International Journal of Computer Vision (2020) 128:2514–2533
https://doi.org/10.1007/s11263-020-01328-9

Towards Image-to-Video Translation: A Structure-Aware Approach via Multi-stage Generative Adversarial Networks

Long Zhao[1] · Xi Peng[2] · Yu Tian[1] · Mubbasir Kapadia[1] · Dimitris N. Metaxas[1]

Received: 28 April 2019 / Accepted: 4 April 2020 / Published online: 28 April 2020

Abstract
In this paper, we consider the problem of image-to-video translation, where one or a set of input images are translated into an output video which contains motions of a single object. Especially, we focus on predicting motions conditioned by high-level structures, such as facial expression and human pose. Recent approaches are either driven by structural conditions or temporal-based. Condition-driven approaches typically train transformation networks to generate future frames conditioned on the predicted structural sequence. Temporal-based approaches, on the other hand, have shown that short high-quality motions can be generated using 3D convolutional networks with temporal knowledge learned from massive training data. In this work, we combine the benefits of both approaches and propose a two-stage generative framework where videos are forecast from the structural sequence and then refined by temporal signals. To model motions more efficiently in the forecasting stage, we train networks with dense connections to learn residual motions between the current and future frames, which avoids learning motion-irrelevant details. To ensure temporal consistency in the refining stage, we adopt the ranking loss for adversarial training. We conduct extensive experiments on two image-to-video translation tasks: facial expression retargeting and human pose forecasting. Superior results over the state of the art on both tasks demonstrate the effectiveness of our approach.

Keywords Image-to-video translation · Video generation · Multi-stage GANs · Motion prediction · Residual learning

1 Introduction

Generative modeling of images and videos is a fundamental but challenging problem in computer vision. Previous methods, such as Variational Auto-Encoders (VAEs) (Kingma et al. 2014; Rezende et al. 2014), adopt probabilistic graphical models to maximize the lower bound of data likelihood. Other methods, such as PixelRNN (van den Oord et al. 2016), aim to model the conditional distribution of the pixel space for image generation. Recent progress made in this field with Generative Adversarial Networks (GANs) (Goodfellow et al. 2014) has attracted a lot of research interests. During the training of GANs, a generator and a discriminator play a zero-sum game: the generator targets at producing samples towards the true data distribution to fool the discriminator, while the discriminator is optimized to distinguish between real and generated samples. GANs have shown promising performance for generative problems, and they can be utilized to synthesize sharp and realistic-looking images for various vision applications (Lu et al. 2017; Ma et al. 2017; Villegas et al. 2017b; Zhang et al. 2017a, 2018a, b).

Compared with synthesizing images, video generation is more challenging since neural networks need to learn the appearance of objects as well as their motions at the same time. We target at a form of classic problems in video generation that can be framed as image-to-video translation. In this

Communicated by Jun-Yan Zhu, Hongsheng Li, Eli Shechtman, Ming-Yu Liu, Jan Kautz, Antonio Torralba.

✉ Long Zhao
lz311@cs.rutgers.edu

Xi Peng
xipeng@udel.edu

Yu Tian
yt219@cs.rutgers.edu

Mubbasir Kapadia
mk1353@cs.rutgers.edu

Dimitris N. Metaxas
dnm@cs.rutgers.edu

1 Rutgers University, Piscataway, NJ 08854, USA

2 University of Delaware, Newark, DE 19716, USA

task, a system receives one or a set of images as the input and translates them into a video containing realistic motions of a single object with a fixed camera viewpoint, which means that the background is always static in the entire output video. Examples include facial expression retargeting (Laine et al. 2017; Olszewski et al. 2017; Thies et al. 2016), future prediction (Tulyakov et al. 2018; Villegas et al. 2017a; Vondrick et al. 2016), and human body pose forecasting (Chao et al. 2017; Fragkiadaki et al. 2015; Villegas et al. 2017b).

There are two major approaches in the field of video generation using GANs. One approach for long-term future prediction (Villegas et al. 2017b; Shi et al. 2015) is to train a transformation network that translates the input image into each future frame separately conditioned by a sequence of structures. It suggests that it is beneficial to incorporate high-level structures during the generative process. However, temporal information is only encoded in structural conditions and there are no additional constraints applied during the training of these methods. As a result, motion artifacts often exist in the final output. In parallel, recent studies (Ji et al. 2013; Simonyan and Zisserman 2014; Tran et al. 2015; Tulyakov et al. 2018; Vondrick et al. 2016) show that temporal visual features are important for modeling actions in the video. Such an approach produces temporally coherent motions with the help of spatiotemporal generative networks (implemented using 3D convolutions) but is poor at long-term conditional motion generation, since no high-level guidance is provided during training.

In this paper, we combine the benefits of these two methods. Our framework includes two motion transformation networks where the entire video is synthesized in a forecasting and then refining manner. In the forecasting stage, the *Motion Forecasting Networks* observe a single frame from the input and generate all future frames individually, which are conditioned by the structural sequence predicted by a *Motion Condition Generator*. This stage aims to generate a coarse video where the spatial structures of the motions are preserved. In the refining stage, the *Motion Refining Networks* based on 3D convolutions are used for refining the output from the previous stage. It performs the generation guided by temporal signals, which targets producing temporally coherent motions.

In the first stage, for more effective motion modeling, the transformation networks are trained in the *residual space*. Rather than learning the mapping from the structural conditions to motions directly, we force the networks to learn the differences between motions occurring in the current and future frames. The intuition is that learning only the residual motion avoids the redundant motion-irrelevant information, such as static backgrounds, which remains unchanged during the transformation. Moreover, we introduce a novel network architecture using *dense connections* for decoders. It encourages reusing spatially different features and thus yields realistic-looking results. In the second stage, the main challenge is to produce future frames with vivid motion while perverse the appearance of the object at the same time. To this end, we introduce 3D convolutional networks for realistic motion generation in the residual space. Furthermore, we present a combination of the adversarial loss and ranking loss to prevent the refining networks from reaching sub-optimal solutions.

We experiment with the proposed approach on two tasks: facial expression retargeting and human pose forecasting as shown in Fig. 1. Success in either task requires reasoning realistic spatial structures as well as temporal semantics of the motions. Note that predicting human poses is more challenging due to the highly non-linear transformations when motion changes are large. Strong performances on both tasks demonstrate the effectiveness of our approach.

The preliminary version of this paper is published in ECCV'18 (Zhao et al. 2018) and we have improved it in four important aspects: giving a more comprehensive implementation of the motion condition generators (Sect. 4), presenting more complete solution for train the refining networks (Sect. 6.2), giving the detailed network architectures of the proposed motion forecasting networks (Sect. 5.1) and refining networks (Sect. 6.1), and conducting thorough studies on the Human3.6M dataset (Ionescu et al. 2014) to evaluate the proposed approach for human pose forecasting (Sect. 7.3). In summary, we show the following *contributions* in this paper.

- We devise a novel two-stage generation framework for image-to-video translation, where the future frames are generated according to the spatial structural sequence and then refined with temporal signals;
- We investigate learning residual motion for video generation, which focuses on the motion-specific knowledge and avoids learning redundant or irrelevant details from the inputs;
- Dense connections between layers of decoders are introduced to encourage spatially different feature reuse during the generation process, which yields more realistic-looking results;
- We conduct extensive experimental validation on standard datasets to quantitatively and subjectively compare our method with the state of the art to demonstrate the effectiveness of the proposed algorithm.

2 Related Work

Video generation has been an important but challenging problem of computer vision in the literature. In this section, we briefly discuss previous works on video generation that are

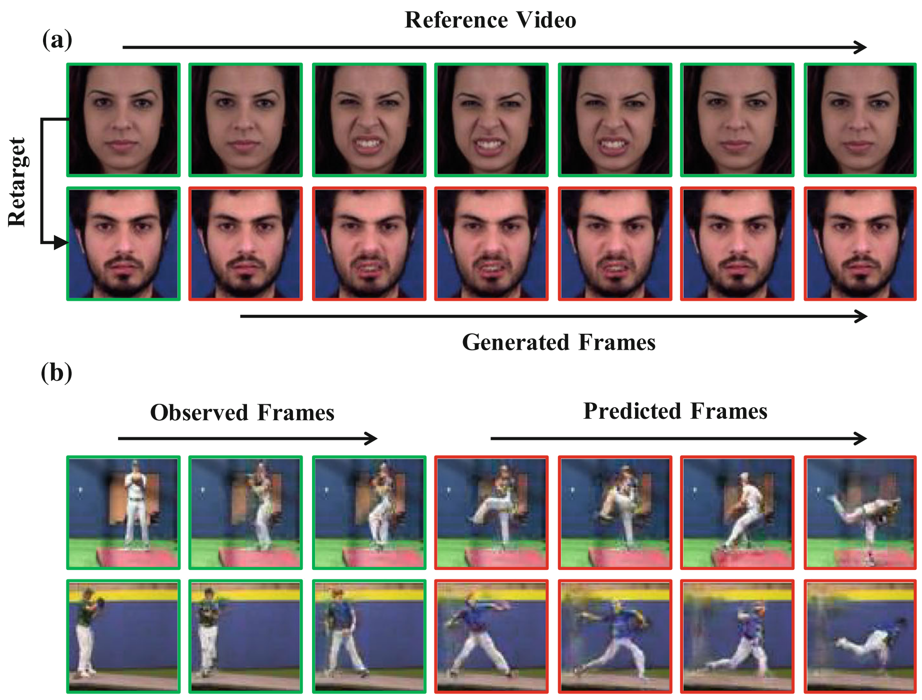

Fig. 1 Two typical problems of image-to-video translation that our approach is able to address: **a** facial expression retargeting and **b** human body pose forecasting

related to our approach, as well as advances in deep learning, such as the development of generative adversarial networks, which has significantly boosted the performance of generative models.

2.1 Generative Adversarial Networks (GANs)

Deep learning techniques have improved the accuracy of various vision systems (Peng et al. 2016, 2015; Tang et al. 2018a, b; Zhao et al. 2019a). Especially, a lot of generative problems (Gulrajani et al. 2017; Peng et al. 2018; Perarnau et al. 2016; Tian et al. 2018) have been tackled by Generative Adversarial Networks (GANs). GANs are first proposed by Goodfellow et al. (2014) to estimate target distribution with a generator and a discriminator. Arjovsky et al. (2017) presented WGAN, a more stable approach to enforce Lipschitz constraint for GANs. Gulrajani et al. (2017) further improved WGAN with gradient penalty. Moreover, Mirza and Osindero (2014) extended traditional GANs by leveraging labeled data other than noise distributions as the input to the generator. Such conditional models are called conditional GANs, which allow flexible control over the output of the model conditioned by various forms of data, including images (Tulyakov et al. 2018; Villegas et al. 2017b; Vondrick et al. 2016), categorical labels (Odena et al. 2017; Tian et al. 2018) and textual descriptions (Reed et al. 2016; Zhang et al. 2017b, c). Our method belongs to the category of conditional video generation with GANs and is built upon the guidance of these previous works.

2.2 Video Generation

Recent methods extend conventional GAN frameworks by leveraging 3D convolutions or recurrent neural networks to model temporal information for video generation. For instance, VGAN (Vondrick et al. 2016) demonstrated that 3D convolutional networks could model scene dynamics from massive data. Saito et al. (2017) proposed TGAN, a GAN architecture which generates multiple frames at the same time. Tulyakov et al. (2018) proposed MoCoGAN combining a recurrent neural network with an adversarial learning scheme to decompose motion and content for video generation. However, due to the unconditional setting, the results of these methods are often with low resolution or short length.

Conditional video generation is closely related to the problem of future frame prediction addressed in Finn et al. (2016), Liang et al. (2017), Liu et al. (2017), Mathieu et al. (2016), Srivastava et al. (2015), Villegas et al. (2017a). These methods aim to synthesize a sequence of images representing a coherent continuation of the given video. Wang et al. (2018) proposed video-to-video synthesis whose goal is to learn a mapping function from an input source video, such as a sequence of semantic segmentation masks, to an output photo-realistic video that precisely depicts the content of the source video. Pan et al. (2019) introduced a task which extends video-to-video synthesis by letting the output video be conditioned on a single semantic label map. Although these methods can handle any datasets with general motions, the main drawback of these approaches is that they can only predict short video clips with the fixed length. Another direction of conditional video generation is to make long-term predictions for videos of specific types such as human poses and faces (Villegas et al. 2017b; Yang et al. 2018; Zhao et al. 2018). These methods rely on a sequence of conditions as the guidance to generate plausible videos. Our study is related to the latter.

