NeMF: Neural Motion Fields for Kinematic Animation – Supplementary Material –

Chengan He
Yale University
chengan.he@yale.edu

Jun Saito
Adobe Research
jsaito@adobe.com

James Zachary
Adobe Research
zachary@adobe.com

Holly Rushmeier
Yale University
holly.rushmeier@yale.edu

Yi Zhou
Adobe Research
yizho@adobe.com

Contents

1 Method Details **1**
1.1 Global Motion Predictor 1
1.2 Network Architecture 2

2 Experiment Details **3**
2.1 Datasets 3
2.2 Training & Optimization 4
2.3 Motion Reconstruction & Synthesis 4
2.4 Qualitative Metrics 4
2.5 Baselines 5

3 Additional Results **5**
3.1 Motion Reconstruction 5
3.2 Motion Composition 6
3.3 Latent Space Interpolation 6
3.4 Time Translation in Latent Space 6

1 Method Details

1.1 Global Motion Predictor

Given the fact that the character's global translation is conditioned on its local poses, similar to [14, 33], we design a fully convolutional network to generate the global translation $\mathbf{r}$ of the root joint based on the local joint positions, velocities, rotations, and angular velocities as inputs. To eliminate ambiguities in the output, instead of generating the root position directly, we try to predict its velocity

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

$\dot{\mathbf{r}}$, which can be integrated using the forward Euler method to compute $\mathbf{r}$:

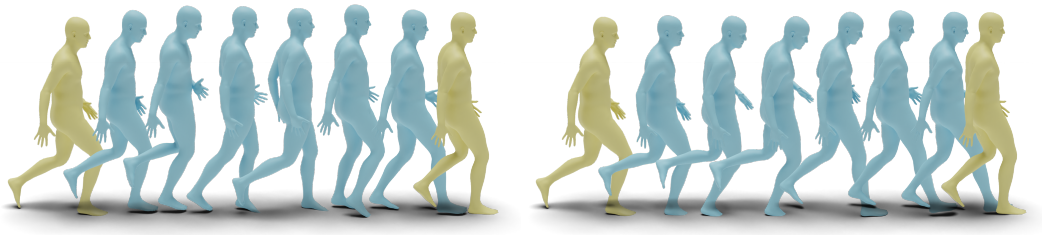
$$\mathbf{r}_{t+1} = \mathbf{r}_t + \dot{\mathbf{r}}_t \Delta t = \mathbf{r}_1 + \sum_{i=1}^{t} \dot{\mathbf{r}}_i \Delta t. \tag{1}$$

However, cumulative errors are inevitable during the integration process, and this becomes more pronounced in the upward direction, where the character gradually moves into the air or under the ground. To avoid this phenomenon, we directly predict the height $\mathbf{r}^h$ of the root joint, which is reasonable since it lies in a region bounded by the height of the character. Then, we measure the differences on the generated velocities and integrated positions as the loss function to minimize:

$$\mathcal{L} = \mathcal{L}_{\text{vel}} + \mathcal{L}_{\text{trans}} \tag{2}$$

$$\mathcal{L}_{\text{vel}} = \sum_{t=1}^{T} \|\dot{\mathbf{r}}_t - \hat{\dot{\mathbf{r}}}_t\|_1, \quad \mathcal{L}_{\text{trans}} = \sum_{t=1}^{T} \|\mathbf{r}_t - \hat{\mathbf{r}}_t\|_1. \tag{3}$$

A Note on An Alternative Integrated Model. A simpler design choice is to predict the global translation, orientation and local motion all at once, and we call it an *integrated model*. However, in our experiments, we observed that this integrated model cannot fully decouple local and global motion, which is more evident when applied in tasks like motion in-betweening. As shown in Figure 1 and our supplemental video, the integrated model tends to generate static local poses with global translation for motion in the interval, thus causing sliding artifacts. While in the separate model, it doesn't have this artifact since this model explicitly decouples local and global motion.



Separate Model Integrated Model

Figure 1: Comparison between separate model and integrated model on motion in-betweening. The two yellow poses are ground truth and the cyan poses are generated results in the interval.

