Predicting Loose-Fitting Garment Deformations Using Bone Driven Motion Networks
——— Supplementary Document ———

1 DATASET

We create a garment animation dataset consisting of three different types of garments, which are driven by the same avatar. To drive the character, we collect 40,000 frames from the dance motions of Miku Miku Dance (MMD) videos [Wikipedia 2021], which are downloaded from YouTube [YouTube 2021] and BiliBili [bil 2021]. Compared to mocap data, e.g., AMASS [Mahmood et al. 2019], the motions from MMD dancing videos are manually edited to be quicker and have wider ranges, which produce more aesthetically pleasing and dynamic simulation results. We split the dancing motions into 500-frame-length segments and obtain 80 segments. We use 75 segments for training and 5 segments for testing.

We use the Houdini Vellum simulation framework [SideFX 2021] to generate ground truth simulation data for loose-fitting garments. Vellum is a simulation framework that uses an extended position-based dynamics approach. To generate a dataset with different simulation parameters, we choose three simulation parameters [SideFX 2021] that may largely affect the simulation results: the cloth's bending stiffness (ranging from $1e-4$ to $1e-9$), density (ranging from 0.1 to 0.004), and the timescale of the simulator (ranging from 0.5 to 1.5). We randomly sample 10 sets of parameters, where 8 sets are used for training and 2 sets for testing. The other simulation parameters are set as constant among all sets of simulation parameters, where stretching stiffness is set to $1e10$, normal drag is set to 60, edge length scale is set to 0.25, and the number of substeps in the simulator is set to 3.

After we get the simulation results, we remove the garments' global transformations and then smooth the meshes and perform skin decomposition. We use Laplacian smoothing to decompose garment deformations to low- and high-frequency parts. We empirically set the smoothing strength to 0.1 and the number of iterations to 10. For skin decomposition, we adopt Smooth Skinning Decomposition with Rigid Bones (SSDR [Le and Deng 2012]). In our work, we mainly change the number of bones and keep the rest of the hyper-parameters fixed, where we set the number of maximum global iterations to 100, the number of clustering update iterations in initialization to 10, the number of weights update iterations per global iteration to 3, and the number of non-zero weights per vertex to 8.

We have also tested our method on the datasets of Santesteban *et al.* [Santesteban et al. 2019] and TailorNet [Patel et al. 2020]. For the dataset of Santesteban *et al.* [Santesteban et al. 2019], we use the simulation result of a T-shirt and a skirt of medium body shape ($\boldsymbol{\beta} = 0$). For the T-shirt/skirt, we split 52/49 clips for training and 4/4 clips for testing. Since the garments closely follow the body, we found the optimal numbers of virtual bones are less than those for garments used in our dataset, and we use 20 bones for T-shirt and skirt. For the dataset of TailorNet [Patel et al. 2020], we use the simulation result for female pants of a medium body shape and medium style ($\boldsymbol{\beta} = 0, \boldsymbol{\gamma} = 0$), and use 16 clips for training and 2 clips for testing. We use 10 virtual bones for the pants. The quantitative results are not comparable, as we only use one shape-style pair and split training/testing sets by clips rather than by frames to fit with the training of GRU. We note that the comparison on the dataset of TailorNet is not really fair, as the dataset is simulated without dynamic effects, while our method is designed for garments with dynamics.

2 NEURAL NETWORKS

We use PyTorch [Paszke et al. 2019] and PyTorch Geometric [Fey and Lenssen 2019] for network implementation, and train the networks on an NVIDIA GeForce RTX 2080Ti GPU. All the networks are optimized using RMSProp optimizer [Tieleman and Hinton 2012].

For the low-frequency module in our motion network, we use a single GRU layer of size 600 followed by a linear layer with *ReLU* activation and 0.5 dropout. To reduce the memory consumption and speed up the training process, we adopt Truncated Back Propagation Through Time (TBPTT [Williams and Peng 1990]) with 50 time steps. We train the network with batches of size 8 and learning rate of $1e-3$, which decays by 70% every 30 epochs. The laplacian weight in the loss is set to 1.

The high-frequency module consists of a global stream and a local stream. The global stream is implemented by a single GRU layer of size 600 followed by a linear layer of size 10, and the local stream is implemented by three stacked EdgeConv layers, where the dimensions of features of a vertex are 3, 8 and 16, respectively. The local and global features of each vertex are concatenated and processed by a shared weight MLP. The network is trained with batches of size 5 and learning rate of $3e-4$, which decays by 70% every 10 epochs. We also adopt TBPTT in training this module with 10 time steps. The weight of the collision term in the loss is set to 0.1.

For the RBF kernel, we use an MLP to project the simulation parameters to the latent space, which has 3 fully connected layers with feature dimensions of [3, 6, 10]. We train the network using learning rate $1e-2$ and batches of size 4. The bandwidth of the RBF kernel $2\sigma^2$ is 1.0 in our work.

