DeePSD: Automatic Deep Skinning And Pose Space Deformation For 3D Garment Animation
SUPPLEMENTARY MATERIAL

Hugo Bertiche[1,2], Meysam Madadi[1,2], Emilio Tylson[1] and Sergio Escalera[1,2]
[1]Universitat de Barcelona and [2]Computer Vision Center, Spain
hugo_bertiche@hotmail.com

In this document we include further discussions and analyses on our methodology. First, we show proof-of-concept computer vision related applications using *DeePSD*. Then, we show some insights on skinning and blend weights. Next, we provide of additional details of our proposed network architecture and training. Later, we discuss model performance in train and test. Finally, we describe the video material submitted along this document.

1. Computer Vision Applications

We believe one of the strongest real applications for *DeePSD* is virtual try-ons. Demand for such application is already present and companies in the private sector are pushing towards this direction. We propose a proof-of-concept application. We use an off-the-shelf CNN to regress SMPL parameters from images [4]. SMPL parameter predictions are combined with template outfits to generate a 3D model of the subject with the desired outfit. Fig.1 shows the results of this experiment (note that some of these outfits do not appear in CLOTH3D dataset). In practice, for virtual try-ons, the methodology to use has to be scalable to unseen and arbitrary outfits. Thus, methodologies limited to individual garments encoded as body homotopies are not suitable [1, 5]. Additionally, we observed how these approaches suffer from texturing artifacts, which further compromises their applicability. Finally, to bring this technology to everyone (portable devices), a model with low computational complexity and memory footprint is necessary. As discussed in the main paper, *DeePSD* has a size of only 4.4MB and generates animated 3D models, which are extremely efficient and compatible with all graphics engines.

We also present an additional proof-of-concept computer vision application for 3D draped human regression from video using *DeePSD*. Similar to the virtual try-on application, we use a CNN[4] to regress SMPL pose. Then, to obtain the outfit from the video sequence, we propose the following approach. First, we use *DeePSD* to compute outfit global descriptors (as defined in the main paper) for each outfit in the training set. Then, we train a VGG-16 to regress outfit global descriptors from static frames. Next, in test, we obtain an estimation of the outfit global descriptor for each frame of a given sequence. We retrieve the 5 nearest neighbours from the training set based on global descriptor Euclidean distance for each frame. We retrieve the outfit with the most appearances. In case of a tie, we choose the one with the lowest Euclidean distance. Finally, once an outfit is retrieved, we combine it with the estimated SMPL pose using *DeePSD* to obtain a final 3D draped human prediction. Fig.2 shows some samples obtained using this approach. Note that, since template outfit is aligned with the body in canonical pose, we retrieve SMPL shape parameters and gender along with the outfit. As it can be seen, these predictions benefit from the physical consistency of *DeePSD* predictions.

2. Blend Weights

We analyze the blend weights distribution for trouser-like garments (body homotopies) and skirt-like garments. Fig. 3 shows the blend weights distribution for SMPL and two different garments (jumpsuit and dress). Note that we do not apply direct supervision to blend weights. Thus, the network predicts blend weights to minimize L2 loss only on final garment vertices. We can observe how the distribution of blend weights is very similar to those of SMPL. This supports the assumption that cloth closely follows body motion, since the network is able to learn by itself this distribution in order to minimize Euclidean error. Nonetheless, we observe a significant difference for skirts, where the predominant blend weights are the ones corresponding to the SMPL root joint. Skirts break the aforementioned assumption and the network learns to avoid relying on body motion by assigning skirts to root joint instead of leg joints.

In the main paper we assume it is known that garment animation on top of a human body cannot be done with skinning alone. First, since cloth behaviour is highly non-linear, skinning is not able to model it properly. Secondly, since