A Note on Our Single-Motion NeMF Model. When training our single-motion NeMF model, we train the MLP to fit both the local and global motion. Therefore, its loss function contains terms both from Equation 5 of our main paper and Equation 2 above. While for our generative model, we observe some artifacts as illustrated in Figure 1 and choose to train a standalone global motion predictor to handle the global motion. Therefore, we drop the loss terms related to global motion when training our VAE.

1.2 Network Architecture

Motion Encoders. We introduce two separate motion encoders to parameterize the latent space of local motion and root orientation respectively. To encode local motion, we adopt the Skeleton Convolution and Skeleton Pooling layers proposed in [1] to build a residual block with PReLU activations [11] and group normalization [31]. The motion encoder contains 4 layers of these skeleton convolution residual blocks with kernel size 4 to extract latent features from local pose parameters, which are followed by 2 fully-connected layers to obtain the mean and variance of $\boldsymbol{z}_l$ in 1024 dimensions. As for root orientation, its residual block is built on 1D convolution and 1D average pooling layers with PReLU activations, and its encoder also contains 4 layers of residual blocks with kernel size 4 to gradually upscale the root orientation in $\mathbb{R}^6$ to latent features in 128, 256, 512, and 512 dimensions. The fully-connected layers then map the latent features to the mean and variance of $\boldsymbol{z}_g$ in 256 dimensions.

MLP Decoder. Similar to [22], we build an MLP to predict local pose parameters and root orientation based on latent variables and temporal coordinates with positional encoding. The MLP contains 11 fully-connected layers with ReLU activations and layer normalization [3]. Each hidden layer has the output size 1024, while a skip connection is introduced in each layer of the MLP to emphasize the importance of the input and to help prevent posterior collapse [16].

Global Motion Predictor. Our global motion predictor has a similar architecture as our motion encoder, where 3 layers of skeleton convolution residual blocks with kernel size 15 are applied to to extract latent features from the local pose parameters. These latent features are then mapped to root velocity and root height with 4 additional residual blocks composed of 1D convolution and 1D average pooling layers with kernel size 15.

2 Experiment Details

2.1 Datasets

We train our model on the AMASS dataset[1] [19] for human motion, which is a motion capture database that aggregates mocap data from multiple datasets and normalizes them into a uniform format. We then leverage the data processing scripts provided by HuMoR [25] to filter some outlier data and unify their frame rates to 30 fps. Here, we plot the duration distribution of the processed AMASS data in Figure 2 and collect some detailed statistics in Table 1.

