basketball and hockey players have very different appearances. Hence, the action category label can be in the content subspace in addition to the motion subspace.

Let \mathbf{z}_A be the action category label. When considering action label as a part of content, we can represent the content vector by the concatenation of \mathbf{z}_C and \mathbf{z}_A . When considering action label as a part of motion, we can input the action label to the recurrent neural network R_M . That is input to R_M is given by $[\epsilon, \mathbf{z}_A]$. This way the action label can influence the generated path in the motion subspace. The generated dynamics will depend on the action. Instead of inputting the action label to the discriminative networks as in [26], we empirically found that jointly training the discriminative networks to differentiate real and synthesized videos and to predict the action category in the videos rendered better video generation results.

5. Implementations

We designed G_I and D_I based on the DCGAN architecture [25], while we designed D_V based on the spatio-temporal CNN in [37]. The recurrent neural network R_M was a simple one-layer GRU network [6]. We trained MoCoGANs using ADAM [17]. We set the learning rate to 0.0002 and momentums to 0.5 and 0.999, respectively, as in [25]. We utilized the one-sided label smoothing trick [29] for robust adversarial training. The training was performed on a Tesla P100 card in an NVIDIA DGX-1 machine.

6. Experiments

We conducted extensive experiments to evaluate the proposed framework. In addition to comparisons to the VGAN framework [37], we also quantitatively evaluated the proposed framework on its ability in 1) generating videos of the same object performing different motion by using a fixed content vector and varying motion trajectories and 2) generating videos of different objects performing the same motion by using different content vectors with the same motion trajectory. Evaluating generative models, especially on generating visual outputs, is a challenging task, since all the popular metrics are subject to flaws [33]. Hence, we reported experiment results on the datasets where we could develop reliable performance metrics.

We used the following datasets in our experiments:

- **Shape motion video generation.** We built a synthetic dataset of video clips of shape motion. The dataset contained two types of shapes (circles and squares) with varying sizes and colors, performing two types of motion: one moving from left to right, the other moving from top to bottom. The motion trajectories were sampled from Bezier curves. There were 4,000 videos in the dataset, where the image resolution was 64×64 and video length was 16.
- **Facial expression video generation.** We used the MUG Facial Expression Database [1] for the experiment. The dataset consisted of 86 subjects (51 male and 34 female) performing various facial expressions. Each video consisted of 50 to 160 frames. We cropped the face regions and resized them to have a resolution of 96×96 . We discarded videos containing fewer than 64 frames and used only the sequences representing one of the six prototypic facial expressions: anger, fear, disgust, happiness, sadness, and surprise. Totally, the training datasets consists of 1254 video clips.
- **Tai Chi video generation.** To show that MoCoGAN can generate challenging video sequences we trained it on a newly collected database of 4500 Tai Chi clips. For each video clip, we applied human pose estimator [4] and cropped the clip so that the performer is in the center.

Table 1: Video content consistency comparison. A smaller ACD means the generated frames in a video are perceptually more similar. We also compute the ACDs for the videos in the training datasets, which serve as the reference shown in the table.

	Shape Motion	Facial Expression
Reference	0	0.116
VGAN [37]	0.055	0.322
MoCoGAN	0.049	0.195

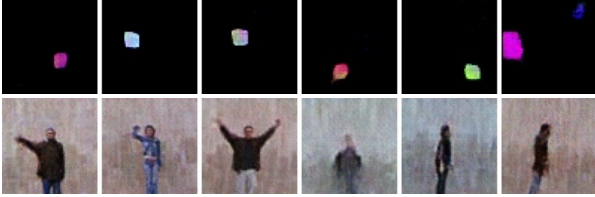


Figure 3: Video generation results. The figure is best viewed via the Acrobat Reader. Click the image to play the video clip.



Figure 4: Examples of influence of motion trajectories on video generation. Two examples (disgust and anger) are shown. Every two rows share the same content representation but have different motion trajectories. Note that different motion trajectories influence the *phase* of the videos, i.e. the time where a facial expression starts.

Videos were rescaled to 64×64 pixels. The database includes more than 300K frames.

