DIFu: Depth-Guided Implicit Fuction for Clothed Human Reconstruction: Supplementary Materials

Dae-Young Song†[1,2], HeeKyung Lee[1], Jeongil Seo[1], Donghyeon Cho*[2]
[1]Electronics and Telecommunications Research Institute, Daejeon, South Korea
[2]Chungnam National University, Daejeon, South Korea
{eadyoung,lhk95,seoji}@etri.re.kr, cdh12242@gmail.com

In this supplementary material, additional about DIFu is provided as follows:

- Detailed descriptions of the DIFu architecture used in the experiments.
- Additional qualitative results.
- An ablation study on the effect of using λ.

1. Architectures

1.1. Network Complexity

First, we report the total number of parameters in each method used for the experiments as shown in Table 1 below.

PIFu [5]	PaMIR [10]	ICON [8]	Ours
24,744,136	79,379,559	430,747,671	154,342,718

Table 1. The total number of each method in our experiments. Each number is the summation of the number of parameters in the pretrained submodules, geometry and texture branches.

1.2. Hallucinator

Residual Block. The structure of a residual block constituting the hallucinator is as shown in Table 2 below.

layer	**k**	**s**	**p**	**group**	**input**
GroupNorm1	-	-	-	8	input
conv1	1	1	0	1	GroupNorm1
conv2	3	1	1	1	conv1
gate1	γ * GELU(conv2) + input				
GroupNorm2	-	-	-	8	gate1
conv3	1	1	0	1	GroupNorm2
conv4	3	1	1	1	GELU(conv3)
gate2	δ * GELU(conv4) + gate1				

Table 2. The architecture of the residual block in our hallucinator, where **k**, **s**, **p**, and **group** are the kernel size, stride, padding, and the number of groups for operation, respectively. Also, γ and δ are the learnable gate parameters, and **input** corresponds to the input of each layer.

Hallucinator. The hallucinator is composed of downsampling convolutional layers, upsampling convolutional layers, and the residual blocks as shown in Table 3 below.

layer	**k**	**s**	**p**	**op**	**group**	C_{in}	C_{out}	**input**
downconv1	3	2	1	-	1	3	64	front-side RGB
residual1	-	-	-	-	-	64	64	downconv1
downconv2	3	2	1	-	1	64	128	residual1
residual2	-	-	-	-	-	128	128	downconv2
downconv3	3	2	1	-	1	128	256	residual2
residual3	-	-	-	-	-	256	256	downconv3
downconv4	3	2	1	-	1	256	512	residual3
residual4	-	-	-	-	-	512	512	downconv4
downconv5	3	2	1	-	1	512	512	residual4
residual5	-	-	-	-	-	512	512	downconv5
residual6	-	-	-	-	-	512	512	residual5
residual7	-	-	-	-	-	512	512	residual6
residual8	-	-	-	-	-	512	512	residual7
upconv1	3	2	1	1	1	1024	512	residual8 $\oplus$ downconv5
residual9	-	-	-	-	-	512	512	upconv1
upconv2	3	2	1	1	1	1024	512	residual9 $\oplus$ downconv4
residual10	-	-	-	-	-	512	512	upconv2
upconv3	3	2	1	1	1	768	256	residual10 $\oplus$ downconv3
residual11	-	-	-	-	-	256	256	upconv3
upconv4	3	2	1	1	1	384	128	residual11 $\oplus$ downconv2
residual12	-	-	-	-	-	128	128	upconv4
upconv5	3	2	1	1	1	192	64	residual12 $\oplus$ downconv1
lconv1	3	1	1	-	1	64	32	upconv5
lconv2	1	1	0	-	1	32	3	lconv1
Sigmoid(lconv2)								

Table 3. The architecture of our hallucinator, where **k**, **s**, **p**, **op**, and **group** are the kernel size, stride, padding, padding for the output, and the number of groups for operation. Also, C_{in} and C_{out} are the number of input and output channels, and **input** corresponds to the input of each layer. The "upconv" layers are transposed convolutional layers, where $\oplus$ means channel-wise concatenation of skip-connections. The structure of "residual" layers is presented in Table 2.

1.3. Depth Estimator

Our depth estimator includes a voxel encoder, a scale regressor, and an U-Net estimator. Note the notations "downconv" and "upconv" are not the same in Section 1.2.