Image-to-video translation aims to predict future frames from input images, which is the extreme case of conditional video generation. Recent methods (Denton and Birodkar 2017; Lotter et al. 2017; Mathieu et al. 2016; Villegas et al. 2017a, b; Shi et al. 2015) train transformation networks that translate the input image into each future frame separately, together with an optional generator predicting the structure sequence which conditions the future frames. However, due to the absence of pixel-level temporal knowledge during the training process, motion artifacts can be observed from the results of these methods. Other approaches explore learning temporal visual features from video with 3D convolutional networks. Ji et al. (2013) showed how 3D convolutions could be applied to human action recognition. Tran et al. (2015) employed 3D convolutions to model features encoded in videos. Vondrick et al. (2016) built a conditional model to generate scene dynamics with 3D generative adversarial networks. Our method differs from the two-stream model of Vondrick et al. (2016) in two aspects. First, our residual motion map disentangles motion from the input: the generated frame is conditioned on the current and future motion structures. Second, we can control object motions in future frames efficiently by using structure conditions. Thus our method can be applied to motion manipulation problems.

2.3 Multi-stage GANs

Conventional GANs usually fail to handle complicated generative problems, e.g., to generate fine-grained images or videos with large motion changes. Recent approaches proved that coarse-to-fine strategy can handle these cases. Our approach is closely related to these GAN-based methods in which the generator produces images or videos in multiple stages. Denton et al. (2015) proposed LAPGAN where a series of GANs are built within a Laplacian pyramid. At each level of the pyramid, the generator produces a coarse image and updates it by using a difference of an initial image. ProgressiveGAN (Karras et al. 2018) generated high-resolution images by incrementally adding more layers to both the generator and the discriminator. Zhang et al. (2017c) introduced StackGAN which generated images from text descriptions through a sketch-refinement process. StackGAN++ (Zhang et al. 2017b) further improved this two-stage framework by synthesizing multi-scale images via multiple generators and discriminators. Li et al. (2018) formulated the multi-frame prediction task as a multi-flow prediction phase followed by a flow-to-frame synthesis phase, which is modeled by spatial-temporal relationships learned through 3D convolutions.

Our model also adopts a multi-stage strategy for video generation. The closest work to our approach is presented by Xiong et al. (2018). Although it generates video within two stages, there are important differences between their work and ours. First, Xiong et al. (2018) is proposed for time-lapse videos while we can generate general videos. Second, we use structure conditions to guide the generation in the first stage but Xiong et al. (2018) performs generation in the same stage with 3D convolutional networks. Third, we can make long-term predictions while Xiong et al. (2018) only generates videos with fixed length.

3 Approach Overview

Our framework consists of three components: a motion condition generator G_C, an image-to-image transformation network G_M for forecasting motion conditioned by G_C to each future frame individually, and a video-to-video transformation network G_R which aims to refine the video clips concatenated from the output of G_M. The pipeline of our approach is illustrated in Fig. 2.

The rest of the paper explains each component in detail which is organized as follows. In Sect. 4, we introduce G_C, a task-specific generator that produces a sequence of structures to condition the motion of each future frame. We present two example implementations of G_C for illustration, where domain knowledge is involved to guide the prediction. In Sects. 5 and 6, the detailed architectures and implementations of G_M and G_R are discussed respectively. Note that two discriminators are utilized for adversarial learning with G_M and G_R, where D_I differentiates real frames from generated ones and D_V is employed for video clips. These networks are trained based on WGAN (Arjovsky et al. 2017; Gulrajani et al. 2017).

4 Motion Condition Generators

In this section, we illustrate how the motion condition generators G_C are implemented for two image-to-video translation tasks: facial expression retargeting and human pose forecasting. One superiority of the proposed G_C is that the domain-specific knowledge, such as 3D morphable model or 2D landmarks, can be leveraged to help the prediction of motion structures.

3DMM for Facial Structure Generator. As shown in Fig. 3, we utilize the 3D Morphable Model (3DMM) (Blanz and Vetter 2003) to model the sequence of expression motions. For the target image and each frame in the reference video, the 3DMM describes the 3D face with PCA as:

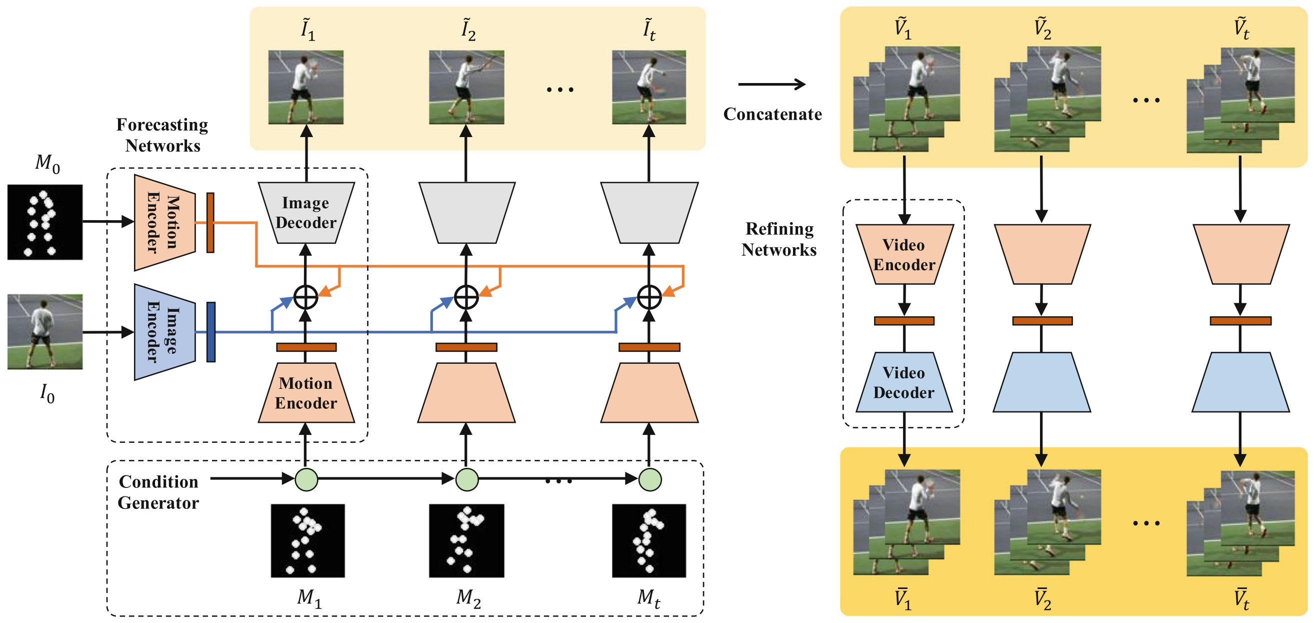

Fig. 2 Overview of our framework. A condition generator predicts a sequence of structural conditions from the input images. Then the video is generated from these conditions with motion forecasting networks and refined by motion refining networks

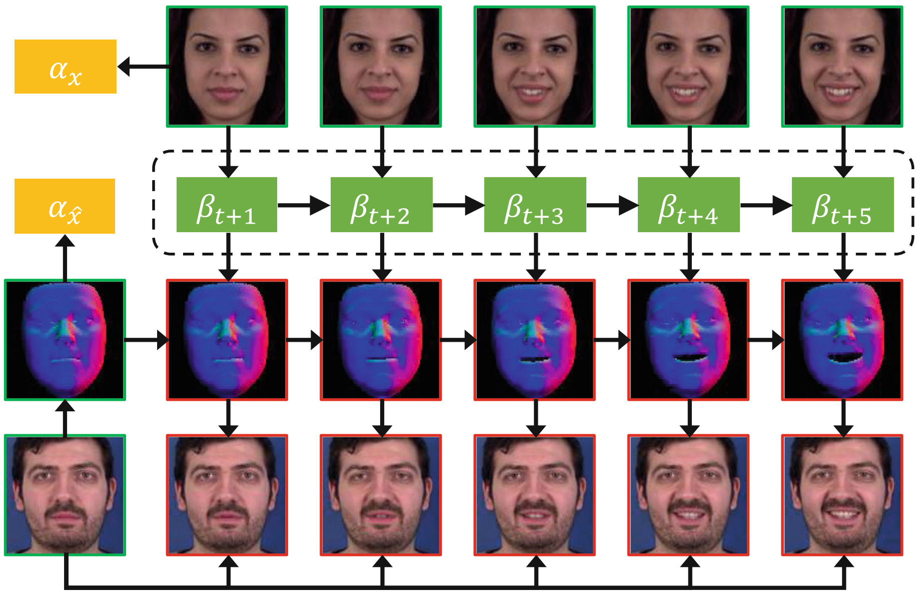

Fig. 3 Illustration of the facial structure generator. 3DMM is leveraged to estimate the facial parameters of the reference video and target image. The retargeted facial structures are generated by transferring expression parameters from the reference frames to the target

$$\mathbf{S} = \bar{\mathbf{S}} + \mathbf{A}_{id}\alpha + \mathbf{A}_{exp}\beta, \tag{1}$$

where $\mathbf{S}$ is the 3D face mesh, $\bar{\mathbf{S}}$ is the mean mesh, $\mathbf{A}_{id}$ and $\mathbf{A}_{exp}$ are the principle axes trained on the 3D face scans for the facial identity and expression respectively, and their corresponding parameters are represented by α and β. We follow Zhu et al. (2016, 2019) to calculate $\mathbf{A}_{id}$ from BFM (Paysan et al. 2009) and $\mathbf{A}_{exp}$ from FaceWarehouse (Cao et al. 2014) respectively. We can then project the 3D face mesh into the image plane by:

$$F = \mathbf{P} \cdot \mathbf{R} \cdot \mathbf{S} + \mathbf{t}, \tag{2}$$

where $\mathbf{P}$ is the orthographic projection matrix, $\mathbf{R}$ represents the rotation matrix, and $\mathbf{t}$ is the 2D translation vector of the mesh respectively.

Given a video containing expression changes of an actor x, it can be parameterized with α_x and $(\beta_t, \beta_{t+1}, \ldots, \beta_{t+k})$ using the 3DMM fitting method as presented by Zhu et al. (2016, 2019), where α_x represents the facial identity and β_t is the expression coefficients in the frame t. In order to retarget the sequence of expressions to another actor $\hat{x}$, we compute the facial identity vector $\alpha_{\hat{x}}$ and combine it with $(\beta_t, \beta_{t+1}, \ldots, \beta_{t+k})$ to reconstruct a new sequence of 3D face models with corresponding facial expressions. The con-

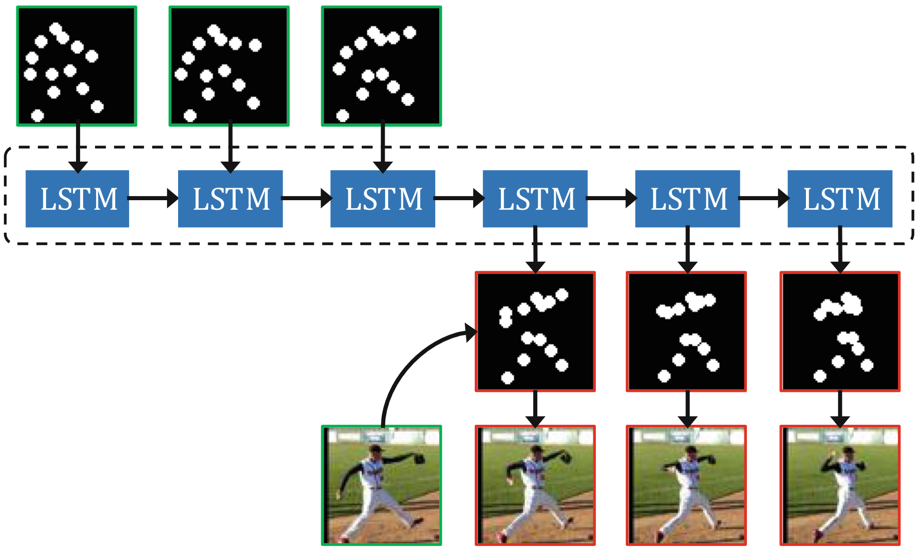

Fig. 4 Illustration of the pose structure predictor. LSTM observes t consecutive human pose inputs and predicts the poses for the next K timesteps. Note that the heatmaps here are used for illustration, but our network observes and outputs 2D coordinates

ditional motion maps are the normal maps calculated from the 3D face models by Eq. (2) respectively.

