Supplementary Material for GenSim: Unsupervised Generic Garment Simulator

Lokender Tiwari Brojeshwar Bhowmick Sanjana Sinha
TCS Research, India
{lokender.tiwari, b.bhowmick, sanjana.sinha}@tcs.com

This supplementary material contains the following:

1. **Implementation Details:** We present $GenSim$'s implementation details in Sec. S1, with the detailed architecture of the PEN module, and additional training details in the Sec. S2.

2. **Ablation and Discussion:** We show ablation and discuss our design choices in Sec. S3.

3. We show additional results and comparison with PBNS in Sec. S4. More results on unseen garments and some large-scale results from the main paper are shown in Fig. S6. Large-scale textured results on the unseen poses from YouTube video frames, unseen garment types, and unseen body shapes are shown in Fig. S7 Fig. S8.

4. A discussion on generalization capability of $GenSim$ along with other SOTA methods is described in Sec. S5

S1. Implementation Details

PEN Architecture: We show the overall architecture of all the learnable blocks of our Physics Enforcing Network (PEN) in Fig. S1. The geometry encoders are PointNet++ [7] blocks that learn per vertex geometric features. The remaining blocks are implemented using MLPs. The number of neurons in each of the MLP's written below their respective linear layers. The dimension of node and edge features is 64. The learnable update functions $\mathcal{N}_{edge}$ and $\mathcal{N}_{node}$ are implemented using two-layer MLPs. The displacement decoder module takes the final updated node features and passes them to two separate sub-decoder MLPs, where one sub-decoder MLP output the magnitude of displacement w. The other sub-decoder MLP is a direction decoder, it produces two coefficients b^1, b^2 of the vector b and the convex combination factor ρ. The coefficients and the ρ are used to construct the direction of displacement $\hat{\mathcal{D}}$.

Pin vertices are computed automatically by finding the vertices closest to the pre-defined garment type aware body vertices where the garment must be pinned.

Mass vertex: we assign the mass of each vertex proportional to the face area. We found this along with the fabric type input is sufficient to capture the fabric-aware mass information.

Motion-aware ARAP Garment Deformation: Our motion-aware ARAP module doesn't have any trainable module. It is an optimization process that can be easily implemented. We take the reference from [1] for our ARAP implementation. We set $nitr = 10$ and $t = 1.0/nitr$, for all our experiments.

Message Passing Graph Network Module: We use a message passing algorithm (Alg. 1) with the latent node and edge features (see Fig. 2 main paper) where every node aggregates the information/features from its incident edges, process them to update its own feature and broadcast to its neighboring nodes. Similarly, every edge takes the features from its end nodes and updates its own features. We repeat the message-passing for L steps and get the final updated encoded garment graph and pass it to the displacement decoder. The node and the edge update function ($\mathcal{N}_{edge}(\cdot)$, $\mathcal{N}_{node}(\cdot)$) in Alg. 1 are approximated using MLP's, and for the aggregation function $\Omega^{agg}(\cdot)$ we use the average operation.

Fabric specific mean and standard deviation: The estimated μ_{fabric} and σ_{fabric} for the fabric types silk, leather, denim, cotton is plotted in Fig. S2.

Algorithm 1: Update Node and Edge Features
Inputs: π, η, **Outputs:** π', η'

for $(i,j) \in \boldsymbol{E}$ **do**
  $\pi'_{ij} \leftarrow \mathcal{N}_{edge}(\pi_{ij}, \eta_i, \eta_j)$
end for
for $k \in \{1, ..., m\}$ **do**
  $\lambda'_k = \{(\pi'_{ij}, i, j)\}_{i=k,\ \forall (i,j) \in \mathbf{E}}$
  $\bar{\pi}'_k = \Omega^{agg}(\lambda'_k)$
  $\eta'_k = \mathcal{N}_{node}(\bar{\pi}'_k, \eta_k)$
end for

