SUPPLEMENTARY MATERIAL FOR DARTS: DEFORMABLE ANIMATION READY TEMPLATES FOR CLOTHING HUMANS

Author(s) Name(s)

Author Affiliation(s)

1. SYNTHETIC DATASET GENERTAION

We generate a synthetic collection of image renderings from garments provided by the DeepFashion-3D dataset [1], incorporating diverse illumination conditions across 50 randomly sampled viewpoints. Each rendering and its corresponding camera parameters are saved relative to the SMPL model's T-pose orientation and the specific garment category. The dataset consists of 60,000 high-quality images rendered using Blender's Cycles renderer. While we primarily generate the dataset to supervise the feature regression network, it can also be utilized for other important tasks related to garment reconstruction. Figure 3(a) illustrates a scene set up with a randomly generated camera trajectory and illuminations. A random number of point lights are initialized at arbitrary locations to introduce illumination diversity. Additionally, as depicted in Figure 3(b), various compositor nodes are configured to save normal maps, ambient occlusion maps, alpha maps, diffuse color (albedo) maps, along with final renderings for each render pass, as shown in Figure 3(c). The entire dataset generation process requires approximately 24 hours on an NVIDIA RTX 4090 GPU system. It is important to note that for training the feature line regression network, we utilize only the diffused renderings or illuminated renderings from this synthetic dataset.

2. TEMPLATES AND FEATURE LINES

For each garment category in the DeepFashion-3D dataset [1], we generate garment-specific templates extracted from a base SMPL model. Figure 4 illustrates all possible templates derived in this process from the SMPL human model in standard T-Pose, with F_g outlined in black. Since the SMPL model is designed as a low-poly mesh, the edge loops formed by the feature lines have a limited number of vertices, which poses challenges for feature-line regression. To address this, we randomly upsample all edge loops before inputting them into the network for training. Specifically, each feature line is upsampled by a random factor in the range $8 \leq N_{up} \leq 18$.

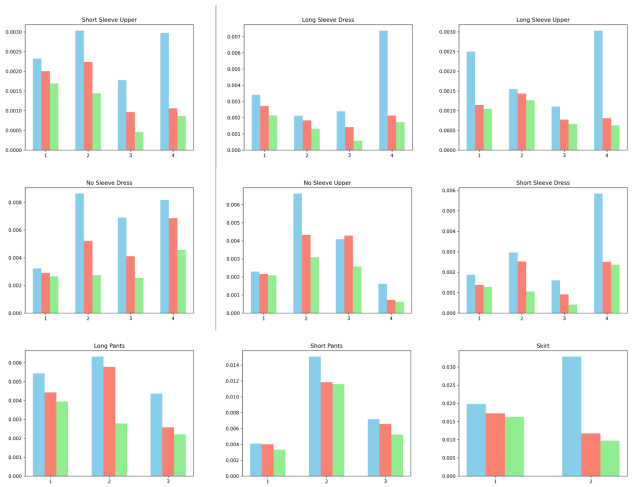


Fig. 1: A quantitative crease-wise comparison of chamfer distance across different garment groups with **P2M + GCN** [1]. **P2M + Im-Guided STM** highlights the impact of incorporating the proposed crease-wise global translations, and **Ours** represents the proposed feature line regression network with GCN-Guided and Image-Guided spatial translations.

3. NETWORK MODULES

Based on the garment category in the input image, *Image Guided STM* module predicts an image-guided spatial translation vector $t_i \in \mathrm{R}^3$ for each feature line $f_i \in \mathrm{R}^{n_i \times 3}$ within F_g. It applies the corresponding translation t_i to all n_i vertices in f_i. Utilizing camera parameters $\mathcal{C}$, we project the translated points onto the spatial resolution of i-th feature map $X_i \in \mathcal{X}_f$ and pool the features for all i, similar to [2]. This process is represented by the *Image-Space Projection* and *Feature Pooling* blocks. Afterward, we concatenate pooled features with vertex locations, resulting in $\mathcal{Z}$, where each vertex in f_i has its corresponding pooled feature of q-dimensionality, such that $z_i \in \mathrm{R}^{n_i \times (q+3)}$. These concatenated features are passed through multiple *Crease*