Voxel Encoder. The voxel encoder embeds a predicted parametric voxel based on depthwise convolutional layers. The dimension of input SMPL voxel is $128 \times 128 \times 128$.

layer	**k**	**s**	**p**	**group**	C_{in}	C_{out}	**input**
depthconv1	3	1	1	128	128	128	voxel
downconv1	3	2	1	1	128	256	depthconv1
depthconv2	3	1	1	256	256	256	downconv1
downconv2	3	2	1	1	256	512	depthconv2
depthconv3	3	1	1	512	512	512	downconv2

Table 4. The architecture of the voxel encoder, where **k**, **s**, **p**, and **group** are the kernel size, stride, padding, and the number of groups for operation, respectively. Also, C_{in} and C_{out} are the number of input and output channels, and **input** corresponds to the input of each layer.

U-Net Estimator. The U-Net estimator simultaneously generates front/back depth maps ranging from 0 to 1.

layer	**k**	**s**	**p**	**group**	C_{in}	C_{out}	**input**
downconv1*	7	2	3	1	6	32	front-/back-side RGB
downconv2*	5	2	2	1	32	64	downconv1*
downconv3*	3	2	1	1	64	128	downconv2*
downconv4*	3	2	1	1	128	256	downconv3*
downconv5*	3	2	1	1	256	512	downconv4*
upconv5†	3	1	1	1	512	512	downconv5*
iconv5†	3	1	1	1	512	512	upconv5†
upconv4†	3	1	1	1	1280	256	iconv5† $\oplus$ downconv4* $\oplus$ depthconv3
iconv4†	3	1	1	1	256	256	upconv4†
upconv3†	3	1	1	1	640	128	iconv4† $\oplus$ downconv3* $\oplus$ depthconv2
iconv3†	3	1	1	1	128	128	upconv3†
upconv2†	3	1	1	1	320	64	iconv3† $\oplus$ downconv2* $\oplus$ depthconv1
iconv2†	3	1	1	1	64	64	upconv2†
upconv1†	3	1	1	1	96	32	iconv2† $\oplus$ downconv1*
iconv1†	3	1	1	1	32	32	upconv1†
conv	3	1	1	1	32	2	iconv1†
Sigmoid(conv)							

Table 5. The architecture of the U-Net depth estimator, where **k**, **s**, **p**, and **group** are the kernel size, stride, padding, and the number of groups for operation, respectively. Also, C_{in} and C_{out} are the number of input and output channels, and **input** corresponds to the input of each layer. Layers marked with a superscript * indicate that they are followed by the group normalization [7] layer with 8 groups and the ELU [1] activation function, while a superscript † means only the ELU follows.

Scale Regressor. The scale regressor takes the encoding feature of the U-Net and the final voxel encoding feature of the voxel encoder.

layer	**k**	**s**	**p**	C_{in}	C_{out}	**input**
scaleconv1†	3	2	1	768	512	downconv4* $\oplus$ depthconv3
scaleconv2†	3	2	1	512	256	scaleconv1†
scaleconv3†	3	2	1	256	128	scaleconv2†
fc1	-	-	-	2048	64	scaleconv3†
fc2	-	-	-	64	32	Dropout(ReLU(fc1))
fc3	-	-	-	32	16	Dropout(ReLU(fc2))
fc4	-	-	-	16	1	Dropout(ReLU(fc3))
Sigmoid(fc4)						

Table 6. The architecture of the scale regressor, where **k**, **s**, and **p** are the kernel size, stride, and padding, respectively. Also, C_{in} and C_{out} are the number of input and output channels, and **input** corresponds to the input of each layer. Layers marked with a superscript † indicate that they are followed by the ELU [1] activation function. The "fc" denotes fully-connected layers.

1.4. Occupancy Prediction Network

Our occupancy prediction network is composed of a 2D encoder, a 3D encoder, and multilayer perceptrons (MLPs). First, we adopt the stacked hourglass networks [4] of PIFu as our 2D encoder. However, since it receives a RGB input and a corresponding depth map, it is modified accordingly to take 4 channels as input. The final 2D feature maps have 256 channels, front-side and back-side, respectively. Meanwhile, The 3D encoder is from PaMIR for the voxel-aligned feature. The final 3D feature maps have 32 channels. Also, our MLPs are based on the MLPs of PaMIR. Since there are front-/back-side RGB inputs and depths, our MLPs have [544(256+256+32), 1024, 512, 256, 128, 1] channel dimensions.

