DOI: 10.1111/cgf.13947
EUROGRAPHICS 2020 / U. Assarsson and D. Panozzo
(Guest Editors)
Volume 39 (2020), Number 2

Motion Retargetting based on Dilated Convolutions and Skeleton-specific Loss Functions

SangBin Kim, Inbum Park, Seongsu Kwon, and JungHyun Han

Department of Computer Science and Engineering, Korea University, Seoul, Korea

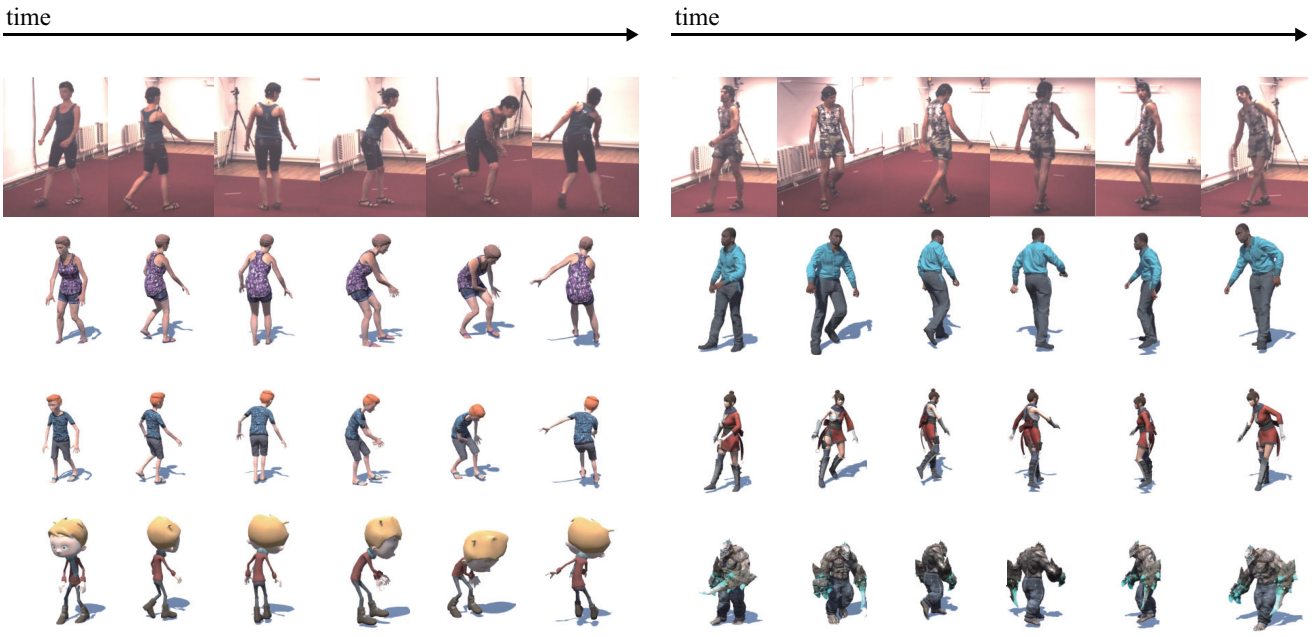

Figure 1: Our motion retargetting model retargets the source motion capture data (top row) to diverse characters with different bone lengths (the other rows). Not only the input motions but also the target characters are unseen during training.

Abstract

Motion retargetting refers to the process of adapting the motion of a source character to a target. This paper presents a motion retargetting model based on temporal dilated convolutions. In an unsupervised manner, the model generates realistic motions for various humanoid characters. The retargetted motions not only preserve the high-frequency detail of the input motions but also produce natural and stable trajectories despite the skeleton size differences between the source and target. Extensive experiments are made using a 3D character motion dataset and a motion capture dataset. Both qualitative and quantitative comparisons against prior methods demonstrate the effectiveness and robustness of our method.

CCS Concepts

• ***Computing methodologies*** → *Neural networks;*

1. Introduction

Motion retargetting is the process of adapting the motion of a *source* character to another called a *target*, whose skeleton size is different from the source's. The retargetted motion should not only look natural but also preserve the features of the source motion.

Deep learning has made a profound impact on numerous areas in science and engineering, but only very recently a few works for

solving motion retargetting in the deep learning framework were reported. The state of the art is the work of Villegas et al. [VYCL18] They proposed an architecture that combines recurrent neural networks (RNNs) with analytic *forward kinematics*, which computes skeleton's joint positions given the joint rotations. They showed promising results but also revealed limitations.

RNNs maintain a hidden state of the entire past. This prevents parallel computation and makes it hard to train RNNs properly [BSF94, PMB13, GAG*17]. In contrast, convolutional neural networks (CNNs) allow to precisely control the maximum length of dependencies to be modeled. In general, motion retargetting may not require a long-term dependency between the frames of a motion sequence, e.g., between the frames longer than four seconds. Therefore, CNNs are the more attractive choice for motion retargetting. In a *dilated* convolutional network [YK16], the filter can skip input frames with a certain step and therefore we can use parameters fewer than regular dense convolutions to model a short-term dependency in the motion sequence.

In addition, the loss functions used in the generative adversarial network (GAN) of Villegas et al. [VYCL18] are proven to be insufficient, i.e., a source motion is not always naturally retargetted to the skeletons of different sizes. Their model also has difficulties in processing a long sequence of motions because it outputs the position offsets of the skeleton's root (hip or pelvis), making errors accumulated.

This paper presents a motion retargetting model built upon temporal dilated convolutional networks, where the receptive fields are defined to be suitable for motion retargetting. We train our model with a skeleton-specific objective function in an unsupervised way. The key elements of our main contributions can be summarized as follows:

- An effective and efficient model based on temporal dilated convolutions, which are tailored to the intrinsic features of motion retargetting.
- A novel objective function designed to meet the basic requirement of motion retargetting, i.e., smooth retargetting to a character of different skeleton size.
- An unsupervised learning framework that works on the typical character animation data with little preprocessing required.
- A solution to the problem of retargetting a long (in principle, unlimitedly long) sequence of motions.

Using our model, we made extensive evaluations including comparisons with the baseline techniques such as the work of Villegas et al. [VYCL18]. Both qualitative and quantitative results prove that our model retargets quite naturally both a virtual character's motion and a real human motion to diverse characters[†].

This paper is organized as follows. Section 2 reviews the related studies. Section 3 presents our motion retargetting model, and Section 4 describes the adversarial learning and loss functions. Section 5 presents the experiment setup, and Section 6 reports the experiment results. Section 7 concludes the paper.

[†] Code is available at http://bit.ly/retargetting-tdcn

2. Related Work

With the success of deep learning, there has been a surge in models that can directly predict 3D poses from images [LC14, PHK16, TKS*16, ZSZ*16, RS16, PZDD17, MRC*17, SSLW17, VRM*17, TRA17]. Many approaches for modeling human motions tried to use temporal information since a model that infers a motion for each frame causes unstable and inconsistent predictions for a sequence. Tekin et al. [TRLF16] proposed a 3D pose regression directly from 3D HOG (Histograms of Oriented Gradients) features of a spatio-temporal volume on which a person is always centered using CNNs. Mehta et al. [MSS*17] devised a real-time system for 3D pose estimation, which utilizes CNNs trained with bone-length constraints and predicts smooth 3D poses with temporal filtering. Lin et al. [LLL*17] used LSTMs [HS97] to estimate 3D poses from a sequence of images. They performed a multi-stage refinement to exploit spatial and temporal constraints. Hossain et al. [HL18] also proposed sequence to sequence learning models using LSTMs to focus on predicting temporally consistent 3D poses by learning the temporal context of a sequence. Katircioglu et al. [KTS*18] used bidirectional LSTMs to improve temporal consistency of 3D poses decoded from the structural latent representation.