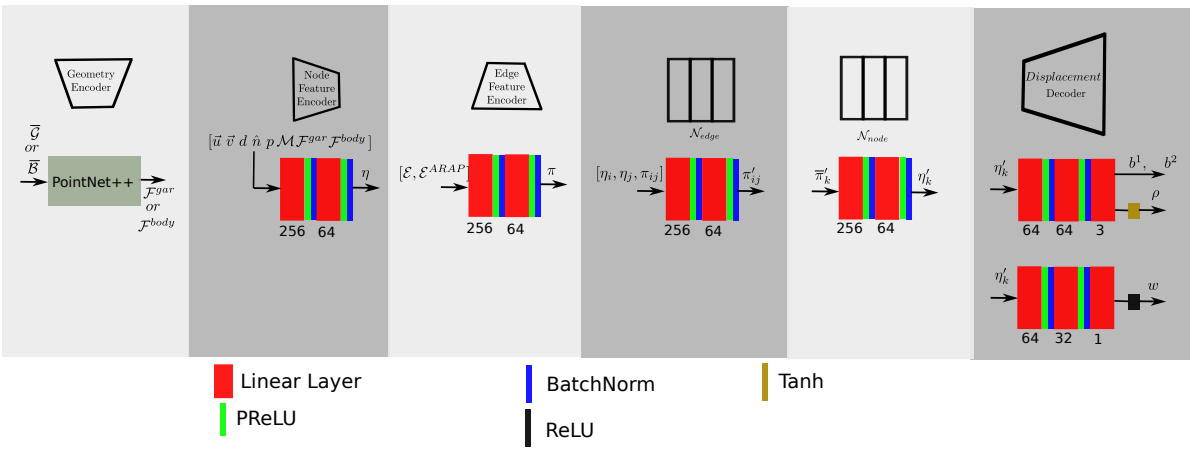

Figure S1. **Physics Enforcing Network:** To encode the geometry of both the garment and the body we use point++ based encoder to get per-vertex geometric features. All other learnable blocks are implemented using the MLPs. The number of neurons in each of the linear layers has been shown below the respective layers. Refer to Sec. S1 for detail.

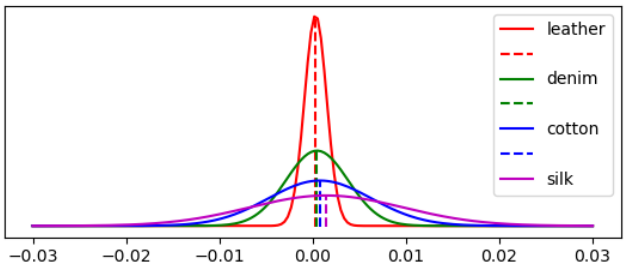

Figure S2. Fabric specific mean μ_{fabric} and standard deviation σ_{fabric} of edge length differences φ. Plotted as a Gaussian distribution. $fabric$ can be leather, denim, cotton, and silk.

S2. Training Details

We train $GenSim$, on full-resolution meshes of the CLOTH3D dataset. We run our experiments on a single $40GB$ A40 Nvidia GPU. We train $GenSim$ using adam optimizer for 30 epochs with an initial learning rate 0.0001 and reduce it by a factor of 10 at 20^{th} epoch. With a batch size of 12, training takes approximately 45 hours. We set $\gamma_1 = \gamma_3 = \gamma_4 = \gamma_6 = 1e2$, $\gamma_2 = \gamma_5 = 1.0$, $\epsilon = -0.5$ and $\epsilon_{coll} = 0.003$. **Note:** though the training time of PBNS is lower than ours, it requires retraining for every new garment or body shape, which is **practically infeasible** for every possible garment-body pair. Our trained model is approximately 1MB. Whereas the size of a PBNS-trained model depends on the number of garment vertices, for a template garment of 12K vertices, the size is 5.8MB. Our model being of the lowest size can be easily ported to low-end devices such as mobiles, tablets, etc.

Training and Inference Time. Total training time is 48 hours. For inferring a 3000 vertices skirt GenSim takes on average $\sim$ 0.9 $secs$ [0.6(ARAP)+ 0.3(PEN)], compared to minutes taken by a physics-based simulator like ArcSim [6]. The number of handle vertices, in this case, is 980, and the average displacement of vertices from the template garment to the ARAP deformed garment is 95mm. The number of iterations is fixed to 10 for all our experiments irrespective of the number of garment vertices and types.

S2.1. Data pre-processing

Since we directly operate on raw meshes, our method doesn't require pre-processing of the garment or the body meshes, such as converting them to any alternate representation like PCA, etc. $GenSim$ can be trained or tested on an arbitrary resolution of body and garment meshes.

