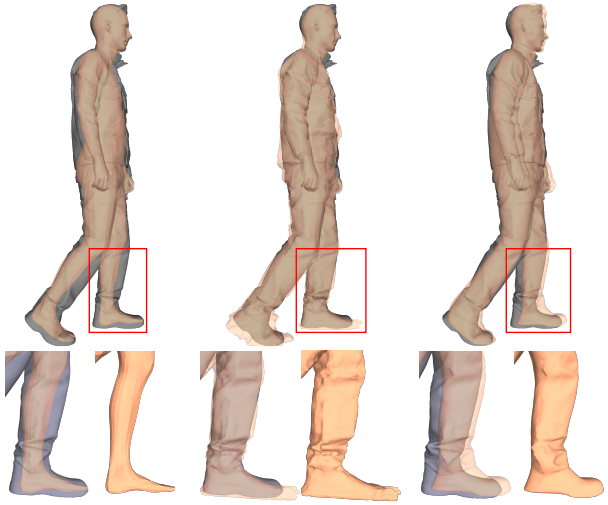
High-fidelity 3D Human Digitization from Single 2K Resolution Images Supplementary

Sang-Hun Han[1], Min-Gyu Park[2], Ju Hong Yoon[2],
Ju-Mi Kang[2], Young-Jae Park[1] and Hae-Gon Jeon[1]
[1]Gwangju Institute of Science and Technology (GIST), [2]Korea Electronics Technology Institute (KETI)
{sanghunhan, youngjae.park}@gm.gist.ac.kr, haegonj@gist.ac.kr
{mpark, jhyoon, yypeip}@keti.re.kr

A. Experiment Detail

A.1. Comparison over ICON

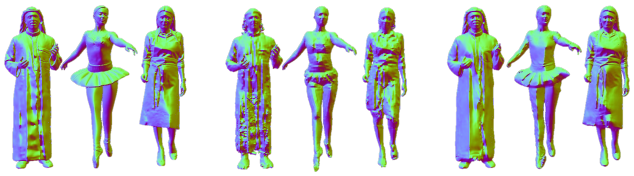


(a) SMPL-X (b) ICON **(c) Ours**

Figure I. Comparison with ICON.

The inconsistency between Table. 2 and Fig. 7 arises from the methodology (ICON [8]: model-based vs. Ours: model-free). When evaluating Table. 2, we used the GT SMPL-X skin body model provided by AGORA [4] for ICON, which shares the initial geometry of the GT clothed human dataset (Fig. I (a)) and is beneficial for the distance metrics: P2S and Chamfer. Since ICON receives the signed distance from SMPL-X skin body to query point as input, ICON results using GT SMPL-X show good performance in distance metrics such as P2S and Chamfer (Fig. I (b)). In contrast, our method predicts a depth from a camera origin, and leads to a depth ambiguity (Fig. I (c)). Due to this, although the mesh with good details is inferred, the distance metrics can be worsen than ICON using a strong geometry prior.

Here, we demonstrate another merit on loose cloth cases of our model-free method. In Fig. II and Table. I, our method consistently shows the high-quality reconstructions while ICON suffers from the loss of details in the skirt or hair.



(a) GT (b) ICON **(c) Ours**

Figure II. RenderPeople results w.r.t. loose clothes test dataset.

Method	P2S↓	Chamfer↓	Normal↓
ICON	1.47	1.63	6.37
2K2K	**1.24**	**1.35**	**3.89**

Table I. Test on 15 loose cloth sets in RenderPeople.

A.2. Comparison with other datasets

We perform an additional experiment to validate the effectiveness of our dataset. Table. II shows the quantitative results on RenderPeople(RP) w.r.t. 3 training scenarios: (1) Ours only, (2) RP only, and (3) Using both of them. As expected, our dataset contributes to the huge improvement in the network training.

Training	P2S↓	Chamfer↓	Normal↓
2K2K	2.20	2.31	5.01
RenderPeople	1.12	0.92	3.64
2K2K+RenderPeople	**0.58**	**0.63**	**3.23**

Table II. Evaluation on RenderPeople test split.

A.3. Details in comparison experiment

For fair comparison evaluating Table. 2, we used the same training set. However, due to the GPU memory issue, all the comparison methods couldn't take 2K input images. For them, we utilized input images with both 512×512 and 1024×1024 resolutions, and reported the best scores of them. Since there is no public code for PIFuHD, we used authors' official pre-trained model. We note that PIFuHD uses 450 models for training, but ours only takes 331 models.