A prevalent way to model 3D human poses with a sequence is to use RNNs [FLFM15, JZSS16, MBR17, GSAH17]. Recently, Aksan et al. [AKH19] proposed a structured prediction layer that could be combined with various architectures while decomposing the body pose predictions into individual joints. On the other hand, there exist multiple cases of successfully modeling sequential networks without using RNNs [vdODZ*16, KES*16, VSP*17]. Butepage et al. [BBKK17] used a feed-forward network for the encoding-decoding framework and compared the results of different temporal encoder structures. Pavllo et al. [PFGA19] proposed a 3D pose estimation network that uses temporal dilated convolutions with excellent performances. Since these networks directly infer the *xyz*-coordinates, however, they are not suitable for motion retargetting which maps a source motion to a target character of different proportions while retaining important constraints such as the root joint trajectory. Further post-processing is also required to meet the bone-length constraints to integrate with character animation.

Gleicher [Gle98] used a spacetime constraints solver to compute motion retargetting while retaining the characteristics of the original motion. Lee and Shin [LS99] proposed a hierarchical approach where motion retargetting was decomposed into per-frame Inverse Kinematics (IK), followed by B-spline curve fitting for smooth results. Choi and Ko [CK00] developed an online retargetting algorithm based on the per-frame IK. Monzani et al. [MBBT00] proposed to use an intermediate simplified skeleton to perform motion retargetting. Tak and Ko [TK05] suggested a per-frame algorithm that filters input motion to obtain a physically plausible one. Villegas et al. [VYCL18] proposed a neural kinematic framework that performs a one-step feed-forward prediction by encoding and decoding temporal information using GRUs [Mai90], whereas the other methods require iterative optimization. Recently, Aberman et al. [AWL*19] used temporally structured representations for video motion retargetting between 2D skeleton poses.

Our motion retargetting framework is designed along the line of

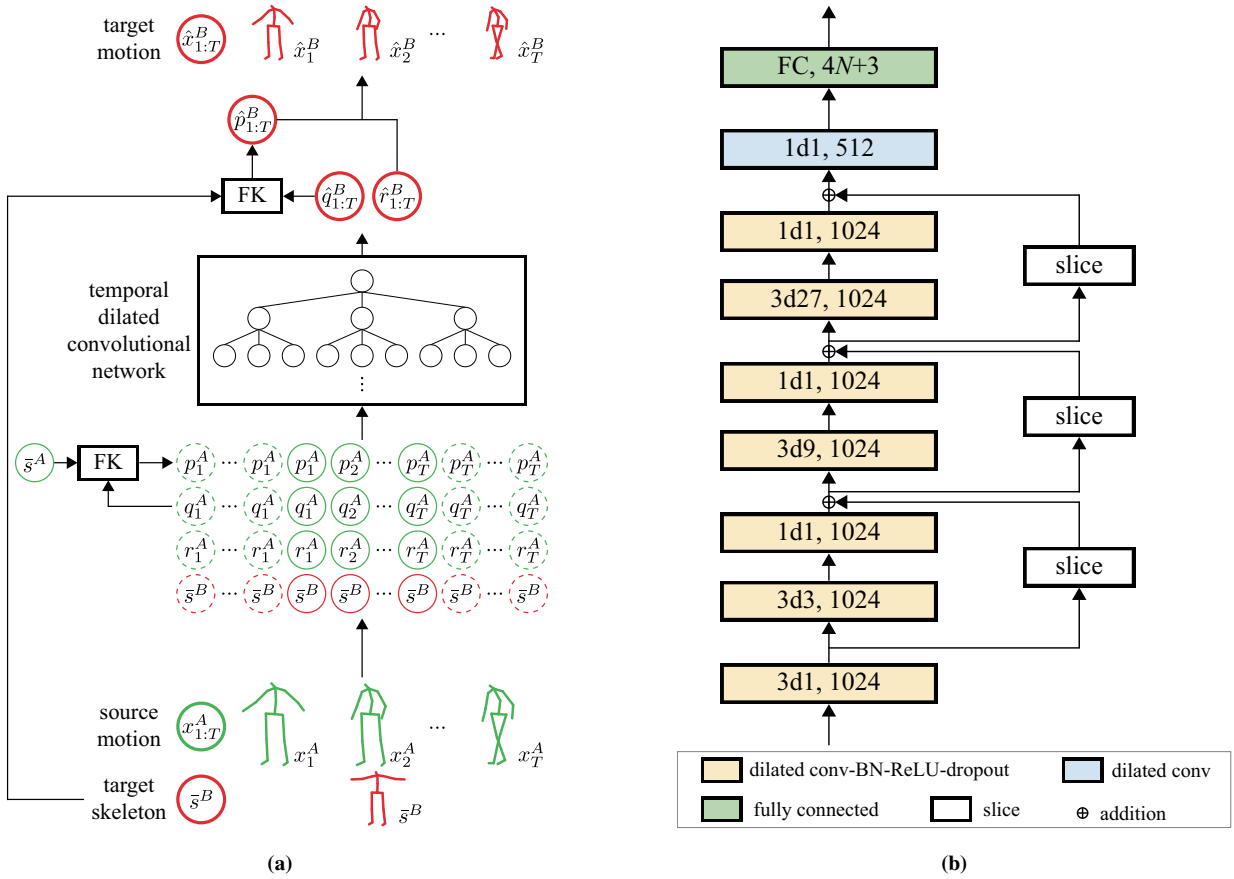

Figure 2: Motion retargetting model based on temporal dilated convolutional network.

Villegas et al. [VYCL18], but we replaced the RNNs with temporal dilated convolutions for stable training. It makes our model more stable for a larger sized mini-batch and augmented dataset during training time. Our model can be used in example-based methods such as motion style transfer or motion synthesis [BH00,GMHP04, HPP05]. We also introduce the PatchGAN [IZZE17] approach into our discriminator to preserve the high-frequency detail of the input motions.

3. Motion Retargetting Model

Section 3.1 describes our neural kinematic framework. The core of the framework is the temporal dilated convolutional network. Section 3.2 presents the network tailored for motion retargetting.

3.1. Neural Kinematic Framework

Figure 2a shows the overall architecture, where a motion sequence performed by a *source* skeleton is mapped to a *target*. We denote the source and target by A and B, respectively. Their skeletons have the same number of joints. The input consists of the source motion sequence, $x^A_{1:T}$ where T is the frame count, and the target skeleton $\bar{s}^B$ in the *default pose*.