Human Pose Predictor with LSTM. We follow Villegas et al. (2017b) to implement an LSTM architecture (Fragkiadaki et al. 2015) as the human pose predictor. The human pose of each frame is represented by the 2D coordinate positions of joints. The LSTM observes consecutive pose inputs to identify the type of motion, and then predicts the poses for the next period of time. An example is shown in Fig. 4. Specifically, the LSTM first encodes the observed motion structures by:

$$[\mathbf{h}_t, \mathbf{c}_t] = \mathrm{LSTM}(\mathbf{x}_t, \mathbf{h}_{t-1}, \mathbf{c}_{t-1}), \tag{3}$$

where $\mathbf{h}_t$ is the hidden state which encodes the observed dynamics up to time t, $\mathbf{c}_t$ represents the memory cell of the LSTM, and $\mathbf{x}_t$ is the human pose at time t. The LSTM needs to observe a few human poses as the input to identify the type of motion so that it can make a reasonable prediction of the future pose by:

$$\mathbf{y}_t = f(\mathbf{w}^\top \mathbf{h}_t), \tag{4}$$

where $\mathbf{w}$ is a learnable matrix which projects the hidden state into the 2D coordinate positions of joints, f is an activation function, and $\mathbf{y}_t$ is the predicted pose.

A sequence-to-sequence approach (Villegas et al. 2017b) is leveraged to predict the future structures of human poses. In particular, we unroll the LSTM for K timesteps to allow it to observe K poses before making the predictions. Therefore, the LSTM is trained to minimize the following loss function:

$$\ell_{pose} = \frac{1}{KL}\sum_{k=1}^{K}\sum_{l=1}^{L} \|\mathbf{y}^l_{t+k} - \mathbf{x}^l_{t+k}\|_2^2, \tag{5}$$

where $\mathbf{y}^l_{t+k}$ and $\mathbf{x}^l_{t+k}$ denote the l-th pose landmark of the prediction and ground truth respectively. We note that the final motion map is calculated by mapping the output 2D coordinates from the LSTM to heatmaps and then concatenating them on depth.

5 Motion Forecasting Networks

Starting from the frame I_t at time t, our network synthesizes the future frame I_{t+k} by predicting the residual motion between them. Previous work (Ma et al. 2017; Shen and Liu 2017) implemented this idea by letting the network estimate a difference map between the input and output, which can be denoted as:

$$I_{t+k} = I_t + G_M(I_t | M_t, M_{t+k}), \tag{6}$$

where M_t is the motion map which conditions I_t. However, this straightforward formulation easily fails when employed to handle videos including large motions. This is due to the fact that learning to generate a combination of residual changes from both dynamic and static contents in a single map is quite difficult. Therefore, we introduce an enhanced formulation where the transformation is disentangled into a residual motion map m_{t+k} and a residual content map c_{t+k} with the following definition:

$$I_{t+k} = \underbrace{m_{t+k} \odot c_{t+k}}_{\text{residual motion}} + \underbrace{(1 - m_{t+k}) \odot I_t}_{\text{static content}}, \tag{7}$$

where both m_{t+k} and c_{t+k} are predicted by G_M, and $\odot$ is element-wise multiplication. Intuitively, $m_{t+k} \in [0, 1]$ can be viewed as a spatial mask that highlights where the motion occurs. c_{t+k} is the content of the residual motions. By sum-

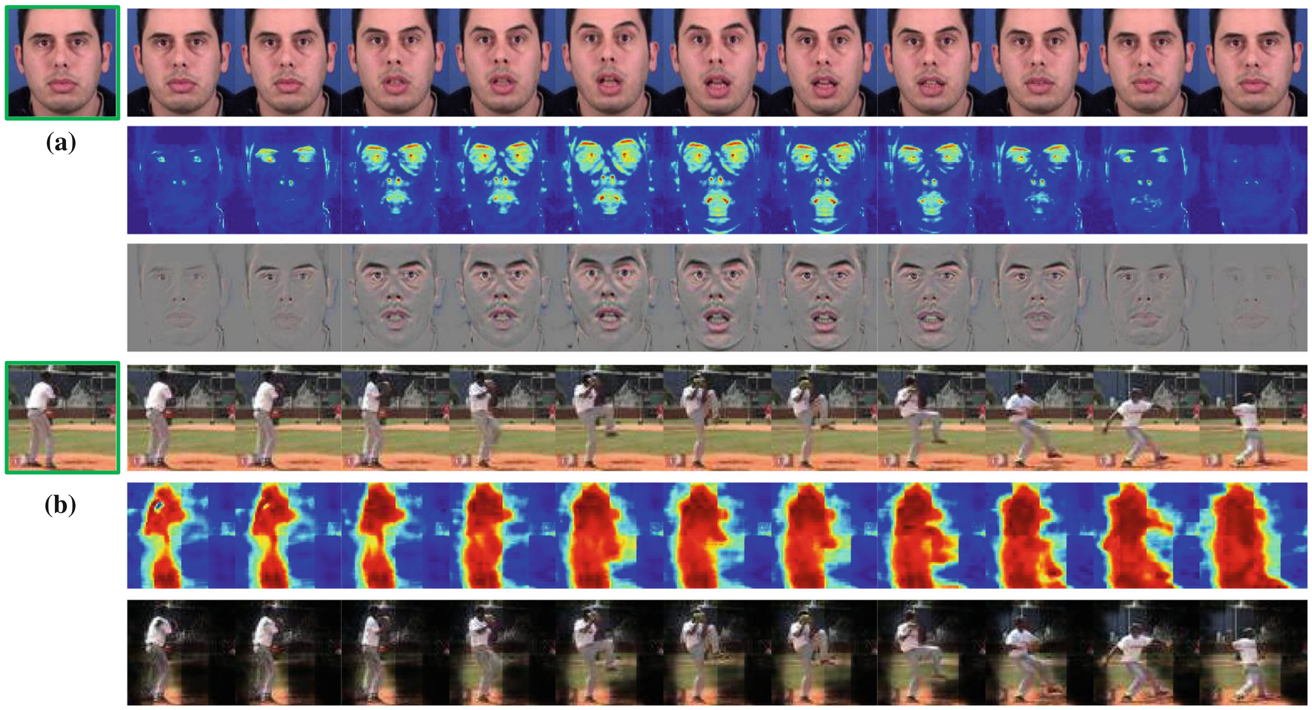

Fig. 5 Illustration of our residual formulation. We disentangle the motion differences between the input and future frames into a residual motion map m_{t+k} (Middle) and a residual content map c_{t+k} (Bottom). We show examples of **a** facial expression retargeting and **b** human body pose forecasting. Compared with the difference map directly computed from them, our formulation makes the learning task much easier

ming the residual motion with the static content, we can obtain the final result. Note that as visualized in Fig. 5, m_{t+k} forces G_M to reuse the static part from the input and concentrate on inferring dynamic motions.

Connection to Attention Mechanism. We note that our residual formulation in Eq. (7) is closely related to the attention module (Vaswani et al. 2017) recently presented for generating facial animation (Pumarola et al. 2018). This can be seen from the fact that at time $t + k$, m_{t+k} defined in Eq. (7) defines a per pixel intensity specifying to those pixels of the frame I_t which will contribute in the future frame I_{t+k}. It shares the same concept of the facial attention mask proposed in Pumarola et al. (2018). As such, our work provides insight by relating the recent attention mechanism to the classic residual formulation in computer vision, and extends the facial animation network in Pumarola et al. (2018) to a more generic motion prediction network for image/video generation problems in computer vision.

Despite the strong relation to Pumarola et al. (2018), we show that the attentional behavior is not limited to producing facial animations which contain local deformations, but it can also model videos with large motions such as human body poses. This is illustrated in Fig. 5.

5.1 The Architecture

Figure 6 shows the architecture of G_M, which is inspired by the visual-structure analogy learning (Reed et al. 2015). Table 1 presents the network specification. The future frame I_{t+k} can be generated by transferring the structural differences between M_t and M_{t+k} to the input frame I_t. We use a motion encoder f_M, an image encoder f_I and a residual content decoder f_D to model this concept. And the residual motion is learned by:

$$\Delta(I_{t+k}, I_t) = f_D(f_M(M_{t+k}) - f_M(M_t) + f_I(I_t)). \tag{8}$$

Intuitively, f_M aims to identify key motion features from the motion map containing high-level structural information; f_I learns to map the appearance model of the input into an embedding space, where the motion feature transformations can be easily imposed to generate the residual motion; f_D learns to decode the embedding. Note that we add skip connections (Ronneberger et al. 2015) between f_I and f_D, which makes it easier for f_D to reuse features of static objects learned from f_I.

Dense Connections for Decoders. Recent studies Huang et al. (2018, 2017) introduced dense connections for image classification, which enhance feature propagation and reuse

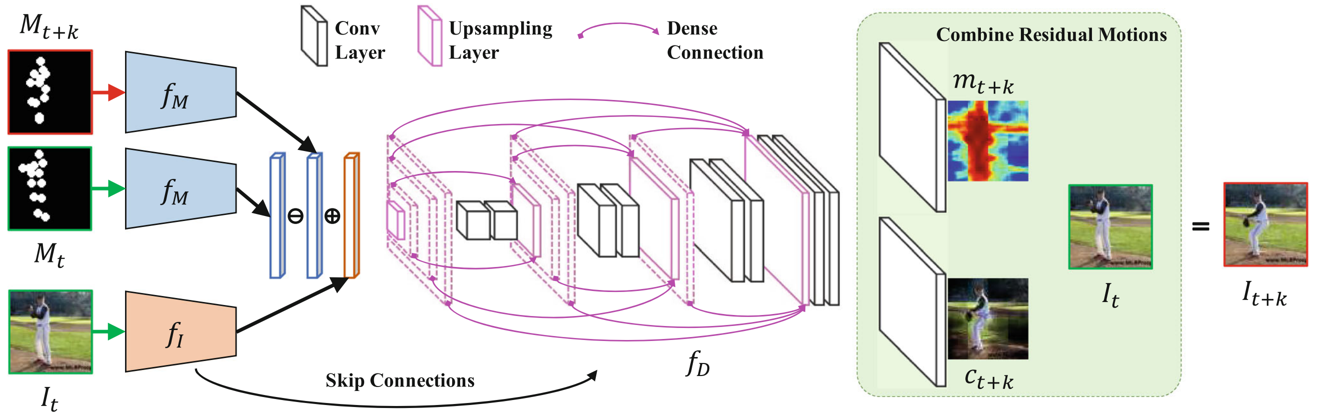

Fig. 6 Architecture of our motion forecasting network G_M. The network observes the input frame I_t with its corresponding motion map M_t, and the motion map of the future frame M_{t+k}. Through analogy learning, the network estimates the *residual motion* between the current frame I_t and future frame I_{t+k}. Note that the *dashed* layers upsample the inputs and connect them to the subsequent dense blocks

in the network. They have proven that dense connections for encoders strengthen feature propagation and also encourage feature reuse. We argue that this is an appealing property for motion transformation networks as well, since in most cases the output frame shares similar high-level structure with the input frame. Especially, dense connections make it easy for the network to reuse features of different spatial positions when large motions are involved in the image.

In this work, we introduce dense connections for decoders. The decoder of our network thus consists of multiple *dense connections*, each of which connects different dense blocks. A dense block contains two 3×3 convolutional layers. The output of a dense block is connected to the first convolutional layers located in *all* subsequent blocks in the network. As dense blocks have different feature resolutions, we upsample feature maps with lower resolutions when we use them as inputs into higher resolution layers.

Compared with multi-scale feature fusion in Liu et al. (2017) where feature maps are only concatenated to the last layer of the network, our dense connections upsample and concatenate feature maps with different scales to all intermediate layers. As a result, our approach is more efficient at feature reuse when utilized for generation, which yields sharper and more realistic-looking results.