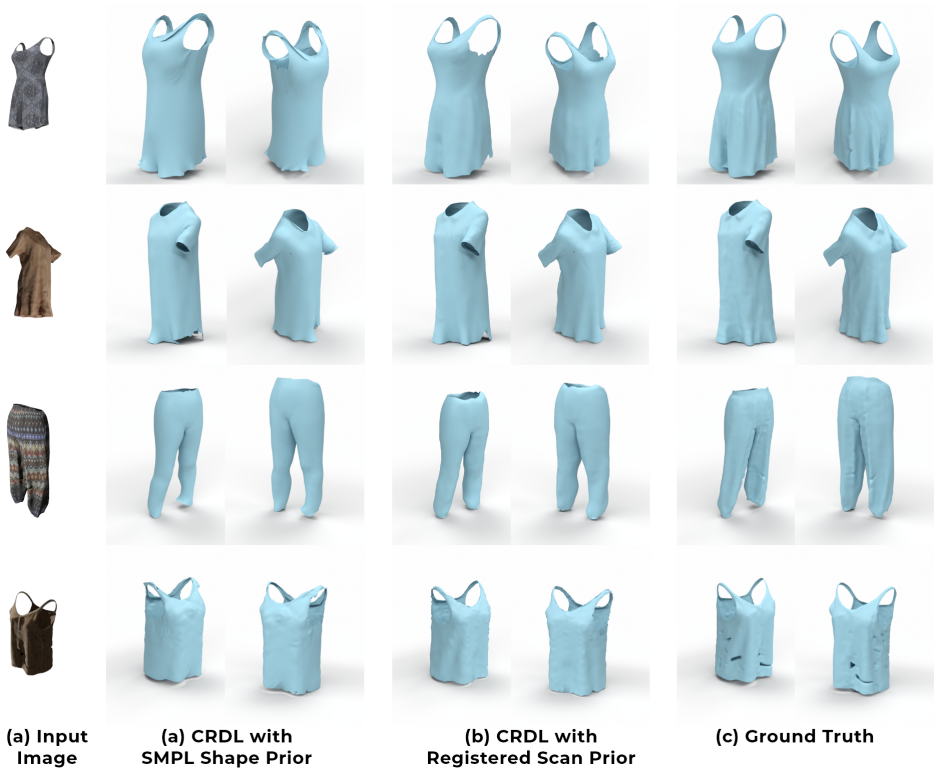


Fig. 2: Deformation of the template mesh T_g using the proposed CRDL-Layer, with the SMPL human body prior and registered scans as shape prior constraint. We use registered scans with CRDL Layer for illustration purposes, highlighting the fitting of fine surface details.

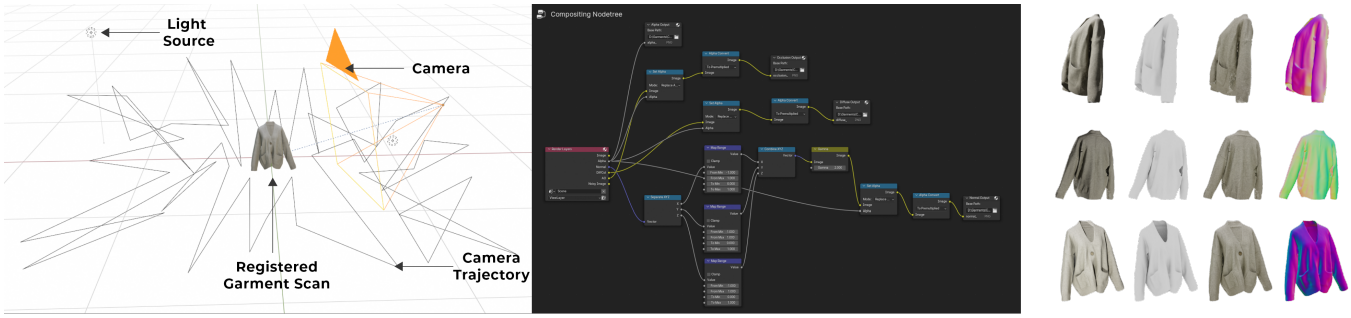
Regressor Blocks to produce the final feature lines. We now describe each module depicted in Figure 1 of main manuscript in detail.

Garment Classification and Template Selection. We fine-tune a ResNet-50 network pre-trained on the ImageNet dataset to address a standard classification task. From our synthetic dataset, we use 90% for training and achieve a classification accuracy of 99.4% on the validation set. The classification output is used to select the appropriate garment template T_g from the template library, along with the template feature lines f_g from f_{smpl}. Note that $|f_{smpl}| = N_f = 13$, and we select only relevant feature lines based on T_g to be processed further, as demonstrated in Figure 1(e) of main manuscript.

