

# Neural Network-Based Human Motion Smoother

Mathias Bastholm<sup>1</sup><sup>a</sup>, Stella Graßhof<sup>1</sup><sup>b</sup> and Sami S. Brandt<sup>1</sup><sup>c</sup>

<sup>1</sup> *Computer Science Department, IT University of Copenhagen, Denmark*

**Keywords:** Recurrent Neural Networks, Reconstruction, Computer Vision, Animation Denoising.

**Abstract:** Recording real life human motion as a skinned mesh animation with an acceptable quality is usually difficult. Even though recent advances in pose estimation have enabled motion capture from off-the-shelf webcams, the low quality makes it infeasible for use in production quality animation. This work proposes to use recent advances in the prediction of human motion through neural networks to augment low quality human motion, in an effort to bridge the gap between cheap recording methods and high quality recording. First, a model, competitive with prior work in short-term human motion prediction, is constructed. Then, the model is trained to clean up motion from two low quality input sources, mimicking a real world scenario of recording human motion through two webcams. Experiments on simulated data show that the model is capable of significantly reducing noise, and it opens the way for future work to test the model on annotated data.

## 1 INTRODUCTION

Humanoid 3D meshes are usually driven by a skeleton. These skeletons are then either animated by hand or by using motion capture (MoCap) to capture real life human motion as digital animations. Animating skeletons by hand is a time-consuming process and requires a skilled animator. Likewise, MoCap requires specialized equipment and often also requires an animator to clean up the recorded data. Animating humanoids thus consumes a lot of time and money for content creators, and might drive them to choose not to include an animation at all.

Recently several solutions allowing for MoCap from a single video camera have been published (Rong et al., 2020; Joo et al., 2020; Shi et al., 2020; Pavllo et al., 2019b). These are not widely used, which is likely because the quality is much lower than that of MoCap and hand-crafted animations. They would require a significant clean up pass by an animator in order to be of use, even for projects with relatively low animation quality requirements.

Recurrent neural networks (RNNs) architectures have made great progress in predicting human motion (Pavllo et al., 2019a; Martinez et al., 2017; Chiu et al., 2018; Gopalakrishnan et al., 2019), and have been shown to work for other tasks that involve generating human motion (Harvey et al., 2020). As such, this work sets out to explore the feasibility of adapting


existing human motion prediction architectures to the task of cleaning up human motion. This would allow for using cheap, realtime MoCap solutions based on webcams to capture human motion.


This paper is organized as follows. First, the related work is described in section 2. Then prior work on predicting human motion is replicated, and extended to the task of motion augmentation in section 3. Results of the proposed models on both prediction and augmentation of human motion can then be seen in section 4. Finally, we conclude this paper by a throughout discussion, and future work in section 5.


## 2 RELATED WORK

Working with human data necessitates choosing a sparse representation of the data, as dense representations make computations infeasible.

The Skinned Multi-Person Linear (SMPL) family of humanoid models (Loper et al., 2015; Romero et al., 2017; Pavlakos et al., 2019; Osman et al., 2020) consists of several models that express the pose and shape of human bodies in a sparse manner. This is accomplished by representing the human as a skinned mesh, with blend shapes representing the shape of the human, and the underlying skeleton of the skinned mesh representing the pose. Having chosen one representation of human motion, new samples can be generated using neural networks based on different kinds of input. For example several methods (Holden et al., 2017; Zhang et al., 2018; Starke et al., 2019; Starke