S3. Ablation and Discussion

GenSim vs. Physical Based Simulation (PBS). To simulate a template garment on an arbitrary *static body pose*, a typical PBS method would require first manual rough alignment of the template garment on the target body pose, followed by an optimization process that minimizes the overall energy. Without alignment, such a method fails to converge (tested using ArcSim and Marvelous software). GenSim does this in an automated fashion where the rough alignment is done by our novel motion-aware ARAP module and PEN adds the correction to make it physically consistent. The time taken by our ARAP module and PEN network is significantly less than the manual effort and the optimization process. An average 3D artist would take 20-30mins for an average-size long skirt to roughly align on the target pose, while our motion-aware ARAP takes on average $\sim 1sec$ for 7000 vertices skirt.

Direct LBS vs Body Motion Aware Garment Deformation: An alternate approach which could be used to roughly align template garment on the target body pose is using lin-

ear blend skinning [5], that deforms the garment vertices by **directly** borrowing the blend weights of the closest body vertices. However, as we can see in Fig. S3, LBS-based deformation doesn't obey underlying body geometry and significantly elongates the edges (mainly for the loose garments). For body-hugging garments LBS approach may work relatively better.

Direct LBS+PEN: We trained our physics enforcing network along with LBS-based initial garment deformation on the training subset. The results on the test set are Edge ($2.8mm$), Smoothness(0.0010), and Collision (6.7%). LBS in many cases produces significant artifacts, hence requiring significant corrections by PEN. While our Body Motion-Aware ARAP garment deformation approach smoothly align the garment on the target pose without any artifact, requiring corrections to be less significant than LBS. Therefore, Body Motion-aware ARAP+PEN performs better than Direct LBS+PEN.

Losses Justification: The potential energy loss will try to push the garment vertices downward due to gravitational force, while the restrain energy loss will resist it by restricting significant change in the edge lengths from their respective template garment. However, edges will stretch/compress depending on the fabric type, keeping the magnitude of change in edge lengths centered around the fabric mean μ_{fabric} with a standard deviation σ_{fabric}. The pin vertices loss will ensure that the garment does not fall down and remains tightly attached to the pin body area. The bending energy loss and mesh smoothing loss will enforce the local surface smoothness. The collision loss applies a penalty on the predicted vertex displacement if the displaced garment vertex collides with the body surface. All the losses are necessary (refer to supplementary for ablation).

Garment-to-Body Collision Penalty In our formulation (Sec. 3.2.3 main paper), we ensure that the predicted direction of displacement for each vertex will not collide with the local body surface. However, there could be a few cases, where the direction of displacement is in the right direction, but due to the large magnitude predicted by the PEN, it may collide with the other non-local body part. For example, as we have shown in Fig.S4, the direction of displacement of a garment vertex which is close to the thigh area of the body is in the right half-space, but due to the high magnitude of the displacement the displaced garment vertex may collide with the non-local region which is the surface near ankle. To avoid such cases we applied the body-garment collision penalty loss. We observed a few such cases in our experiments. Due to our local body-geometry-aware direction predictions, by construction collisions are majorly avoided, except in such few cases.

S4. Additional Results

S4.1. Comparison with PBNS

We show more comparison results with the closest unsupervised method PBNS in Fig. S5. PBNS trains a model which is specific to an **outfit-body** pair. We train three different PBNS models for each template garment shown in Fig.S5 (top row). $GenSim$ is **trained once** using the tops and skirts of the cloth3D dataset. Unlike, PBNS $GenSim$ doesn't require retraining a new model with every new garment or body variation.

We can observe a squeezing v-$shaped$ artifact between the leg area present in all the predictions made by PBNS. This artifact has also been mentioned by the respective authors in their papers, especially in the skirts. While, they claim this issue is resolved and have visually shown results on a *small template skirt* with a large separation between legs, we observe the artifact is still visible, especially for long skirts. **Note:** we train PBNS using officially released code [2] by the authors, following the protocol mentioned in their paper. The $Tops$ results of both PBNS and $GenSim$ are visually similar, but $GenSim$ clearly outperforms the PBNS in loose skirts. To measure the extent of the v-$shaped$ distortion we compute the smoothing error/loss mentioned in (Eq. 7, main paper), the value for tops and skirts are mentioned below each example. The numbers in **bold** are the best. **Lower value** indicates a smooth surface i.e., no v-shaped or any arbitrary unnatural surface distortions.

S4.2. Additional Evaluation on Unseen Garments

We show additional qualitative colored and textures rendering for unseen garments on random body shapes and poses in Fig. S6.

S5. Generalization Capability of $GenSim$ and other SOTA Methods

PBNS [3]: PBNS is an *unsupervised* method, it trains an **outfit-body pair**, specific model, hence it cannot generalize to garment and body other than it is trained on.