1.5. Texture Prediction Network

Our texture prediction network is composed of a 2D encoder, a 3D encoder, and MLPs, which is similar to Section 1.4. The 2D encoder is from the CycleGAN [11] same as PIFu, and the 3D encoder is same as Section 1.4. The final 2D front-/back-side feature maps and 3D feature maps have 256, 256, and 32 channel dimensions, respectively. With the feature maps of the texture prediction branch, the MLPs use the 2D feature maps from Section 1.4. Therefore, the MLPs have [1056(512+512+32), 1024, 512, 256, 128, 5] channel dimensions.

2. Additional Qualitative Results

In this section, we present additional results of state-of-the-art methods for unseen data in Figure 1–6. All methods requiring the SMPL [3] model use a pre-trained GCMR [2]. The texture branch of ICON is implemented in that of PaMIR, because ICON use the SMPL model. Also, Pix2PixHD [6] is adopted for clothed normal generation networks of ICON. All geometry branches were trained for 10 epochs, while the texture branches for 5 epochs, and 495 scans from THuman2.0 [9] were used for training dataset. The pre-/post-optimization algorithms of ICON are adopted and they are repeated 200 and 300 iterations with the stochastic gradient descent optimizer, respectively. For other methods, we inspect faces of output mesh and remove isolated noise. Vertices connected less than 20% of the total number of vertices are considered noise and deleted.

3. Projection without λ

We introduce λ in the main manuscript to project the D^F and D^B into the center of the V. Without the scale regressor reported in Table 6, gap incurs between D^F and D^B as shown in Figure 7. Since our model is dependent on V, training becomes difficult without the scale regressor and λ.

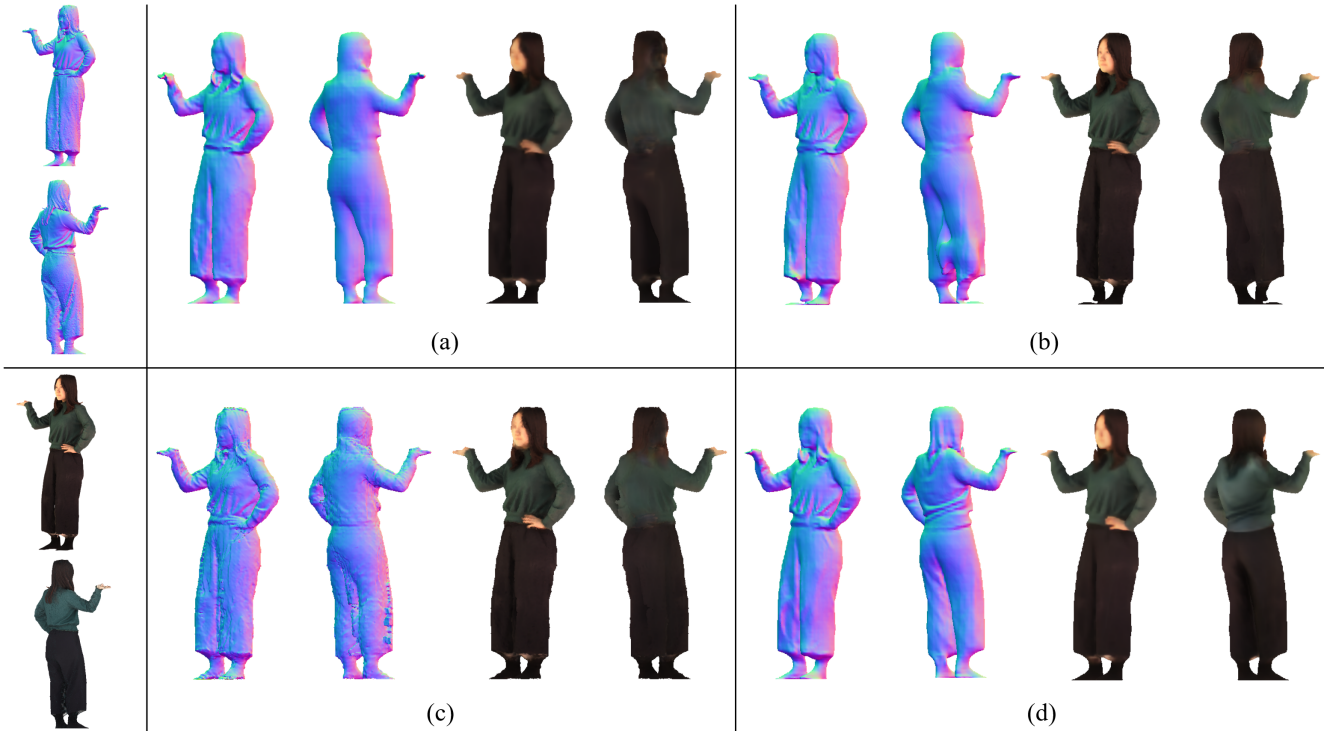

Figure 1. Additional Results of the state-of-the-art methods using a single-view RGB input. (left) Front-/back-side normal maps and RGB images from GT Mesh. (a) PIFu. (b) PaMIR. (c) ICON. (d) Ours.