A.4. Comparison of SOTA normal prediction

We compare our normal prediction method with those of the previous works, PIFuHD [7] and HDNet [1], in Fig. III. While other results show artifacts or blurs, we infer the high-quality normal.

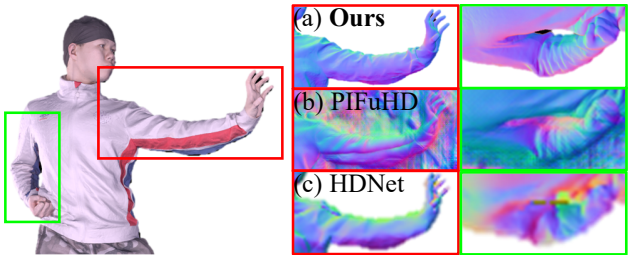


Figure III. Comparison of the normal prediction.

B. Method Detail

B.1. Body part extraction detail

We compute the part-wise transformation matrix $\mathbf{M}_i$ with human joints by using the linear least squares (LLS). The goal of this process to transform the input image into a target body part. After that, we crop the transformed image to a predefined size. The similarity transformation matrix is computed based on the target position (x', y') and ith part joint position $j_i(x, y) \in \mathcal{J}$,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & -b & c \\ b & a & d \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{1}$$

where a, b, c, and d are an element of the similarity matrix. To solve the above equation, we need at least two joints because the degree of freedom of the similarity matrix is 4. After arranging the above matrix for elements, the equation for each point is stacked in row as follows,

$$\begin{bmatrix} x_1 & -y_1 & 1 & 0 \\ y_1 & x_1 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_N & -y_N & 1 & 0 \\ y_N & x_N & 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} x'_1 \\ y'_1 \\ \vdots \\ x'_N \\ y'_N \end{bmatrix}, \tag{2}$$

where N is the number of joints that we use to compute the transformation matrix $\mathbf{M}$.

For detail, the joints $j_i(x, y) \in \mathcal{J}$ and the pre-defined size for cropping are shown in the Table. III. Each joint position is a value that maps the upper left to [-1, -1] and the lower right to [1, 1]. The detail process of transforming and cropping the image is as shown in Fig. IV. This process has the advantage of requiring few computation cost.

	# of Keypoint	Keypoint $j_i(x, y)$	Crop size s_i
Head	2	(-1/12, 1/24), (1/12, 1/24)	[368, 320]
Body	4	(-1/9, -1/6), (1/9, -1/6), (-1/12, 1/6), (1/12, 1/6)	[528, 336]
Upper arms	2	(0, -3/16), (0, 1/16)	[352, 224]
Lower arms	3	(0, -2/9), (0, 0), (0, 1/18)	[352, 224]
Left upper legs	2	(-3/128, -5/32), (-3/128, 3/32)	[352, 224]
Right upper legs	2	(3/128, -5/32), (3/128, 3/32)	[352, 224]
Left lower legs	2	(-3/128, -9/64), (-3/128, 9/64)	[272, 256]
Right lower legs	2	(3/128, -9/64), (3/128, 9/64)	[272, 256]
Feet	4	(0, 1/24), (0, 1/24), (0, -1/24), (0, -1/24)	[176, 128]

Table III. In this work, we set five parts with twelve predefined positions and sizes; Head: two keypoints from left/right ears, Body: four keypoints from left/right shoulders and left/right heaps, Arms: two keypoints from a shoulder and an elbow, three keypoints from an elbow, an wrist, and a finger, Legs: two keypoints from a heap and a knee, two keypoints from a knee and am ankle, Feet: four keypoints from an ankle, a sole, and two toes.