For each frame t in $[1,T]$, x^A_t is a combination of q^A_t and r^A_t, where q^A_t ($\in \mathbb{R}^{4N}$ for N joints) represents the *unit quaternions* that describe the *joint rotations* of the source skeleton and r^A_t ($\in \mathbb{R}^3$) represents the root joint position. Taking q^A_t and $\bar{s}^A$ (the source skeleton in the default pose) as input, the forward kinematics (FK) module, f_{FK}, outputs p^A_t ($\in \mathbb{R}^{3N}$), which represents the joints' local coordinates with respect to the root:

$$p^A_t = f_{FK}(q^A_t, \bar{s}^A) \tag{1}$$

The input to the temporal dilated convolutional network (TDCN) is a sequence of frames, each of which contains p^A_t, q^A_t and r^A_t of the source and $\bar{s}^B$ of the target. In order to generate a retargetted motion for *every* input frame, the left end of the input sequence is padded with p^A_1, q^A_1 and r^A_1, and the right end with p^A_T, q^A_T and r^A_T. The padded sequence, denoted as $\tilde{x}^A_{1:T}$, and the target skeleton $\bar{s}^B$ are then fed to TDCN, f_{TDCN}. It outputs the target motion sequence, $\hat{x}^B_{1:T}$:

$$\hat{x}^B_{1:T} = f_{TDCN}(\tilde{x}^A_{1:T}, \bar{s}^B) \tag{2}$$

where each component $\hat{x}^B_t$ is a combination of the root joint's global position $\hat{r}^B_t$ ($\in \mathbb{R}^3$) and the unit quaternions, $\hat{q}^B_t$ ($\in \mathbb{R}^{4N}$). The FK module converts $\hat{q}^B_t$ to $\hat{p}^B_t$, which represents each joint's local coordinates with respect to the root:

$$\hat{p}^B_{1:T} = f_{FK}(\hat{q}^B_{1:T}, \bar{s}^B) \tag{3}$$

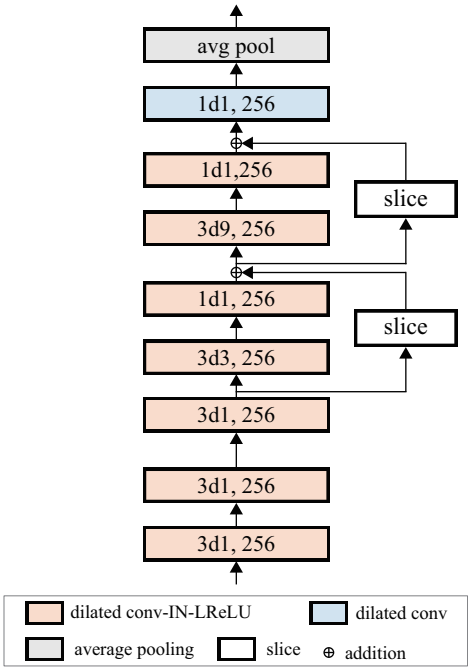

Figure 3: The discriminator's structure.

It is combined with $\hat{r}_t^B$ to determine the joint's global coordinates in the retargetted motion.

3.2. Temporal Dilated Convolutional Network

We have tailored the TDCN proposed by Pavllo et al. [PFGA19] to the need of motion retargetting. Figure 2b shows its structure. Our TDCN first applies a convolutional layer to the input frame of p_t^A, q_t^A, r_t^A and s^B. The first convolutional layer is denoted as "3d1, 1024." The first element implies that the filter size is three, i.e., three frames are convolved, and the *dilation factor* is one. The second element denotes 1024 output channels.

The convolution is followed by three ResNet-style blocks. Each block is surrounded by a skip-connection that slices the residuals symmetrically and adds them to subsequent features [HZRS16]. For the i-th block, the dilation factor for convolution is 3^i. For example, it is $3^2 = 9$ for the second block. In each block, such a convolution is followed by a linear projection denoted as 1d1.

Note that, in our TDCN, the *receptive field* is limited to 81 frames, i.e., the TDCN can see at a time a local sequence of 81 frames. Conceptually speaking, a window slides over $x_{1:T}^A$, and $\hat{x}_t^B$ is generated using the input motions that can be seen through the window.

Each of the seven layers up to this point (colored in yellow in Figure 2b) is followed by batch normalization [IS15], leaky rectified linear units [XWCL15], and dropout [SHK*14]. For the sake of simplicity, however, they are not depicted in Figure 2b.

The features produced by the last block are converted to high-level features by an additional convolutional layer (colored in cyan). Finally, a simple fully-connected layer maps the high-level features to $\hat{q}_t^B$ and $\hat{r}_t^B$.

4. Unsupervised Motion Retargetting

Our unsupervised motion retargetting adopts adversarial cycle consistency training [ZPIE17]. Section 4.1 presents adversarial learning for motion retargetting, and Section 4.2 presents cycle consistency training and loss functions.

4.1. Discriminator for Adversarial Learning

For adversarial learning, the network in Figure 2b works as the *generator*. It provides the retargetted motion sequence for the *discriminator*, which is also a temporal dilated convolutional network. See Figure 3. Its structure is similar to the generator's shown in Figure 2b, but the *receptive field* is limited to 31 frames.

In the framework proposed by Villegas et al. [VYCL18], the entire sequence of motions was input to the discriminator. In this way, however, the high-frequency detail of the local motions may not be properly generated. Advance to Figure 5a and see the characters in the last column. The target's limb joints do not correctly follow the source's.

In the context of generating realistic high-frequency images using GANs, *patch-based discriminators* [IZZE17, LW16, ZPIE17] have been proposed to address a similar problem. They classify the *local* image patches as either real or fake. By the same token, we provide a local sequence of the motions for the discriminator.

4.2. Cycle Training and Loss Functions

Let G and D denote the generator and discriminator, respectively. For cycle consistency training, G first retargets A's motion sequence, $x_{1:T}^A$, to B to produce $\hat{x}_{1:T}^B$, and then retargets $\hat{x}_{1:T}^B$ back to A to produce $\hat{\hat{x}}_{1:T}^A$:

$$\hat{x}_{1:T}^B = G(x_{1:T}^A, s^B) \tag{4}$$

$$\hat{\hat{x}}_{1:T}^A = G(\hat{x}_{1:T}^B, s^A) \tag{5}$$

Figure 4 illustrates the cycle. We have six loss terms: (1) $\mathcal{L}_c$ stands for the cycle consistency loss, (2) $\mathcal{L}_t$ for the joint twist loss, (3) $\mathcal{L}_h$ for the height loss, (4) $\mathcal{L}_a$ for the adversarial loss, (5) $\mathcal{L}_r$ for the regularization loss, and (6) $\mathcal{L}_o$ for the orientation loss. Our full training objective is defined as follows:

$$\min_G \max_D \quad \mathcal{L}_c + \lambda_t \mathcal{L}_t + \lambda_h \mathcal{L}_h + \lambda_a \mathcal{L}_a + \lambda_r \mathcal{L}_r + \lambda_o \mathcal{L}_o \tag{6}$$

where λ_* represents the weight of each loss term.

Cycle consistency loss. $\mathcal{L}_c$ is the standard term in cycle training, which minimizes the difference between $x_{1:T}^A$ and $\hat{\hat{x}}_{1:T}^A$:

$$\mathcal{L}_c(x_{1:T}^A, \hat{\hat{x}}_{1:T}^A) = \|x_{1:T}^A - \hat{\hat{x}}_{1:T}^A\|_2^2 \tag{7}$$

where x_t^A includes p_t^A as well as q_t^A and r_t^A which are all defined in Section 3.1. Similarly, $\hat{x}_t^A$ includes $\hat{p}_t^A$ as well as $\hat{q}_t^A$ and $\hat{r}_t^A$.

Joint twist loss. There is no explicit label for retargetted rotation

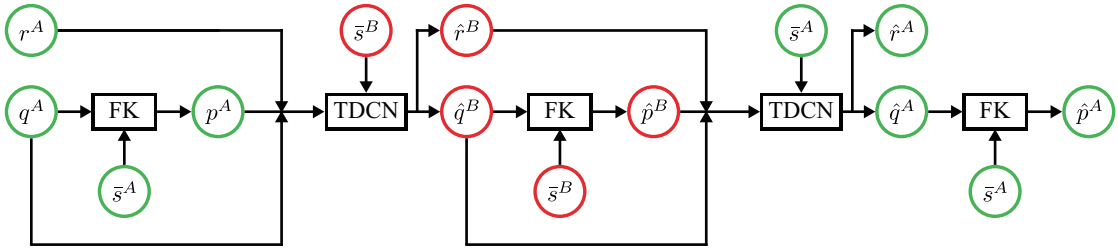

Figure 4: Adversarial cycle consistency training.

in the unsupervised learning framework. Consequently, we may often encounter excessive twisting of a joint. $\mathcal{L}_t$ constrains the joint rotation:

$$\mathcal{L}_t(\hat{q}^B_{1:T}, \hat{q}^A_{1:T}) = \|max(\mathbf{0}, |\mathcal{E}(\hat{q}^B_{1:T}) - \alpha|)\|_2^2 + \|max(\mathbf{0}, |\mathcal{E}(\hat{q}^A_{1:T}) - \alpha|)\|_2^2 \quad (8)$$

where $\mathcal{E}(\cdot)$ converts a quaternion into a rotation angle. Any angle exceeding α is penalized. In the current implementation, $\alpha = 100°$.

