Supplementary Materials for IntegratedPIFu: Integrated Pixel Aligned Implicit Function for Single-view Human Reconstruction

Kennard Yanting Chan[1,2,3], Guosheng Lin[3], Haiyu Zhao[2], and Weisi Lin[1,3]

[1] S-Lab, Nanyang Technological University
[2] SenseTime Research
[3] Nanyang Technological University
kenn0042@e.ntu.edu.sg, zhaohaiyu@sensetime.com, {gslin,wslin}@ntu.edu.sg

1 Results of IntegratedPIFu on Real Internet Images

We provide the results of IntegratedPIFu on real Internet Images sourced from Shutterstock in Fig. 1. The obtained results is consistent with results shown in the **main paper**. Unlike PIFuHD, our IntegratedPIFu does not produce wavy lines on the reconstructed mesh, and it also avoids producing floating artifacts surrounding the reconstructed mesh. Moreover, fine features such as human fingers are correctly reconstructed by IntegratedPIFu but not PIFuHD.

2 Comparison between SDF and DOS

In this section, in order to facilitate a simpler and more direct comparison between SDF and DOS, we modify the range of values that our labels can take in DOS scheme from [0.0, 1.0] to [-0.5, 0.5]. While this range of values differs from the one mentioned in our **main paper**, the difference is trivial and will not affect our conclusions. Our Depth-Oriented Sampling (DOS) drew inspiration from Signed Distance Function (SDF) used in DeepSDF [1]. The similarities and distinctions between SDF and DOS can be seen in Fig. 2. The key difference is that, in DOS, distance is calculated based on the camera direction, but that is not the case for SDF.

A more detailed comparison between SDF and DOS is given in Fig. 3 (See p_0, p_2, p_3, p_4, p_5). The maximum and minimum values given to a point's label are 0.5 and -0.5 respectively. Sample points that are beyond these two thresholds are discarded. We will explain the magenta fonts in this figure later.

In Fig. 2 of the **main paper**, we see that PIFu uses an encoder to encode its inputs (e.g. RGB image) into a set of 128x128 2D feature maps. In Fig. 4, we see the mapping between a pixel in the RGB image, a cell on a 2D feature map (for simplicity, we illustrate only one 2D feature map, but extrapolation is trivial), and a cuboid of space in the 3D camera space (where the reconstructed mesh is generated). The cuboid's depth is aligned with the camera's direction (i.e. its depth runs along the camera direction). Theoretically, this cuboid of space

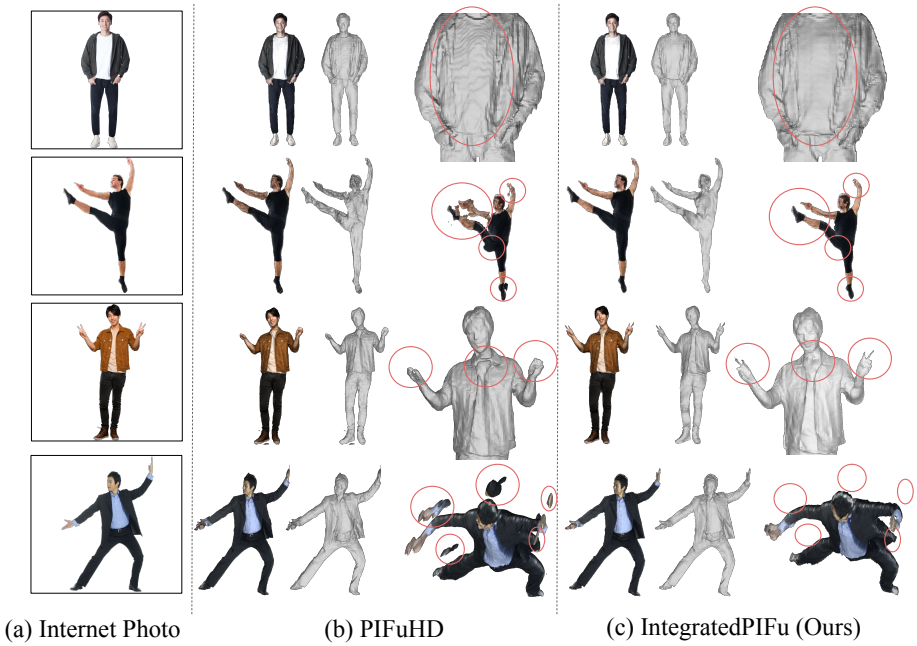

Fig. 1. Results of IntegratedPIFu on Internet Images from Shutterstock, compared with the same produced by PIFuHD

would have infinite depth. But in practice, the depth of this cuboid is finite and predefined.