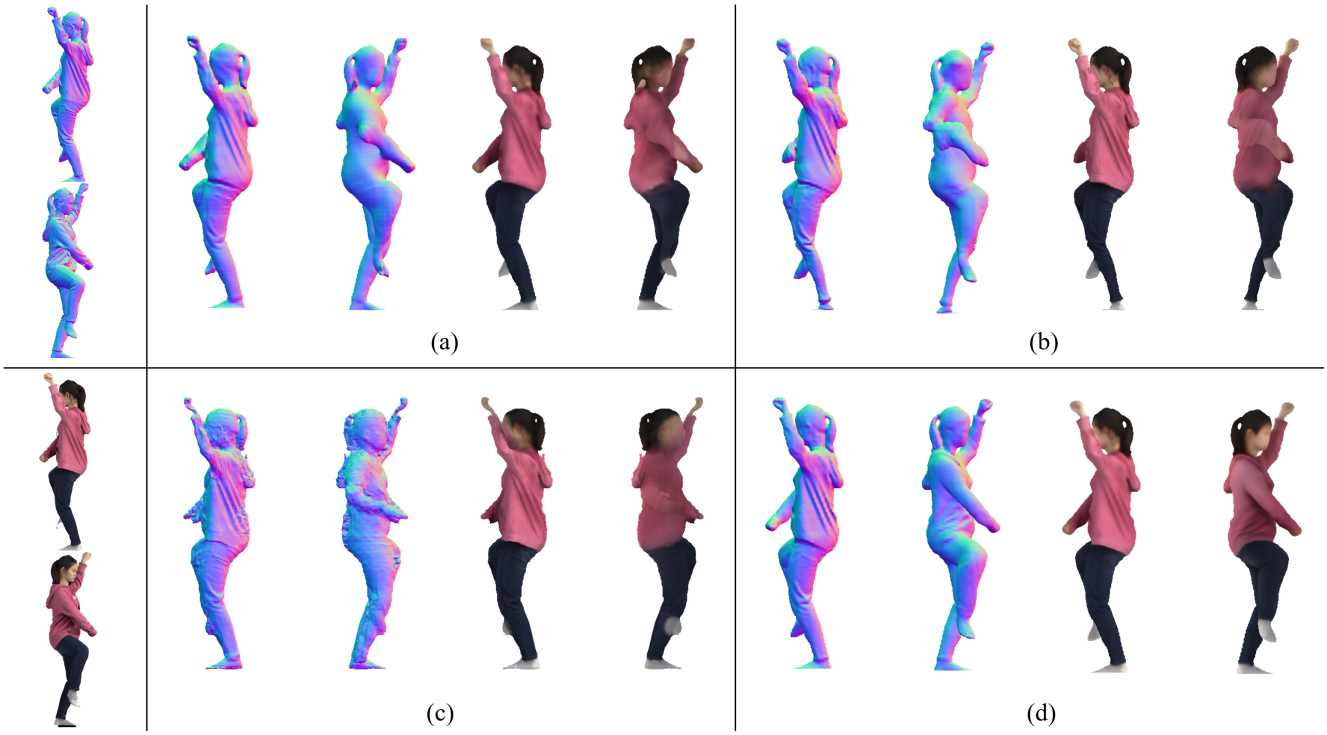

Figure 2. Additional Results of the state-of-the-art methods using a single-view RGB input. (left) Front-/back-side normal maps and RGB images from GT Mesh. (a) PIFu. (b) PaMIR. (c) ICON. (d) Ours.