<sup>a</sup>  <https://orcid.org/0000-0001-9897-0002>

<sup>b</sup>  <https://orcid.org/0000-0002-6791-7425>

<sup>c</sup>  <https://orcid.org/0000-0003-2141-9815>

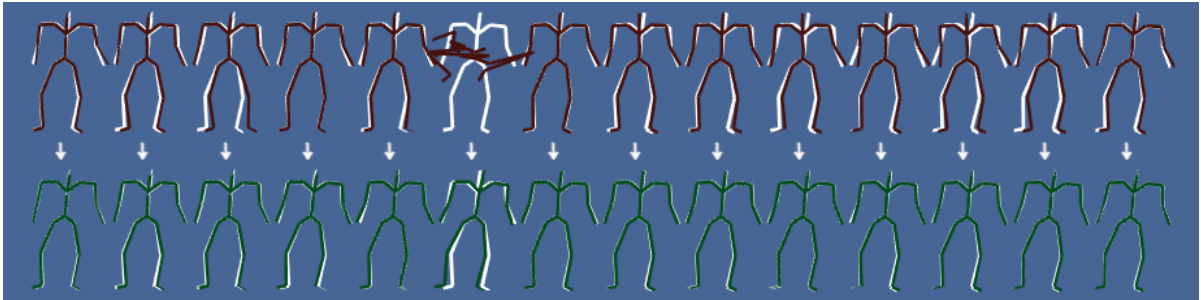


Figure 1: The top row shows input data in red, the bottom row shows the output of our model in green, and both rows show ground truth in white.

et al., 2020; Ling et al., 2020; Holden et al., 2020) exist that generate humanoid motion based on user input, such as a joystick or a goal position. This is particularly relevant for the video game industry, where player controlled characters are common. However, this setting is not directly applicable to the task of cleaning up human motion.

Instead of using a skinned mesh, so-called motion capture markers (MoCap markers), can be used to represent human body motion by a sparse set of points. MoCap markers represent physical points in space traditionally recorded by a Mocap system. E.g., a marker could be one small white ball attached to a key point, i.e., one joint, on the captured person. Holden (Holden, 2018) used a multi-layer perceptron (MLP) with skip connections inspired by the ResNet (He et al., 2015) architecture, to remove common noise on MoCap marker position introduced by most motion capture setups. The model is not easily integrated into existing workflows that operate on humanoid motion, as it operates on MoCap markers, which means that a separate post-processing step is needed in order to obtain a humanoid skeleton.

Another branch of methods that deal with generating human motion is human motion prediction. One such method is QuaterNet (Pavlo et al., 2019a) as proposed by Pavlo et al., which consists of a 2-layer RNN predicting future human motion from past motion, using a forwards kinematics (FK) loss. In that work the authors represent rotations as quaternions, opposed to previous work where Euler angles or exponential maps are frequently employed. The choice is motivated by the fact that Euler angles and axis-angle representations come with several problems: non-uniqueness, discontinuity in the representation space, and singularities, which are not exhibited by quaternions. QuaterNet also introduces a normalization loss, as normalized quaternions are required to represent valid rotations. The FK loss is calculated by performing FK and then taking the positional loss of the joints. FK is when the joint positions are cal-

culated from the joint rotations using the pre-defined skeleton. FK loss helps against the positional error introduced on the outer limbs by rotational error on the inner limbs, as the positional error of the outer limbs is affected by the rotational error of all parent limbs in the kinematic chain.

Building on previous work on motion prediction, such as QuaterNet, Harvey et al. (Harvey et al., 2020) propose a model that can fill in gaps of missing motion in a given motion sequence. It takes past motion and a target frame as input, and then generates the frames in-between using an RNN. To help the model maintain temporal coherency a time-to-arrival embedding is added to the input frames, which tells the model how many frames are left before the target frame is reached. This is the same approach as the positional embeddings in transformers (Vaswani et al., 2017). They introduce an adversarial loss based on Least Squares Generative Adversarial Network (Mao et al., 2017) (LSGAN), which is applied in order to create realistic looking and temporally coherent motion. A foot contact loss is also introduced, which gives an indication of whether each foot is touching the ground. This information stabilizes the feet as a post-processing step, which helps to combat a phenomenon commonly known as foot sliding. Foot loss can also be found in other recent work involving human motion, such as MotioNet (Shi et al., 2020).

### 3 METHODS

First, a model for prediction is constructed as a baseline model for dealing with human motions, for which we employ existing knowledge about RNNs for motion prediction. In the following step the prediction model is extended to be able to perform motion augmentation instead of prediction. This builds on a model architecture that is known to handle humanoid motion well in a prediction context, but now augments frames instead of predicting them.