5.2 Network Training

Given a video clip, we train our network to perform random jumps in time to learn forecasting motion changes. To be specific, for every iteration at training time, we sample a frame I_t and its corresponding motion map M_t given by G_C at time t, and then force it to generate frame I_{t+k} given motion map M_{t+k}. Note that in order to let our network perform learning in the entire residual motion space, k is also randomly defined for each iteration. On the other hand, learning with jumps in time can prevent the network from falling into suboptimal parameters as well (Villegas et al. 2017b). Our network is trained to minimize the following objective function:

$$\ell_{G_M} = \ell_{rec}(I_{t+k}, \tilde{I}_{t+k}) + \ell_r(m_{t+k}) + \ell_{gen}. \tag{9}$$

ℓ_{rec} is the reconstruction loss defined in the image space which measures the pixel-wise differences between the predicted and target frames:

$$\ell_{rec}(I_{t+k}, \tilde{I}_{t+k}) = \|I_{t+k} - \tilde{I}_{t+k}\|_1, \tag{10}$$

where $\tilde{I}_{t+k}$ denotes the frame predicted by G_M. The reconstruction loss intuitively offers guidance for our network in making a rough prediction that preserves most content information of the target image. More importantly, it leads the result to share similar structure information with the input image. ℓ_r is an L-1 norm regularization term defined as:

$$\ell_r(m_{t+k}) = \|m_{t+k}\|_1, \tag{11}$$

where m_{t+k} is the residual motion map predicted by G_M. It forces the predicted motion changes to be sparse, since dynamic motions always occur in local positions of each frame while the static parts (e.g., background objects) should be unchanged. Furthermore, ℓ_{gen} is the adversarial loss that enables our model to generate realistic frames and reduce blurs, and it is defined as:

$$\ell_{gen} = -D_I([\tilde{I}_{t+k}, M_{t+k}]), \tag{12}$$

where D_I is the discriminator for images in adversarial learning. We concatenate the output of G_M and motion map M_{t+k} as the input of D_I and make the discriminator conditioned on the motion (Mirza and Osindero 2014).

Table 1 Specific design of the motion forecasting network G_M: block name (Top), feature map dimension (Middle), and layer configuration (Bottom)

E_0	E_1	E_2	E_3	E_4	D_4	D_3	D_2	D_1	D_0
64×64	32×32	16×16	8×8	4×4	8×8	16×16	32×32	64×64	64×64
$1\times$ Conv	$2\times$ Conv	$2\times$ Conv	$2\times$ Conv	$2\times$ Conv	Unpooling	Unpooling	Unpooling	Unpooling	–
$\begin{bmatrix} 3 \times 3, 64 \end{bmatrix}$	$\begin{bmatrix} 3 \times 3, 64 \end{bmatrix}$	$\begin{bmatrix} 3 \times 3, 128 \end{bmatrix}$	$\begin{bmatrix} 3 \times 3, 256 \end{bmatrix}$	$\begin{bmatrix} 3 \times 3, 512 \end{bmatrix}$	$2\times$ Conv	$2\times$ Conv	$2\times$ Conv	$2\times$ Conv	$1\times$ Conv
–	Pooling	Pooling	Pooling	Pooling	$\begin{bmatrix} 3 \times 3, 512 \end{bmatrix}$	$\begin{bmatrix} 3 \times 3, 256 \end{bmatrix}$	$\begin{bmatrix} 3 \times 3, 128 \end{bmatrix}$	$\begin{bmatrix} 3 \times 3, 64 \end{bmatrix}$	$\begin{bmatrix} 3 \times 3, 3 \end{bmatrix}$

$\begin{bmatrix} 3 \times 3, 64 \end{bmatrix}$ means there are 64 filters (channels) and each has a size of 3×3. Pooling or unpooling operations are performed after or before each block. The pooling window is 2×2 with a stride of 2. ReLU is employed in both the encoder (E_*) and decoder (D_*) after each convolution, while batch normalization is only used in D_*. E_1 to D_1 are all residual blocks as in Gulrajani et al. (2017). f_M, f_I and the image discriminator D_I share the same design of E_*

Note that we follow WGAN (Arjovsky et al. 2017; Gulrajani et al. 2017) to train D_I to measure the Wasserstein distance between distributions of the real images and results generated from G_M. During the optimization of D_I, the following loss function is minimized:

$$\ell_{D_I} = D_I([\tilde{I}_{t+k}, M_{t+k}]) - D_I([I_{t+k}, M_{t+k}]) + \lambda \cdot \ell_{gp}, \quad (13)$$

$$\ell_{gp} = (\|\nabla_{[\hat{I}_{t+k}, M_{t+k}]} D_I([\hat{I}_{t+k}, M_{t+k}])\|_2 - 1)^2, \quad (14)$$

where λ is experimentally set to 10. Note that ℓ_{gp} is the gradient penalty term proposed by Gulrajani et al. (2017) where $\hat{I}_{t+k}$ is sampled from the interpolation of I_{t+k} and $\tilde{I}_{t+k}$, and we extend it to be conditioned on the motion M_{t+k} as well. The adversarial loss in combination with the rest of loss terms allows our network to generate high-quality frames given the conditions which encode motion structures.

6 Motion Refining Networks

Let $\tilde{V}_t = [\tilde{I}_{t+1}, \tilde{I}_{t+2}, \ldots, \tilde{I}_{t+K}]$ be the video clip with length K temporally concatenated from the outputs of G_M. The goal of the motion refining network G_R is to refine $\tilde{V}_t$ to be more temporally coherent, which is achieved by performing pixel-level refinement with the help of spatiotemporal generative networks. We extend Eq. (7) by adding one additional temporal dimension to let G_R estimate the residual between the real video clip V_t and $\tilde{V}_t$, which is defined as:

$$V_t = m_t \odot c_t + (1 - m_t) \odot \tilde{V}_t, \quad (15)$$

where m_t is a spatiotemporal mask which selects either to be refined for each pixel location and timestep, while c_t produces a spatiotemporal cuboid which stands for the refined motion content masked by m_t.

6.1 The Architecture

Our motion refining network roughly follows the guideline of Vondrick et al. (2016). As shown in Fig. 7, we do not use pooling layers, instead strided and fractionally strided 3D convolutions are utilized for in-network downsampling and upsampling. Note that we concatenate the frames with their corresponding conditional motion maps as the inputs to guide the refinement. We also add skip connections to encourage feature reuse. However, when training the network, we find that adding skip connections everywhere in the network will easily lead to suboptimal results. It suggests that skip connections mainly contribute to generate structural content but are not helpful for temporal motion generation. This observation is also consistent with (Xiong et al. 2018). Therefore, we remove the skip connections from the first two layers of

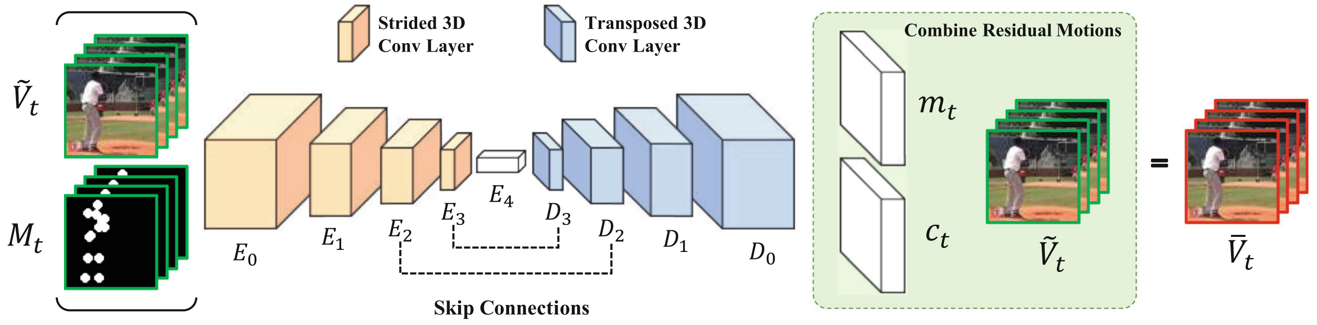

Fig. 7 Architecture of our motion refining network G_R. The network receives temporally concatenated frames generated by G_M together with their corresponding conditional motion maps as the input and aims to refine the video clip to be more temporally coherent. It performs learning in the residual motion space as well. Note that skip connections are only employed in deeper layers

the network while keep the rest in deeper layers as shown in Fig. 7. Table 2 presents the network specification.

6.2 Network Training

The key requirement for G_R is that the refined video should be temporal coherent in motion while preserving the annotation information from the input. To this end, we propose to train the network by minimizing a combination of three losses which is similar to Eq. (9):

$$\ell_{G_R} = \ell_{rec}(V_t, \bar{V}_t) + \ell_r(m_t) + \bar{\ell}_{gen} + \gamma \cdot \ell_{rank}(V_t, \tilde{V}_t, \bar{V}_t), \tag{16}$$

where $\bar{V}_t$ is the output of G_R. ℓ_{rec} and ℓ_r share the same definition with Eqs. (10) and (11) respectively. ℓ_{rec} is the reconstruction loss that aims at refining the synthesized video towards the ground truth with minimal error. Compared with the self-regularization loss proposed by Shrivastava et al. (2017), we argue that the sparse regularization term ℓ_r is also efficient to preserve the annotation information (e.g., the facial identity and the type of pose) during the refinement, since it forces the network to only modify the essential pixels.

$\bar{\ell}_{gen}$ is the adversarial loss which is defined as:

$$\bar{\ell}_{gen} = -D_V([\bar{V}_t, \mathbf{M}_t]) - \frac{1}{K}\sum_{i=1}^{K} D_I([\bar{I}_{t+i}, M_{t+i}]), \tag{17}$$

where $\mathbf{M}_t = [M_{t+1}, M_{t+2}, \ldots, M_{t+K}]$ is the temporally concatenated condition motion maps, and $\bar{I}_{t+i}$ is the i-th frame of $\bar{V}_t$. In the adversarial learning term $\bar{\ell}_{gen}$, both D_I and D_V play the role to judge whether the input is a real video clip or not, providing criticisms to G_R. The image discriminator D_I criticizes G_R based on individual frames, which is trained to determine if each frame is sampled from a real video clip. At the same time, the video discriminator D_V provides criticisms to G_R based on the whole video clip, which takes a fixed length video clip as the input and judges if a video clip is sampled from a real video as well as evaluates the motions contained. As suggested by Tulyakov et al. (2018), although D_V alone should be sufficient, D_I significantly improves the convergence and the final results of G_R.

One major challenge of training the motion refining network G_R in this stage is that the vanilla adversarial loss usually leads G_R to suboptimal solutions. Since $\tilde{V}_t$ (i.e., the input of G_R) encodes the similar motion structure with V_t, G_R trends to learn an identity mapping between $\tilde{V}_t$ and $\bar{V}_t$ if no additional constraints are employed to guide the network training. Our main idea is to optimize over distance comparisons between refined video clips $\bar{V}_t$ with those from the input $\tilde{V}_t$ and target V_t. We consider that the motion feature representation of refined result $\bar{V}_t$ should be closer to those of real data V_t than that of the input $\tilde{V}_t$. In this sense, we adopt the contrasting lose proposed in Liang et al. (2018) to compute the distance of motion features between video clips. Therefore, the ranking loss ℓ_{rank} is defined as:

$$\begin{aligned}&\ell_{rank}(V_t, \tilde{V}_t, \bar{V}_t)\\&= -\log \frac{e^{-\|f_{D_V}(\bar{V}_t) - f_{D_V}(V_t)\|_1}}{e^{-\|f_{D_V}(\bar{V}_t) - f_{D_V}(V_t)\|_1} + e^{-\|f_{D_V}(\bar{V}_t) - f_{D_V}(\tilde{V}_t)\|_1}},\end{aligned} \tag{18}$$

where f_{D_V} is a differentiable function which computes the motion feature via D_V. Following the guidance of Xiong et al. (2018), we implement f_{D_V} as the Gram matrix (Gatys et al. 2015) computed from the feature maps in different layers of f_{D_V}. During the training process, G_R minimizes the ranking loss ℓ_{rank} so that the feature distance between the refined $\bar{V}_t$ and the real V_t is encouraged to be smaller, while the distance between $\bar{V}_t$ and $\tilde{V}_t$ should be larger. By optimizing the network in such a manner, G_R is able to improve the quality of $\tilde{V}_t$.

