Self-Supervised Collision Handling via Generative 3D Garment Models for Virtual Try-On

— Supplementary Document —

Igor Santesteban[1] Nils Thuerey[2] Miguel A. Otaduy[1] Dan Casas[1]

[1]Universidad Rey Juan Carlos, Spain

[2]Technical University of Munich, Germany

first.last@{urjc.es}{tum.de}

In this document we provide additional details, including dataset description, training parameters, and further evaluation, of our paper "Self-Supervised Collision Handling via Generative 3D Garment Models for Virtual Try-On".

1. Dataset

1.1. Physical Simulation

In this work we use physical simulation to generate the ground-truth data to train and validate our model. Specifically, we use ARCSim [6], an open-source cloth simulator.

We use a fixed timestep of 3.33ms in all our simulations, and store the results from 1 out of every 10 time steps, to match the frame rate of the character animations. We use a diverse set of 56 animations from the CMU Motion Capture Database in the AMASS dataset, which cover a wide range of motions, and use the rest of sequences of the dataset for the tests shown in the video.

We train our model using 17 different body shapes β. Specifically, we generate 4 samples for each of the first 4 principal components of the shape space in SMPL, leaving the rest of β parameters in 0. Additionally, we use the average shape $\beta = \mathbf{0}$.

To initialize the simulations with a valid cloth configuration we always start at t-pose and progressively interpolate towards the initial pose and body shape. Once we reach the initial state, we simulate the garment until it reaches equilibrium, and then we run the animation.

For garment materials, we use polyester knit fabrics from the set of measured materials [11]. Specifically, we use the `gray-interlock` material for the t-shirt and the `navy-sparkle-sweat` material for the dress. In both cases we set the damping coefficient to 0.01.

1.2. Projection to Canonical Space

Once we have the simulation data, we run the optimization described in Section 4.3 to obtain the projection in the canonical space. To this end, we solve Equation (9) with a L-BFGS method and gradients computed with automatic differentiation in TensorFlow [1]. The weights of the energy function are set to $\omega_1 = 1\text{e}{-4}$, $\omega_2 = 1\text{e}{-2}$ for all meshes.

2. Neural Networks

We use TensorFlow [1] to implement all our networks. All networks use Adam optimizer [3].

2.1. Diffused Human Model

To generate the training data for the different components of our diffused human model (*e.g.,* blendshape displacements, rigging weights), we exhaustively sample the SMPL model as follows. We sample 10,000 points, and we use libigl [2] to find the closest point in the body surface. Then, from this closest point, we compute ground truth value (*e.g.*, distance, blendshape displacement, skinning weights, depending on the network we are training) using barycentric interpolation. For each epoch, we dynamically update the set of samples. Since the garment can move far from the body surface, we sample points uniformingly to have similar accuracy in all the 3D space. All terms of our diffused human model are trained with neural networks with these parameters: initial learning-rate 1e-4, batch size 256.

Signed Distance Network. The network that approximates the distance field of the body template follows the architecture of SIREN [10], which uses $sine$ activation functions. The network consists of 5 fully-connected hidden lay-

ers with 256 neurons each. We train the network by minimizing the L1 error between the predicted distance and the ground-truth distance.

Blendshape Networks. For the networks that approximate the body blendshapes for any arbitrary 3D points, we use a fully-connected network of 5 hidden layers with 256 neurons. We use ReLU activations. We also tried to use SIREN for these networks, but had better results using the architecture of DeepSDF [7].

Skinning Weight Network. Same as blendshape networks, but the last activation is Softmax (this makes the output sum 1 always 1, which is necessary for the skinning weights). Instead of minimizing the L1 norm of the error, we minimize the KL divergence, as done in the work of Liu *et al.* [5],

2.2. Deformation Subspace

The generative space of garment deformations is obtained by using a Variational Autoencoder (VAE) [4]. Both the encoder and the decoder share the same architecture: 4 dense layers of size 2000 with $ReLU$ activations, and layer normalization after each activation. To train the autoencoder we first train it during 100 epochs with the learning rate set to $1e{-}4$ and the weights set to $\lambda_1 = 100$, $\lambda_2 = 0$ and $\lambda_3 = 1$ (*i.e.*, we don't optimize collisions during this step). Then we lower the learning rate to $1e{-}5$ and we progressively raise the KL and collision loss until we reach the final weights $\lambda_1 = 100$, $\lambda_2 = 10$ and $\lambda_3 = 1$, which are maintained without change until the network converges.