Height loss. Suppose that a tall source character (A) makes a stride and this is retargetted to a short target character (B). Then, B's motions should be smaller than A's. The height loss is in charge of achieving this effect.

With the local coordinates of the joints in A and B, i.e., $p^A_{1:T}$ and $\hat{p}^B_{1:T}$, the local motion differences are defined between two adjacent frames:

$$d^A_{2:T} = p^A_{2:T} - p^A_{1:T-1} \quad (9)$$

$$\hat{d}^B_{2:T} = \hat{p}^B_{2:T} - \hat{p}^B_{1:T-1} \quad (10)$$

Let us normalize their magnitudes using the heights of A and B, which are denoted as $\mathcal{H}(\bar{s}^A)$ and $\mathcal{H}(\bar{s}^B)$, respectively:

$$l^A_t = \frac{\|d^A_t\|}{\mathcal{H}(\bar{s}^A)} \quad (11)$$

$$\hat{l}^B_t = \frac{\|\hat{d}^B_t\|}{\mathcal{H}(\bar{s}^B)} \quad (12)$$

Then, a loss is defined as follows:

$$\text{smooth}_{L_1}(l^A_{2:T} - \hat{l}^B_{2:T}) \quad (13)$$

where

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2, & \text{if } |x| < 1 \\ |x| - 0.5, & \text{otherwise} \end{cases} \quad (14)$$

is a robust L_1 loss that is less sensitive to outliers than the L_2 loss.

Without loss of generality, we assume that the motions are made around or across the origin of the global coordinate system. Then, the magnitudes of the root position vectors, r^A_t and $\hat{r}^B_t$, can be normalized:

$$m^A_t = \frac{\|r^A_t\|}{\mathcal{H}(\bar{s}^A)} \quad (15)$$

$$\hat{m}^B_t = \frac{\|\hat{r}^B_t\|}{\mathcal{H}(\bar{s}^B)} \quad (16)$$

With m^A_t and $\hat{m}^B_t$, the loss in Equation (13) is extended to define the height loss:

$$\mathcal{L}_h(x^A_{1:T}, \hat{x}^B_{1:T}) = \text{smooth}_{L_1}(l^A_{2:T} - \hat{l}^B_{2:T}) + \lambda_g \text{smooth}_{L_1}(m^A_{2:T} - \hat{m}^B_{2:T}) \quad (17)$$

In the current implementation, $\lambda_g = 5$.

Adversarial loss. Consider the normalized magnitudes of the root motion differences between two adjacent frames:

$$\delta_{2:T} = \frac{\|r_{2:T} - r_{1:T-1}\|}{\mathcal{H}(\bar{s})} \quad (18)$$

The discriminator, D, computes the scores for A's *real* motion sequence and B's *fake* one:

$$h^A_{2:T} = D(l^A_{2:T}, \delta^A_{2:T}, \mathcal{B}^A) \quad (19)$$

$$h^B_{2:T} = D(\hat{l}^B_{2:T}, \hat{\delta}^B_{2:T}, \mathcal{B}^B) \quad (20)$$

where l^A_t and $\hat{l}^B_t$ are defined in Equations (11) and (12), respectively, and $\mathcal{B}^A$ and $\mathcal{B}^B$ represent the *bone lengths* of A and B, respectively.

As we randomly sample the skeletons during training, B can be identical to A. Then, $x^A_{1:T} = \hat{x}^B_{1:T}$. The adversarial loss is defined by distinguishing between two cases:

$$\mathcal{L}_a(x^A_{1:T}, \hat{x}^B_{1:T}) = \begin{cases} (h^A_{2:T})^2 + (1 - h^B_{2:T})^2 & \text{if } A \neq B \\ \|x^A_{1:T} - \hat{x}^B_{1:T}\|_2^2 & \text{if } A = B \end{cases} \quad (21)$$

where x^A_t includes p^A_t as well as q^A_t and r^A_t, and $\hat{x}^B_t$ includes $\hat{p}^B_t$ as well as $\hat{q}^B_t$ and $\hat{r}^B_t$.

$\mathcal{L}_a$ is basically taken from Villegas et al. [VYCL18] but is different from two aspects: (i) In regular GANs, the sigmoid cross entropy loss function often leads to the vanishing gradient problem [MLX*17]. In order to mitigate this problem and also improve the training stability, we use the least square loss function. (ii) The inputs to the discriminator, i.e., l^A_t, δ^A_t, $\hat{l}^B_t$ and $\hat{\delta}^B_t$, are normalized, and the discriminator uses shorter clips than the generator, as mentioned in Section 4.1.

Regularization loss. Inspired by the work of Pavllo et al. [PGA18], we use a penalty term with respect to the *unit* quaternions.

$$\mathcal{L}_r(\hat{q}^B_{1:T}, \hat{q}^A_{1:T}) = (1 - \|\hat{q}^B_{1:T}\|)^2 + (1 - \|\hat{q}^A_{1:T}\|)^2 \quad (22)$$

This loss function acts as a regularizer that leads to better training stability.

Orientation loss. The height loss $\mathcal{L}_h$ accounts for each joint's

	p_t		r_t	
training dataset	*M*	*SD*	*M*	*SD*
Villegas et al. [VYCL18]	5.95	36.95	40.06	80.30
ours (balanced)	5.74	**45.12**	51.59	**106.33**

Table 1: Dataset statistics: *M* stands for mean and *SD* for standard deviation.

translation or displacement, but does not handle the characters' orientation. Let θ_t denote the root joint's quaternion. Then, the orientation loss $\mathcal{L}_o$ optimizes the following objective:

$$\mathcal{L}_o(\theta^A_{1:T}, \hat{\theta}^B_{1:T}) = \text{smooth}_{L_1}(\mathcal{E}(\theta^A_{1:T}) - \mathcal{E}(\hat{\theta}^B_{1:T})) \quad (23)$$

5. Experiment Setup

For training our model, we used the Mixamo dataset [MIX], which contains approximately 2400 motion clips for 71 characters. For test, we used not only the Mixamo dataset but also Human3.6M motion capture dataset [IPOS14], which is extracted from 15 kinds of actions made by 7 subjects. Focusing on the Mixamo dataset, this section briefly presents the dataset used for training and test. As will be presented in Section 6, the baseline models used for comparisons were built upon the work of Villegas et al. [VYCL18] and therefore this section also presents their dataset.

Training dataset. We used 1646 non-overlapping motion clips for nine characters in Mixamo (AJ, Big Vegas, Kaya, Malcolm, Peasant Man, Regina, Remy, Shae, and Warrok Kurniawan). The dataset size was the same as that of Villegas et al. [VYCL18] but we used two more characters to make the data better balanced. We also performed random scaling, i.e., we scaled p_t and r_t with random factors in $[0.5, 1.5]$. Table 1 compares the statistics of two datasets. The larger the standard deviation is, the better balanced the dataset is. Section 6 presents the benefit brought by this balanced dataset.