Table 2 Specific design of the motion refining network G_R: block name (Top), feature map dimension (Middle), and layer configuration (Bottom)

E_0	E_1	E_2	E_3	E_4	D_3	D_2	D_1	D_0
$8 \times 32 \times 32$	$4 \times 16 \times 16$	$2 \times 8 \times 8$	$1 \times 4 \times 4$	$1 \times 1 \times 1$	$1 \times 4 \times 4$	$2 \times 8 \times 8$	$4 \times 16 \times 16$	$8 \times 32 \times 32$
Conv (128)	Conv (256)	Conv (512)	Conv (1024)	Conv (1024)	ConvT (1024)	ConvT (512)	ConvT (256)	ConvT (128)
$[4 \times 4 \times 4, 2]$	$[4 \times 4 \times 4, 2]$	$[4 \times 4 \times 4, 2]$	$[4 \times 4 \times 4, 2]$	$[1 \times 4 \times 4, 1]$	$[1 \times 4 \times 4, 1]$	$[4 \times 4 \times 4, 2]$	$[4 \times 4 \times 4, 2]$	$[4 \times 4 \times 4, 2]$
ReLU	ReLU	ReLU	ReLU	–	ReLU, BN	ReLU, BN	ReLU, BN	ReLU, BN

Conv (128) means the 3D convolutional layer with 128 filters (channels). ConvT denotes the transposed 3D convolutional layer. $[4 \times 4 \times 4, 2]$ means the filter has a size of $4 \times 4 \times 4$ with stride 2. ReLU is employed in both the encoder (E_*) and decoder (D_*) after each convolution. Batch normalization is only used in D_*. The video discriminator D_V shares the same design of E_*

We follow the same strategy as introduced in Eq. (13) to optimize D_I. Note that in each iteration, one pair of real and generated frames is randomly sampled from V_t and $\bar{V}_t$ to train D_I. On the other hand, training D_V is also based on the WGAN framework, where we extend it to spatiotemporal inputs. Therefore, D_V is optimized by minimizing the following loss function:

$$\ell_{D_V} = D_V([\bar{V}_t, \mathbf{M}_t]) - D_V([V_t, \mathbf{M}_t]) + \lambda \cdot \ell_{gp} - \gamma \cdot \ell_{rank}, \tag{19}$$

$$\ell_{gp} = (\|\nabla_{[\hat{V}_t, \mathbf{M}_t]} D_V([\hat{V}_t, \mathbf{M}_t])\|_2 - 1)^2, \tag{20}$$

where $\hat{V}_t$ is sampled from the interpolation of V_t and $\bar{V}_t$. D_V also maximizes the ranking loss ℓ_{rank}. The intuition is that if D_V is updated by expecting that the distance between $\bar{V}_t$ and V_t is not small enough, then G_R is encouraged to generate the result that is closer to the real data and further away from the input in the next iteration. Note that G_R, D_I and D_V are trained alternatively. To be specific, we update D_I and D_V in one step while fixing G_R; in the alternating step, we fix D_I and D_V while updating G_R.

7 Experiments

We perform experiments on two image-to-video translation tasks: facial expression retargeting and human pose forecasting. For facial expression retargeting, we demonstrate that our method is able to combine domain-specific knowledge, such as 3DMM, to generate realistic-looking results. For human pose forecasting, experimental results show that our method yields high-quality videos when applied for video generation tasks containing complex motion changes.

7.1 Settings and Datasets

To train our networks, we use Adam (Kingma and Ba 2014) for optimization with a learning rate of 0.0001 and momentums of 0.0 and 0.9. We first train the forecasting networks, and then train the refining networks using the generated coarse frames. The batch size is set to 32 for all networks. We empirically set λ to 10 and γ to 1. We list configurations and setups of each dataset employed in the experiments in Table 3.

We use the *MUG Facial Expression Database* (Aifanti et al. 2010) to evaluate our approach on facial expression retargeting. It is composed of 86 subjects (35 women and 51 men). We crop the face regions with regards to the ground truth landmarks and scale them to 128×128. To train our networks, we use only the sequences representing one of the six facial expressions: anger, fear, disgust, happiness, sadness, and surprise. We evenly split the database into three groups

Table 3 The video datasets used for training and evaluation in the experiments

	MUG (Kingma and Ba 2014)	BU-4DFE (Zhang et al. 2013b)	Penn Action (Zhang et al. 2013a)	Human3.6M (Ionescu et al. 2014)
Data content	Facial expression	Facial expression	Human pose	Human pose
Video number	876	606	2326	21,840
Subject number	86	101	–	7
Motion category number	6	6	8	15
Used in training	✓	✓	✓	✓
Used in evaluation	✓	–	✓	✓

Note that no subject is classified for each video in Penn Action. Thus we assume that each video has a different subject. BU-4DFE is employed only for training

according to the subjects. Two groups are used for training G_M and G_R respectively, and the results are evaluated on the last one. To boost the performance of our networks, we also employ the *BU-4DFE Dataset* (Zhang et al. 2013b) for training. This dataset contains 606 facial expression sequences captured from 101 subjects of various ethnic backgrounds. It includes six universal expressions, i.e., angry, disgust, fear, happiness, sadness, and surprise. We evenly split the dataset into two parts for training G_M and G_R respectively. The 3D Basel Face Model (Paysan et al. 2009) serves as the 3D morphable model to fit the facial identities and expressions for the condition generator G_C. We use (Zhu et al. 2016, 2019) to compute the 3DMM parameters for each frame. Note that we train G_R to refine the video clips every 32 frames.

We evaluate our method on human pose forecasting with the following two dataset. The *Human3.6M Dataset* (Ionescu et al. 2014) consists of 3.6 million 3D human poses with 32 joints and corresponding images taken from 11 professional actors in 15 actions. The provided 2D pose projections are used for evaluation. We follow the same evaluation protocol in Villegas et al. (2017b). For training, we use subjects S1, S5, S6, S7 and S8, and test on subjects S9 and S11. We crop video frames based on temporal tubes to remove as much background as possible while we ensure that the human actions are in all frames, and then we scale each cropped frame to 128×128. We evenly split the training set into two parts according to the subjects. G_M and G_R are trained with them respectively. We employ the same strategy as Villegas et al. (2017b) to train the LSTM pose generator. It is trained to observe 10 inputs and predict 64 steps, and tested on predicting 128 steps.

The *Penn Action Dataset* (Zhang et al. 2013a) consists of 2326 video sequences of 15 different human actions. For each action sequence in the dataset, 13 human joint annotations are provided as the ground truth. To remove very noisy joint ground-truth in the dataset, we follow the setting of Villegas et al. (2017b) to sub-sample the actions. Therefore, 8 actions including baseball pitch, baseball swing, clean and jerk, golf swing, jumping jacks, jump rope, tennis forehand, and tennis serve are used for training our networks. We evenly split the standard dataset into three sets. G_M and G_R are trained in the first two sets respectively, while we evaluate our models in the last set. The LSTM pose generator is trained to observe 10 inputs and predict 32 steps. Note that for both datasets containing human body poses, G_R is trained to refine the video clips with the length of 16.

7.2 Evaluation on Facial Expression Retargeting

We compare our method to MCNet (Villegas et al. 2017a), MoCoGAN (Tulyakov et al. 2018) and Villegas et al. (2017b) on the MUG Database. For each facial expression, we randomly select one video as the reference, and retarget it to all the subjects in the testing set with different methods. Each method only observes the input frame of the target subject, and performs the generation based on it. Our method and Villegas et al. (2017b) share the same 3DMM-based condition generator as introduced in Sect. 4.

Quantitative Comparison. The quality of a generated video are measured by the Average Content Distance (ACD) as introduced in Tulyakov et al. (2018). For each generated video, we make use of OpenFace (Amos et al. 2016), which outperforms human performance in the face recognition task, to measure the video quality. OpenFace produces a feature vector for each frame, and then the ACD is calculated by measuring the L-2 distance of these vectors. We introduce two variants of the ACD in this experiment. The ACD-I is the average distance between each generated frame and the original input frame. It aims to judge if the facial identity is well-preserved in the generated video. The ACD-C is the average pairwise distance of the per-frame feature vectors in the generated video. It measures the content consistency of the generated video.

Table 4 summarizes the comparison results. From the table, we find that our method achieves ACD scores both lower than 0.2, which is substantially better than the baselines. One interesting observation is that Villegas et al.

Table 4 Video generation quality comparison on the MUG Dataset (Aifanti et al. 2010)

Methods	ACD-I	ACD-C
MCNet (Villegas et al. 2017a)	0.545	0.322
Villegas et al. (2017b)	0.683	0.130
MoCoGAN (Tulyakov et al. 2018)	0.291	0.205
Ours	**0.184**	**0.107**
Reference	0.109	0.098

Bold values indicate the best performance
We also compute the ACD scores for the training set, which is the reference

Table 5 Average user preference score (the average number of times, a user prefers our result to the competing one) on the MUG Dataset (Aifanti et al. 2010)

Methods	Preference (%)
Ours/MCNet (Villegas et al. 2017a)	**84.2**/15.8
Ours/Villegas et al. (2017b)	**74.6**/25.4
Ours/MoCoGAN (Tulyakov et al. 2018)	**62.5**/37.5

Bold values indicate the best performance
Our results own higher preference scores compared with the others

(2017b) has the worst ACD-I but its ACD-C is the second best. We argue that this is due to the high-level information offered by our 3DMM-based condition generator, which plays a vital role for producing content consistency results. Our method outperforms other state-of-the-art approaches, since we utilize both domain knowledge (i.e., 3DMM) and temporal signals for video generation. We show that it is greatly beneficial to incorporate both factors into the generative process.

We also conduct a user study to quantitatively compare these methods. For each method, we randomly select 10 videos for each expression. We then randomly pair the videos generated by ours with the videos from one of the competing methods to form 54 questions. For each question, 3 users are asked to select the video which is more realistic. To be fair, the videos from different methods are shown in random orders. We report the average user preference scores (the average number of times, a user prefers our result to the competing one) in Table 5. We find that the users consider the videos generated by ours more realistic most of the time. This is consistent with the ACD results in Table 4, in which our method substantially outperforms the baselines.

Visual Results. In Fig. 8, we show the visual results (the expressions of happiness, surprise and disgust) generated by our method. We observe that our model is able to generate realistic motions while the facial identities are well-preserved. We hypothesize that the domain knowledge (i.e., 3DMM) employed serves as a good prior which improves the generation.

7.3 Evaluation on Human Pose Forecasting

Following the settings of Villegas et al. (2017b), we engage the feature similarity loss term ℓ_{feat} for our motion forecasting network G_M to capture the appearance (C_1) and structure (C_2) of the human action. This loss term is added to Eq. (9), which is defined as follows:

$$\ell_{feat} = \sum_{i=1}^{N} \|C_i(I_{t+k}) - C_i(\tilde{I}_{t+k})\|_2^2, \tag{21}$$

where we use the last convolutional layer of the VGG16 Network (Simonyan and Zisserman 2015) as C_1, and the last layer of the Hourglass Network (Newell et al. 2016) as C_2. Note that we compute the bounding box according to the ground truth to crop the human of interest for each frame, and then scale it to 224×224 as the input of the VGG16 Network.

Results on Human3.6M Dataset. On Human3.6M Dataset, we aim to compare our method to the state of the art in terms of short-term and long-term future prediction. We use Peak Signal-to-Noise Ratio (PSNR) for motion-based pixel-level quantitative evaluation. To evaluate the performance of short-term prediction, we measure the next-frame prediction performance with PSNR (PSNR-Next) after inputting the driven frames for each approach. On the other hand, the average PSNR of 120 generated frames (PSNR-120) is computed for measuring the performance of long-term video generation. We compare our model with four LSTM-based approaches (Denton and Birodkar 2017; Lotter et al. 2017; Villegas et al. 2017b; Shi et al. 2015), one multi-scale method (Mathieu et al. 2016) and one using 3D convolutions (Li et al. 2018). Here we implement a modified network architecture of Li et al. (2018) which predicts the future in next 32 timesteps given the starting frame. To get long-term future predictions, we train their model and iteratively treat the last predicted frame as the input to get the next 32-timestep prediction based on the previous output. Given the input frames, all other methods are trained to predict future frames recursively, one by one, according to their default settings.