Basically, each cell on the 2D feature map (more accurately, the set of 2D feature maps) is responsible for reconstructing the mesh vertices in a corresponding cuboid. This cuboid represents the 3D space in which the 2D feature map cell is tasked to reconstruct in. Each 2D feature map cell is assigned to a different cuboid of space.

We will now explain the magenta fonts in Fig. 3, which illustrates why SDF will not work well in a pixel-aligned implicit model. This figure shows two 2D feature map cells (c_1 and c_2) and they correspond to the yellow 2D feature map cells that we have seen in Fig. 4. The space between each pair of dotted magenta lines represent the space occupied by a cuboid illustrated in Fig. 4.

As aforementioned, DOS only considers distance in the camera direction. Thus, unlike DOS, SDF would use p_2 to reveal to feature map cell c_1 the presence of the protrusion, but c_1 is not responsible for reconstructing the space (recall the cuboid of space in Fig. 4) containing the protrusion. This problem is worsened when there is a cliff (See c_2 and p_4). There is no mesh surface or mesh vertices for c_2 to reconstruct, but p_4 is saying otherwise. Concretely, sample point p_4's label in SDF is saying there is a mesh surface that is moderately near to this sample point, and c_2 is advised by p_4 to be prepared to reconstruct this mesh surface that is supposedly 'near' p_4. But as we can see in Fig. 3, the cuboid of

space that corresponds to feature map cell c_2 is totally empty, and c_2 should not be advised to reconstructed anything.

These issues are not encountered in DOS, where point p_2 correctly inform cell c_1 of the location of the mesh surface that the cell is supposed to reconstruct. Similarly, in DOS, p_4 tells c_1 that there is likely to be nothing to reconstruct.

To see concrete results, refer to Fig. 5 and Tab. 1. Fig. 5 shows that a PIFuHD trained with SDF would generate spike-like artefacts near the boundary of the reconstructed human mesh, and this is consistent with our theoretical analysis of using SDF in a pixel-aligned implicit model (recall the aforementioned issue with p_2 and p_4 in Fig. 3 when we use SDF). In contrast, a PIFuHD trained with DOS will not produce such spike-like artefacts. In Tab. 1, we observe that SDF performed significantly worse compared to DOS. The reason behind this is due to the spike-like artefacts generated when SDF is used.

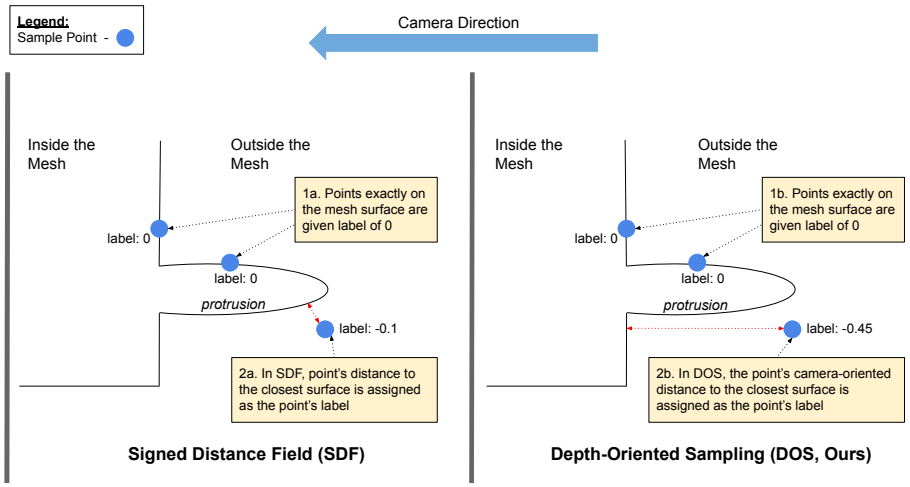

Fig. 2. Comparison between SDF and DOS. In both SDF and DOS, sample points that are outside of the mesh are given negative labels, and sample points inside the mesh are given positive labels. The figure assumes that both SDF labels and DOS labels range from -0.5 to 0.5

Table 1. Quantitative comparison between a PIFuHD trained using SDF and using DOS in the THuman2.0 test set

Methods	CD(10^{-4})	P2D(10^{-4})
PIFuHD with SDF	4.609	6.481
PIFuHD with DOS	**2.638**	**2.167**