Image Guided STM. Since our feature lines F_{smpl} are derived from the posed SMPL model, we focus specifically on learning the translations of each feature line so that the translated f_i first aligns with the respective boundary in the input image. To accomplish this, we use spatial transformers [3] configured specifically to learn translation vectors. Each feature line should naturally shift in a specific direction guided by the input image $\mathcal{I}$. Therefore, we define $|F_{smpl}| = N_f$ spatial transformer modules (STMs), denoted as $\mathcal{S}$, where each S_i predicts a global translation for all n_i vertices within a given feature line f_i, defined in Equation 1. Consequently, all N_f crease lines have individual branches dedicated to spatial translation. After translation, all f_i^{trans} are stacked together to obtain F_g^{trans}. To facilitate image-guided learning within the spatial transformers $\mathcal{S}$, we use the output of the final layer from the feature extractor $X_k \in \mathcal{X}_f$ to predict a translation vector t_i for each f_i. We first process X_k through a couple of convolutional layers and then flatten the features to predict t_i employing Linear layers, as illustrated in Figure 5(a). It is important to highlight that in Figure 5(a), we only predict translations for the feature lines that are part of F_g, deduced based on selected template T_g. Following this step, the translated feature lines shift toward the approximate boundaries reflected in the input image. For example, the hemline in Figure 1(f) of the main manuscript demonstrates how this adjustment would greatly aid the projection and feature pooling stage, playing a critical role in enhancing prediction accuracy.

$$f_i^{trans} = S_i(X_k) + f_i \quad \forall f_i \in F_g \tag{1}$$

Projection and Feature Pooling. Let the projection be defined as $\Pi : \mathrm{R}^{N\times 3} \rightarrow \mathrm{R}^{H_i \times W_i}$, which maps every



(a) Scene Setup (b) Compositing Setup (c) Renderings

Fig. 3: For each registered garment scan from [1], we generate a random camera trajectory around the garment with randomized illumination (a) in Blender software. We sample 50 points along this trajectory, ensuring the camera always points at the object's center and renders the scene. The compositor setup for each render pass is shown in (b), and the resulting images are displayed in (c), with each column showcasing the final render, ambient-occlusion map, albedo map, and normal map, respectively.

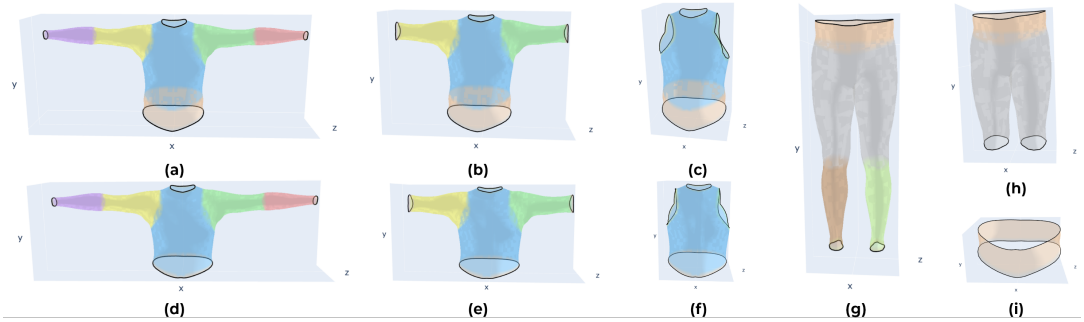


Fig. 4: Templates derived from SMPL human models in a T-pose: (a) Long Sleeve Dress, (b) Short Sleeve Dress, (c) No Sleeve Dress, (d) Long Sleeve Upper, (e) Short Sleeve Upper, (f) No Sleeve Upper, (g) Long Pants, (h) Short Pants, and (i) Skirts. The initial crease lines, F_g, are indicated on each template, T_g, by black lines.

vertex $\mathbf{x} \in F_g$ on the image space of i-th feature map $X_i \in \mathcal{X}$, based on camera parameters $\mathcal{C}$. We sample the features from X_i for all feature maps in $\mathcal{X}$ concatenate all sampled features with their respective vertex locations, and represent the result as $\mathcal{Z}$ as shown in Equation 2.

$$\mathcal{Z} = \left[\mathbf{x}, \bigcup_{X_i \in \mathcal{X}} \phi(X_i, \Pi(\mathbf{x}, \mathcal{C}))\right] \tag{2}$$

Here, $\bigcup$ denotes concatenation operation, and ϕ represents the grid sampling of features from the feature space of X_i.