Results are shown in Table 6. First, we can find that our approach matches the state-of-the-art performance in terms of shot-term future prediction. Specifically, 3D convolution-based models (i.e., Li et al. 2018 and ours) achieve the top performance, as 3D convolutions are able to produce more temporally coherent motions than other LSTM-based approaches. On the aspect of long-term prediction, the pro-

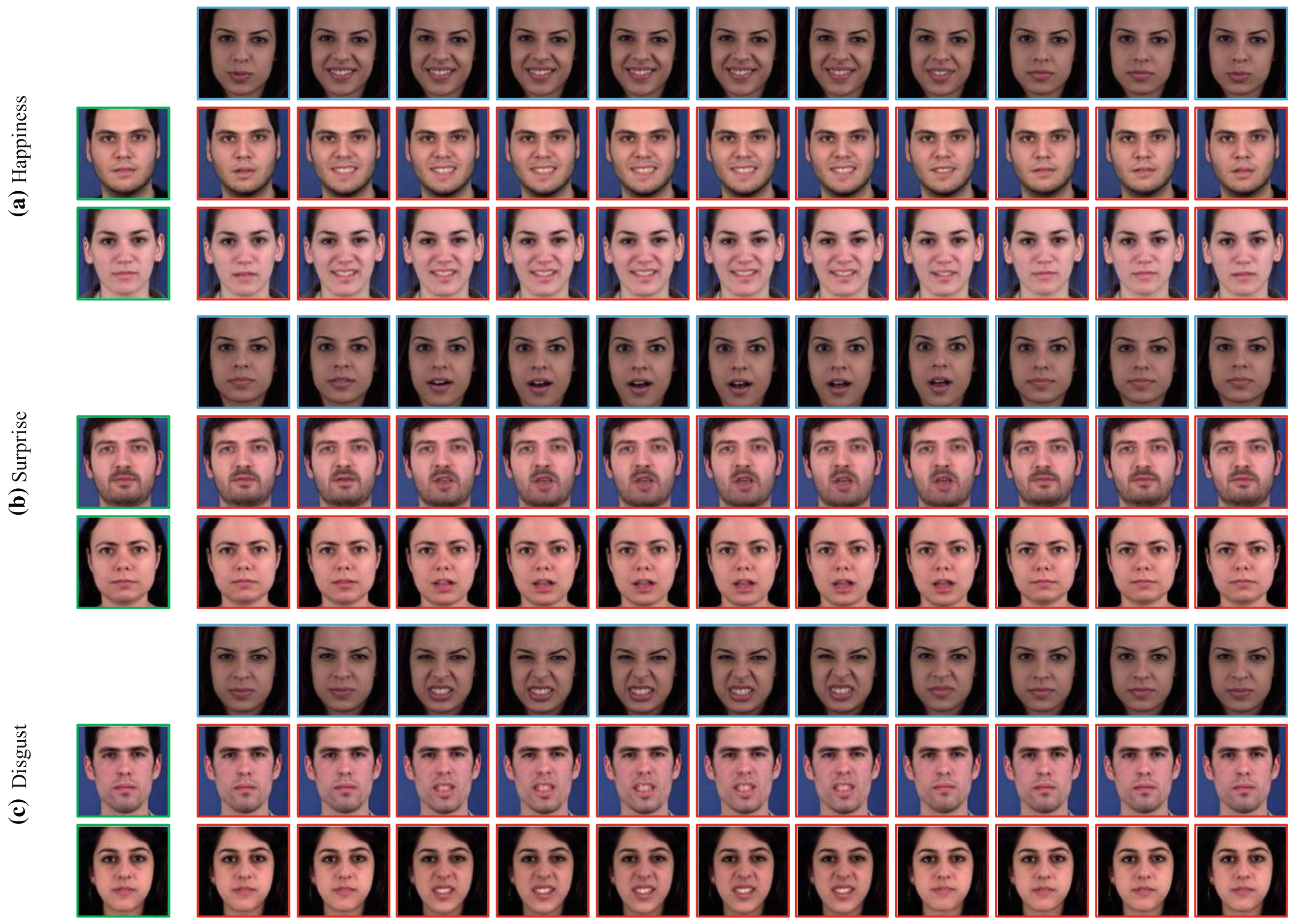

Fig. 8 Examples of facial expression retargeting using our algorithm on the MUG Database (Aifanti et al. 2010). We show three expressions as an illustration: **a** happiness, **b** surprise, and **c** disgust. The reference video and the input target images are highlighted in *blue* and *green*, respectively, while the generated frames are highlighted in *red*. The results are sampled every 8 frames. Best viewed in color and zoom in for details (Color figure online)

Table 6 Comparison of state-of-the-art methods on the Human3.6M Dataset (Ionescu et al. 2014). A larger PSNR score means better performance

Methods	# of Input Frames	PSNR-Next	PSNR-120
Conv LSTM (Shi et al. 2015)	10	28.5	18.4
Mathieu et al. (2016)	4	26.7	16.6
PredNet (Lotter et al. 2017)	1	38.9	17.8
Villegas et al. (2017b)	10	40.1	19.2
DRNet (Denton and Birodkar 2017)	10	41.1	18.1
Li et al. (2018)	1	**42.8**	16.4
Ours	10	**42.8**	**22.6**

Bold values indicate the best performance

posed model substantially outperforms other state-of-the-art methods. We can see that LSTM-based methods which encode structure conditions, including Conv LSTM (Shi et al. 2015; Villegas et al. 2017b), DRNet (Denton and Birodkar 2017) and ours, obtain better results. Note that Li et al. (2018) succeeded in shot-term prediction but can not achieve competitive results when applied to long-term motion generation. This is due to the absence of structural information during the their generative process. In contrast, our approach combines the benefits of structural prediction and 3D convolutions within a two-stage framework and thus obtains the best performance in both tasks.

To further evaluate the performance of long-term video generation, we compare our method against convolutional

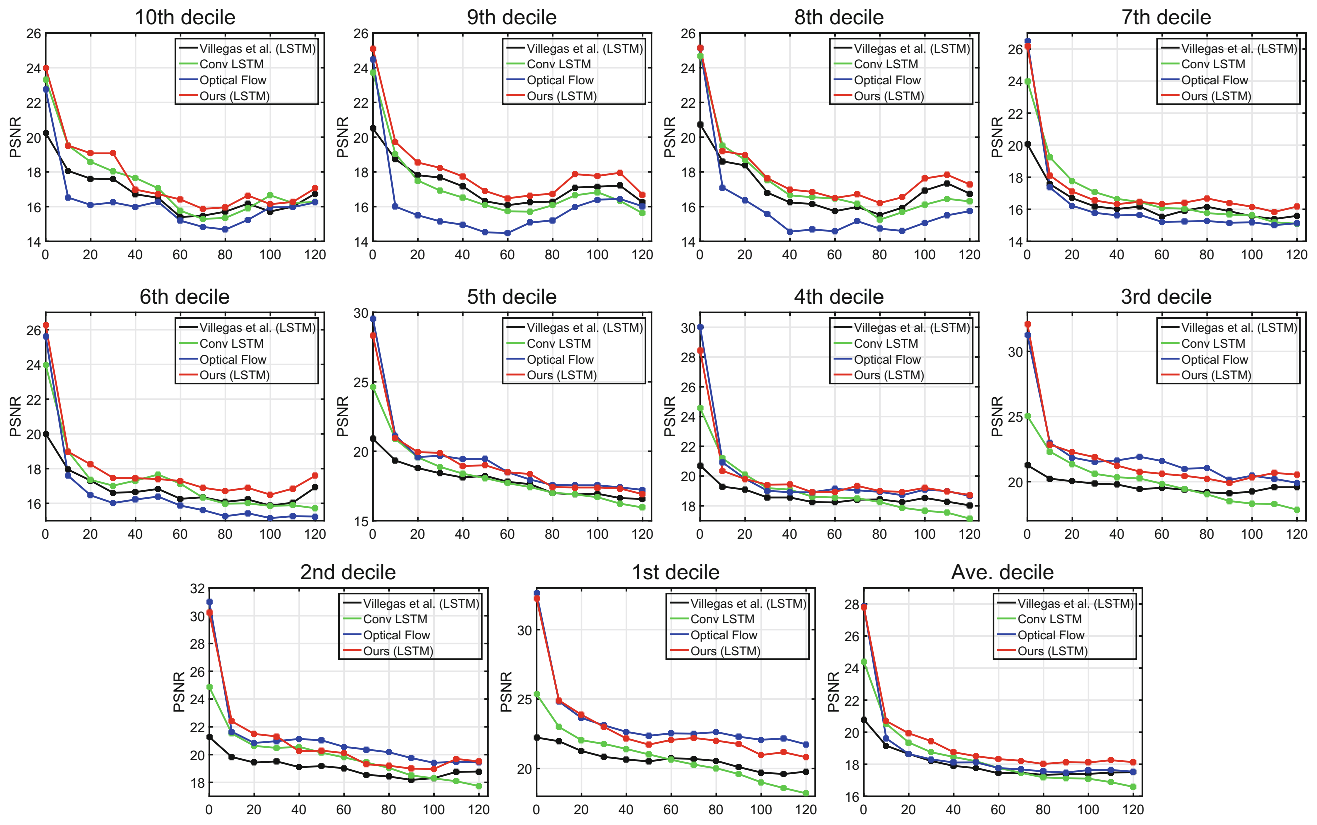

Fig. 9 Comparison of state-of-the-art algorithms using Peak Signal-to-Noise Ratio (PSNR) on different motion deciles from Human3.6M Dataset (Ionescu et al. 2014). Note that the results of ours and (Villegas et al. 2017b) use the condition motion maps computed from LSTM

LSTM (Conv LSTM) (Shi et al. 2015), Villegas et al. (2017b), and one additional baseline based on optical flow (Farnebäck 2003). Following the same setting of Villegas et al. (2017b), we use PSNR for quantitative evaluation as well. Moreover, we evaluate the predictions by deciles of motion similar to Villegas et al. (2017a). To be specific, the test videos are separated into deciles based on the computed average L-2 norms, and then we measure the image quality on each decile. The 1st decile contains videos with the least overall of motion (i.e., frames that show the smallest motion changes over time), and the 10th decile contains videos with the most overall motion (i.e., frames that show the largest motion changes over time). All the results of these baseline methods are collected from Villegas et al. (2017b). Note that for both Villegas et al. (2017b) and our approach, we report the predictions made from the pre-trained LSTM pose generator implemented according to Villegas et al. (2017b).

Results are reported in Fig. 9. We find that our structure-aware approach obtains the best PSNR performance among all the methods in the average motion deciles. In specific, our method and Villegas et al. (2017b) tend to achieve PSNR performance that is better than optical flow and Conv LSTM in the motion deciles with larger changes (e.g., from the 10th decile to the 7th decile). This is due to the fact that both our method and Villegas et al. (2017b) make predictions based on structure conditions. However, optical flow and Conv LSTM outperform Villegas et al. (2017b) in those deciles with smaller changes (e.g., the 2nd decile and the 1st decile). But our method is still comparable to them, since our residual formulation is more robust to deal with unchanged backgrounds. Note that there is higher uncertainty of the actions being performed in the Human3.6M Dataset compared to Penn Action Dataset. As a result, when training the LSTM which is leveraged to generate motion conditions, even plausible future predictions can still deviate significantly from the ground-truth future trajectory, which can penalize PSNRs during evaluation.

Results on Penn Action Dataset. On Penn Action Dataset, we compare our work with VGAN (Vondrick et al. 2016), Mathieu et al. (2016) and Villegas et al. (2017b). We produce the results of their models according to their papers or reference codes. For fair comparison, we generate videos with 32 generated frames using each method, and evaluate them starting from the first frame. Note that we train an individual VGAN for different action categories with randomly picked

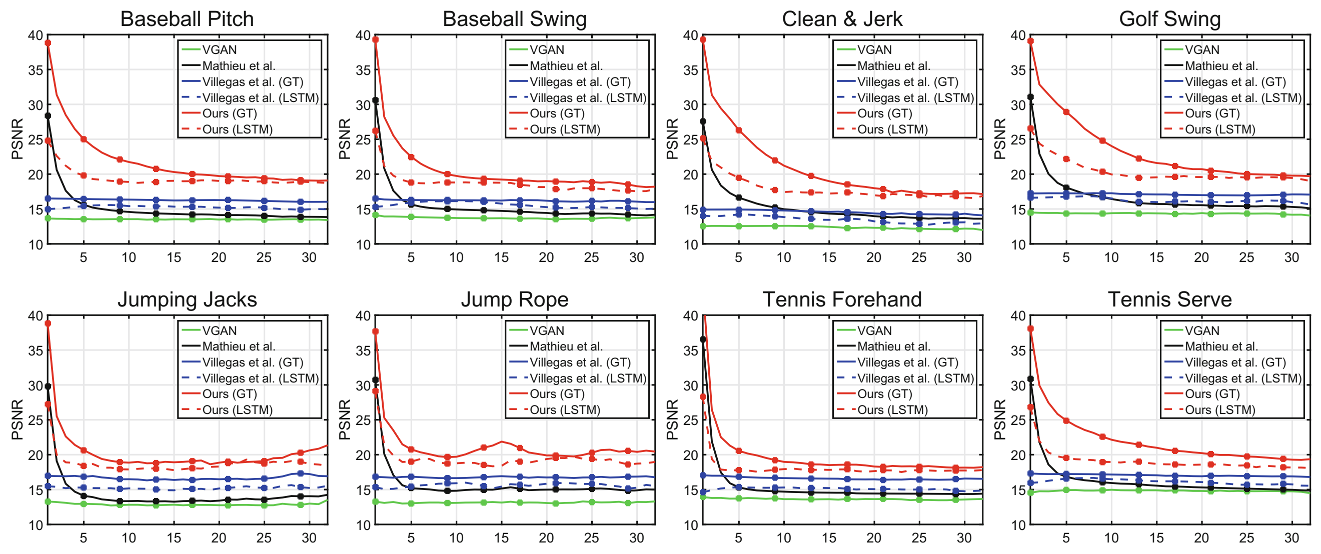

Fig. 10 Comparison of state-of-the-art methods using Peak Signal-to-Noise Ratio (PSNR) on different action categories from the Penn Action Dataset (Zhang et al. 2013a). We show the results of ours and Villegas et al. (2017b) using the condition motion maps computed from the ground truth (GT) or LSTM

Table 7 Comparison of state-of-the-art algorithms on the Penn Action Database (Zhang et al. 2013a)

Methods	MSE	MSE (GT)
VGAN (Vondrick et al. 2016)	0.047	–
Mathieu et al. (2016)	0.041	–
Villegas et al. (2017b)	0.030	0.025
Ours	**0.023**	**0.011**

A smaller MSE score means better performance

video clips from the dataset, while one network among all categories are trained for every other method. Both Villegas et al. (2017b) and ours perform the generation based on the pre-trained LSTM provided by Villegas et al. (2017b), and we train Villegas et al. (2017b) through the same way of our motion forecasting network G_M.