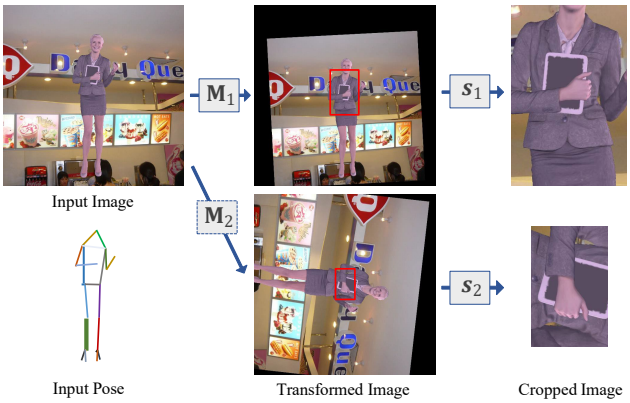


Figure IV. Body part extraction from input image. Through the part-wise transformation matrix $\mathbf{M}$ calculated using the input human joint $j(x, y)$, we transform the target part so that it is centered on the image. After that, we crop it to a fixed size $\mathbf{s}$ and extract the part image

B.2. Network description

In this section, we provide detailed descriptions of our network in Tables IV and V where we indicate the input in blue and the output in red. In the INPUT column, element-wise sum denotes '+', and $\oplus$ denotes channel concatenation operator.

IDX	STRUCTURE		KERNEL	INPUT	OUTPUT
1	Up_block	Up + Conv + BN + ReLU	3 × 3	256 × 256 × 2	512 × 512 × 32
2	Conv_block	Conv + BN + ReLU	3 × 3	512 × 512 × 6	512 × 512 × 32
3	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX1_OUT ⊕ IDX2_OUT	512 × 512 × 64
4	Conv	Conv	1 × 1	IDX3_OUT	512 × 512 × 64
5	Up_block	Up + Conv + BN + ReLU	3 × 3	IDX4_OUT	1024 × 1024 × 32
6	Conv_block	Conv + BN + ReLU	3 × 3	1024 × 1024 × 6	1024 × 1024 × 32
7	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX5_OUT ⊕ IDX6_OUT	512 × 512 × 64
8	Conv	Conv	1 × 1	IDX7_OUT	512 × 512 × 64
9	Up_block	Up + Conv + BN + ReLU	3 × 3	IDX8_OUT	2048 × 2048 × 32
10	Conv_block	Conv + BN + ReLU	3 × 3	2048 × 2048 × 6	2048 × 2048 × 32
11	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX9_OUT ⊕ IDX10_OUT	2048 × 2048 × 64
12	Conv_out	Conv + ReLU	1 × 1	IDX11_OUT	2048 × 2048 × 3

Table IV. Details of a high-resolution depth network. We estimate high-resolution front and back depth maps in a three-step cascade structure. In each cascade block, low resolution depth maps are up-sampled to higher resolution depth maps with a corresponding normal feature map.

IDX	STRUCTURE		KERNEL	INPUT	OUTPUT
1	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	256 × 256 × 3	256 × 256 × 32
2	Maxpool	Maxpool	2 × 2	IDX1_OUT	128 × 128 × 32
3	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX2_OUT	128 × 128 × 64
4	Maxpool	Maxpool	2 × 2	IDX3_OUT	64 × 64 × 64
5	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX4_OUT	64 × 64 × 128
6	Maxpool	Maxpool	2 × 2	IDX5_OUT	32 × 32 × 128
7	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX6_OUT	32 × 32 × 256
8	Maxpool	Maxpool	2 × 2	IDX7_OUT	16 × 16 × 256
9	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX8_OUT	16 × 16 × 256
10	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	256 × 256 × 6	256 × 256 × 32
11	Maxpool	Maxpool	2 × 2	IDX10_OUT	128 × 128 × 32
12	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX11_OUT	128 × 128 × 64
13	Maxpool	Maxpool	2 × 2	IDX12_OUT	64 × 64 × 64
14	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX13_OUT	64 × 64 × 128
15	Maxpool	Maxpool	2 × 2	IDX14_OUT	32 × 32 × 128
16	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX15_OUT	32 × 32 × 256
17	Maxpool	Maxpool	2 × 2	IDX16_OUT	16 × 16 × 256
18	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX17_OUT	16 × 16 × 512
19	Up_block	Up + Conv + BN + ReLU	3 × 3	IDX9_OUT ⊕ IDX18_OUT	32 × 32 × 512
20	3*Attention_block	Conv + BN	1 × 1	IDX19_OUT	32 × 32 × 256
21		Conv + BN	1 × 1	IDX7_OUT ⊕ IDX16_OUT	32 × 32 × 256
22		ReLU + Conv + BN + Sigmoid	1 × 1	IDX20_OUT + IDX21_OUT	32 × 32 × 1
23	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX19_OUT ⊕ IDX22_OUT	32 × 32 × 512
24	Up_block	Up + Conv + BN + ReLU	3 × 3	IDX23_OUT	64 × 64 × 256
25	3*Attention_block	Conv + BN	1 × 1	IDX24_OUT	64 × 64 × 128
26		Conv + BN	1 × 1	IDX5_OUT ⊕ IDX14_OUT	64 × 64 × 128
27		ReLU + Conv + BN + Sigmoid	1 × 1	IDX25_OUT + IDX26_OUT	64 × 64 × 1
28	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX24_OUT ⊕ IDX27_OUT	64 × 64 × 256
29	Up_block	Up + Conv + BN + ReLU	3 × 3	IDX28_OUT	128 × 128 × 128
30	3*Attention_block	Conv + BN	1 × 1	IDX29_OUT	128 × 128 × 64
31		Conv + BN	1 × 1	IDX3_OUT ⊕ IDX12_OUT	128 × 128 × 64
32		ReLU + Conv + BN + Sigmoid	1 × 1	IDX30_OUT + IDX31_OUT	128 × 128 × 1
33	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX29_OUT ⊕ IDX32_OUT	128 × 128 × 128
34	Up_block	Up + Conv + BN + ReLU	3 × 3	IDX33_OUT	256 × 256 × 64
35	3*Attention_block	Conv + BN	1 × 1	IDX34_OUT	256 × 256 × 32
36		Conv + BN	1 × 1	IDX1_OUT ⊕ IDX10_OUT	256 × 256 × 32
37		ReLU + Conv + BN + Sigmoid	1 × 1	IDX35_OUT + IDX36_OUT	256 × 256 × 1
38	Conv_block × 2	(Conv + BN + ReLU) × 2	3 × 3	IDX34_OUT ⊕ IDX37_OUT	256 × 256 × 64
39	Conv_out	Conv	1 × 1	IDX38_OUT	256 × 256 × 3