GCN Guided STM. The architecture of the GCN-guided STM follows a similar structure to that of the Image-guided STM, with the key difference being that we predict vertex-wise translations for all vertex points in $\mathcal{P}_g$. Here, $\mathcal{P}_g$ is a high dimensional feature vector of all vertices in $\mathcal{Z}$ obtained after a few graph convolution operations. In contrast to $\mathcal{X}_f$, $\mathcal{Z}_h$ is a node-wise feature map. To accommodate this, we replace the convolutional layers in the Image-guided STM with linear layers, as shown in Figure 5(b). We observe that applying per-vertex translations, predicted by the proposed GCN-guided STM on $\mathcal{P}_g$, enables the vertices of feature lines to move more freely, allowing them to capture high-frequency variations along the crease.

Crease Regression Block. The Crease Regression Block (CRB) takes the pooled features $\mathcal{Z}$ as input and applies graph convolutions employing multiple *GResNet Block*, leveraging the adjacency information of the feature lines $f_i \in F_g$ for feature aggregation. We denote the output of the last GResNet block as $\mathcal{Z}_h$, which is then fed into the GCN-guided STM. Additionally, it is concatenated with the pooled features and passed to the subsequent CRB block. Each CRB block outputs P_g^{trans}, on which we apply loss functions to supervise the training of the feature line regression network. The number of CRB blocks significantly affects the convergence rate and final accuracy.

3.1. Loss Functions

We impose losses on the translated outputs from the STMs, specifically on F_g^{trans} and P_g^{trans} from each

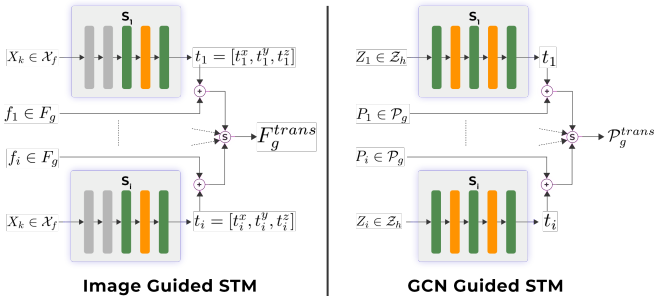


Fig. 5: The architecture of the Image-Guided STM and GCN-Guided STM modules: We use 13 crease-wise STM modules, one for each feature in $|F_{smpl}|$, where each module learns 3 translation parameters. The Image-Guided GCN predicts a global translation for all vertices within a feature line $f_i \in F_g$. In contrast, the GCN-guided STM predicts translations independently for each vertex. This allows for more flexible vertex movement to capture fine boundary details as reflected in the input image $\mathcal{I}$.
Legend: ■ 2D Convolution Layer, ■ Linear Layer and ■ ReLU activation. S and + indicate stack and addition operations.

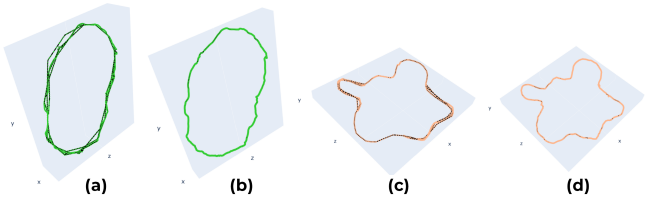


Fig. 6: The result of training with the proposed circularity loss is shown in (b) and (d). Since chamfer and edge regularization losses do not penalize loop formation by design, the network tends to converge to outputs that produce a spiraling effect, as seen in (a) and (b), which is undesirable for handle-based mesh deformation.

Crease Regressor Block. Let there be N_y CRB blocks. We define the set of all predictions as X_g such that X_g^i represents predictions including F_g^{trans} and P_g^{trans} of all N_y CRB blocks. We minimize the Chamfer distance between the predicted and ground-truth vertex locations of the feature lines. Additionally, we apply an edge regularization term to ensure smoothness and promote uniform edge lengths, as shown in Equation 3.

$$L_{ce} = \sum_{i=0}^{N_y+1} L_{chamfer}(F_g, X_g^i) + \lambda \sum_{i=0}^{N_y+1} L_{edge}(X_g^i) \quad (3)$$

In the initial phase of training, when predicted vertices are dispersed, the Chamfer loss and edge regularization terms in L_{ce} alone struggle to prevent spiraling in the output, as they lack mechanisms to explicitly penalize loop formations. Consequently, with these losses, output tends to get stuck in spiraled outputs, an effect we term the *spiraling effect*, as depicted in Figure 6(a) and 6(c).