- **Human action video generation.** We used the Weizmann Action database [12], containing 81 videos of 9 people performing 9 actions including jumping-jack and waving-hands. We resized the videos to have a resolution of 96×96 . Due to the small size, we did not conduct quantitative evaluation using the dataset. Instead, we provided visualization results.

Performance Metrics. We used the following performance metrics for comparing video generation methods.

- **Average Content Distance (ACD).** We used the ACD to measure content consistency of a generated video. For

shape motion, we first computed average color of the generated shape in a frame. Each frame was, hence, represented by a 3-dimensional vector, and a video was represented by a set of vectors. We then computed pairwise L2 distances between the vectors. The average of the distances, termed as the ACD, was used as the performance metric. The smaller the ACD, the more consistent the video frames were. For facial expression video generation, the average color representation would be problematic, since this was not invariant to facial expression changes. For measuring if the generated faces in a facial expression video were all from the same person, we employed OpenFace [2], which surpassed human-level performance for face verification [32]. OpenFace extracted a feature vector for a face image and had the property that vectors of the face images from the same person were similar. We used the vectors to compute the ACD to quantify whether the generated faces in a video clip corresponds to the same person.

- **Motion Control Score (MCS).** We used the MCS to evaluate action-conditioned video generation performance. We trained a spatio-temporal CNN classifier for action recognition using the labeled training dataset. We split the training dataset into a training set and a validation set for determining the training hyperparameters. In the test time, the MoCoGAN generated a video clip based on the input action label, and we used the classifier to verify whether the generated video contained the action. The MCS was then given by the ratio of the generated videos that the classifier considered having the actions that the generative network was asked to generate. The larger the MCS, the better control of the motion the generative network has.
- **Content Control Score (CCS).** We used the CCS to evaluate the performance of the proposed framework on action-conditioned video generation. We generated pairs of videos, using the same action label and motion trajectory but two different content vectors. We then extracted feature vectors from the frames in the generated videos (e.g. using OpenFace). The average pairwise L2 distance between vectors in different videos in a pair was computed. The CCS was given by the average of the distances over all pairs. The larger the CCS, the more difference the content in a pair of videos. The CCS measured how well the content vector impacted the generated video content.

We trained the proposed and VGAN frameworks for the shape motion and facial expression generation tasks, respectively, for comparing their video content generation consistency. For VGAN, we used the implementation provided by the authors for training and randomly generated 256 videos from the trained models to compute the ACDs. For the proposed framework, we also randomly generated 256 videos using the trained models to compute the ACDs.

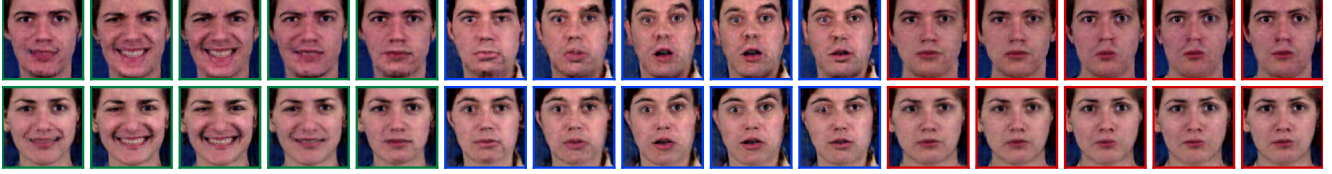


Figure 5: Two examples of generating face videos of different identities with changing facial expressions. We changed the expression label from smile to fear through surprise.

Table 2: Performance on action-conditioned facial expression video generation with various MoCoGAN settings.

Settings		MCS	ACD	CCS
\bar{D}_T	$\mathbf{z}_A \rightarrow G_I$	0.472	1.115	1.123
\bar{D}_T	$\mathbf{z}_A \rightarrow R_M$	0.491	1.073	1.080
D_I	$\mathbf{z}_A \rightarrow G_I$	0.355	0.738	1.333
D_I	$\mathbf{z}_A \rightarrow R_M$	0.581	0.606	1.269

The comparison results are given in Table 1. From the table, we found that the content of the videos generated by the proposed framework are more consistent. This was particularly the case when dealing with the more difficult facial expression video generation task. The face images in the generated facial expression video from the proposed framework looked more like they were from the same person. In Figure 3, we visualized the video generation results on the shape motion and human action video generation tasks and found that the proposed MoCoGAN learned to generate various shape motion and human action.