3 IMPLEMENTATIONS OF COMPARISON METHODS

In order to compare with prior methods, we use the available source code and adjust them to fit our data: TailorNet [Patel et al. 2020] and DNG [Zhang et al. 2021]. For Santesteban *et al.* [Santesteban et al. 2019], which only released the code of inference, we referred to their code and implemented the training part. For the method described in [Chen et al. 2021], there is no public source code, so we re-implemented it using the hyper-parameters provided in that paper.

During experiments, we found that some of the hyper-parameters provided by the comparison methods did not match our dataset, causing the results to be much lower quality on specific poses compared with results in the original paper. For a fair comparison, we optimized some of the comparison methods' hyper-parameters in the experiments of our dataset. The changes include:

- For Santesteban *et al.* [Santesteban et al. 2019], we use the learning rate 1e-3 which decays by 70% every 30 epochs, and add a Laplacian term to the loss function to avoid jittering on meshes when avatars have large or swift motions. The network trained with new parameters generates results with better visual quality on specific poses and slightly higher quantitative results, compared with the network trained with hyper-params in the paper.
- For DNG [Zhang et al. 2021], we use the learning rate 1e-3, which decays by 70% every 30 epochs when training the *Joint2Coarse* network. The quantitative result is similar to results generated by the network trained with parameters in the original paper, and the meshes have higher quality on specific poses.

4 ADDITIONAL EVALUATIONS

4.1 Additional Results

In Fig. 1, we present a side-by-side comparison of our result and the ground truth, along with RMSE maps, where the garments have complex deformations. The largest errors are mainly on the bottoms of the garments, as they are the most loose-fitting parts, and it is difficult for the networks to learn their deformations. Even with these errors, our estimated deformations are visually plausible, which demonstrates that our approach can predict high-quality loose garment animations. Please refer to supplementary videos for qualitative comparisons of our method with competitive methods [Patel et al. 2020; Santesteban et al. 2019; Zhang et al. 2021], where our results are significantly closer to the ground truth than other methods.

4.2 Additional Comparisons

In Table 1, we provide a quantitative comparison of the overall performance of our motion network with other methods. The errors follow a similar trend as those in the low-frequency module, where the RMSE of our method is about 20% lower and the Hausdorff distance and STED are about 10% lower than the best of them.

In Fig. 2, we depict RMSE curves of a motion sequence of the low- and high-frequency modules in the motion network and other competitive methods. On the top row, the sequence contains large motions on frames from 80 to 220, which causes the garments to have large-scale deformations, where our method has the lowest RMSE because the low-frequency module uses rigid transformation of bones as deformation representations. On the bottom row, our method tends to have the lowest RMSE as the high-frequency module leverages both the global information of virtual bones' motions and local information on low-frequency meshes.

In Table 2. we provide a quantitative comparison on garments from public datasets. These garments are relatively tighter than garments in our dataset, which means they deform closely following the body and do not have complex garment dynamics. Our method generates comparable results. Specifically, for the T-shirt and pants, TailorNet [Patel et al. 2020] has the best performance; for the skirt, our method has better results. The deformations of these garments closely follows the body, which can be more efficiently captured by body bones than virtual bones.

Table 1. Comparison to other methods on the overall performance of the motion network.

		TailorNet [Patel et al. 2020]	*DNG [Zhang et al. 2021]*	*[Santesteban et al.]*	*Ours*
Dress1	RMSE ↓	33.73	24.94	22.67	**17.38**
	Hausdorff ↓	92.16	81.99	80.29	**69.07**
	STED ↓	0.09892	0.09678	0.08929	**0.07863**
Dress2	RMSE ↓	41.59	37.13	31.92	**27.10**
	Hausdorff ↓	133.26	120.57	111.36	**98.93**
	STED ↓	0.09296	0.11071	0.07860	**0.07328**
Dress3	RMSE ↓	36.71	25.65	27.05	**21.93**
	Hausdorff ↓	91.87	75.24	80.53	**62.30**
	STED ↓	0.10629	0.09770	0.09393	**0.08073**

Table 2. Comparison to other methods on the overall performance of the motion network on the dataset of Santesteban *et al.* [Santesteban et al. 2019] and TailorNet [Patel et al. 2020]

		TailorNet [Patel et al. 2020]	[Santesteban *et al.*]	Ours
T-shirt	RMSE ↓	**9.90**	10.25	10.52
	Hausdorff ↓	**27.02**	29.56	31.51
	STED ↓	**0.04183**	0.04490	0.04528
Skirt	RMSE ↓	22.95	20.96	**19.91**
	Hausdorff ↓	**76.80**	87.01	83.39
	STED ↓	0.07578	0.07452	**0.07221**
Pants	RMSE ↓	**4.84**	4.91	5.08
	Hausdorff ↓	**14.46**	14.87	18.75
	STED ↓	**0.01279**	0.01294	0.01665

4.3 Performance

We list the mesh statistics and run time performances in Table 3. A single motion network takes about 2.5 ms to infer a garment consisting of about 10K vertices and 20K faces, reaching 400 fps. For simulation parameter variation, we use 8 motion networks; thus the overall time is eight times the time of the motion network with the inference time of the RBF kernel. Our approach can obtain an interactive frame rate (about 40 fps on a computer with two NVIDIA GeForce RTX 2080Ti GPUs).