Circularity Loss. As template deformation relies on the predicted positions of feature lines, the spiraling effect severely compromises the accuracy of the deformation. We introduce a simple circularity loss term alongside L_{ce} to address this, effectively reducing the spiraling issue. Consider a predicted feature line forming an edge loop, denoted as $f_i \in X_g$, with n_i vertices, represented by $f_i = (v_i, e_i)$, where v_i contains vertex positions and e_i contains edge connectivity information. Let $\mathbf{c}$ be the centroid of the loop f_i. We calculate angle spacings, θ_k, between consecutive vertices along the edges with respect to centroid $\mathbf{c}$, in a clockwise direction. To encourage uniform spacing, we impose a penalty if the angles deviate from the ideal equal spacing of $\frac{2\pi}{n_i}$ radians. The circularity loss for a feature line f_i is formulated as in Equation 4. Finally, we compute total loss for all feature lines $f_i \in X_g$ as $L_{circ} = \sum_i L_{circ}^i$. Applying this loss function during training leads to the formation of a single loop in the output, as shown in Figure 6(b) and 6(d).

$$L_{circ}^i = \frac{1}{n_i} \sum_k (\theta_k - \frac{2\pi}{n_i}) + \text{var}(\theta_k) \quad (4)$$

$$L = w_{ce} L_{ce} + w_{circ} L_{circ} \quad (5)$$

3.2. Implementation Details.

In our implementation, we use $N_y = 3$ number of CRB blocks, each containing $N_x = 6$ GResNet Blocks. The edge regularization weight $\lambda = 0.2$ in Equation 3, and $w_{ce} = 5$ in Equation 4. We apply the circularity loss with a weight $w_{circ} = 1$ for the first five epochs, then reduce w_{circ} to 0 once the feature lines have converged to form a stable single closed loop. During these initial epochs, the circularity loss encourages the vertices to organize into a continuous loop. After this convergence, removing the loss allows for more flexible vertex movement to capture finer boundary details. In total, we train for 30 epochs on a machine with an NVIDIA RTX 4090 GPU.

4. AS-RIGID-AS-POSSIBLE DEFORMATION

We propose a differentiable CRDL-Layer that minimizes energy $\mathcal{E}$ given in Equation 6. Unlike [5], all vertices $v_T \in T_g$ are learnable parameters of CRDL Layer, with deformation driven by handle vertices, which correspond to the regressed feature line vertices $v \in F_g$. As discussed in the main manuscript, this process is further constrained by additional losses, such as L_{fit} and L_{sil}.

$$\mathcal{E} = \sum_i \sum_{j \in \mathcal{N}(i)} w_{ij} ||(p_i' - p_j') - R_i(p_i - p_j)||^2 \quad (6)$$

In this section, we outline the approximation of R_i ensuring that the deformation of each i-th cell in the template