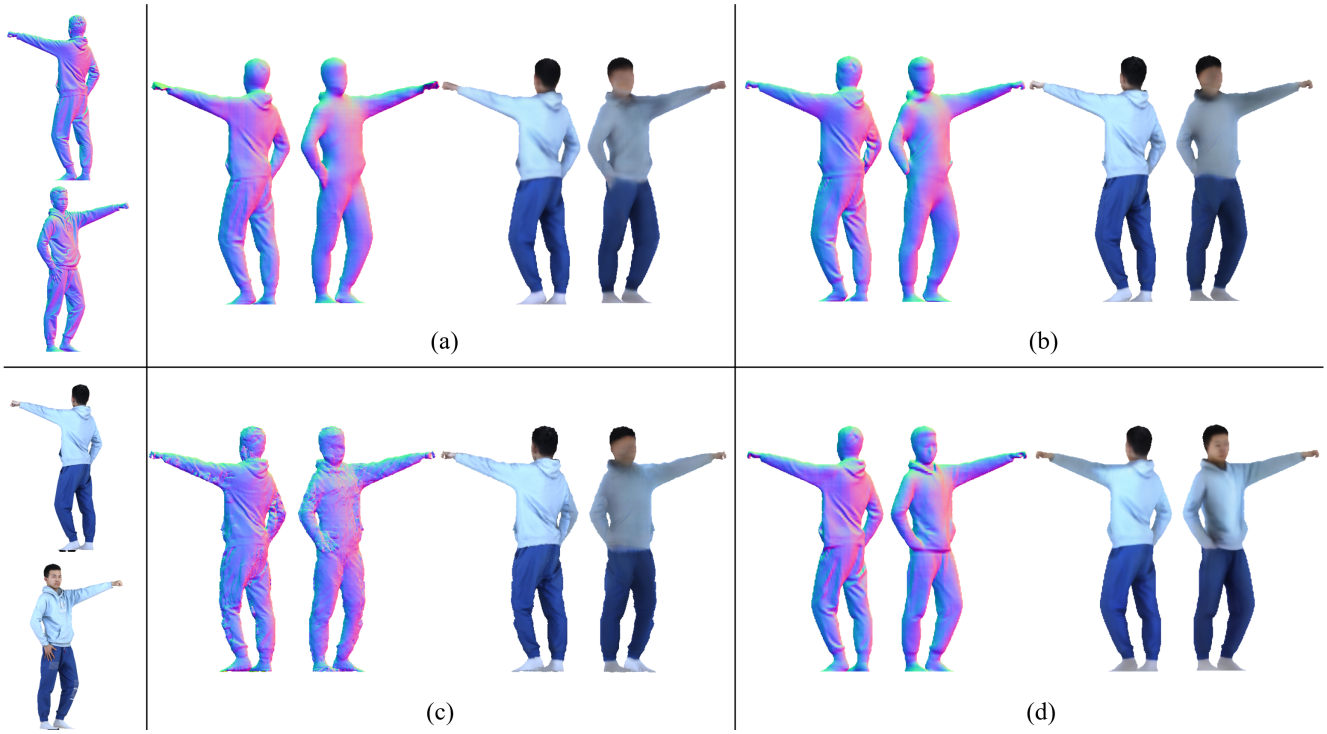

Figure 3. Additional Results of the state-of-the-art methods using a single-view RGB input. (left) Front-/back-side normal maps and RGB images from GT Mesh. (a) PIFu. (b) PaMIR. (c) ICON. (d) Ours.