Test dataset. We collected motion sequences of six characters (Malcolm, Mutant, Warrok Kurniawan, Sporty Granny, Claire, and Liam) from the Mixamo website, which stores motions in 52 pages. Table 2 lists the character-page combinations. The test dataset was collected along the guideline by Villegas et al. [VYCL18]:

1. Both the input motion and the target character are seen during training.
2. The input motion is seen during training but the target character is not.
3. The input motion is not seen during training but the target character is seen.
4. Neither the input motion nor the target character is seen during training.

The specific combinations of the input motion and target character are listed in Table 3. For evaluations, we also collected "as groundtruth" the Mixamo motions made by the target characters.

Data preprocessing. The characters in the Mixamo dataset have different numbers of joints. For both training and test, we selected the following 22 joints: Root, Spine, Spine1, Spine2, Neck, Head, LeftUpLeg, LeftLeg, LeftFoot, LeftToeBase, RightUpLeg, RightLeg, RightFoot, RightToeBase, LeftShoulder, LeftArm, LeftForeArm, LeftHand, RightShoulder, RightArm, RightForeArm, and RightHand.

test dataset	
character	page
Malcolm	28,51
Warrok W Kurniawan	18,52
Liam	23,45
Mutant	33,45,52
Claire	52
Sporty Granny	51

Table 2: Animation pages for test dataset.

scenario	input→target	page
(1)	Kaya→Warrok W Kurniawan	18
	Big Vegas→Malcolm	28
(2)	Peasant Man→Liam	23
	AJ→Mutant	33
(3)	Sporty Granny→Malcolm	51
	Claire→Warrok W Kurniawan	52
(4)	Mutant→Liam	45
	Claire→Mutant	52

Table 3: Combinations of the input motion and target character for each test scenario.

Training detail. Every motion sequence used for training our model was composed of 81 consecutive frames, which were randomly sampled from the raw motion clips of Mixamo. For training the discriminator, we sampled the motion clips performed by the character, which was taken as the target by the generator. We used the Adam optimizer [KB15] with a learning rate of $1e^{-4}$ and momentum parameters, $\beta_1 = 0.5$ and $\beta_2 = 0.999$.

We trained our model with a batch size of 128 and a learning rate of $1e^{-4}$ using PyTorch. Each batch had 128 pairs of a source character's motion sequence and a target character. In 50% of the pairs, we made the target identical to the source, i.e., $A = B$. We used a dropout rate of 0.1 for the generator. For the objective function presented in Equation (6) of Section 4.2, $\lambda_t = 10$, $\lambda_h = 10$, $\lambda_a = 1$, $\lambda_r = 0.1$, and $\lambda_o = 1$.

6. Evaluation

For comparisons, we used four baseline models. (i) We took the original work of Villegas et al. [VYCL18] It is called 'Baseline.' (ii) We replaced RNNs in Baseline with *dense convolutions*. It is called 'Baseline-dense.' (iii) We replaced RNNs in Baseline with TDCNs. It is called 'Baseline-dilated.' (iv) We added the height loss function ($\mathcal{L}_h$) to Baseline, where the root-position offsets are

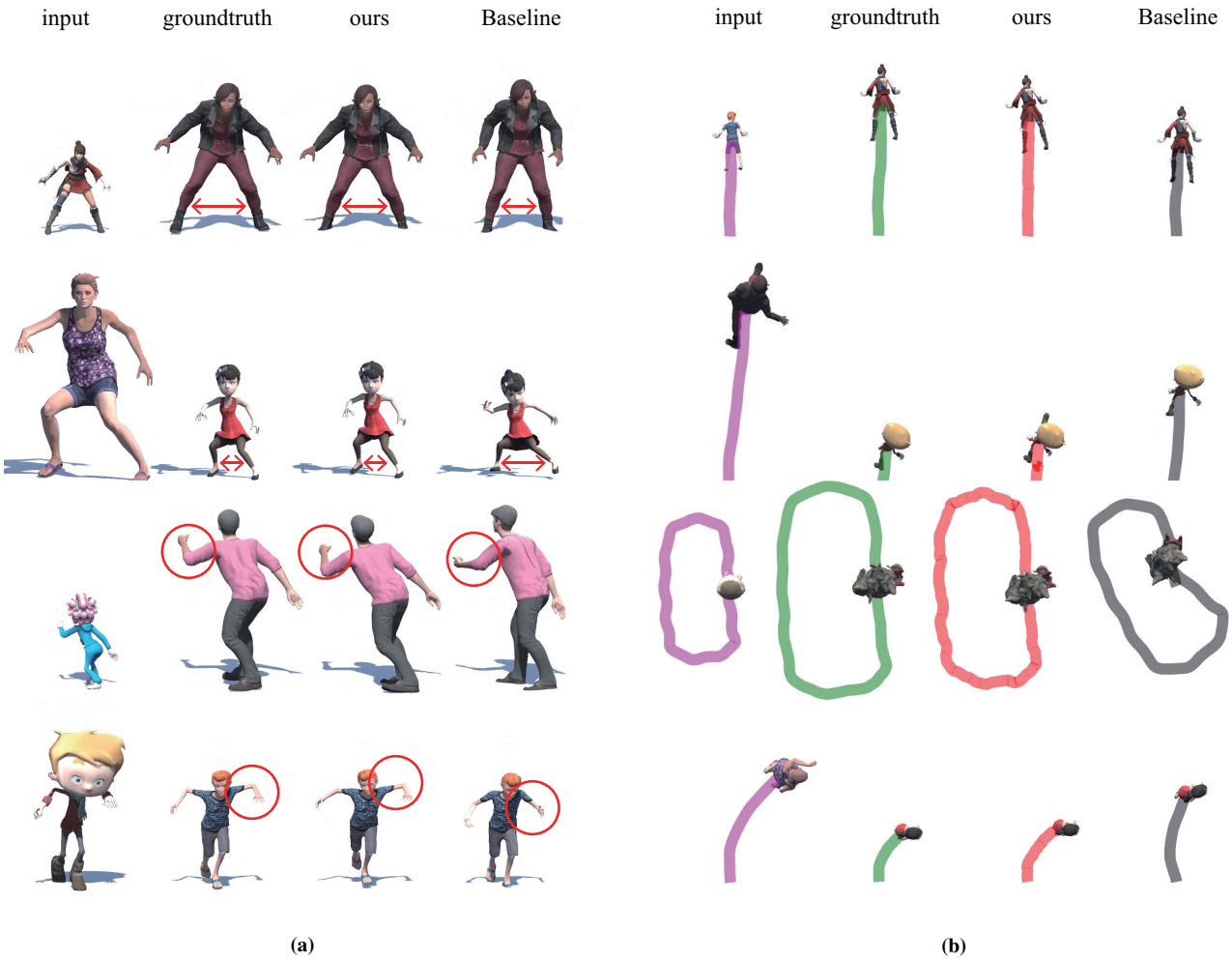

Figure 5: Qualitative comparisons: (a) The front view of the retargetted motions. (b) The top view of the root joint's trajectories.

normalized. It is called 'Baseline-height.' All baseline models were trained from scratch. Section 6.1 quantitatively compares the results of our model and four baseline models using the Mixamo dataset. Section 6.2 qualitatively compares the results of our model and Baseline using the Human 3.6M dataset.