2.3. Recurrent Regressor

We implement our recurrent regressor $R()$ with 2 GRU layers of size 500. For the recurrent steps we use a $sigmoid$ activation whereas for the final output of each layer we apply the $tanh$ function. We train the network with an initial learning rate of $1e{-}4$ and batches of size 8, and we set the weights of the velocity and acceleration losses to $\rho_1 = 0.5$ and $\rho_2 = 0.1$, respectively.

3. Additional Evaluation

3.1. Performance

To further validate the advantage of our model with respect to existing methods that apply a postprocess step to fix the problematic vertices, we compare the runtime performance of both strategies. As we show in Table 1 of this document, the cost of evaluating the networks required by our approach (*i.e.*, the networks of the diffused human model) is significantly lower than the cost of postprocessing required by [8] and [9].

	Ours	[9]	[8]
T-shirt (8,710 triangles)	**1.4 ms**	3.0 ms	210 ms
Dress (23,949 triangles)	**2.9 ms**	6.9 ms	698 ms

Table 1: Evaluation time of our networks required to account for body-garment collisions (*i.e.*, diffused body model to unproject vertices to posed state) *vs.* postprocessing time for [8] and [9] using original authors implementation.

3.2. Unposing Evaluation

A fundamental step of our approach is the novel optimization-based strategy to unpose ground-truth simulated garments described in Section 4.3 of the main document. Here we provide additional quantitative evaluation of this step, and we compare it to state-of-the-art methods that are based on fixed rigging weights [9, 8], and to a baseline strategy that uses a rigging weight assignment based on the per-frame nearest vertex between the simulated garment and underlying body. To this end, we provide two metrics: the average triangle strain of the unposed meshes, which measures the overall distortion with respect to the template mesh; and the number of collisions in the unposed mesh. Our strategy leads to significantly lower errors in both metrics, indicating that unposed meshes have significantly fewer collisions and do not suffer from undesired distortions. This is qualitative shown in Figure 3 of the main document, and in the supplementary video.

	Constant weights [9, 8]	Nearest vertex	Ours
Strain	5.5	190.8	**0.07**
Num collisions	218.6	48.6	**12.1**

Table 2: Strain and average number of collisions in rest pose.

3.3. Quantitative Comparison

Finally, in Table 3 we quantitatively evaluate our method and compare it to the approach of Santesteban *et al.* [9], which is the only existing work that also models dynamics as a function of body shape and motion. To this end, we use 5 test sequences and 17 shapes, totaling 12,155 frames, and we compute error metrics based on per-vertex Euclidean error, average triangle strain, and average number of collisions. Our method is on par with the per-vertex surface error of [9], while we significantly reduce the number of collisions.

	Santesteban [9]	Ours
Error (cm)	**2.9**	3.1
Strain	**0.006**	0.007
Num collisions	80.0	**1.3**

Table 3: Quantitative evaluation of our approach in 5 test sequences and 17 body shapes.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 1

[2] Alec Jacobson, Daniele Panozzo, et al. libigl: A simple C++ geometry processing library, 2018. https://libigl.github.io/. 1

[3] Diederick P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 1

[4] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[5] Lijuan Liu, Youyi Zheng, Di Tang, Yi Yuan, Changjie Fan, and Kun Zhou. Neuroskinning: Automatic skin binding for production characters with deep graph networks. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 38(4), July 2019. 2

[6] Rahul Narain, Armin Samii, and James F O'brien. Adaptive Anisotropic Remeshing for Cloth Simulation. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 31(6):1–10, 2012. 1

[7] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[8] Chaitanya Patel, Zhouyingcheng Liao, and Gerard Pons-Moll. The Virtual Tailor: Predicting Clothing in 3D as a Function of Human Pose, Shape and Garment Style. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[9] Igor Santesteban, Miguel A. Otaduy, and Dan Casas. Learning-Based Animation of Clothing for Virtual Try-On. *Computer Graphics Forum (Proc. Eurographics)*, 38(2), 2019. 2, 3

[10] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit Neural Representations with Periodic Activation Functions. In *Proc. NeurIPS*, 2020. 1

[11] Huamin Wang, James F O'Brien, and Ravi Ramamoorthi. Data-Driven Elastic Models for Cloth: Modeling and Measurement. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 30(4):1–12, 2011. 1