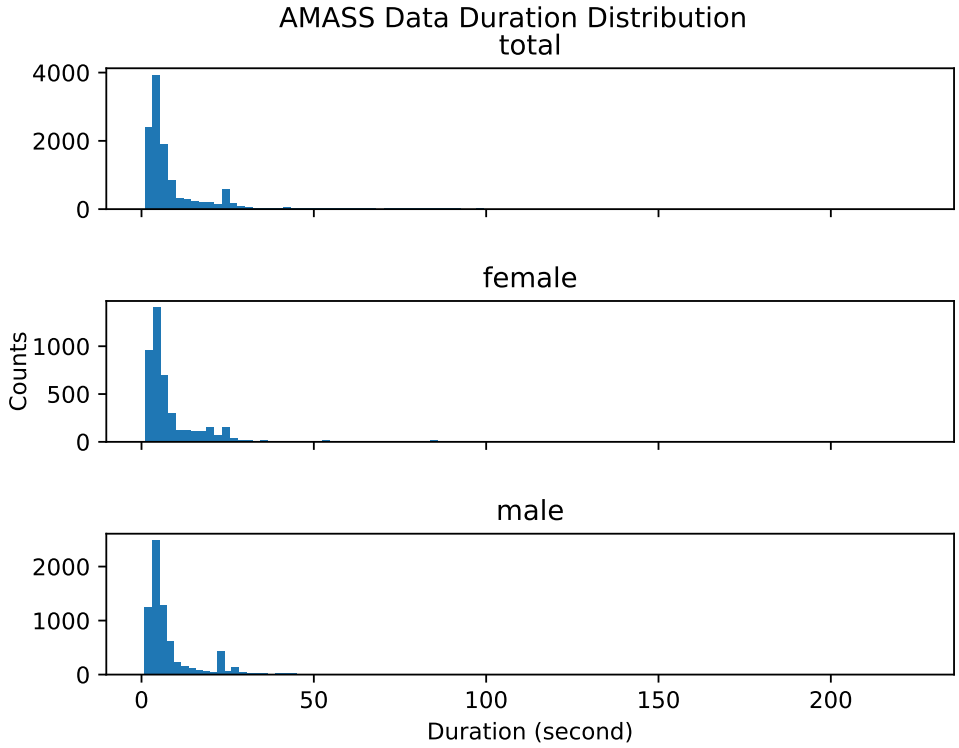


Figure 2: Duration distribution of AMASS data.

From the statistics we collect, the duration of AMASS data has a large variance while most of them are between 0 and $50s$. To be more specific, only more than 50.98% of the data are longer than $5s$. Therefore, to fully utilize the AMASS data, we choose to chop the sequences into clips with 128 frames ($4.3s$) and set the batch size to be 16 throughout experiments. Then we split these processed

[1]For the license of AMASS, please check: https://amass.is.tue.mpg.de/license.html.

data into training, validation and testing sets, where the training set contains data from CMU [29], MPI Limits [2], TotalCapture [28], Eyes Japan [18], KIT [20], BMLrub [27], BMLmovi [8], EKUT [20], ACCAD [6], BMLhandball [12], DanceDB [5], DFaust [4], and SSM [19], the validation set contains data from MPI HDM05 [23], SFU [30], and MPI Mosh [17], and the testing set contains data from HumanEva [26] and Transitions [19]. This split results in $11,642$ sequences in the training set, $1,668$ sequences in the validation set and 164 sequences in the testing set, roughly 20 hours in total for use.

We additionally train our model for the reconstruction experiments on a quadruped motion dataset [32], which contains 30 minutes of dog motion capture. Similar to [13], we manually filter those clips on uneven terrain and the remaining data are all in 60 fps with various lengths from 155 to $13,399$ frames.

Table 1: Detailed statistics collected from the AMASS data.

AMASS Data Statistics	
Total motion sequences	$11,831$
Total Duration	$119,661.40s$
Minimal Duration	$0.97s$
Maximal Duration	$224.57s$
Average Duration	$10.11s$
Sequences longer than $5s$	$6,031$ (50.98%)
Sequences longer than $10s$	$2,739$ (23.15%)
Male Data	$7,400$ (Duration: $73,989.10s$)
Female Data	$4,431$ (Duration: $45,672.30s$)

2.2 Training & Optimization

We train our model and conduct all the experiments on a cluster with 8 Intel® Xeon® Gold 6136 CPUs @ 3.00GHz, 64GB memory, and 2 NVIDIA Tesla V100 GPUs. Our code is implemented with Python 3.9.7 and PyTorch 1.9.0.

Training. We employ Adam optimizer throughout the training for all NeMF architectures with the learning rate of 0.0001. We train our single-motion NeMF for 500 iterations to fit a 32-frame sequence, and scale the number of iterations proportionally as the sequence length increases to make sure that our model is sufficiently trained for each length of sequences. As for our generative NeMF and global motion predictor, we train their architectures for $1,000$ epochs with weight decay 0.0001.

Test-Time Optimization. Our test-time optimization utilizes Adam with the initial learning rate of 0.1. In all experiments, our method converges within 600 iterations with proper initialization, and we decay the learning rate to 0.07 and 0.049 at iteration 200 and 400, respectively.

Hyperparameters. In all of our experiments, we set the weights λ_{rot} to 1.0, λ_{ori} to 1.0, and λ_{pos} to 10.0. In training our generative NeMF, we initially set λ_{KL} to $1e^{-5}$. To combat posterior collapse, we adopt the cyclical annealing schedule [7] to linearly anneal λ_{KL} from $1e^{-7}$ to its full value every 50 epochs. For the energy functions formulated during test-time optimization, we set the weights λ_{trans} to 1.0, λ_{sim} to 0.5, λ_{traj} to 1.0, and λ_{angle} to 1.0.