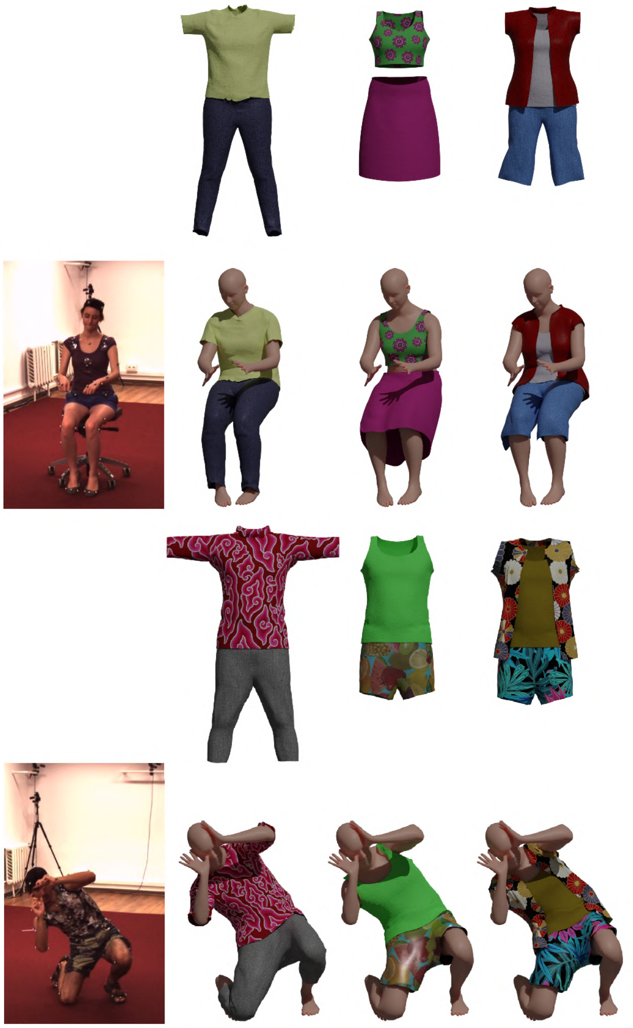

Figure 1: Virtual try-on. Combining DeePSD with a body pose and shape recovery CNN we obtain an effectively working virtual try-on. With the obtained SMPL parameters and a digital wardrobe (3D outfits in canonical pose) it is possible to generate draped 3D models in the corresponding pose. Images extracted from Human3.6M[3, 2]

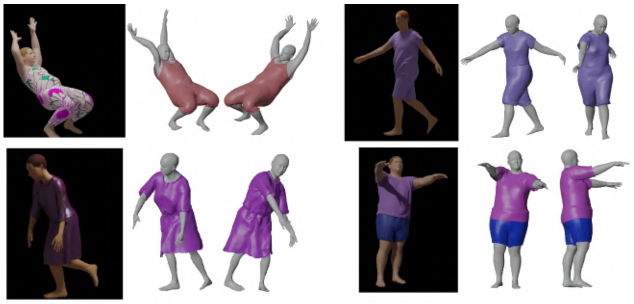

Figure 2: Proof-of-concept computer vision application. We combine an outfit retrieval approach with SMPL parameter regression to obtain 3D predictions of draped humans.

	Euclidean error	Edge	Bend	Collision
Data	36.73	1.16	0.015	15.4%
Phys	37.67	1.17	0.013	12.9%

Table 1: Results without PSD. First row corresponds to training with L2 loss on PBS data only. Second row is trained with L2 combined with $\mathcal{L}_{phys}$. We observe how performance is greatly compromised without PSD.

SMPL itself has PSD, it would be impossible to achieve physically consistent predictions without PSD as well in the outfit. Nonetheless, to further prove this, we perform two additional experiments with no PSD (no $\mathbf{D}_{data}$ nor $\mathbf{D}_{phys}$). Tab. 1 shows the obtained results. In the first experiment we train only with L2 loss against PBS data. On the second one, we apply both, L2 and $\mathcal{L}_{phys}$ losses. As it can be seen, the capacity of the model is highly compromised and, thus, Euclidean error is much higher than *DeePSD*. Then, we observe that applying physical consistency loss has almost no effect, yielding a very high number of collisions. Nonetheless, since templates already have cloth consistency (edge and bend), we see how linear transformations (skinning) are able to maintain such constraints.

3. Network Architecture and Training

In this section with further detail network architecture and training process. Note that code is provided along with supplementary material to ease reproducibility and understanding of the model. In the main paper we describe the input as $\mathbf{T} \in \mathbb{R}^{N\times 3}$. We empirically observed an slight increase in performance by concatenating vertex normals to each vertex. Therefore, the final input is defined as $\mathbf{T} \in \mathbb{R}^{N\times 6}$. Then, Φ is defined as 4 layers of graph convolutions with a perceptive field of $K = 1$ each and dimensionalities 32, 64, 128 and 256. Then, Φ also includes a fully-connected layer to compute global descriptor with dimensionality 256. Global descriptor is concatenated with each vertex local descriptor to form per-vertex feature vectors with $F = 512$. Then, Ω, Ψ and χ are composed of 4 fully-connected layers each, applied to vertices (vertices are *samples*). Then, Ω has dimensionalities 128, 64, 32 and 24. Ψ and χ have a similar architecture with dimensionalities 256, 256, 256 and $P \times 3$ (where $P = 128$ is the dimensionality of the high-level pose embedding Θ). Finally, the MLP used for obtaining Θ from θ is composed of 4 fully-connected layers with 256 dimensions each, except the output layer, with dimensionality $P = 128$. We empirically observed that normalizing Θ as $\bar{\Theta}_i = \Theta_i / \sum_k \Theta_k$ heavily increases training stability.