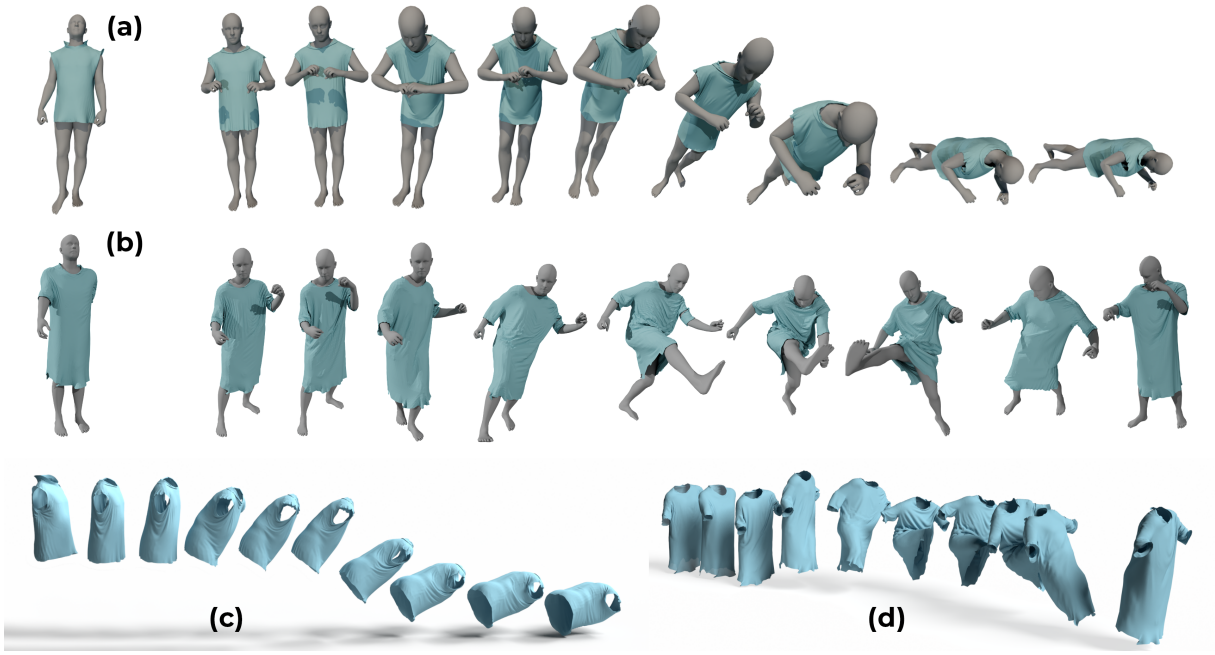


Fig. 7: Simulation results obtained within Maya software using AMASS [4] sequences, initialized with the templates shown in (a) and (b). The extracted garment geometry for the frames in sequences is presented in (c) and (d).

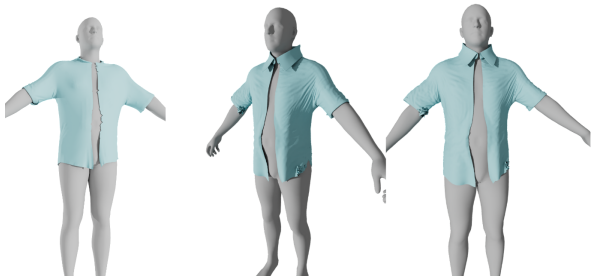


Fig. 8: Addition of a collar accessory to the predefined neck seam on DARTs and corresponding simulation results.

mesh T_g remains as rigid as possible, following [5]. Let $\mathcal{C}_i$ be a cell representing all the triangular faces incident on a vertex $v_i \in T_g$. The corresponding cell in the deformed template mesh at a given iteration is denoted by $\mathcal{C}'_i$. If the deformation $\mathcal{C} \rightarrow \mathcal{C}'$ is rigid, then there exists a rotation matrix R_i satisfying Equation 7. We define a covariance matrix S_i for vertex v_i to capture the local distribution and relationships among its neighboring vertices, as defined in Equation 8.

$$p'_i - p'_j = R_i(p_i - p_j), \quad \forall\, j \in \mathcal{N}(i) \tag{7}$$

$$S_i = \sum_{j \in \mathcal{N}(i)} w_{ij}(p_i - p_j)(p'_i - p'_j)^T \tag{8}$$

The rotation matrix R_i in Equation 6 can be derived by applying singular value decomposition to the covariance matrix $S_i = U_i \Sigma_i V_i^T$, yielding $R_i = V_i U_i^T$. To ensure a valid rotation matrix, we additionally flip the sign of the column in U_i corresponding to the smallest singular value, ensuring that $\det(R_i) > 0$.

5. QUANTITATIVE EVALUATION

We expand on the results presented in the main manuscript, where garment categories were grouped into three classes: **upper**, consisting of long/short/no sleeve dresses and upper garments, **lower**, consisting of long/short pants, and **skirts**. Here, we provide a detailed breakdown of the chamfer distance across individual garment categories, analyzed crease-wise, as shown in Figure 1.