Table V. Details of dual-encoder AU-Net. The first encoder, indexed by 1 to 9, takes a low-resolution image as input. The second encoder, indexed by 10 to 18, takes predicted low-resolution front and back normal maps as input. Finally, a decoder, indexed by 19 to 39, fuses features of the two encoders and predicts front and back depth maps.

B.3. Coarse-to-fine vs. Part-wise for normal prediction

(a) GT (b) C2F (c) Ours

Figure V. Comparison with coarse-to-fine method.

We adopted a coarse-to-fine normal refinement module of DeepHuman [11] after changing the resolution of the input to 2K in Fig. V. The performance was worse (0.890, 47GB in batch-size 1) than ours (0.679, 39GB in batch-size 3) in terms of normal error and memory requirement. The part-wise model is a practical solution for 2K or even larger resolution images because it does not extract global image features for normal map prediction, while the coarse-to-fine approach requires additional convolution layers to enlarge receptive fields for high resolution images.

B.4. Poisson reconstruction

For fair comparison, we first convert the predicted depth map into the depth point cloud. We then overlap the point-cloud (Black) and the mesh (Gray) generated by a screened Poisson surface construction [2]. Fig. VI shows that they are completely overlapped, which means our high-resolution reconstruction does not cause a blur artifact.

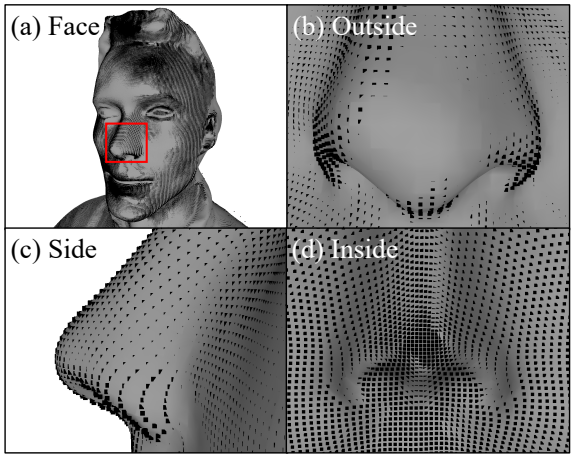


Figure VI. Comparison of pointcloud and mesh using the Poisson.

B.5. Gaussian blending

Fig. VII compares normal maps merged by averaging and Gaussian blending. Averaging approach shows streaking artifacts at the boundary whereas ours does not.