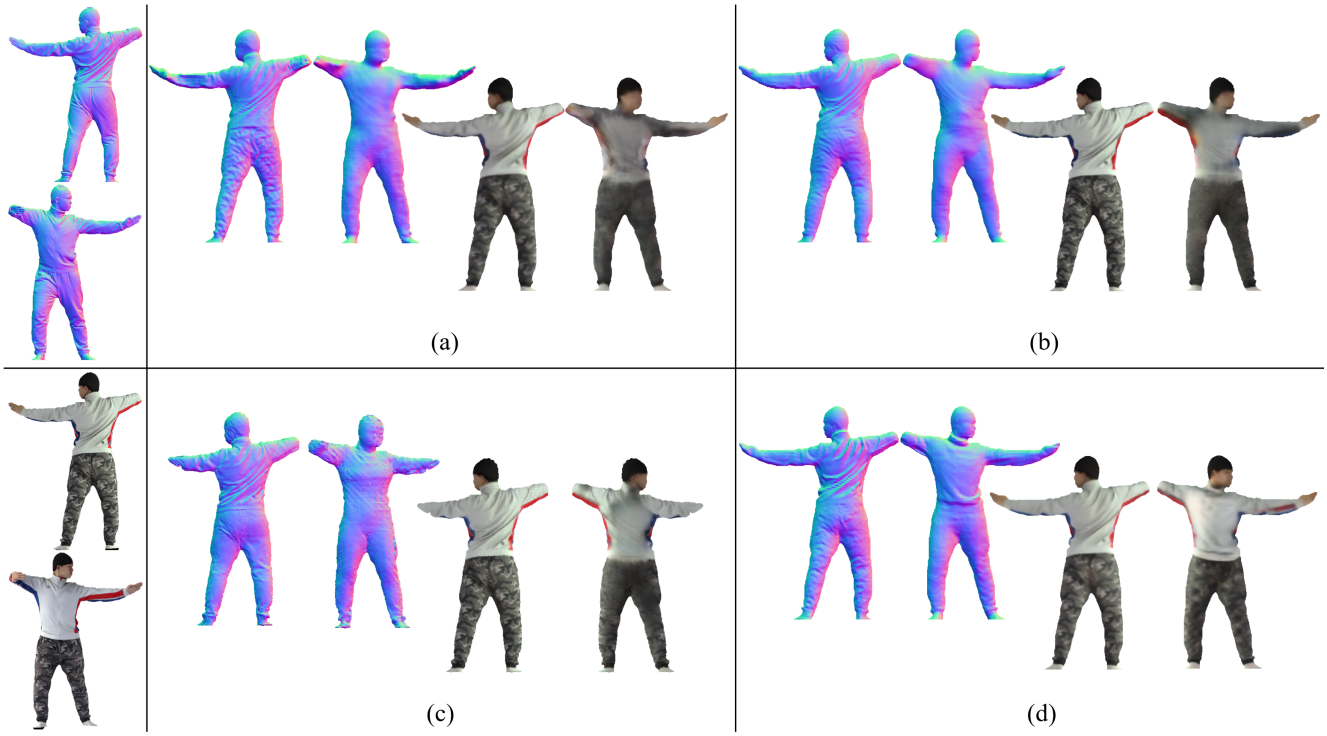

Figure 4. Additional Results of the state-of-the-art methods using a single-view RGB input. (left) Front-/back-side normal maps and RGB images from GT Mesh. (a) PIFu. (b) PaMIR. (c) ICON. (d) Ours.