In Figure 4, we visualized the facial expression videos generated by the proposed framework. To generate the videos, we fixed the content vector \mathbf{z}_C and randomly sampled $[\epsilon^{(1)}, \dots, \epsilon^{(K)}]$ to generate different motion trajectories, $[\mathbf{z}_M^{(1)}, \dots, \mathbf{z}_M^{(K)}]$. The concatenation of the content and motion vectors were sequentially passed to G_I to generate the facial expression videos. From the figure, we found that the generated videos corresponded to the same person performing the same expression with different speeds.

6.1. Action-Conditioned Video Generation

We trained the MoCoGAN framework for the action-conditioned facial expression video generation task and used the trained network to generate long facial expression videos (96 frames). During the generation, we changed the action category \mathbf{z}_A every 16 frames and ensured that all 6 expressions were covered. Hence, a generated video would correspond to a person performed 6 different facial expressions. To evaluate the performance, we computed the ACD of the generated videos. A smaller ACD value meant the generated faces over the 96 frames were more likely to be from the same person. Note that the ACD values reported in this subsection were generally larger than the ACD values reported in Table 1, because the generated videos in the table contained only 1 expression while the generated videos

Table 3: Amazon Mechanical Turk workers preferences.

	Facial expressions	Tai Chi
VGAN [37]	15.8%	42.1%
MoCoGAN	84.2%	57.9%

in this subsection contained 6 different expressions. We also used the CCS metric to evaluate MoCoGAN’s capability in content generation control. Specifically, we generated a set of 96-frames videos with the same motion trajectory and action category sequence but different content vectors. We then computed the CCS metric to measure how difference the generated faces were across the videos.

We evaluated different conditioning schemes. In one scheme, the action category label was directly fed to the image generative network, termed $\mathbf{z}_A \rightarrow G_I$. In another scheme, the action category label was fed to the recurrent neural network, termed $\mathbf{z}_A \rightarrow R_M$. In addition, we evaluated the impact of the image discriminative network D_I for the action-conditioned video generation task, where we considered training the MoCoGAN framework without D_I .

We reported the experiment results in Table 2. From the table, we found that the models trained with D_I consistently rendered better performances on various metrics. We also found that $\mathbf{z}_A \rightarrow R_M$ rendered a better performance. Figure 5 shows two videos from the best model in Table 2. We observed that by fixing the content vector but changing the expression label, it generated videos of the same person performing different expressions. And similarly, by changing the content vector and providing the same motion trajectory, we generated videos of different people showing the same expression sequence.

6.2. Motion and Content Subspace Dimensions

We conducted an experiment to empirically determine the dimension of the content vector \mathbf{z}_C and the dimension of the motion vectors $\mathbf{z}_M^{(t)}$ ’s, which are referred to as d_C and d_M . In the experiment, we fixed the sum of the dimensions to 60 (i.e., $d_C + d_M = 60$) and changed the value of d_C from 10 to 50, with a step size of 10. For each combination, we trained a MoCoGAN network using the MUG Facial Expression Database [1]. The rest of the experiment setup strictly followed those described in Section 6.1 in the main paper. We evaluated these MoCoGAN network based on the Motion Control Score (MCS), the Average Content

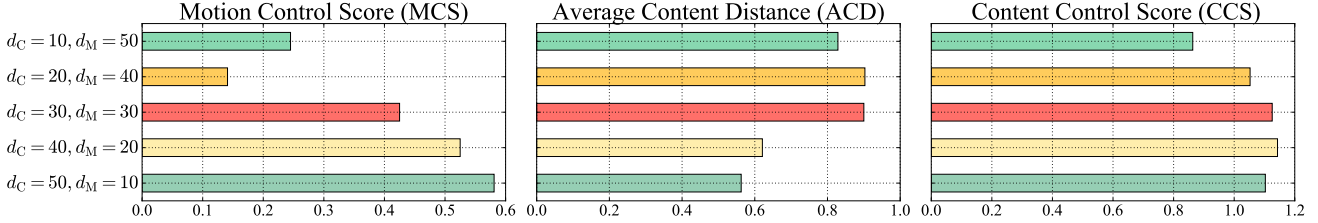


Figure 6: Performance of action-conditioned MoCoGAN models with varying d_C and d_M on facial expression generation.