6.1. Quantitative Comparison

For quantitative evaluation, we used mean square error (MSE) between the joints' global coordinates of the retargetted character and those of the groundtruth. Table 4 shows the results. The numbers in parentheses represent the four scenarios presented in Table 3. Table 4 reports the MSEs for *short* and *long* motion sequences. A *short* sequence was composed of 120 frames (for 4 seconds). In the work of Villegas et al. [VYCL18], every test sequence was *short*. However, motion retargetting in reality requires us to take the 'entire' sequence of the source motions. It is more difficult to perform retargetting with longer sequences. In our test, *long* sequences had at maximum 1130 frames and at minimum 121 frames. Their mean was 228.

Table 4 shows that our model outperformed all baseline models for both *short* and *long*. Note that our model showed similar performances for *short* and *long*. In contrast, the baseline models significantly degraded for *long*, including Baseline-height, where the root-position offsets are normalized. We believe that the baseline models suffer from the artifact because they output the position offsets of the skeleton's root, making errors accumulated, whereas our model directly outputs the root positions.

Baseline-height performed the best among the baseline models. This proves the strong impact of the height loss ($\mathcal{L}_h$). It is interesting to find that Baseline-dense and Baseline-dilated performed mostly worse than Baseline. This indicates that simply modifying the architecture of the generator does not guarantee performance improvements.

We conducted an ablation study in order to validate the effectiveness of several features of our model. In Table 4, A1 through A5 denote our models with a feature's absence or modification.

- A1: TDCN was replaced by the typical dense convolutional network, which had 6.5 times more parameters than TDCN. Com-

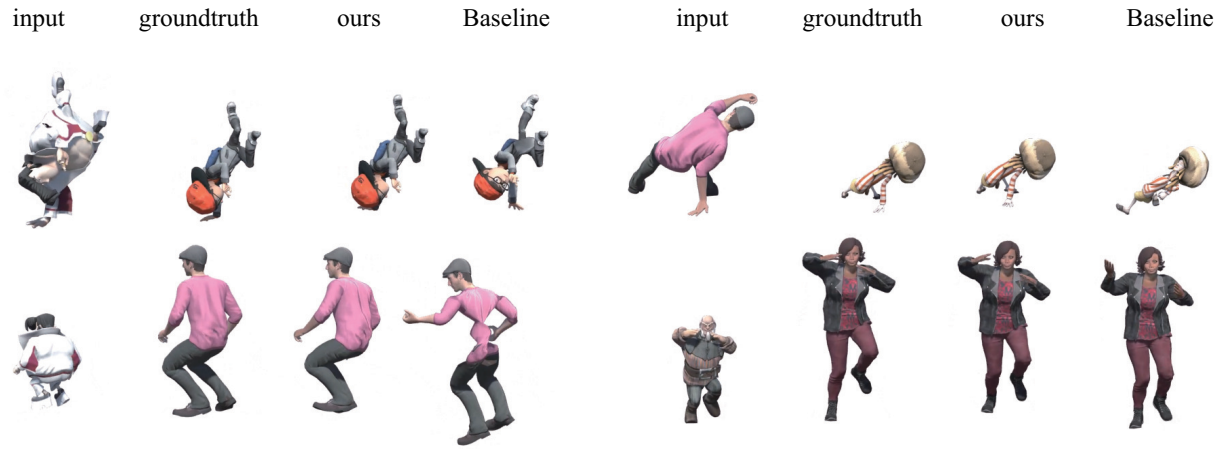

Figure 6: Dancing motion retargetting.

	short					*long*				
model	(1)	(2)	(3)	(4)	avg.	(1)	(2)	(3)	(4)	avg.
Baseline: Villegas et al. [VYCL18]	4.79	1.22	3.99	18.31	7.08	26.14	4.64	8.38	45.34	21.12
Baseline-dense: Baseline w/ dense conv.	4.45	1.73	3.85	19.71	7.44	23.66	4.75	9.14	35.37	18.23
Baseline-dilated: Baseline w/ TDCNs	7.92	4.53	7.63	26.26	11.59	30.14	9.41	11.24	49.43	25.06
Baseline-height: Baseline w/ $\mathcal{L}_h$	2.72	0.88	3.29	6.63	3.38	14.86	2.65	9.13	14.89	10.38
ours	2.21	**0.82**	**2.70**	**4.06**	**2.45**	1.65	**0.78**	**3.00**	**4.49**	**2.48**
A1: ours w/ dense conv.	**1.58**	0.99	3.00	4.21	**2.45**	**1.63**	0.80	3.49	6.33	3.06
A2: ours trained w/ unbalanced dataset	3.74	2.71	3.71	9.60	4.94	3.99	2.80	4.89	13.14	6.20
A3: ours w/o $\mathcal{L}_h$	6.90	1.70	5.26	25.23	9.77	12.27	2.25	6.23	51.66	18.10
A4: ours w/o patch-based discriminator	2.31	1.28	3.70	4.30	2.90	1.97	1.24	4.23	5.31	3.19
A5: ours w/ causal conv	1.65	1.15	2.86	5.54	2.80	2.03	0.83	3.58	8.39	3.71

Table 4: Quantitative comparisons using normalized mean square error (MSE).

pared with our full model, A1 showed similar performances for *short* but significantly degraded for *long*. As A1 was trained with 81-frame sequences, we argue that A1 was *overfitted* to short sequences. It is interesting to find that A1 excelled ours only for scenario (1), which consists of known motions and known skeletons. We argue that A1 was *overfitted* also to known motions and skeletons. In contrast, dilated convolutions counteract overfitting, as reported by Pavllo et al. [PFGA19] in their ablation studies.

- A2: Our model was trained not with the balanced dataset but with the dataset used by Villegas et al. [VYCL18] A2's performances were degraded by more than 50% for both *short* and *long*. Note however that A2 performed better than Baseline. We also tested the reverse, i.e., Villegas et al. [VYCL18] was trained with the balanced dataset, but the resulting performance was too poor to be worth being reported. Our speculation is twofold: (1) The vanilla GAN requires a vast amount of hyperparameter tuning for the new (balanced) dataset because it uses Jensen-Shannon divergence as the loss function. (2) Our model uses the least squares loss function and so it performs more stably.
- A3: The height loss ($\mathcal{L}_h$) was removed from the objective function of our model. From the outset, $\mathcal{L}_h$ was designed to handle the height differences among characters. Its strong impact was clearly proved in our ablation study.
- A4: We added a convolution layer to our discriminator such that it sees the same number of frames (81 frames) as the generator. Being forced to see longer sequences (than ours), the discriminator often misses high-frequency details. A4's performances degraded more for *long* because the missed details, i.e., the errors, are accumulated.
- A5: Note that our model uses not only the past frames but also the future ones. In order for a model to be used for live or real-time motion retargetting, it should use only the past frames. To this end, we tested our TDCN with a *causal convolution*. The performance was worse than our original TDCN but better than Baseline.

6.2. Qualitative Comparison

Figure 5a shows that our model successfully retargetted the motions of the source to the target despite the differences in their skeleton sizes. Figure 5b shows the top views of the characters' root joint trajectories. Given the same stride count, the trajectory of a tall character should be longer than that of a short character. Our model

Figure 7: Motion capture data retargetting.

successfully produced such differences. Figure 6 compares another set of results with an acrobatic dancer.

The earlier version of our model did not take as input the positions, p_t^A. It implies that the model was designed to 'learn' the forward kinematics (FK). The results were not satisfactory. We speculated that it is hard to learn FK in an unsupervised way. Then, the model was modified to take the positions, not the rotations, q_t^A, but suffered from the same problems of Villegas et al. [VYCL18] shown in Figure 6: The target's limb joints did not correctly follow the source's, and complex or rapid motions were not smoothly retargetted. In contrast, the results were satisfactory when we used both positions and rotations as input to our model. It would be because the skip-connections (presented in Figure 2b) help our model learn how to generate the output rotations, $\hat{q}_t^B$.