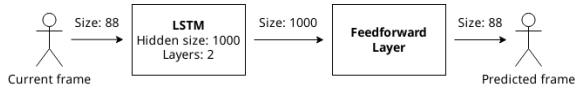


Figure 2: The architecture of the prediction model.

### 3.1 Prediction Model

The prediction model is trained to predict human motion, which means given a past frame of human motion it predicts the next frame. The model is based on the short term version of QuaterNet (Pavlo et al., 2019a), with two notable differences. First, a long short-term memory (LSTM) network is used in place of a gated recurrent unit (GRU) network, motivated by results from Harvey et al. (Harvey et al., 2020). Secondly, the rotational loss is calculated as the L1 loss of the quaternions, as opposed to taking the L1 loss of the Euler angles constructed from the quaternions. This rotational loss combines rotational error and quaternion normalization error, eliminating the need for an explicit quaternion normalization loss. This means that for a predicted sequence  $\hat{X}$  and the ground truth  $X$  the loss is defined as

$$L_{\text{prediction}} = \frac{1}{T} \sum_{t=0}^T \sum_{j=0}^J \|\hat{X}_{t,j} - X_{t,j}\|_1, \quad (1)$$

where  $T$  is the sequence length, and  $J$  is amount of joints in the skeleton. The prediction model consists of a two layer LSTM encoder with a hidden size of 1000 and a decoder. The decoder consists of a simple feedforward layer, added to convert from the hidden size of 1000 to the target output size of 88. The network is given 50 past frames and then outputs a single predicted frame. During training, this process is repeated to generate 10 predicted frames, with the past frames containing the past model outputs. See Figure 2 for a visualization of this architecture.

### 3.2 Augmentation Model

The augmentation model augments human motion, which means given a noisy frame it outputs a frame without noise. The model is based on the prediction model, and uses a two layer LSTM with a hidden size of 1000 for encoding and a feedforward layer for decoding.

For augmentation two input sources are used, reflecting a real world usage of having two different cameras from different angles with slightly different error patterns. This is handled by adding different noise to the same input frame. The corresponding two frames are then concatenated together increasing the input size of the LSTM to twice the size of a single frame. The loss is the same as that of the prediction model, as shown in Equation 1.

The network is then given all frames from each input source one pair at a time, and for each pair of frames given to the network it outputs an augmented frame corresponding to that pair of noisy frames. When training the size of each input source is limited for batching purposes, but for evaluation the network runs on all the frames.

### 3.3 Datasets

This work uses two datasets, the first is the Archive of Motion Capture as Surface Shapes (AMASS) database (Mahmood et al., 2019), which consists of many datasets, but for this work only the CMU subset is used (Carnegie Mellon University, 2003). The dataset consists of 2605 humanoid motions across 106 subjects, totalling 552 minutes of motions at varying framerates and 3.5M frames. We use Fairmotion (Gopinath and Won, 2020) to load and manipulate the motions from this dataset. The data is split so that 90% of the motions are used for training, 5% for validation, and 5% for testing. A single training step through each of the models require at least 60 frames, as such motions with less than 60 frames have been excluded. The second dataset used is the Human3.6M dataset (Ionescu et al., 2014; Catalin Ionescu, 2011), which consists of 210 motions across 7 subjects, totaling 176 minutes of motions at 50 frames per second and 0.5M frames. The dataset is split into training and test data by using all motions from subject 5 as test data and the rest as training data, as in previous work (Martinez et al., 2017; Pavlo et al., 2019a; Harvey et al., 2020).

The framerate differs between motions in the datasets used. To prevent this from affecting the model all motions are resampled to 25 fps. This is done by either discarding frames or interpolating frames, depending on whether downsampling or up-sampling is needed.