SNUG [9] : SNUG is a *self-supervised* method, it trains a garment template-specific mode, it can generalize to other body shapes and poses, **but only for the garment template it is trained on**.

VTON [8]: VTON is a *supervised* method, it trains a garment template-specific model, which can generalize to other body shapes and poses, but only for the garment template it is trained on.

VTOColl [10]: VTOColl learns garment deformation model in a *supervised* manner, and collision handling in a *self-supervised* manner. But, their training is garment template specific.

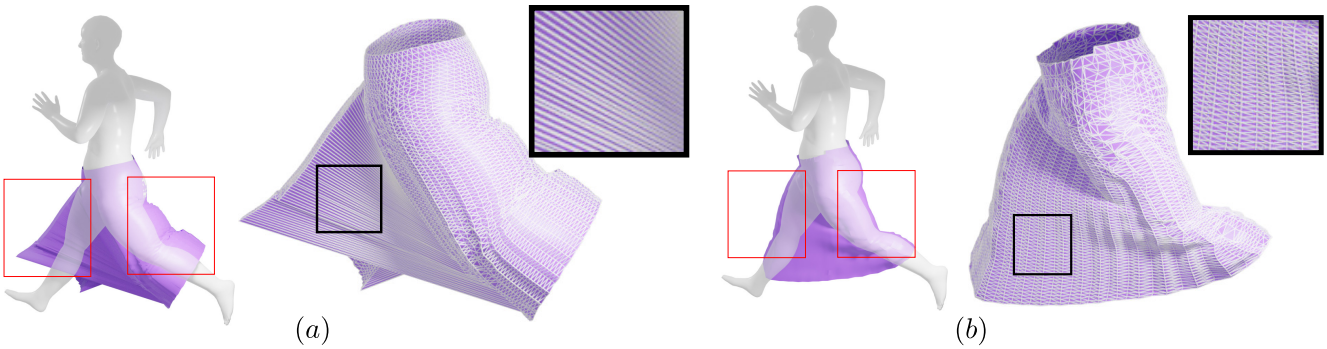

Figure S3. (a) Direct LBS vs. (b) Motion-aware ARAP-based deformation.

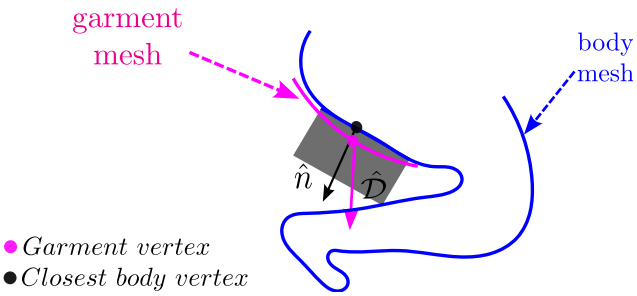

Figure S4. Collision of Garment Vertex with Non-Local Body Surface. Refer to Sec. S3 for detail.

DeePSD [4]: DeePSD learns a garment deformation model in a *supervised* manner using multiple type of garments. This method generalizes to other body shapes and poses, and the types of garments used in the training. The authors have mentioned in their paper the limitation of their method on loose garments such as skirts.

The major limitation of SOTA methods is in their inability to generalize to other garment types (other than training garment templates), or they fail to give physically plausible garment deformation for loose garments. Another limitation is the requirement of ground-truth PBS data. *GenSim* being a completely unsupervised method generalizes to arbitrary garment types (including loose garments), body shapes, and poses. A comparison in Fig. S9 shows our method performs comparatively better than the competing methods (maintaining the fitting of template garments). Since we do not model temporal information either in the network architecture or in the losses, our method can show some inconsistency when the results are viewed in a motion sequence. We believe the ability of *GenSim* to accurately deform arbitrary garments on *arbitrary body shapes and poses* is a significant step for physically plausible garment deformation. Adding temporal consistency can be a future direction to improve *GenSim* for temporal consistent garment deformations.

S6. Code Availability

Due to the policy of our funding agencies, on acceptance of the paper, we can make our trained model available for evaluation by researchers at universities for non-commercial research purposes. The model will be available on an access-controlled cloud environment which can be granted on a request basis. Access to code/models will be restricted to others.