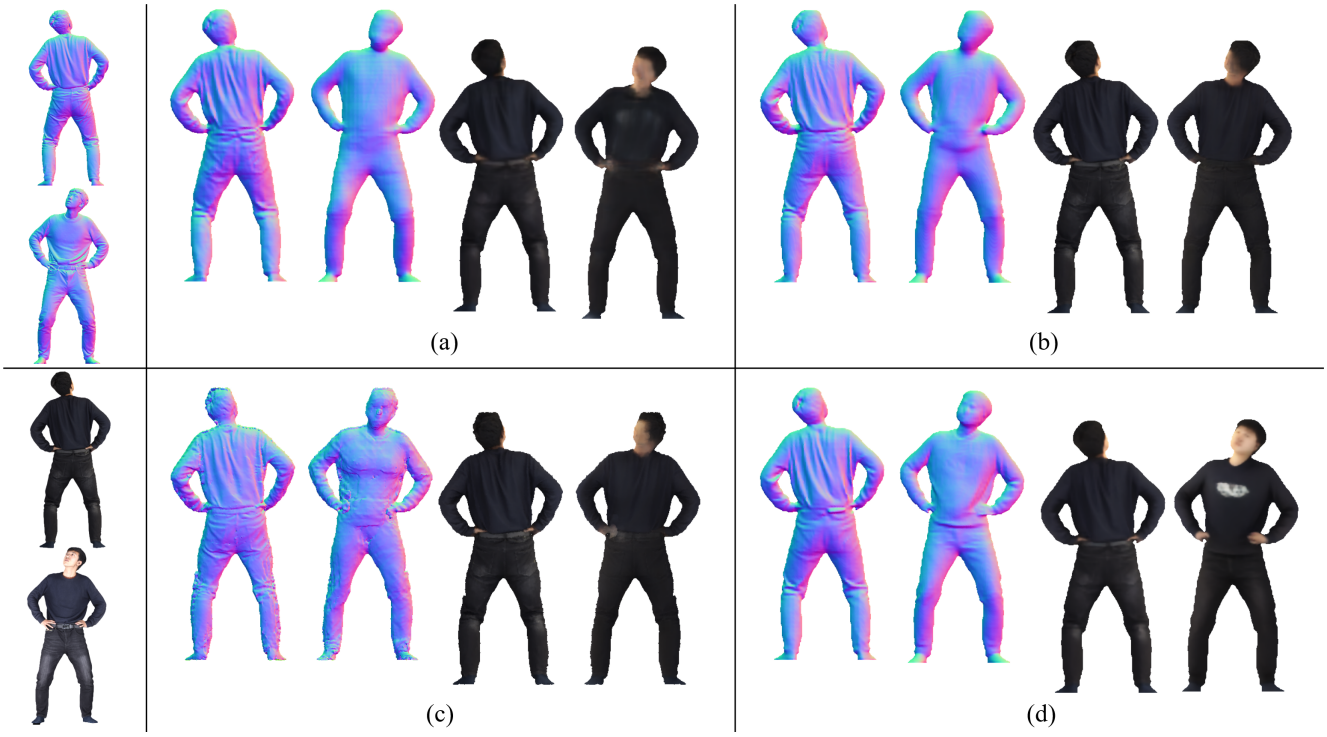

Figure 5. Additional Results of the state-of-the-art methods using a single-view RGB input. (left) Front-/back-side normal maps and RGB images from GT Mesh. (a) PIFu. (b) PaMIR. (c) ICON. (d) Ours.

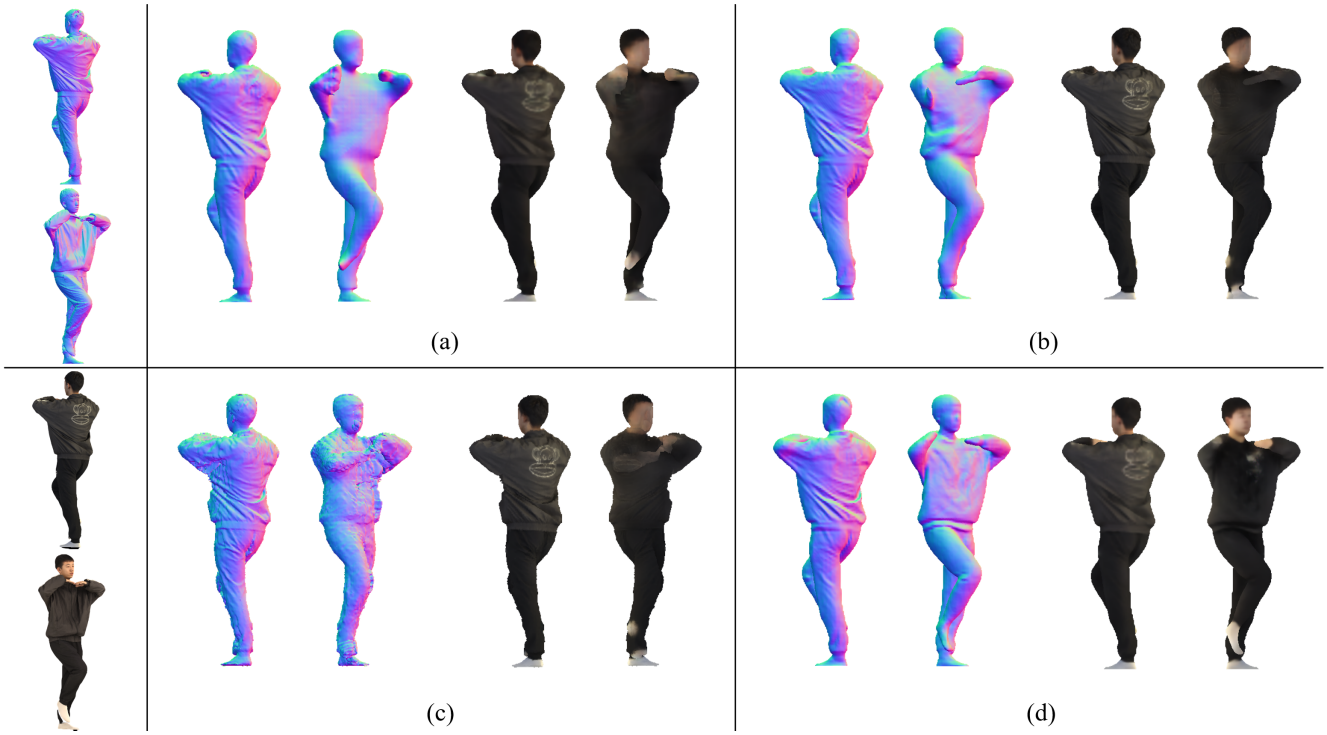

Figure 6. Additional Results of the state-of-the-art methods using a single-view RGB input. (left) Front-/back-side normal maps and RGB images from GT Mesh. (a) PIFu. (b) PaMIR. (c) ICON. (d) Ours.