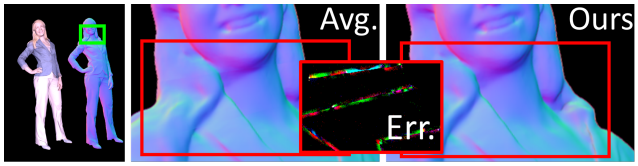
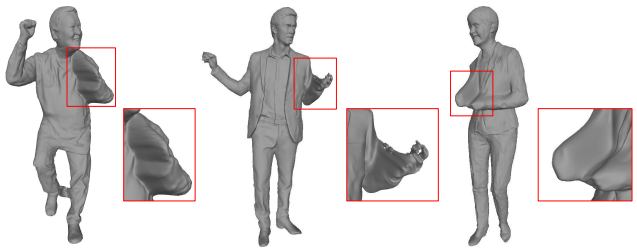


Figure VII. Comparison of averaging and Gaussian blending.

B.6. Failure case

Fig. VIII shows some failure cases. 3D recovery using only the front and back view depth maps shows that it is vulnerable to occlusions caused by objects or human body parts. In Fig. VIII (a), the correct shape could not be created because the spatial information between the arm and the torso could not be known only from the front and back depths. In addition, the network has difficulty in distinguishing it from garment texture and creating a clear depth boundary. Fig. VIII (b) shows an unnatural appearance as the boundary between the holding cup and the human body is smooth. We expect that the limitations can be improved by using a skinned human model [3].



(a) Occlusion (b) Ambiguity of boundary

Figure VIII. Failure cases caused by occlusions and boundary ambiguities.

C. Dataset Detail

C.1. Feasibility of rendering at 2K resolution

We took 6,000×4,000 resolution images using 80 multi-view DSLR cameras. Sharing whole original images requires 1,850TB. Instead, since each body model in our dataset has 1M vertices and 2M faces, it is enough to be synthesized into 2K images as shown in Fig. IX.

Figure IX. Rendered multi-view images example.

C.2. Multi-view dataset capture

Thanks to the high-quality of our 3D models, we can render images from arbitrary viewpoints, whose example of an original and rendered images at 2K res. in Fig. IX. To acquire synchronized images, we used a VILTROX VL-500RT controller to generate trigger signals. Then, an Arduino board receives the signals and broadcasts them to the connected devices including the strobe lights and the DSLR cameras.

C.3. UV unwrapping

With our dataset, we are able to implement UV unwrapping modules for DensePose and SMPL template models (see Fig. X). Note that the texture map can be noisy if the model wears loose clothes (<5%).

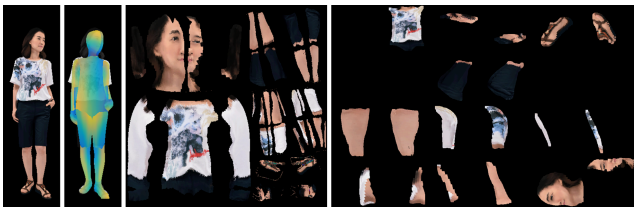


Figure X. UV texture maps for DensePose and SMPL.

C.4. Dataset appearance example

Fig. XI shows appearance, mesh and normal examples of our dataset. Our dataset contains subjects of various genders, ages, objects, poses, and cloths with high-resolution scans.