2.3 Motion Reconstruction & Synthesis

In the ablation study and comparison, we evaluate both the reconstruction and synthesis capability of our generative NeMF. For motion reconstruction, we use the trained network to directly infer the 164 samples in the testing set and compute some deterministic metrics to measure the reconstruction errors. For motion synthesis, we generate 400 samples through latent space sampling and introduce three additional metrics to measure the quality of motion.

2.4 Qualitative Metrics

In our experiments, we employ the following three metrics to measure the motion characteristics that reconstruction errors cannot capture, namely Fréchet Inception Distance (**FID**), diversity (**Diversity**) and foot skating (**FS**).

Fréchet Inception Distance (FID). FID is a statistical metric which has been widely used for measuring the image quality, while Guo et al. [9] and Petrovich et al. [24] have transferred it to the motion domain and employ it in tasks such as action recognition. To evaluate FID, we use a

pre-trained feature extractor to extract motion features from real and generated motions, then the FID is computed from the distribution of these feature vectors.

Diversity. Diversity was first introduced by Guo et al. [9] to measure the variance of generated motions. To evaluate diversity, we randomly split all generated data into two subsets with equal size. Feature vectors are then extracted from them respectively, and diversity is computed as the mean Euclidean distance between these feature vectors.

Foot Skating (FS). To measure the foot skating artifact, we use the metric proposed in [16, 32]. To be specific, this metric measures the accumulated drift on the ankle and toe joints when their height h is within a certain threshold H. Their velocity is first projected onto the horizontal plane to compute the magnitude v, which is further weighted with the formula $s = v(2 - 2^{\frac{h}{H}})$. In our experiments, we set H according to the values provided by HuMoR [25], which are $4cm$ for toe joints and $8cm$ for ankle joints. This parameter setting leads to an average foot skate of $0.512cm$ per frame in the ground truth data.

To build the feature extractor for FID and diversity, we train an auto-encoder that maps the input motion parameters to feature vectors. The auto-encoder has a similar architecture as our VAE, except that it takes both local and global motion as input and the fully-connected layer outputs latent vectors directly instead of their mean and variance. Similar to [9, 24], we randomly pick 100 samples from the generated data to evaluate these plausible metrics in each iteration, and perform 20 iterations with different random seeds. We then report the mean value of these metrics in our tables.

2.5 Baselines

HM-VAE [14]. We use the pre-trained HM-VAE model released by the authors. The model was trained on the AMASS dataset with a sequence length of 64. To accommodate longer sequences, we use the concatenation method in their open-source code[2] to connect each sub-sequence.

HuMoR [25]. We use the pre-trained HuMoR model released by the authors[3].

Robust Motion In-betweening (RMI) [10]. We choose an open-source implementation of RMI[4] since Ubisoft does not release their official code. We modified this unofficial implementation and trained it on the AMASS dataset to fit our experiment setup.

3 Additional Results

3.1 Motion Reconstruction

In Figure 3 we show our reconstruction result on the quadruped motion. This sequence contains $4,336$ frames at 60 fps ($73s$), which takes $8,000$ iterations to converge. From the visualization in Figure 3 and our supplemental video, our predicted motion is almost identical to the ground truth.

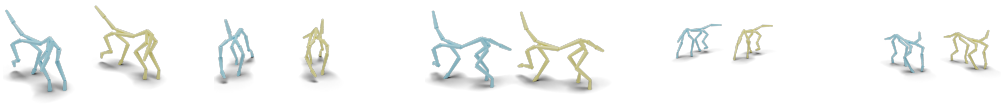
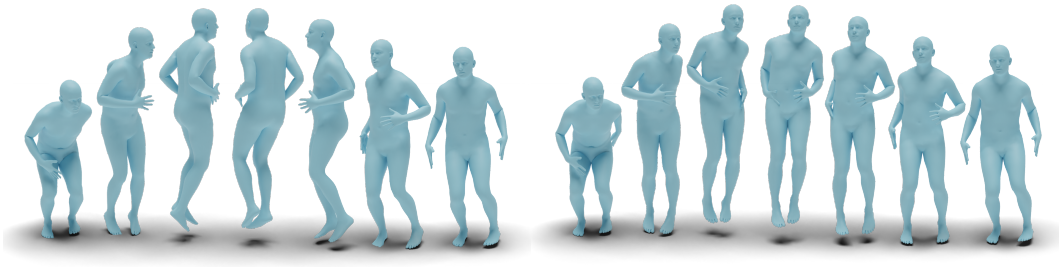


Figure 3: Quadruped motions generated from single-motion NeMF, where the cyan skeleton indicates our generated result and the yellow skeleton is the ground truth motion.