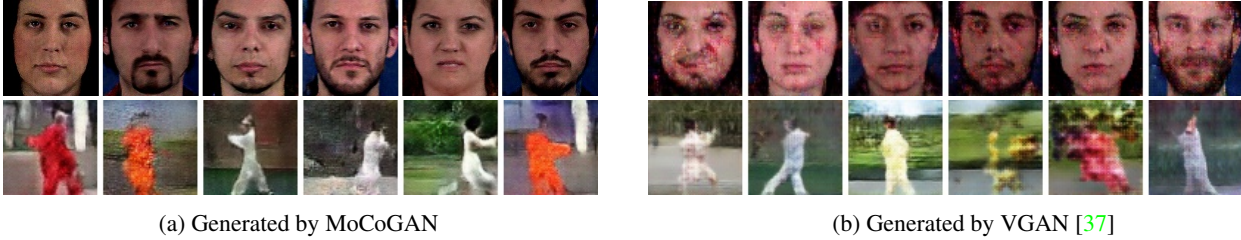


Figure 7: Randomly selected video clips used in the user study. The figure is best viewed via the Acrobat Reader. Click each image to play the video clip.

Distance (ACD), and the Content Control Score (CCS) as described in the main paper.

Figure 6 shows the results. We found that when d_C was large, the trained MoCoGAN has a small ACD. This meant that a video generated by the MoCoGAN resembled to the same person performed different expressions. The generated content was uniform. We also found that a larger d_C rendered a larger CCS, which meant that the MoCoGAN had a better control in the generated content. Different content vectors with the same motion vectors corresponded to different people performing the same expression.

We were expecting that a larger z_M would lead to a larger MCS but found the contrary. As visualizing the generated videos, we found that when d_M was large (i.e., d_C was small), the MoCoGAN failed to generate recognizable faces, which resulted in a poor MCS since the calculation of the MCS was based on a facial expression recognition network. If the generated faces were not recognizable, the facial expression recognition network could only perform a random guess on the expression and score poorly. Based on the experiment results, we adopt the setting of $d_C = 50$ and $d_M = 10$ as the preferred setting, which was used in all the other experiments.

6.3. User Study

We performed user-centric comparison of the proposed MoCoGAN framework and the VGAN work [37]. We trained both MoCoGAN and VGAN and performed a user study in which we asked Amazon Mechanical Turk (AMT) workers with approval rate $>95\%$ to select a more realistic video between the two. We randomly generated 80 videos using each algorithm and randomly paired the videos for the workers to choose. This constituted 80 questions. For

each question, we gathered answers from 3 different workers. This experimental evaluation was performed for facial expression and Tai Chi video clip generation datasets.

Table 3 presents the preferences of the AMT workers. It shows that the majority of the workers specified the presented framework as being able to generate more realistically looking video sequences. Examples of randomly selected image pairs are given in Fig. 7. We observe that MoCoGAN generates video sequences with much higher fidelity, explaining AMT workers preferences.

7. Conclusion

We presented the MoCoGAN framework for motion and content decomposed video generation. Given sufficient video training data, MoCoGAN automatically learns to disentangle motion from content in an unsupervised manner. For instance, given videos of people performing different facial expressions, MoCoGAN learns to separate a person’s identity from their expression, thus allowing us to synthesize a new video of a person performing different expressions, or fixing the expression and going through various identities. This is enabled by a new generative adversarial network, which generates a video clip by sequentially generating video frames. Each video frame is generated from a random vector, which consists of two parts, one signifying content and one signifying motion. The content subspace is modeled with a Gaussian distribution, whereas the motion subspace is modeled with a recurrent neural network. We sample this space in order to synthesize each video frame. We have validated through experiments that the proposed MoCoGAN framework is superior to current state-of-the-art video generation methods.