References

[1] libigl - a simple c++ geometry processing library. https://libigl.github.io/. 1

[2] Pbns code repository, howpublished = https://github.com/hbertiche/pbns (accessed on 1/nov/2022),. 3, 5

[3] Hugo Bertiche, Meysam Madadi, and Sergio Escalera. Pbns: physically based neural simulation for unsupervised garment pose space deformation. *ACM Transactions on Graphics (TOG)*, 40(6):1–14, 2021. 3

[4] Hugo Bertiche, Meysam Madadi, Emilio Tylson, and Sergio Escalera. Deepsd: Automatic deep skinning and pose space deformation for 3d garment animation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5471–5480, 2021. 4

[5] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):1–16, 2015. 3

[6] Rahul Narain, Armin Samii, and James F O'brien. Adaptive anisotropic remeshing for cloth simulation. *ACM transactions on graphics (TOG)*, 31(6):1–10, 2012. 2

[7] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 1

[8] Igor Santesteban, Miguel A Otaduy, and Dan Casas. Learning-based animation of clothing for virtual try-on. In *Computer Graphics Forum*, volume 38, pages 355–366. Wiley Online Library, 2019. 3, 7

[9] Igor Santesteban, Miguel A Otaduy, and Dan Casas. Snug: Self-supervised neural dynamic garments. In *Proceedings of*