Figure 7 shows the results of retargetting the motion capture data of Human3.6M dataset to six virtual characters of different skeleton sizes. Recall that our model was designed to work on 22-joint characters and were trained using the Mixamo dataset only. The Mixamo dataset has 25 frames per second. In Human3.6M, the motion data captured at 50 fps has 32 joints. In a preprocessing stage, the number of joints was reduced to 22, and the frames were downsampled by half, i.e., to 25 fps. It is important to note that the human actors used for test were never seen during training, i.e., our model generalizes to such new motions.

7. Conclusion and Future Work

This paper presents a motion retargetting model based on temporal dilated convolutional networks. It is trained with adversarial cycle consistency objective in an unsupervised manner to overcome the lack of training pairs. The success factors of our proposed model can be listed as follows: (1) Temporal dilated convolutions make our model more stable and robust when training with various characters. (2) The loss functions make our model reflect the skeleton size differences quite effectively. (3) The limited receptive fields of our discriminator allow to capture the high-frequency detail of input motions.

Our model also has limitations. First of all, our model assumes that the source and target skeletons have the same number of joints. A solution to retarget motions between heterogeneous skeletons would be to project the joint or vertex positions onto voxels and use 3D CNN to retarget them. Another solution would be to adopt the existing techniques of learning or building a mapping function between different character morphologies [YAH10,SOL13,RTIK*14]. However, the techniques were developed in a supervised way, and therefore we should extend them to our unsupervised learning framework.

Secondly, our model currently does not take into account the end-effectors such as hand and feet. In order to generate naturally retargetted motions of such end-effectors, e.g., to avoid the footskating artifact, the objective function should be extended to have the loss terms elaborately designed for the end-effectors because people are quite sensitive to the hand and feet motions. Rhodin et al. [RTK*15] showed that footskating artifact could be handled using a weighted vote based on foot contact database. We envision that, if our network predicts the foot contact in the retargetted motion, the artifact can be handled using their method.

Thirdly, our model uses the future frames as well as the past ones. As discussed in Section 6, it prevented our model from being used for live or real-time motion retargetting. (In contrast, RNNs work in an online manner.) The causal model, A5, presented in Table 4 showed a reasonable performance for *short*, but the performance for *long* requires improvement. We envision that inverse kinematics supported in real-time game engines can be adopted for the improvement. Our future work will focus on overcoming the limitations.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korea government (MSIT) (NRF-2017M3C4A7066316 and No. NRF2016-R1A2B3014319).

References

[AKH19] AKSAN E., KAUFMANN M., HILLIGES O.: Structured prediction helps 3d human motion modelling. In *Proceedings of the IEEE International Conference on Computer Vision* (2019), pp. 7144–7153. 2

[AWL*19] ABERMAN K., WU R., LISCHINSKI D., CHEN B., COHEN-OR D.: Learning character-agnostic motion for motion retargeting in

2d. *ACM Trans. Graph. 38*, 4 (July 2019). URL: https://doi.org/10.1145/3306346.3322999, doi:10.1145/3306346.3322999. 2

[BBKK17] BUTEPAGE J., BLACK M. J., KRAGIC D., KJELLSTROM H.: Deep representation learning for human motion prediction and classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 6158–6166. 2

[BH00] BRAND M., HERTZMANN A.: Style machines. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., pp. 183–192. doi:10.1145/344779.344865. 3

[BSF94] BENGIO Y., SIMARD P., FRASCONI P.: Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks 5*, 2 (March 1994), 157–166. doi:10.1109/72.279181. 2

[CK00] CHOI K.-J., KO H.-S.: Online motion retargetting. *The Journal of Visualization and Computer Animation 11*, 5 (2000), 223–235. doi:10.1002/1099-1778(200012)11:5<223::AID-VIS236>3.0.CO;2-5. 2

[FLFM15] FRAGKIADAKI K., LEVINE S., FELSEN P., MALIK J.: Recurrent network models for human dynamics. In *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 4346–4354. 2

[GAG*17] GEHRING J., AULI M., GRANGIER D., YARATS D., DAUPHIN Y. N.: Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70* (2017), ICML'17, JMLR.org, pp. 1243–1252. 2

[Gle98] GLEICHER M.: Retargetting motion to new characters. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1998), SIGGRAPH '98, ACM, pp. 33–42. doi:10.1145/280814.280820. 2

[GMHP04] GROCHOW K., MARTIN S. L., HERTZMANN A., POPOVIĆ Z.: Style-based inverse kinematics. *ACM Transactions on Graphics (TOG) 23*, 3 (Aug. 2004), 522–531. doi:10.1145/1015706.1015755. 3

[GSAH17] GHOSH P., SONG J., AKSAN E., HILLIGES O.: Learning human motion models for long-term predictions. In *2017 International Conference on 3D Vision (3DV)* (2017), IEEE, pp. 458–466. 2

[HL18] HOSSAIN M. R. I., LITTLE J. J.: Exploiting temporal information for 3d human pose estimation. In *European Conference on Computer Vision* (2018), Springer, pp. 69–86. 2

[HPP05] HSU E., PULLI K., POPOVIĆ J.: Style translation for human motion. *ACM Transactions on Graphics (TOG) 24*, 3 (July 2005), 1082–1089. doi:10.1145/1073204.1073315. 3

[HS97] HOCHREITER S., SCHMIDHUBER J.: Long short-term memory. *Neural Computation 9*, 8 (1997), 1735–1780. doi:10.1162/neco.1997.9.8.1735. 2

[HZRS16] HE K., ZHANG X., REN S., SUN J.: Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016), pp. 770–778. doi:10.1109/CVPR.2016.90. 4

[IPOS14] IONESCU C., PAPAVA D., OLARU V., SMINCHISESCU C.: Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence 36*, 7 (July 2014), 1325–1339. doi:10.1109/TPAMI.2013.248. 6

[IS15] IOFFE S., SZEGEDY C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning* (Lille, France, 07–09 Jul 2015), Bach F., Blei D., (Eds.), vol. 37 of *Proceedings of Machine Learning Research*, PMLR, pp. 448–456. 4

[IZZE17] ISOLA P., ZHU J., ZHOU T., EFROS A. A.: Image-to-image translation with conditional adversarial networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017), pp. 5967–5976. doi:10.1109/CVPR.2017.632. 3, 4

[JZSS16] JAIN A., ZAMIR A. R., SAVARESE S., SAXENA A.: Structural-rnn: Deep learning on spatio-temporal graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 5308–5317. 2

[KB15] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)* (2015). 6

[KES*16] KALCHBRENNER N., ESPEHOLT L., SIMONYAN K., VAN DEN OORD A., GRAVES A., KAVUKCUOGLU K.: Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099* (2016). URL: https://arxiv.org/abs/1610.10099. 2

[KTS*18] KATIRCIOGLU I., TEKIN B., SALZMANN M., LEPETIT V., FUA P.: Learning latent representations of 3d human pose with deep neural networks. *International Journal of Computer Vision 126*, 12 (2018), 1326–1341. 2

[LC14] LI S., CHAN A. B.: 3d human pose estimation from monocular images with deep convolutional neural network. In *Asian Conference on Computer Vision* (2014), Springer, pp. 332–347. 2

[LLL*17] LIN M., LIN L., LIANG X., WANG K., CHENG H.: Recurrent 3d pose sequence machines. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017), pp. 5543–5552. doi:10.1109/CVPR.2017.588. 2

[LS99] LEE J., SHIN S. Y.: A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1999), SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., pp. 39–48. doi:10.1145/311535.311539. 2

[LW16] LI C., WAND M.: Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European Conference on Computer Vision* (2016), Springer, pp. 702–716. 4