D. More Results

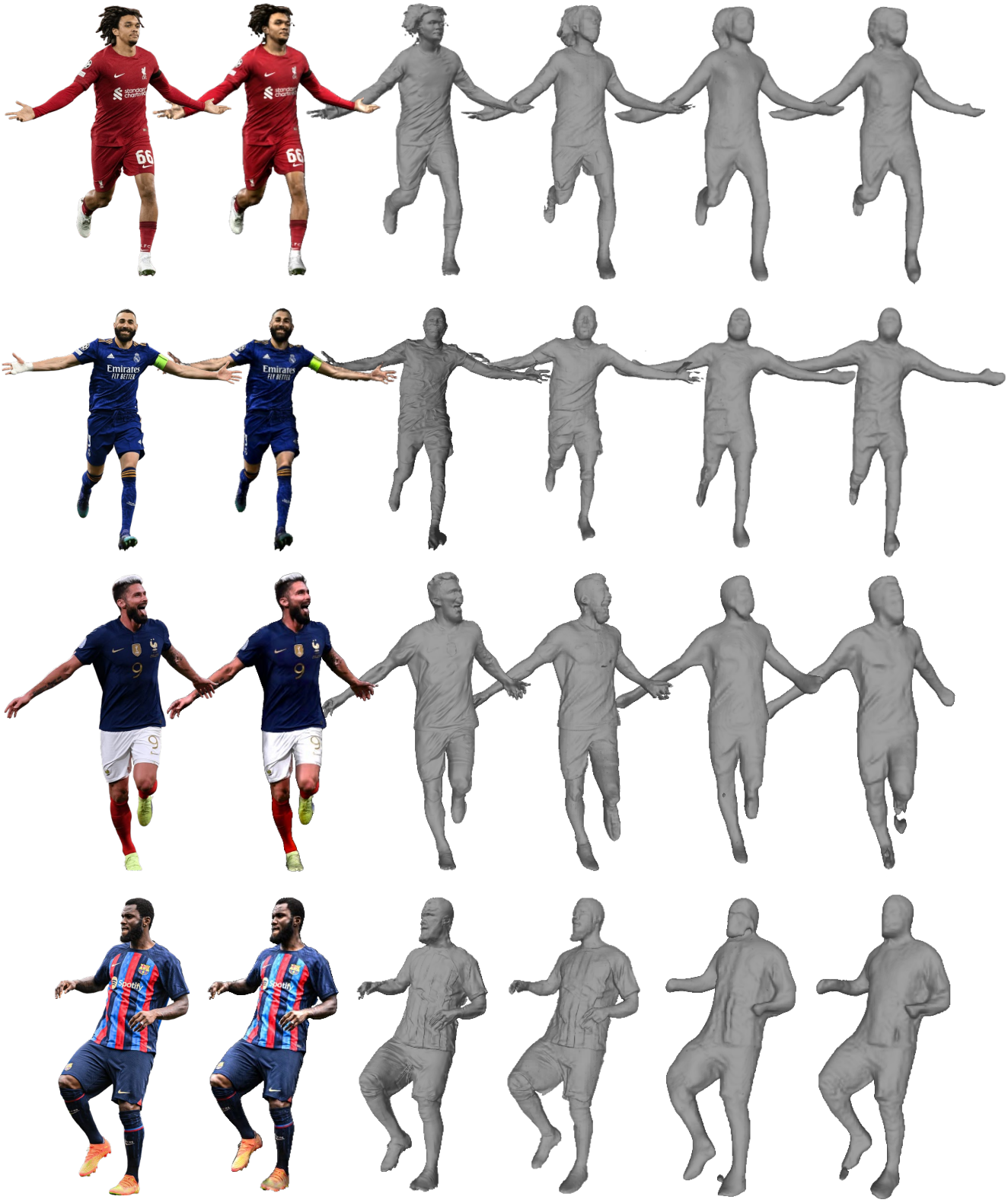
D.1. In-the-wild qualitative results

Fig. XII, XIII shows the qualitative result made from in-the-wild internet photos. Although there is noise taken with a camera, our result forms a detailed human body shape compared to other methods [6,7,10]. Additionally, we reconstruct a sequence of images taken with a DSLR and make it as a video.

D.2. Dataset qualitative results

Fig. XIV, XV shows the qualitative comparison results of RenderPeople [5] and THuman2.0 [9] dataset. Compared to other methods [6,7,10], we reconstruct in a more clean and detailed surface when expressing the shape of a face, cloth, etc.

Figure XI. Example appearance, mesh and normal of our dataset. Each subject has high-resolution geometry features.



(a) Input (b) Ours (c) PIFuHD (d) PIFu (e) PaMIR

Figure XII. Qualitative comparison of in-the-wild images with other methods.

(a) Input (b) Ours (c) PIFuHD (d) PIFu (e) PaMIR

Figure XIII. Qualitative comparison of in-the-wild images with other methods.