References

- [1] N. Aifanti, C. Papachristou, and A. Delopoulos. The mug facial expression database. In *Image Analysis for Multimedia Interactive Services (WIAMIS), 2010 11th International Workshop on*, pages 1–4. IEEE, 2010. 6, 8
- [2] B. Amos, B. Ludwiczuk, and M. Satyanarayanan. Openface: A general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, CMU School of Computer Science, 2016. 7
- [3] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017. 2
- [4] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Computer Vision and Pattern Recognition*, 2017. 6
- [5] X. Chen, Y. Duan, R. Houthoofd, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2016. 2
- [6] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 4, 6
- [7] E. L. Denton, S. Chintala, R. Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in Neural Information Processing Systems*, 2015. 2
- [8] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *International Journal of Computer Vision*, 2003. 2
- [9] I. Durugkar, I. Gemp, and S. Mahadevan. Generative multi-adversarial networks. *International Conference on Learning Representation*, 2017. 5
- [10] C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in Neural Information Processing Systems*, 2016. 3
- [11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014. 2, 3
- [12] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *PAMI*, 29(12):2247–2253, 2007. 7
- [13] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra. Draw: A recurrent neural network for image generation. *International Conference on Machine Learning*, 2015. 3
- [14] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 1997. 3, 4
- [15] D. J. Im, C. D. Kim, H. Jiang, and R. Memisevic. Generating images with recurrent adversarial networks. *arXiv preprint arXiv:1602.05110*, 2016. 3
- [16] N. Kalchbrenner, A. v. d. Oord, K. Simonyan, I. Danihelka, O. Vinyals, A. Graves, and K. Kavukcuoglu. Video pixel networks. *arXiv preprint arXiv:1610.00527*, 2016. 3
- [17] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 6
- [18] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2
- [19] Y. Li, K. Swersky, and R. S. Zemel. Generative moment matching networks. In *International Conference on Machine Learning*, 2015. 2
- [20] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. *arXiv preprint arXiv:1703.00848*, 2017. 2
- [21] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In *Advances in Neural Information Processing Systems*, 2016. 2
- [22] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015. 3
- [23] A. Nguyen, J. Yosinski, Y. Bengio, A. Dosovitskiy, and J. Clune. Plug & play generative networks: Conditional iterative generation of images in latent space. *arXiv preprint arXiv:1612.00005*, 2016. 2
- [24] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh. Action-conditional video prediction using deep networks in atari games. In *Advances in Neural Information Processing Systems*, 2015. 3
- [25] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*, 2016. 2, 6
- [26] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In *International Conference on Machine Learning*, 2016. 6
- [27] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and variational inference in deep latent gaussian models. In *International Conference on Machine Learning*, 2014. 2
- [28] M. Saito and E. Matsumoto. Temporal generative adversarial nets. *arXiv preprint arXiv:1611.06624*, 2016. 2
- [29] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, 2016. 2, 6
- [30] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using lstms. In *International Conference on Machine Learning*, 2015. 3
- [31] M. Szummer and R. W. Picard. Temporal texture modeling. In *International Conference on Image Processing*, 1996. 2
- [32] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, pages 1701–1708, 2014. 7
- [33] L. Theis, A. v. d. Oord, and M. Bethge. A note on the evaluation of generative models. *International Conference on Learning Representations*, 2016. 6
- [34] J. van Amersfoort, A. Kannan, M. Ranzato, A. Szlam, D. Tran, and S. Chintala. Transformation-based models of video sequences. *arXiv preprint arXiv:1701.08435*, 2017. 3
- [35] A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in Neural Information Processing Systems*, 2016. 2

- [36] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee. Decomposing motion and content for natural video sequence prediction. In *International Conference on Learning Representation*, 2017. [3](#)
- [37] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, 2016. [2](#), [3](#), [6](#), [7](#), [8](#), [9](#)
- [38] L.-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *SIGGRAPH Conference*, 2000. [2](#)
- [39] T. Xue, J. Wu, K. Bouman, and B. Freeman. Probabilistic modeling of future frames from a single image. In *Advances In Neural Information Processing Systems*, 2016. [3](#)
- [40] H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *arXiv preprint arXiv:1612.03242*, 2016. [2](#)