Both the prediction and augmentation model are trained on data from AMASS, whereas the prediction model is also trained and evaluated on the Human3.6M dataset. This is done in order to compare with previous work, as all previous work evaluates on the Human3.6M dataset, but only some use the AMASS dataset. This is necessary, as a model cannot be trained on the AMASS dataset and then evaluated on Human3.6M dataset, as the two datasets use different skeletons with a different joint count.

Table 1: Results of our prediction model on the Human3.6M dataset compared to reported results of Zero-velocity (Martinez et al., 2017), QuaterNet (Pavlo et al., 2019a), TP-RNN (Chiu et al., 2018), ERD-QV (Harvey et al., 2020), and VGRU-rl (Gopalakrishnan et al., 2019). The values are the mean squared loss after converting the rotations to euler angles, as described by Equation 11.

milliseconds	Walking				Eating				Smoking				Discussion			
	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Zero-velocity	0.39	0.68	0.99	1.15	0.27	0.48	0.73	0.86	0.26	0.48	0.97	0.95	0.31	0.67	0.94	1.04
QuaterNet	0.21	<b>0.34</b>	<b>0.56</b>	<b>0.62</b>	0.20	0.35	0.58	0.70	0.25	0.47	0.93	0.90	0.26	0.60	<b>0.85</b>	<b>0.93</b>
TP-RNN	0.25	0.41	0.58	0.65	0.20	<b>0.33</b>	<b>0.53</b>	0.67	0.26	0.47	0.88	0.90	0.30	0.66	0.96	1.04
ERD-QV	<b>0.20</b>	<b>0.34</b>	<b>0.56</b>	0.64	<b>0.18</b>	<b>0.33</b>	<b>0.53</b>	<b>0.63</b>	<b>0.23</b>	0.47	0.96	0.99	<b>0.23</b>	<b>0.59</b>	0.86	<b>0.93</b>
VGRU-rl	0.34	0.47	0.64	0.72	0.27	0.40	0.64	0.79	0.36	0.61	<b>0.85</b>	0.92	0.46	0.82	0.95	1.21
Our model	0.24	0.40	0.61	0.68	0.21	0.37	0.57	0.69	<b>0.23</b>	<b>0.44</b>	0.89	<b>0.88</b>	0.26	0.64	0.93	1.00

### 3.4 Generating Data for Supervised Learning

The datasets do not have any annotations, which means that in order to perform supervised learning target data has to be generated. For the prediction task this is done by taking 60 frames from a motion and then splitting it into 50 past and 10 future frames. The past frames are the features, and the future frames are the targets.

For the augmentation task, target data is generated by taking 60 frames from a motion and then splitting it into 50 past and 10 future frames. Noise is then applied to both the past and future frames. The past and future frames with added noise are the input features, and the future frames without noise are the target outputs. The noise defined as

$$N = B + I + L_1 L_2, \quad (2)$$

and consists of three kinds of noise. Let  $\mathcal{N}$  denote the normal distribution and  $\mathcal{B}$  the Bernoulli distribution. The bias

$$B \sim \mathcal{N}(0, \theta_B^2), \quad (3)$$

$$\theta_B \sim \mathcal{N}(\mu_B, \sigma_B^2), \quad (4)$$

represents that joint rotations captured through webcam pose detection models usually have a constant bias, depending on the subject captured. Imprecision noise

$$I \sim \mathcal{N}(0, \theta_I^2), \quad (5)$$

$$\theta_I \sim \mathcal{N}(\mu_I, \sigma_I^2), \quad (6)$$

represents small differences from the ground truth that occurs in the joint rotations captured through webcam pose detection models. Lost tracking noise with

$$L_1 \sim \mathcal{B}(p), \quad (7)$$

$$L_2 \sim \mathcal{N}(0, \theta_L^2), \quad (8)$$

$$\theta_L \sim \mathcal{N}(\mu_L, \sigma_L^2), \quad (9)$$

represents that sometimes a joint is not recognized, giving completely arbitrary values for that joint rotation. This noise consists of the probability that a given is frame suffers from lost tracking  $L_1$ , and the noise applied if the frame does suffer from lost tracking  $L_2$ . Then if  $q$  denotes the motions of the dataset, the input features  $F$  are then defined as

$$F_{m,t,j,a} = q_{m,t,j,a} + N \quad (10)$$

where  $m$  is the human motion,  $t$  is the frame,  $j$  is the joint, and  $a$  is the axis.