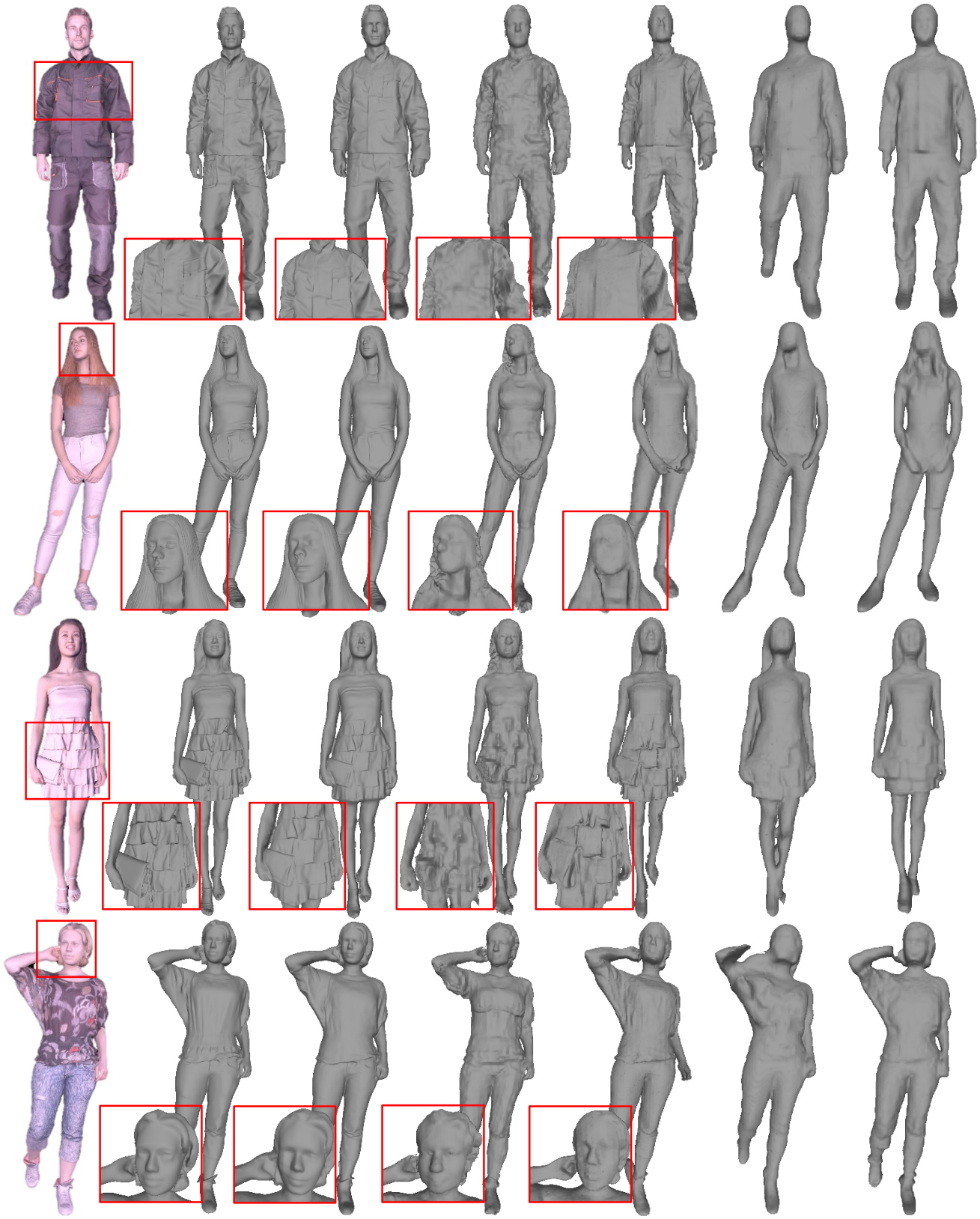


Figure XIV. Qualitative comparisons of reconstructed results on the RenderPeople dataset.