[Mai90] MAILLOT P.-G.: Graphics gems. Academic Press Professional, Inc., San Diego, CA, USA, 1990, ch. Using Quaternions for Coding 3D Transformations, pp. 498–515. 2

[MBBT00] MONZANI J.-S., BAERLOCHER P., BOULIC R., THALMANN D.: Using an intermediate skeleton and inverse kinematics for motion retargeting. In *Computer Graphics Forum* (2000), vol. 19, Wiley Online Library, pp. 11–19. 2

[MBR17] MARTINEZ J., BLACK M. J., ROMERO J.: On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 2891–2900. 2

[MIX] Adobe's Mixamo. Accessed: 2019-01-08. URL: https://www.mixamo.com. 6

[MLX*17] MAO X., LI Q., XIE H., LAU R. Y. K., WANG Z., SMOLLEY S. P.: Least squares generative adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)* (Oct 2017), pp. 2813–2821. doi:10.1109/ICCV.2017.304. 5

[MRC*17] MEHTA D., RHODIN H., CASAS D., FUA P., SOTNYCHENKO O., XU W., THEOBALT C.: Monocular 3d human pose estimation in the wild using improved cnn supervision. In *2017 International Conference on 3D Vision (3DV)* (Oct 2017), pp. 506–516. doi:10.1109/3DV.2017.00064. 2

[MSS*17] MEHTA D., SRIDHAR S., SOTNYCHENKO O., RHODIN H., SHAFIEI M., SEIDEL H.-P., XU W., CASAS D., THEOBALT C.: Vnect: Real-time 3d human pose estimation with a single rgb camera. *ACM Transactions on Graphics (TOG) 36*, 4 (July 2017), 44:1–44:14. doi:10.1145/3072959.3073596. 2

[PFGA19] PAVLLO D., FEICHTENHOFER C., GRANGIER D., AULI M.: 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 7753–7762. 2, 4, 8

[PGA18] PAVLLO D., GRANGIER D., AULI M.: Quaternet: A quaternion-based recurrent model for human motion. In *British Machine Vision Conference (BMVC)* (2018). 5

[PHK16] PARK S., HWANG J., KWAK N.: 3d human pose estimation using convolutional neural networks with 2d pose information. In *European Conference on Computer Vision* (2016), Springer, pp. 156–169. 2

[PMB13] PASCANU R., MIKOLOV T., BENGIO Y.: On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning* (Atlanta, Georgia, USA, 17–19 Jun 2013), Dasgupta S., McAllester D., (Eds.), vol. 28 of *Proceedings of Machine Learning Research*, PMLR, pp. 1310–1318. 2

[PZDD17] PAVLAKOS G., ZHOU X., DERPANIS K. G., DANIILIDIS K.: Coarse-to-fine volumetric prediction for single-image 3d human pose. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017), pp. 1263–1272. doi:10.1109/CVPR.2017.139. 2

[RS16] ROGEZ G., SCHMID C.: Mocap-guided data augmentation for 3d pose estimation in the wild. In *Advances in Neural Information Processing Systems 29*, Lee D. D., Sugiyama M., Luxburg U. V., Guyon I., Garnett R., (Eds.). Curran Associates, Inc., 2016, pp. 3108–3116. 2

[RTIK*14] RHODIN H., TOMPKIN J., IN KIM K., VARANASI K., SEIDEL H.-P., THEOBALT C.: Interactive motion mapping for real-time character control. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 273–282. 9

[RTK*15] RHODIN H., TOMPKIN J., KIM K. I., DE AGUIAR E., PFISTER H., SEIDEL H.-P., THEOBALT C.: Generalizing wave gestures from sparse examples for real-time character control. *ACM Transactions on Graphics (TOG) 34*, 6 (2015), 181. 9

[SHK*14] SRIVASTAVA N., HINTON G., KRIZHEVSKY A., SUTSKEVER I., SALAKHUTDINOV R.: Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research 15* (2014), 1929–1958. 4

[SOL13] SEOL Y., O'SULLIVAN C., LEE J.: Creature features: online motion puppetry for non-human characters. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2013), ACM, pp. 213–221. 9

[SSLW17] SUN X., SHANG J., LIANG S., WEI Y.: Compositional human pose regression. In *2017 IEEE International Conference on Computer Vision (ICCV)* (Oct 2017), pp. 2621–2630. doi:10.1109/ICCV.2017.284. 2

[TK05] TAK S., KO H.-S.: A physically-based motion retargeting filter. *ACM Transactions on Graphics (TOG) 24*, 1 (Jan. 2005), 98–117. doi:10.1145/1037957.1037963. 2

[TKS*16] TEKIN B., KATIRCIOGLU I., SALZMANN M., LEPETIT V., FUA P.: Structured prediction of 3d human pose with deep neural networks. In *Proceedings of the British Machine Vision Conference (BMVC)* (September 2016), Richard C. Wilson E. R. H., Smith W. A. P., (Eds.), BMVA Press, pp. 130.1–130.11. doi:10.5244/C.30.130. 2

[TRA17] TOME D., RUSSELL C., AGAPITO L.: Lifting from the deep: Convolutional 3d pose estimation from a single image. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017), pp. 5689–5698. doi:10.1109/CVPR.2017.603. 2

[TRLF16] TEKIN B., ROZANTSEV A., LEPETIT V., FUA P.: Direct prediction of 3d body poses from motion compensated sequences. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016), pp. 991–1000. doi:10.1109/CVPR.2016.113. 2

[vdODZ*16] VAN DEN OORD A., DIELEMAN S., ZEN H., SIMONYAN K., VINYALS O., GRAVES A., KALCHBRENNER N., SENIOR A., KAVUKCUOGLU K.: Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499* (2016). URL: https://arxiv.org/abs/1609.03499. 2

[VRM*17] VAROL G., ROMERO J., MARTIN X., MAHMOOD N., BLACK M. J., LAPTEV I., SCHMID C.: Learning from synthetic humans. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017), pp. 4627–4635. doi:10.1109/CVPR.2017.492. 2

[VSP*17] VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER L. U., POLOSUKHIN I.: Attention is all you need. In *Advances in Neural Information Processing Systems 30*, Guyon I., Luxburg U. V., Bengio S., Wallach H., Fergus R., Vishwanathan S., Garnett R., (Eds.). Curran Associates, Inc., 2017, pp. 5998–6008. 2

[VYCL18] VILLEGAS R., YANG J., CEYLAN D., LEE H.: Neural kinematic networks for unsupervised motion retargetting. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (June 2018), pp. 8639–8648. doi:10.1109/CVPR.2018.00901. 2, 3, 4, 5, 6, 7, 8, 9

[XWCL15] XU B., WANG N., CHEN T., LI M.: Empirical evaluation of rectified activations in convolutional network. In *ICML Deep Learning Workshop* (2015). URL: https://arxiv.org/abs/1505.00853. 4

[YAH10] YAMANE K., ARIKI Y., HODGINS J.: Animating non-humanoid characters with human motion data. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2010), Eurographics Association, pp. 169–178. 9

[YK16] YU F., KOLTUN V.: Multi-scale context aggregation by dilated convolutions. In *International Conference on Learning Representations (ICLR)* (2016). 2

[ZPIE17] ZHU J., PARK T., ISOLA P., EFROS A. A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)* (Oct 2017), pp. 2242–2251. doi:10.1109/ICCV.2017.244. 4

[ZSZ*16] ZHOU X., SUN X., ZHANG W., LIANG S., WEI Y.: Deep kinematic pose regression. In *European Conference on Computer Vision* (2016), Springer, pp. 186–201. 2