[2] https://github.com/lijiaman/hm-vae
[3] https://github.com/davrempe/humor
[4] https://github.com/xjwxjw/Pytorch-Robust-Motion-In-betweening

3.2 Motion Composition

Since we disentangle the latent space for local motion and root orientation, we can create interesting motion editing results as in Figure 4, where we cancel the spinning motion of a pirouette jump by assigning different $\boldsymbol{z}_g$ to the same $\boldsymbol{z}_l$.



$\boldsymbol{z}_l(\text{jumping}) + \boldsymbol{z}_g(\text{spinning})$ $\boldsymbol{z}_l(\text{jumping}) + \boldsymbol{z}_g(\text{no spinning})$

Figure 4: Orientation editing by assigning different $\boldsymbol{z}_g$ to the same $\boldsymbol{z}_l$.

3.3 Latent Space Interpolation

To examine the smoothness of the latent space and see whether our model can blend different styles of motion at the sequence level, we linearly interpolate $\boldsymbol{z}$ from two existing motion sequences and infer the novel ones as shown in Figure 5.

Figure 5: Latent space interpolation.

3.4 Time Translation in Latent Space

As a motion prior, different motion clips will be mapped to different positions in the latent space. Therefore, it would be interesting to examine the latent patterns formed by those clips which share a large portion of overlap while containing some temporal offsets.

We then set up an experiment by first picking 3 different sequences from AIST++ [15], each containing about 200 to 300 frames. For each sequence, we use a sliding window with an offset of 10 to obtain clips with 118-frame overlap, and then encode these clips with our encoder. The latent variables are projected to 3D with UMAP [21] and visualized in Figure 6. In the cases we test, the clips with overlapping frames are mapped to nearby positions and even form some interesting patterns, thus suggesting that they maintain certain connection in the latent space.

References

[1] Kfir Aberman, Peizhuo Li, Sorkine-Hornung Olga, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Skeleton-aware networks for deep motion retargeting. *ACM Transactions on Graphics (TOG)*, 39(4):62, 2020. 2

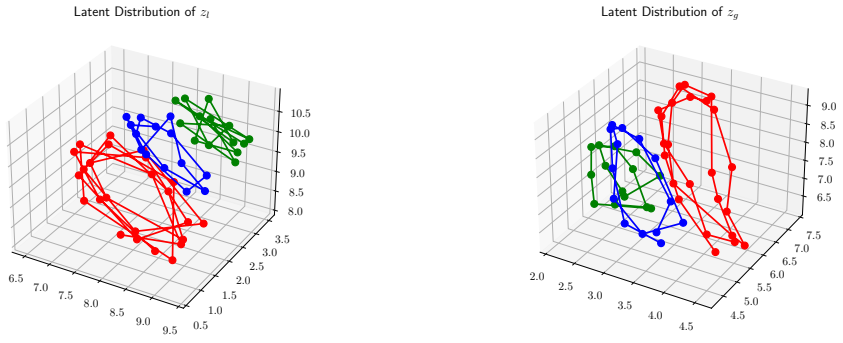


Figure 6: Latent distribution of clips with time translation. Different colors refer to different AIST++ sequences.

[2] Ijaz Akhter and Michael J. Black. Pose-conditioned joint angle limits for 3D human pose reconstruction. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2015*, June 2015. 4

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 3

[4] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST: Registering human bodies in motion. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 4

[5] DanceDB. Dance motion capture database. 4

[6] Advanced Computing Center for the Arts and Design. Accad mocap dataset. 4

[7] Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. Cyclical annealing schedule: A simple approach to mitigating kl vanishing. *arXiv preprint arXiv:1903.10145*, 2019. 4

[8] Saeed Ghorbani, Kimia Mahdaviani, Anne Thaler, Konrad Kording, Douglas James Cook, Gunnar Blohm, and Nikolaus F Troje. Movi: A large multipurpose motion and video dataset. *arXiv preprint arXiv:2003.01888*, 2020. 4