## 4 EXPERIMENTS

### 4.1 Prediction Model

The prediction model is evaluated using the same loss function as used by Quaternet (Pavlo et al., 2019a), which is defined as

$$L_{\text{mae}} = \frac{1}{T} \sum_{t,j} \|(\Phi(\hat{X}_{t,j}) - \Phi(X_{t,j}) + \pi) \bmod 2\pi - \pi\|_1, \quad (11)$$

where  $\hat{X}$  is the predicted sequence, and  $X$  is the ground truth,  $T$  is the sequence length,  $J$  is the number of joints in the skeleton, and  $\Phi$  is a function converting quaternions into Euler angles. The model is trained and evaluated on the Human3.6M dataset, of which the evaluation results can be seen in Table 1.

### 4.2 Augmentation Model

The model is trained and evaluated on the CMU dataset with data generated according to subsection 3.4, with  $\mu_B = \mu_I = 0.005$ ,  $\sigma_B = \sigma_I = 0.002$ ,  $\mu_L = 1$ ,  $\sigma_L = 0.01$ , and  $p = 0.01$ . Samples are only drawn from  $\theta$  once per motion, to ensure that each motion has unique distributions of noise, so that the model learns to remove general noise, and is not tied

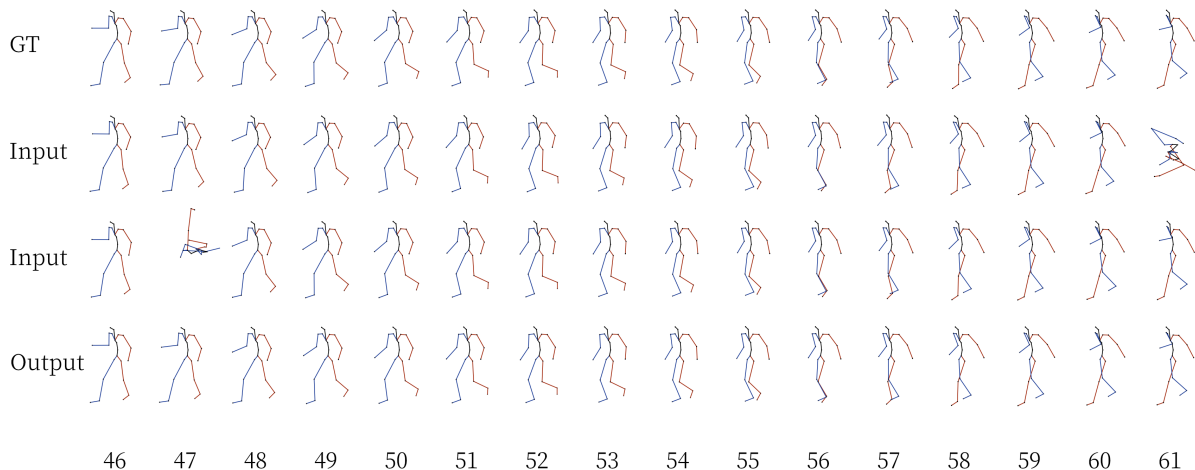


Figure 3: Visualization showing 15 frames of subject 6, trial 5 from the CMU dataset. The first row shows the ground truth pose. The second and third rows show the input to the model. For each frame the values from the second and third rows are concatenated and input to the model, the output is then displayed on the fourth row. Notice that the model is robust to lost frames, as seen in frame 47 and 61. A video showcasing the results of the model is available at <https://i.imgur.com/gS3Pin8.mp4>.