We implemented our model using TensorFlow. We train each model for 10 epochs. Starting with a batch size of 4 and doubling it every 2 epochs. We use Adam optimizer

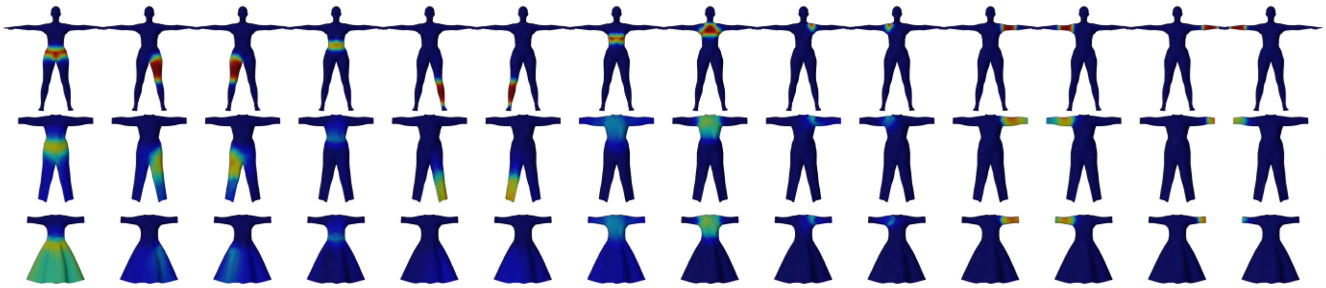

Figure 3: Blend weights comparison after training. First row: SMPL. Middle row: trouser-like. Last row: skirt-like.

with an initial learning rate of 0.001. As stated in the main paper, it is not useful to train χ before the rest of the model has achieved convergence. Thus, we train χ independently during 10 epochs after training the rest of the model (Φ, Ω and Ψ). Regarding Ω without direct supervision, it will converge to the weights presented in Sec. 2 of this document. Nonetheless, we empirically observed that it is not always guaranteed, most likely due to sensitivity to initialization, batch order or data bias. Thus, during the first epoch only, we apply a L2 loss based on a prior distribution. This prior relies again on the assumption that cloth follows the body. Then, Ω uses the nearest body vertex (in canonical pose) blend weights as labels. This ensures a correct initialization and slightly speeds up convergence. We further complement this prior with a prior on deformations. We assume that deformations are small. Again, for the first epoch only, we apply an L2 regularization on PSD.

4. Performance

We train our model in a GTX1080Ti for 10 epochs, taking around 8-16 hours (unsupervised losses are computationally expensive). Our model has a size of 4.4Mb. For deployment, we run the model once per outfit to obtain animated 3D models in standard format. This format is highly computationally efficient due to exhaustive optimization of current graphics engines and GPUs. The only extra computational cost is obtaining the high-level pose embedding. Since an MLP is the most basic deep-learning model, it is unlikely to find a more efficient deep-based approach for 3D animation. In practice, we get 3-6ms per sample and around 0.1ms for batched samples, depending on vertex count, within TensorFlow pipeline. Proper integration into commercial graphics engine might further increase efficiency.

5. Video

We complement the supplementary material with a video. In this video we briefly describe the main characteristics of our methodology and discuss its benefits. Additionally, we show qualitative results in unseen pose sequences. Note how our approach is able to keep temporal consistency despise not being trained specifically for it.

References

[1] Hugo Bertiche, Meysam Madadi, and Sergio Escalera. Cloth3d: Clothed 3d humans. In *European Conference on Computer Vision*, pages 344–359. Springer, 2020.

[2] Cristian Sminchisescu Catalin Ionescu, Fuxin Li. Latent structured models for human pose estimation. In *International Conference on Computer Vision*, 2011.

[3] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. 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):1325–1339, jul 2014.

[4] Meysam Madadi, Hugo Bertiche, and Sergio Escalera. Smplr: Deep learning based smpl reverse for 3d human pose and shape recovery. *Pattern Recognition*, page 107472, 2020.

[5] Chaitanya Patel, Zhouyingcheng Liao, and Gerard Pons-Moll. 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*, pages 7365–7375, 2020.