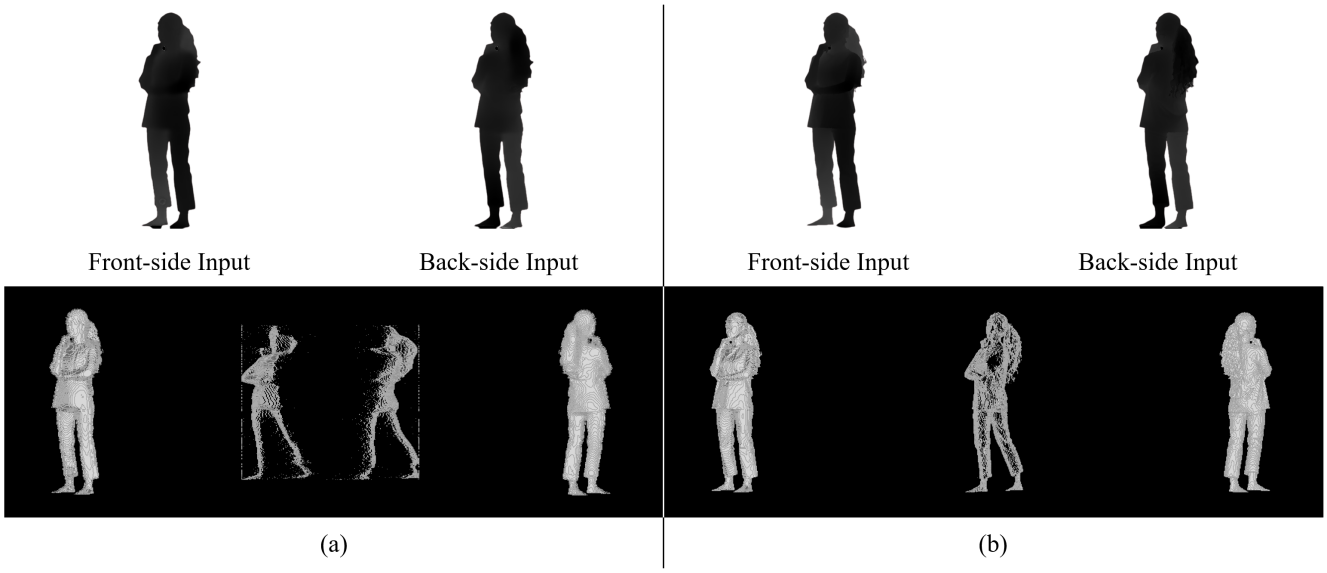

Figure 7. An Ablation experiment of λ and the scale regressor. (a) Depth projection without the scale regressor. (b) Depth projection with the scale regressor.

References

[1] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *International Conference on Learning Representation (ICLR)*, 2016. 3

[2] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. Convolutional mesh regression for single-image human shape reconstruction. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 4501–4510, 2019. 4

[3] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM Trans. on Graph. (ToG)*, 34(6):1–16, 2015. 4

[4] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *Proc. of European Conf. on Computer Vision (ECCV)*, pages 483–499. Springer, 2016. 3

[5] 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 *Proc. of Int'l Conf. on Computer Vision (ICCV)*, pages 2304–2314, 2019. 1

[6] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 8798–8807, 2018. 4

[7] Yuxin Wu and Kaiming He. Group normalization. In *Proc. of European Conf. on Computer Vision (ECCV)*, pages 3–19, 2018. 3

[8] Yuliang Xiu, Jinlong Yang, Dimitrios Tzionas, and Michael J Black. Icon: Implicit clothed humans obtained from normals. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 13286–13296. IEEE, 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 *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 5746–5756, 2021. 4

[10] Zerong Zheng, Tao Yu, Yebin Liu, and Qionghai Dai. Pamir: Parametric model-conditioned implicit representation for image-based human reconstruction. *IEEE Trans. on Pattern Anal. Mach. Intell. (TPAMI)*, 44(6):3170–3184, 2021. 1

[11] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proc. of Int'l Conf. on Computer Vision (ICCV)*, pages 2223–2232, 2017. 4