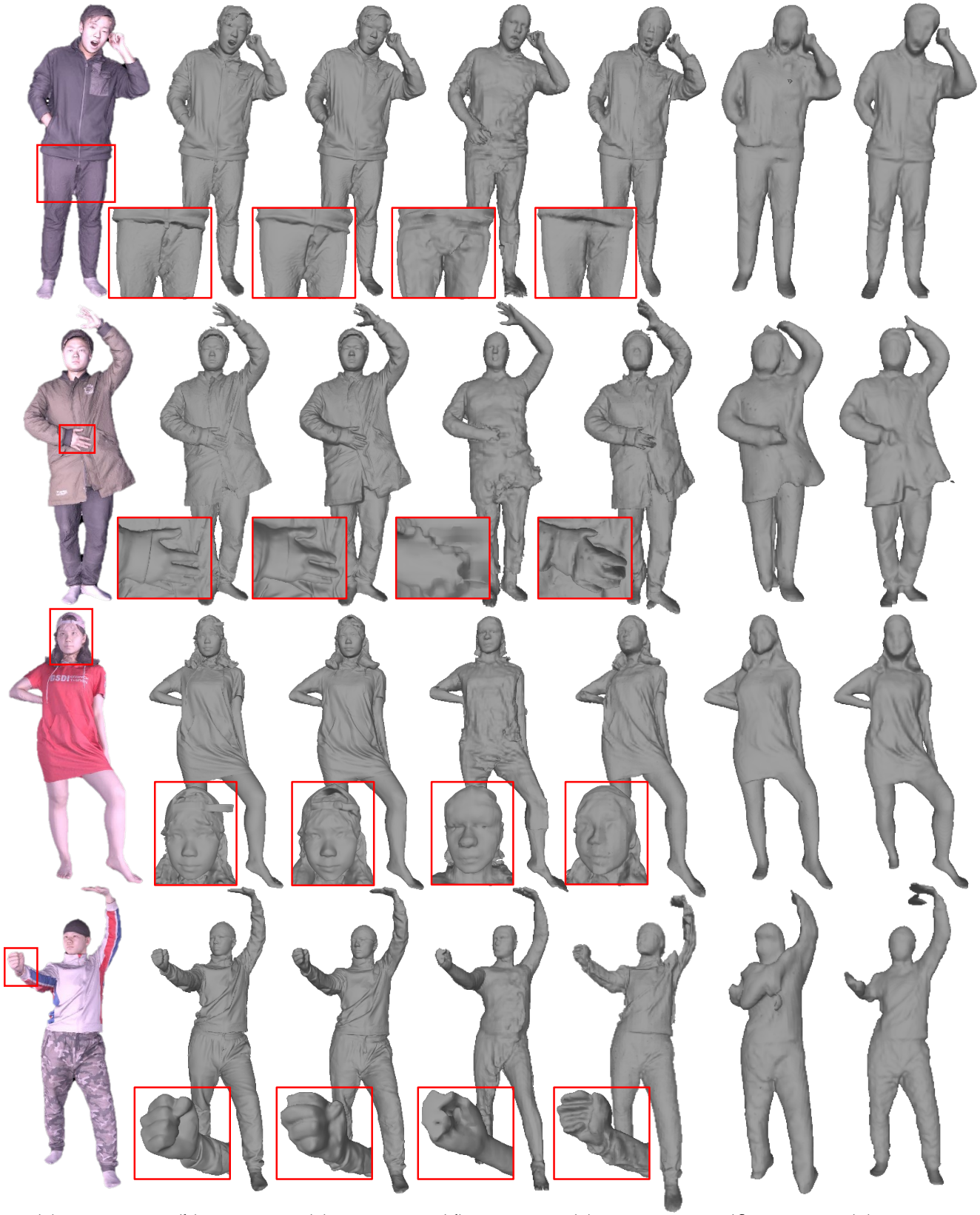


Figure XV. Qualitative comparisons of reconstructed results on the THuman2.0 dataset.

References

[1] Yasamin Jafarian and Hyun Soo Park. Learning high fidelity depths of dressed humans by watching social media dance videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[2] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013. 3

[3] 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)*, 34:1–16, 2015. 3

[4] Priyanka Patel, Chun-Hao P. Huang, Joachim Tesch, David T. Hoffmann, Shashank Tripathi, and Michael J. Black. AGORA: Avatars in geography optimized for regression analysis. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 1

[5] Renderpeople. https://renderpeople.com/. 4

[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 (ICCV)*, 2019. 4

[7] Shunsuke Saito, Tomas Simon, Jason Saragih, and Hanbyul Joo. Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 4

[8] Yuliang Xiu, Jinlong Yang, Dimitrios Tzionas, and Michael J Black. Icon: Implicit clothed humans obtained from normals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1

[9] Tao Yu, Zerong Zheng, Kaiwen Guo, Pengpeng Liu, Qionghai Dai, and Yebin Liu. Function4d: Real-time human volumetric capture from very sparse consumer rgbd sensors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 4

[10] Zerong Zheng, Tao Yu, Yebin Liu, and Qionghai Dai. PaMIR: Parametric Model-Conditioned Implicit Representation for Image-based Human Reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 44:3170–3184, 2021. 4

[11] Zerong Zheng, Tao Yu, Yixuan Wei, Qionghai Dai, and Yebin Liu. Deephuman: 3d human reconstruction from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 3