Fig. 1. Additional results and corresponding RMSE maps, where the garments have complex deformations. Ours are close to the ground truth with RMSEs on most vertices lower than 8cm.

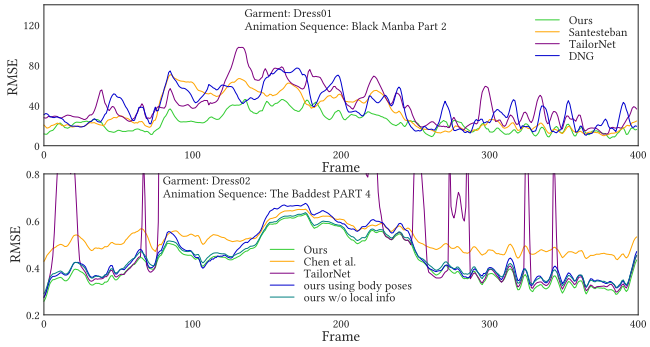

Fig. 2. Quantitative comparison of low- and high-frequency modules to different methods (top and bottom). Our method achieves lower errors than competitive methods.

Table 3. Performance breakdown for running time (ms).

	Vert	Face	Motion Network	Overall
Dress1	10822	21183	2.59	23.72
Dress2	12273	24140	2.83	25.64
Dress3	8744	17236	2.47	22.76
T-shirt	4424	8710	2.36	-
Skirt	12146	23949	2.84	-
Pant	4718	9283	2.43	-

4.4 Failure Cases

Our method may fail when the avatar has some extreme poses where severe penetrations may arise, as shown in Fig. 3. It is an interesting direction to avoid collisions on such poses.

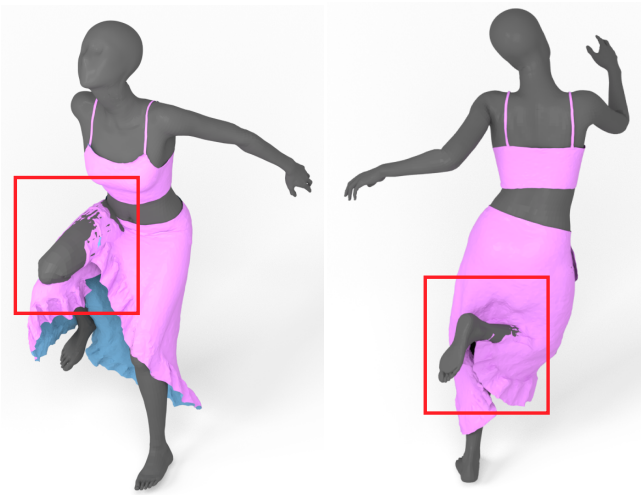

Fig. 3. Our method may fail on some extreme poses where severe penetrations may arise.

REFERENCES

2021. bilibili. https://www.bilibili.com/
Lan Chen, Lin Gao, Jie Yang, Shibiao Xu, Juntao Ye, Xiaopeng Zhang, and Yu-Kun Lai. 2021. Deep Deformation Detail Synthesis for Thin Shell Models. *CoRR* abs/2102.11541 (2021).
Matthias Fey and Jan E. Lenssen. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.
Binh Le and Zhigang Deng. 2012. Smooth Skinning Decomposition with Rigid Bones. *ACM Transactions on Graphics (TOG)* 31, 6 (2012).
Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. 2019. AMASS: Archive of Motion Capture as Surface Shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 5442–5451.
Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan

Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Chaitanya Patel, Zhouyingcheng Liao, and Gerard Pons-Moll. 2020. TailorNet: Predicting Clothing in 3D as a Function of Human Pose, Shape and Garment Style. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 7363–7373.

Igor Santesteban, Miguel A. Otaduy, and Dan Casas. 2019. Learning-Based Animation of Clothing for Virtual Try-On. *Computer Graphics Forum (CGF)* 38, 2 (2019), 355–366.

SideFX. 2021. *Houdini Vellum*.

T. Tieleman and G. Hinton. 2012. Lecture 6.5—RmsProp: Divide the Gradient by a Running Average of its Recent Magnitude. COURSERA: Neural Networks for Machine Learning.

Wikipedia. 2021. MikuMikuDance, Wikipedia. https://en.wikipedia.org/wiki/MikuMikuDance [Online; accessed 2021-12-13].

Ronald J. Williams and Jing Peng. 1990. An Efficient Gradient-Based Algorithm for On-Line Training of Recurrent Network Trajectories. *Neural Computation* 2, 4 (1990), 490–501.

YouTube. 2021. YouTube. https://www.youtube.com/ [Online; accessed 2021-12-13].

Meng Zhang, Tuanfeng Y. Wang, Duygu Ceylan, and Niloy J. Mitra. 2021. Dynamic Neural Garments. *ACM Transactions on Graphics (TOG)* 40, 6, Article 235 (2021), 15 pages.