We evaluate the predictions using Peak Signal-to-Noise Ratio (PSNR) and Mean Square Error (MSE). Both metrics perform pixel-level analysis between the ground truth frames and the generated videos. We also report the results of our method and Villegas et al. (2017b) using the condition motion maps computed from the ground truth joints (GT). The results are shown in Fig. 10 and Table 7 respectively. From these two scores, we discover that the proposed method achieves better quantitative results which demonstrates the effectiveness of our algorithm.

Visual comparison of our method with Villegas et al. (2017b) are shown in Fig. 11. We can find that the predicted future of our method is closer to the ground-truth future. To be specific, our model yields more consistent motions and keeps human appearances as well. By contrast, artifacts can be observed from the results of Villegas et al. (2017b) especially when there are large motion changes.

7.4 Ablation Study

Evaluation of Main Modules. Our method consists of three main modules: residual learning, dense connections for the decoder and the two-stage generation schema. Without residual learning, our network decays to Villegas et al. (2017b). As we can see in Sects. 7.2 and 7.3, our framework outperforms Villegas et al. (2017b) which demonstrates the effectiveness of residual learning. To verify the rest modules, we train one partial variant of G_M, where the dense connections are not employed in the decoder f_D. Then we evaluate three different settings of our method on both tasks: G_M without dense connections, using only G_M for generation and our full model. Note that in order to get rid of the influence from the LSTM, we report the results using the conditional motion maps calculated from the ground truth on the Penn Action Dataset. Results are shown in Table 8. Our approach with more modules performs better than those with less components, which suggests the effectiveness of each part of our algorithm.

Evaluation of Motion Refining Networks. Notice that the design of our motion refining networks G_R is general and does not rely on the implementation of the motion forecasting networks G_M. This implies that the proposed G_R can be easily extended to refine the output of other methods. Therefore, we turn to conduct experiments to show how G_R can be applied to improve the performance of differ-

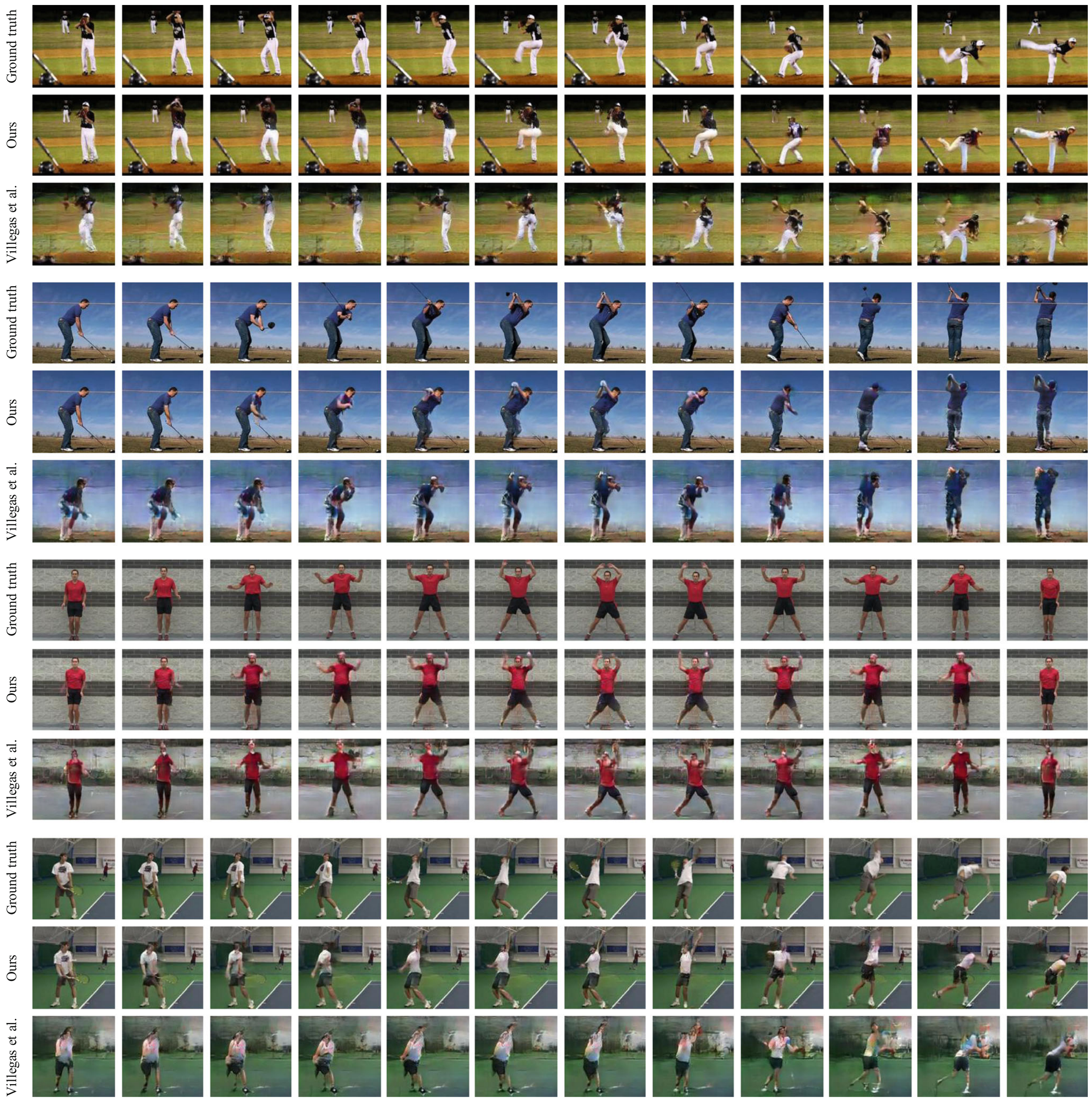

Fig. 11 Visual comparison of our method with Villegas et al. (2017b) on the Penn Action Dataset (Zhang et al. 2013a). From top to bottom, examples are from the actions of baseball, golf, jumping and tennis, respectively. The results are taken every 5 frames. Best viewed in color and zoom in for details

ent state-of-the-art approaches. On the MUG Database, we choose MCNet (Villegas et al. 2017a), Villegas et al. (2017b) and MoCoGAN (Tulyakov et al. 2018) as the baseline methods and compute the performance improvements measured by ACD-C. On the Human3.6M Dataset, the performance improvements measured by PSNR-120 of three baselines, PredNet (Lotter et al. 2017), Villegas et al. (2017b) and DRNet (Denton and Birodkar 2017), are reported. We train an individual G_R for each method using the same data splits as described in Sect. 7.1. The output frames of each method are concatenated with the corresponding motion maps produced by the motion condition generators and then fed into G_R. The results in Fig. 12 show that all baseline methods on both image-to-video translation tasks are improved by the proposed G_R with a large margin, which demonstrates that

Table 8 Quantitative results of ablation study

Methods	ACD-I	ACD-C	MSE
G_M	0.459	0.155	0.027
G_M (Dense)	0.252	0.140	0.014
G_M (Dense), G_R	**0.184**	**0.107**	**0.011**

Bold values indicate the best performance
We report the ACD scores on the MUG Database (Aifanti et al. 2010) and MSE scores on the Penn Action Dataset (Zhang et al. 2013a)

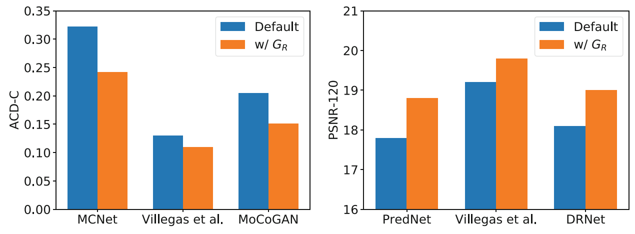

Fig. 12 Performance improvements of different state-of-the-art methods on the MUG Dataset (Left) and Human3.6M Dataset (Right) after their results are refined by the proposed motion refining networks G_R

our motion refining networks can be effectively generalized to different generative models.

8 Limitations

There are some notable limitations to our work. First of all, the success of our approach relies on incorporating high-level structure information, such as 3D face morphable models or human body landmarks, into the generative process. However, the major limitation of this structure-driven formulation is that it cannot be directly applied to general natural videos that do not contain human faces or body poses, especially, when object structures involved in the video are difficult to be predicted. This problem can be tackled by estimating a more general form of structural representations [e.g., optical flows (Li et al. 2018), motion vectors (Zhang et al. 2016) and semantic label maps (Pan et al. 2019)] using neural networks to encode the motion of foreground. Another promising direction is to capture high-level structures of objects in a latent space with unsupervised learning. However, this is more challenging than learning a explicit representation as described in Sect. 4, and we leave it as future work.

Another limitation of this work is that we assume a static background contained in the input, and hence our model cannot handle object motions occurring in the background. An example is illustrated in the first case of Fig. 11, where the baseball player in the background is blurred and even "disappeared" during the generation. However, we note that this is a highly challenging problem since background objects sometimes come in and out of sight. As a result, predicting background motion requires a generative model that is able to infer unseen objects or background movements from the input. One promising solution will be training an auxiliary generative model to predict the evolution of the background movements, and then combining it with the motions of the foreground object.

Finally, our model is limited by generating only a single future trajectory. As discussed in Villegas et al. (2017b), for an agent to make a better estimation of possible futures, we expect a generative model which is able to produce more than one generated future. To address this limitation, future work in this direction can leverage more powerful models as the motion condition generators, such as probabilistic sequence models or graph models (Tian et al. 2019; Yan et al. 2019; Zhao et al. 2019b).

9 Conclusions

In this paper, we combine the benefits of high-level structure conditions and spatiotemporal generative networks for image-to-video translation by synthesizing videos in a generation and then refinement manner. We have applied this method to facial expression retargeting where we show that our method is able to engage domain knowledge for realistic video generation, and to human pose forecasting where we demonstrate that our method achieves higher performance than state of the art when generating videos involving large motion changes. We also incorporate residual learning and dense connections to produce high-quality results. In the future, we plan to further explore the use of our framework for other image or video generation tasks.

Acknowledgements This work is partly supported by NSF 1763523, 1747778, 1733843 and 1703883 Awards. This work was also funded in part by grant BAAAFOSR-2013-0001 to Dimitris N. Metaxas. Mubbasir Kapadia has been funded in part by NSF IIS-1703883, NSF S&AS-1723869, and DARPA SocialSim-W911NF-17-C-0098.

References

Aifanti, N., Papachristou, C., & Delopoulos, A. (2010). The MUG facial expression database. In *International workshop on image analysis for multimedia interactive services (WIAMIS)*.

Amos, B., Ludwiczuk, B., & Satyanarayanan, M. (2016). OpenFace: A general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, CMU School of Computer Science.

Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning (ICML)*.

Blanz, V., & Vetter, T. (2003). Face recognition based on fitting a 3D morphable model. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *25*(9), 1063–1074.

Cao, C., Weng, Y., Zhou, S., Tong, Y., & Zhou, K. (2014). Facewarehouse: A 3D facial expression database for visual computing. *IEEE*

Transactions on Visualization and Computer Graphics (TVCG), *20*(3), 413–425.