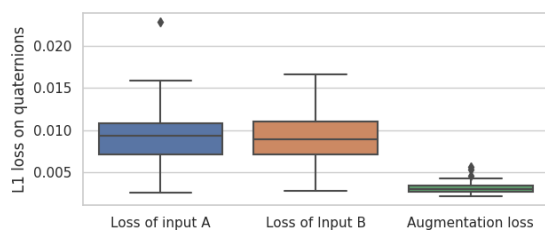


Figure 4: Comparison of the L1 loss of the input data and the augmentation model for the test data of the CMU dataset. Input A and B refers to the two views of the input before concatenation.

to a specific distribution. A summarized result of evaluation on test data can be seen in Figure 4 (c.f. Figure 5). The figure shows the L1 loss on rotations represented as quaternions computed for the entire sequence. These results show that the model is able to greatly reduce the noise, confirming that an LSTM-based model is able to perform noise reduction on human motion. Note that the test data is generated using the same noise function as in training, and that an out-of-bounds annotated dataset would give a clearer picture how well the model would perform in the wild. subsection 4.3 elaborates on why an out-of-bounds dataset is preferable. Several frames of input, output, and ground truth data are visualized in Figure 3.

### 4.3 Suitability of Evaluation Data

To the best of our knowledge, no dataset exists for the task of augmenting or cleaning up skinned human motion. This by extension means that no dataset

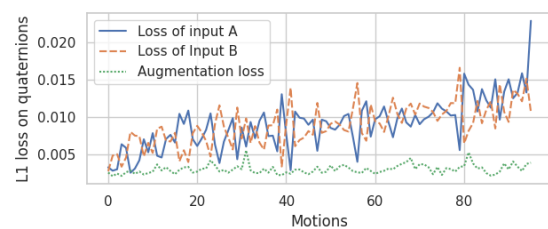


Figure 5: The L1 loss of the augmentation model for the test data of the CMU dataset compared to the loss of the input sequences. The x-axis represents test motions sorted by the average loss across the two input sequences for that motion. Input A and B refers to the two views of the input before concatenation.

exists for augmenting skinned human motion, that originates from pose detection performed on webcam videos using neural networks. Furthermore, creating such a dataset requires deep knowledge of human animation, thus making the creation of such a dataset out of scope for this work.

This is not a problem when training the network, as training data can be generated. It is however a problem when evaluating the network, because neural networks are susceptible to shortcut learning (Geirhos et al., 2020), where they take shortcuts instead of learning the intended generalized solution. For example, a neural network trained to classify objects might erroneously take the background into account, leading to mislabellings.

One way to mitigate shortcut learning is not to evaluate the network on data from the same dataset used for training the network. In other words, it is not

enough to split a single dataset into training, validation and testing, as this makes the test set independent and identically distributed (i.i.d.) with regards to the training set. Instead, one ought to use one or several datasets for testing and evaluation, that have systematic differences from the training dataset, making them out-of-distribution (o.o.d.).

Now, as mentioned previously this is unfeasible, which means that the final model evaluation is susceptible to shortcut learning. The evaluation data is related to the training data in two ways. First off, the evaluation data comes from the same dataset making it i.i.d. with respect to the motions it contains. Secondly, the source of the noise used to generate the input motions is not the actual noise introduced, when going through a webcam-based pose estimation pipeline, but instead the same noise estimation used as when training the network.

As such, the validation data used represent the best possible effort, given the limited data availability for this task. However, should datasets of skinned human motion augmentation become generally available, it would then be desirable to re-evaluate the model on those o.o.d. datasets.

## 5 CONCLUSION

An LSTM-based prediction model is constructed and shown to be competitive with prior work on the task of predicting human motion. The same approach is then used to train an augmentation model, that is capable of cleaning up and merging two noisy motions into a single motion. This shows that an LSTM-based architecture is viable for augmenting human motions, when evaluated on generated data. The lack of annotated data to evaluate on, means that it is unclear how the model performs on real life data. Overcoming this limitation and implementing various potential improvements is a topic for future work.