5.1. Ablation Studies

Depth Ablation. As shown in Table 1, we run several experiments by varying the number of GResNet blocks within the CRB blocks and the total number of CRB blocks. All networks are trained for 15 epochs to ensure unbiased comparisons, with $w_c irc = 0$ applied after five epochs, as outlined in the main paper. The inclusion of CRB blocks incorporates perceptual feature pooling [2], with the loss functions applied at the output of each N_y CRB-Blocks. Notably, even when the number of graph convolutions facilitated by the GResNet blocks remains the same across different N_x, N_y configurations, we observe an improvement due to the feature pooling of newly

Model	N_x	N_y	CD	Model	N_x	N_y	CD
	3	1	0.673		3	1	0.617
	6	1	0.648		6	1	0.546
	9	1	0.581		9	1	0.496
Without	3	2	0.526	**With**	3	2	0.393
GCN-Guided	6	2	0.402	**GCN-Guided**	6	2	0.375
STM	9	2	0.393	**STM**	9	2	0.367
	3	3	0.417		3	3	0.368
	6	3	**0.382**		6	3	0.346
	9	3	0.383		9	3	**0.342**

Table 1: Ablation Study on Number of GResNet-Blocks (N_x) and Number of CRB Blocks (N_y) in the proposed feature line regression network. We compare models trained with and without GCN-guided spatial transformers, observing improved performance when applying per-vertex translations predicted by GCN-Guided STM. Here, CD refers to chamfer distance across all categories of garments in our validation split.

predicted points at the outputs of each CRB block. Including GCN-guided STM-based per-vertex translations in each CRB block allows for more precise vertex movements, enabling the capture of finer crease details and further enhancing regression accuracy. Based on the model complexity and accuracy results from Table 1, we choose the values of $N_x = 6$ and $N_y = 3$ for the proposed feature line regressor network.

On Spatial Rotations. Our network utilizes spatial transformers [3] in two stages: first, guided by the image to align feature lines with those in the input images globally, and second, guided by pooled per-vertex GCN features for finer adjustments. Since we extract templates from a posed SMPL human model $M(\beta, \theta)$, the feature lines are already rotationally aligned with respect to the image. Furthermore, when training the Image-Guided STM with learnable parameters for the rotation matrix $R \in \mathrm{R}^{3\times3}$, the matrix R gradually converges to an approximate identity matrix. Including R as a learnable parameter in STM prolongs the convergence process while contributing only to a near-identity rotation matrix. Therefore, we train the STM exclusively for translations. In the second stage, we predict per-vertex translations. Since rotating a single vertex lacks a meaningful reference point, applying only translations is sufficient.

5.2. Qualitative Results

Figure 2 showcases additional qualitative results across various garment categories, highlighting the deformations achieved using the proposed CRDL Layer. In Figure 2(a), we employ fitting loss L_{fit} with SMPL human body as the prior shape, while in Figure 2(b), we use registered scan mesh as prior for fitting. We demonstrate the use of the CRDL Layer on registered scans for illustration purposes, showing its potential to be seamlessly integrated with methods that learn implicit representations, such as [6, 7], similar to that of [1, 8].

6. DARTS SIMULATION

Starting with the deformed template on the leftmost side, Figure 7 illustrates clothing simulation across different garment categories and motion sequences from AMASS [4]. We developed DARTs using templates extracted from the SMPL human body model, incorporating predefined seams and spring constraints to streamline the simulation process. Figure 8 demonstrates the addition of a collar to the template mesh with predefined seams on DARTs, along with the corresponding simulation results.

7. REFERENCES

[1] Heming Zhu, Yu Cao, Hang Jin, Weikai Chen, Dong Du, Zhangye Wang, Shuguang Cui, and Xiaoguang Han, "Deep fashion3d: A dataset and benchmark for 3d garment reconstruction from single images," in *Computer Vision–ECCV 2020: 16th European Conference, UK, August 23–28*. Springer, 2020.

[2] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang, "Pixel2mesh: Generating 3d mesh models from single rgb images," in *European conference on computer vision (ECCV)*, 2018.

[3] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al., "Spatial transformer networks," *Advances in neural information processing systems*, vol. 28, 2015.

[4] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black, "Amass: Archive of motion capture as surface shapes," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 5442–5451.

[5] Olga Sorkine and Marc Alexa, "As-rigid-as-possible surface modeling," in *Symposium on Geometry processing*. Citeseer, 2007, vol. 4, pp. 109–116.

[6] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li, "Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 2304–2314.

[7] Yuliang Xiu, Jinlong Yang, Dimitrios Tzionas, and Michael J Black, "Icon: Implicit clothed humans obtained from normals," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2022, pp. 13286–13296.

[8] Heming Zhu, Lingteng Qiu, Yuda Qiu, and Xiaoguang Han, "Registering explicit to implicit: Towards high-fidelity garment mesh reconstruction from single images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.