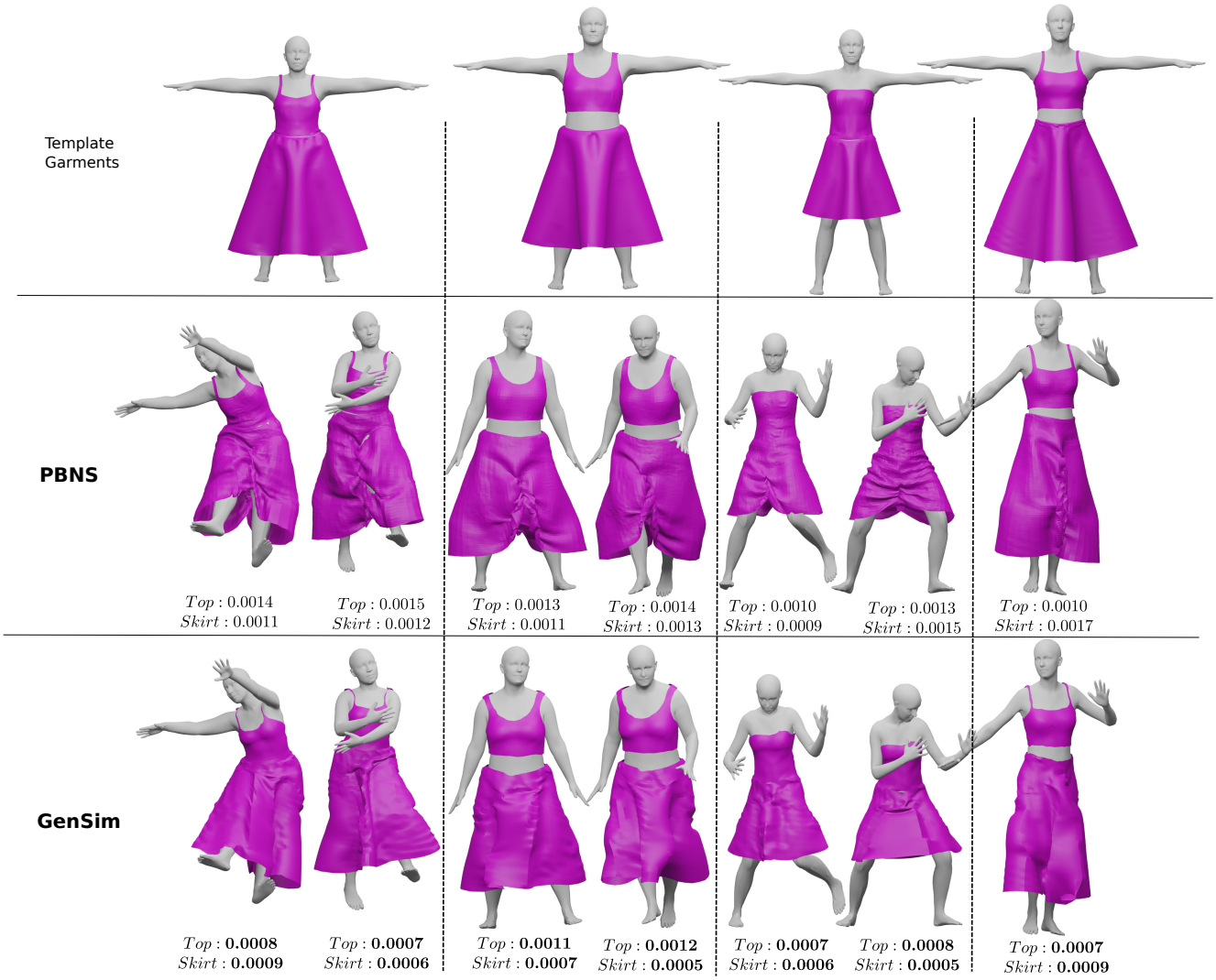

Figure S5. **Additional Comparison with PBNS:** PBNS is an outfit-body pair specific model. It is also an unsupervised method. Unlike PBNS, which can be evaluated on the same outfit and body shape on which it is trained, $GenSim$ once trained can be evaluated on any body shape or any garment of variable type, topology, size, etc. We can observe a **v-shaped** squeezing artifact between the legs in all the outputs of PBNS. This issue is also mentioned by the authors in their paper, especially in the skirts. Although the author claimed in the paper the issue is resolved, we observe the same artifact, especially in the long skirts. **Note:** we train PBNS using officially released code [2] by the authors, following the protocol mentioned in the paper. They clearly mentioned this is more prominent in the case of skirts. However, $GenSim$ doesn't have this artifact and can be seen a more realistic natural garment deformation of loose skirts. To quantify the level of distortion due to these artifacts, we compute the cot Laplacian (Eq. 7 main paper). The lower the number the better the method. The bold number indicates the best method. It can be seen that $GenSim$ outperforms the PBNS. Refer to Sec. S4.1 for more detail.

the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 8140–8150, 2022. 3, 7

[10] Igor Santesteban, Nils Thuerey, Miguel A Otaduy, and Dan Casas. Self-supervised collision handling via generative 3d garment models for virtual try-on. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11763–11773, 2021. 3, 7

Figure S6. *GenSim* evaluation on unseen garments, on random body shapes and poses.

Figure S7. Large scale image of Fig. 10 main paper.

Figure S8. Large scale image of Fig. 10 main paper.

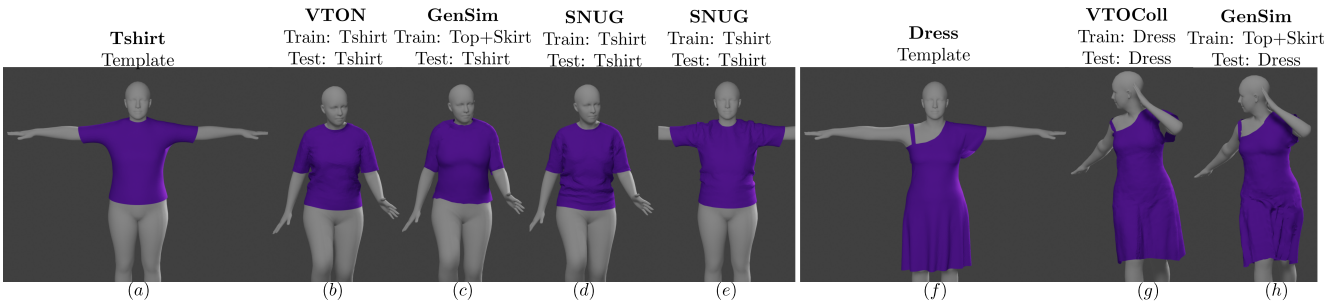

Figure S9. **Generalization Comparison with Unseen T-shirt and Dress**. *GenSim* which is an **unsupervised** method trained on only tops and skirts, generalizes well on unseen garments, and produces physically plausible simulations; whereas, VTON [8], SNUG [9], and VTOCOLL [10] are trained and tested on the same garment template (a) and (f). VTON [8] is a **supervised** method, whereas SNUG [9] is a self-supervised method and requires *garment vertices locations from the previous two inferred frames* to compute their velocities for self-supervision. VTOCOLL [10] also requires ground truth data for training, however, its collision handling module is self-supervised. *GenSim* **keeps the fitting of the template t-shirt (tight) and dress (loose) intact even after the deformations**. While SNUG **generates unnecessary wrinkles and folds**. To verify this we evaluate SNUG on the same template body with zero motion, the result in (e) shows that without any body motion, it generates wrinkles around the shoulder and the waist. Refer. Fig. S6, Fig. S7, and Fig. S8 to see results of *GenSim* on *unseen garment types pants, shorts, and tank.*