## REFERENCES

Carnegie Mellon University (2003). CMU MoCap Dataset.  
 Catalin Ionescu, Fuxin Li, C. S. (2011). Latent structured models for human pose estimation. In *International Conference on Computer Vision*.  
 Chiu, H., Adeli, E., Wang, B., Huang, D.-A., and Niebles, J. C. (2018). Action-agnostic human pose forecasting.  
 Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. (2020). Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Gopalakrishnan, A., Mali, A., Kifer, D., Giles, C. L., and Ororbia, A. G. (2019). A neural temporal model for human motion prediction.  
 Gopinath, D. and Won, J. (2020). fairmotion - tools to load, process and visualize motion capture data. Github.  
 Harvey, F. G., Yurick, M., Nowrouzezahrai, D., and Pal, C. (2020). Robust motion in-betweening. *ACM Transactions on Graphics*, 39(4).  
 He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition.  
 Holden, D. (2018). Robust solving of optical motion capture data by denoising. *ACM Trans. Graph.*, 37(4).  
 Holden, D., Kanoun, O., Perepichka, M., and Popa, T. (2020). Learned motion matching. *ACM Trans. Graph.*, 39(4).  
 Holden, D., Komura, T., and Saito, J. (2017). Phase-functioned neural networks for character control. *ACM Trans. Graph.*, 36(4).  
 Ionescu, C., Papava, D., Olaru, V., and 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*.  
 Joo, H., Neverova, N., and Vedaldi, A. (2020). Exemplar fine-tuning for 3d human model fitting towards in-the-wild 3d human pose estimation.  
 Ling, H. Y., Zinno, F., Cheng, G., and Van De Panne, M. (2020). Character controllers using motion vaes. *ACM Trans. Graph.*, 39(4).  
 Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., and Black, M. J. (2015). SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16.  
 Mahmood, N., Ghorbani, N., Troje, N. F., Pons-Moll, G., and Black, M. J. (2019). AMASS: Archive of motion capture as surface shapes. In *International Conference on Computer Vision*, pages 5442–5451.  
 Mao, X., Li, Q., Xie, H., Lau, R. Y. K., Wang, Z., and Smolley, S. P. (2017). Least squares generative adversarial networks.  
 Martinez, J., Black, M. J., and Romero, J. (2017). On human motion prediction using recurrent neural networks.  
 Osman, A. A. A., Bolkart, T., and Black, M. J. (2020). STAR: A sparse trained articulated human body regressor. In *European Conference on Computer Vision (ECCV)*, pages 598–613.  
 Pavlakos, G., Choutas, V., Ghorbani, N., Bolkart, T., Osman, A. A. A., Tzionas, D., and Black, M. J. (2019). Expressive body capture: 3D hands, face, and body from a single image. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 10975–10985.  
 Pavllo, D., Feichtenhofer, C., Auli, M., and Grangier, D. (2019a). Modeling human motion with quaternion-based neural networks. *International Journal of Computer Vision*, 128(4):855–872.  
 Pavllo, D., Feichtenhofer, C., Grangier, D., and Auli, M. (2019b). 3d human pose estimation in video with temporal convolutions and semi-supervised training. In

*Conference on Computer Vision and Pattern Recognition (CVPR).*

- Romero, J., Tzionas, D., and Black, M. J. (2017). Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6).
- Rong, Y., Shiratori, T., and Joo, H. (2020). Frankmocap: Fast monocular 3d hand and body motion capture by regression and integration.
- Shi, M., Aberman, K., Aristidou, A., Komura, T., Lischinski, D., Cohen-Or, D., and Chen, B. (2020). Motionet: 3d human motion reconstruction from monocular video with skeleton consistency.
- Starke, S., Zhang, H., Komura, T., and Saito, J. (2019). Neural state machine for character-scene interactions. *ACM Trans. Graph.*, 38(6).
- Starke, S., Zhao, Y., Komura, T., and Zaman, K. (2020). Local motion phases for learning multi-contact character movements. *ACM Trans. Graph.*, 39(4).
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need.
- Zhang, H., Starke, S., Komura, T., and Saito, J. (2018). Mode-adaptive neural networks for quadruped motion control. *ACM Trans. Graph.*, 37(4).