[9] Chuan Guo, Xinxin Zuo, Sen Wang, Shihao Zou, Qingyao Sun, Annan Deng, Minglun Gong, and Li Cheng. Action2motion: Conditioned generation of 3d human motions. In *Proceedings of the 28th ACM International Conference on Multimedia (MM '20)*, 2020. 4, 5

[10] Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. Robust motion in-betweening. *ACM Trans. Graph.*, 39(4), jul 2020. 5

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015. 2

[12] Fabian Helm, Nikolaus F Troje, and Jörn Munzert. Motion database of disguised and non-disguised team handball penalty throws by novice and expert performers. *Data in brief*, 15:981–986, 2017. 4

[13] Gustav Eje Henter, Simon Alexanderson, and Jonas Beskow. Moglow: Probabilistic and controllable motion synthesis using normalising flows. *ACM Trans. Graph.*, 39(6), nov 2020. 4

[14] Jiaman Li, Ruben Villegas, Duygu Ceylan, Jimei Yang, Zhengfei Kuang, Hao Li, and Yajie Zhao. Task-generic hierarchical human motion prior using vaes. 2021. 1, 5

[15] Ruilong Li, Shan Yang, David A. Ross, and Angjoo Kanazawa. Ai choreographer: Music conditioned 3d dance generation with aist++, 2021. 6

[16] Hung Yu Ling, Fabio Zinno, George Cheng, and Michiel Van De Panne. Character controllers using motion VAEs. *ACM Trans. Graph.*, 39(4):40:1–40:12, July 2020. 3, 5

[17] Matthew M. Loper, Naureen Mahmood, and Michael J. Black. MoSh: Motion and shape capture from sparse markers. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 33(6):220:1–220:13, November 2014. 4

[18] Eyes JAPAN Co. Ltd. Eyes japan mocap dataset. 4

[19] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *International Conference on Computer Vision*, pages 5442–5451, October 2019. 3, 4

[20] Christian Mandery, Ömer Terlemez, Martin Do, Nikolaus Vahrenkamp, and Tamim Asfour. The kit whole-body human motion database. In *International Conference on Advanced Robotics (ICAR)*, pages 329–336, 2015. 4

[21] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29):861, 2018. 6

[22] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 3

[23] Meinard Müller, Tido Röder, Michael Clausen, Bernhard Eberhardt, Björn Krüger, and Andreas Weber. Documentation mocap database hdm05. 2007. 4

[24] Mathis Petrovich, Michael J. Black, and Gül Varol. Action-conditioned 3D human motion synthesis with transformer VAE. In *International Conference on Computer Vision (ICCV)*, pages 10985–10995, October 2021. 4, 5

[25] Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J. Guibas. Humor: 3d human motion model for robust pose estimation. In *International Conference on Computer Vision (ICCV)*, 2021. 3, 5

[26] L. Sigal, A. Balan, and M. J. Black. HumanEva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal of Computer Vision*, 87(1):4–27, March 2010. 4

[27] Nikolaus F Troje. Decomposing biological motion: A framework for analysis and synthesis of human gait patterns. *Journal of vision*, 2(5):2–2, 2002. 4

[28] Matt Trumble, Andrew Gilbert, Charles Malleson, Adrian Hilton, and John Collomosse. Total capture: 3d human pose estimation fusing video and inertial sensors. In *2017 British Machine Vision Conference (BMVC)*, 2017. 4

[29] Carnegie Mellon University. Cmu graphics lab motion capture database. 4

[30] Simon Fraser University and National University of Singapore. Sfu motion capture database. 4

[31] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 2

[32] He Zhang, Sebastian Starke, Taku Komura, and Jun Saito. Mode-adaptive neural networks for quadruped motion control. *ACM Trans. Graph.*, 37(4), jul 2018. 4, 5

[33] Yi Zhou, Jingwan Lu, Connelly Barnes, Jimei Yang, Sitao Xiang, et al. Generative tweening: Long-term inbetweening of 3d human motions. *arXiv preprint arXiv:2005.08891*, 2020. 1