Chao, Y. W., Yang, J., Price, B., Cohen, S., & Deng, J. (2017). Forecasting human dynamics from static images. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Denton, E., & Birodkar, V. (2017). Unsupervised learning of disentangled representations from video. In *Annual conference on neural information processing systems (NeurIPS)* (pp. 4414–4423).

Denton, E., Chintala, S., Szlam, A., & Fergus, R. (2015). Deep generative image models using a Laplacian pyramid of adversarial networks. In *Annual conference on neural information processing systems (NeurIPS)*.

Farnebäck, G. (2003). Two-frame motion estimation based on polynomial expansion. In *Scandinavian conference on Image analysis* (pp. 363–370).

Finn, C., Goodfellow, I., & Levine, S. (2016). Unsupervised Learning for Physical Interaction through Video Prediction. In *Annual conference on neural information processing systems (NeurIPS)*.

Fragkiadaki, K., Levine, S., Felsen, P., & Malik, J. (2015). Recurrent network models for human dynamics. In *IEEE international conference on computer vision (ICCV)* (pp. 4346–4354).

Gatys, L., Ecker, A. S., & Bethge, M. (2015). Texture synthesis using convolutional neural networks. In *Annual conference on neural information processing systems (NeurIPS)* (pp. 262–270).

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al. (2014). Generative adversarial nets. In *Annual conference on neural information processing systems (NeurIPS)* (pp. 2672–2680).

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). Improved training of Wasserstein GANs. In *Annual conference on neural information processing systems (NeurIPS)*.

Huang, G., Liu, S., van der Maaten, L., & Weinberger, K. Q. (2018). Condensenet: An efficient densenet using learned group convolutions. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Ionescu, C., Papava, D., Olaru, V., & Sminchisescu, C. (2014). Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *36*(7), 1325–1339.

Ji, S., Xu, W., Yang, M., & Yu, K. (2013). 3D convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *35*(1), 221–231.

Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2018). Progressive growing of GANs for improved quality, stability, and variation. In *International conference on learning representations (ICLR)*.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. In *International conference on learning representations (ICLR)*.

Kingma, D. P., & Welling, M. (2014). Auto-encoding variational bayes. In *International conference on learning representations (ICLR)*.

Laine, S., Karras, T., Aila, T., Herva, A., Saito, S., Yu, R., Li, H., & Lehtinen, J. (2017). Production-level facial performance capture using deep convolutional neural networks. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*.

Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., & Yang, M. H. (2018). Flow-grounded spatial-temporal video prediction from still images. In *European conference on computer vision (ECCV)*.

Liang, X., Lee, L., Dai, W., & Xing, E. P. (2017). Dual motion GAN for future-flow embedded video prediction. In *IEEE international conference on computer vision (ICCV)*.

Liang, X., Zhang, H., Lin, L., & Xing, E. (2018). Generative semantic manipulation with mask-contrasting GAN. In *European conference on computer vision (ECCV)* (pp. 558–573).

Liu, Z., Yeh, R. A., Tang, X., Liu, Y., & Agarwala, A. (2017). Video frame synthesis using deep voxel flow. In *IEEE international conference on computer vision (ICCV)*.

Lotter, W., Kreiman, G., & Cox, D. (2017). Deep predictive coding networks for video prediction and unsupervised learning. In *International conference on learning representations (ICLR)*.

Lu, J., Issaranon, T., & Forsyth, D. (2017). SafetyNet: Detecting and rejecting adversarial examples robustly. In *IEEE international conference on computer vision (ICCV)*.

Ma, L., Jia, X., Sun, Q., Schiele, B., Tuytelaars, T., & Van Gool, L. (2017). Pose guided person image generation. In *Annual conference on neural information processing systems (NeurIPS)* (pp. 405–415).

Mathieu, M., Couprie, C., & LeCun, Y. (2016). Deep multi-scale video prediction beyond mean square error. In *International conference on learning representations (ICLR)*.

Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. arXiv:1411.1784

Newell, A., Yang, K., & Deng, J. (2016). Stacked hourglass networks for human pose estimation. In *European conference on computer vision (ECCV)* (pp. 483–499).

Odena, A., Olah, C., & Shlens, J. (2017). Conditional image synthesis with auxiliary classifier GANs. In *International conference on machine learning (ICML)*.

Olszewski, K., Li, Z., Yang, C., Zhou, Y., Yu, R., Huang, Z., Xiang, S., Saito, S., Kohli, P., & Li, H. (2017). Realistic dynamic facial textures from a single image using GANs. In *IEEE international conference on computer vision (ICCV)*.

Pan, J., Wang, C., Jia, X., Shao, J., Sheng, L., Yan, J., & Wang, X. (2019). Video generation from single semantic label map. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 3733–3742).

Paysan, P., Knothe, R., Amberg, B., Romdhani, S., & Vetter, T. (2009). A 3D face model for pose and illumination invariant face recognition. In *IEEE international conference on advanced video and signal based surveillance (AVSS) for security, safety and monitoring in smart environments*.

Peng, X., Feris, R. S., Wang, X., & Metaxas, D. N. (2016). A recurrent encoder-decoder network for sequential face alignment. In *European conference on computer vision (ECCV)* (pp. 38–56).

Peng, X., Huang, J., Hu, Q., Zhang, S., Elgammal, A., & Metaxas, D. (2015). From circle to 3-Sphere: Head pose estimation by instance parameterization. *Computer Vision and Image Understanding (CVIU)*, *136*, 92–102.

Peng, X., Tang, Z., Yang, F., Feris, R. S., & Metaxas, D. (2018). Jointly optimize data augmentation and network training: Adversarial data augmentation in human pose estimation. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 2226–2234).

Perarnau, G., van de Weijer, J., Raducanu, B., & Álvarez, J. M. (2016). Invertible conditional GANs for image editing. In *NeurIPS workshop on adversarial training*.

Pumarola, A., Agudo, A., Martinez, A., Sanfeliu, A., & Moreno-Noguer, F. (2018). GANimation: Anatomically-aware facial animation from a single image. In *European conference on computer vision (ECCV)*.

Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., & Lee, H. (2016). Generative adversarial text-to-image synthesis. In *International conference on machine learning (ICML)*.

Reed, S. E., Zhang, Y., Zhang, Y., & Lee, H. (2015). Deep visual analogy-making. In *Annual conference on neural information processing systems (NeurIPS)*.

Rezende, D. J., Mohamed, S., & Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning (ICML)*.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention (MICCAI)*.

Saito, M., Matsumoto, E., & Saito, S. (2017). Temporal generative adversarial nets with singular value clipping. In *IEEE international conference on computer vision (ICCV)*.

Shen, W., & Liu, R. (2017). Learning residual images for face attribute manipulation. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W. K., & Woo, W.-c. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Annual conference on neural information processing systems (NeurIPS)* (pp. 802–810).

Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., & Webb, R. (2017). Learning from simulated and unsupervised images through adversarial training. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Simonyan, K., & Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. In *Annual conference on neural information processing systems (NeurIPS)* (pp. 568–576).

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *International conference on learning representations (ICLR)*.

Srivastava, N., Mansimov, E., & Salakhutdinov, R. (2015). Unsupervised learning of video representations using LSTMs. In *International conference on machine learning (ICML)*.

Tang, Z., Peng, X., Geng, S., Wu, L., Zhang, S., & Metaxas, D. N. (2018a). Quantized densely connected U-nets for efficient landmark localization. In *European conference on computer vision (ECCV)*.

Tang, Z., Peng, X., Geng, S., Zhu, Y., & Metaxas, D. (2018b). CU-Net: Coupled U-nets. In *British machine vision conference (BMVC)*.

Thies, J., Zollhöfer, M., Stamminger, M., Theobalt, C., & Nießner, M. (2016). Face2Face: Real-time face capture and reenactment of RGB videos. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Tian, Y., Peng, X., Zhao, L., Zhang, S., & Metaxas, D. N. (2018). CR-GAN: Learning complete representations for multi-view generation. In *International joint conference on artificial intelligence (IJCAI)* (pp. 942–948).

Tian, Y., Zhao, L., Peng, X., & Metaxas, D. N. (2019). Rethinking kernel methods for node representation learning on graphs. In *Annual conference on neural information processing systems (NeurIPS)*.

Tran, D., Bourdev, L., Fergus, R., Torresani, L., & Paluri, M. (2015). Learning spatiotemporal features with 3D convolutional networks. In *IEEE international conference on computer vision (ICCV)* (pp. 4489–4497).

Tulyakov, S., Liu, M. Y., Yang, X., & Kautz, J. (2018). MoCoGAN: Decomposing motion and content for video generation. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

van den Oord, A., Kalchbrenner, N., & Kavukcuoglu, K. (2016). Pixel recurrent neural networks. In *International conference on machine learning (ICML)*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. In *Annual conference on neural information processing systems (NeurIPS)*.

Villegas, R., Yang, J., Hong, S., Lin, X., & Lee, H. (2017a). Decomposing motion and content for natural video sequence prediction. In *International conference on learning representations (ICLR)*.

Villegas, R., Yang, J., Zou, Y., Sohn, S., Lin, X., & Lee, H. (2017b). Learning to generate long-term future via hierarchical prediction. In *International conference on machine learning (ICML)*.

Vondrick, C., Pirsiavash, H., & Torralba, A. (2016). Generating videos with scene dynamics. In *Annual conference on neural information processing systems (NeurIPS)*.

Wang, T. C., Liu, M. Y., Zhu, J. Y., Liu, G., Tao, A., Kautz, J., & Catanzaro, B. (2018). Video-to-video synthesis. In *Annual conference on neural information processing systems (NeurIPS)* (pp. 1144–1156).

Xiong, W., Luo, W., Ma, L., Liu, W., & Luo, J. (2018). Learning to generate time-lapse videos using multi-stage dynamic generative adversarial networks. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Yan, S., Li, Z., Xiong, Y., Yan, H., & Lin, D. (2019). Convolutional sequence generation for skeleton-based action synthesis. In *IEEE international conference on computer vision (ICCV)*.

Yang, C., Wang, Z., Zhu, X., Huang, C., Shi, J., & Lin, D. (2018). Pose guided human video generation. In *European conference on computer vision (ECCV)*.

Zhang, B., Wang, L., Wang, Z., Qiao, Y., & Wang, H. (2016). Real-time action recognition with enhanced motion vector CNNs. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Zhang, H., Sindagi, V., & Patel, V. M. (2017a). Image de-raining using a conditional generative adversarial network. arXiv:1701.05957.

Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., & Metaxas, D. (2017b). StackGAN++: Realistic image synthesis with stacked generative adversarial networks. arXiv:1710.10916.

Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., & Metaxas, D. (2017c). StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *IEEE international conference on computer vision (ICCV)*.

Zhang, W., Zhu, M., & Derpanis, K. (2013a). From actemes to action: A strongly-supervised representation for detailed action understanding. In *IEEE international conference on computer vision (ICCV)*.

Zhang, X., Yin, L., Cohn, J. F., Canavan, S., Reale, M., Horowitz, A., et al. (2013b). A high-resolution spontaneous 3D dynamic facial expression database. In *IEEE international conference and workshops on automatic face and gesture recognition (FG)* (pp. 1–6).

Zhang, Z., Xie, Y., & Yang, L. (2018a). Photographic text-to-image synthesis with a hierarchically-nested adversarial network. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Zhang, Z., Yang, L., & Zheng, Y. (2018b). Translating and segmenting multimodal medical volumes with cycle-and shape consistency generative adversarial network. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Zhao, L., Han, F., Peng, X., Zhang, X., Kapadia, M., Pavlovic, V., et al. (2019a). Cartoonish sketch-based face editing in videos using identity deformation transfer. *Computers & Graphics*, *79*, 58–68.

Zhao, L., Peng, X., Tian, Y., Kapadia, M., & Metaxas, D. (2018). Learning to forecast and refine residual motion for image-to-video generation. In *European conference on computer vision (ECCV)* (pp. 387–403).

Zhao, L., Peng, X., Tian, Y., Kapadia, M., & Metaxas, D. N. (2019b). Semantic graph convolutional networks for 3D human pose regression. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 3425–3435).

Zhu, X., Lei, Z., Liu, X., Shi, H., & Li, S. (2016). Face alignment across large poses: A 3D solution. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Zhu, X., Liu, X., Lei, Z., & Li, S. Z. (2019). Face alignment in full pose range: A 3D total solution. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *41*(1), 78–92.