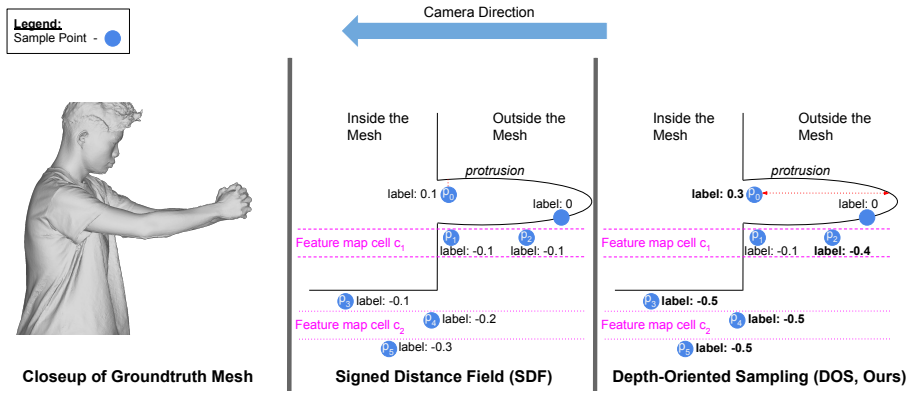

Fig. 3. A more detailed comparison between SDF and DOS. Some labels are bold-faced to show the difference between SDF and DOS

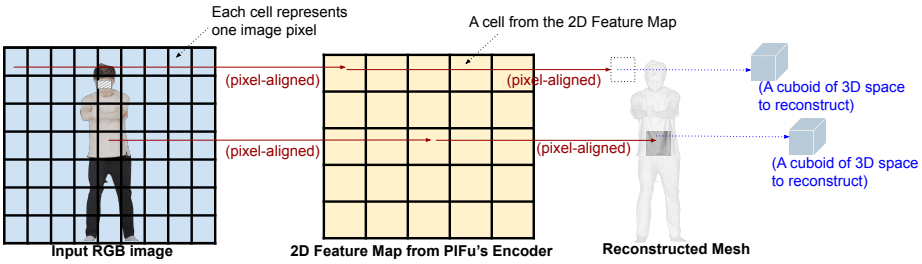

Fig. 4. Meaning of pixel alignment in pixel-aligned implicit models

3 More on the Effect of Depth maps and Human parsing maps in IntegratedPIFu

In the ablation studies done in the main paper, we showed that incorporating depth and human parsing maps into the backbone of an IntegratedPIFu can make a significant difference in terms of the structural fidelity of the reconstructed human meshes generated by the backbone. We did that by showing readers the meshes reconstructed by backbones that are given different combinations of depth and human parsing maps. We assured readers that the problems in the backbones will almost always be propagated to the final reconstructions produced by IntegratedPIFu (or PIFuHD). Due to space constraints in that paper, we are unable to provide comprehensive evidence on this claim. Thus, here, we will show the final human meshes that are reconstructed by an IntegratedPIFu (not by just its backbone) when it is given the different backbones. We show the results in Fig. 6. The human subjects used in this figure are deliberately chosen to align with the human subjects used in Fig. 5 in the **main paper**. We do this so that readers can use the two figures and see for themselves that when

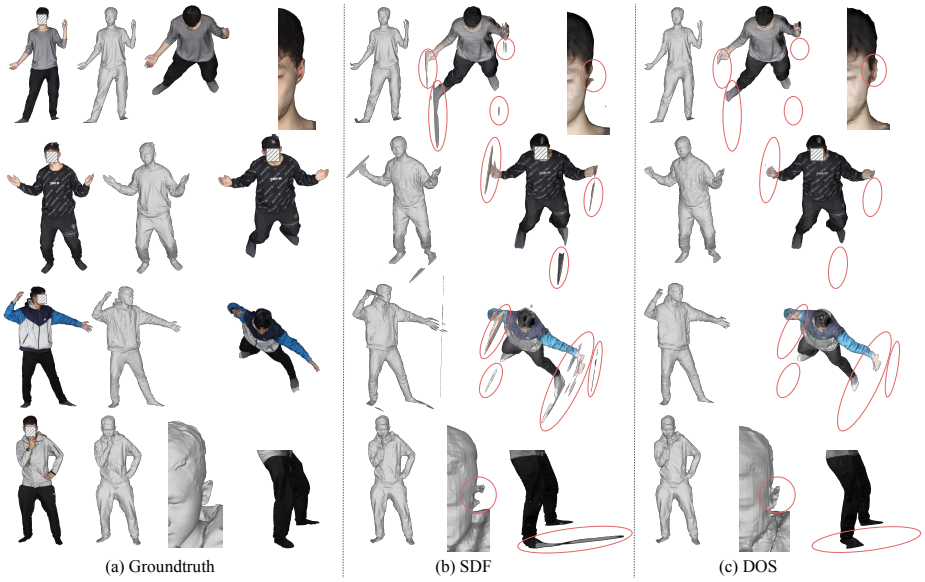

Fig. 5. Qualitative Comparison between using SDF and DOS on a PIFuHD

the backbone of IntegratedPIFu produces an error, that error would usually be replicated by the IntegratedPIFu itself.

In the first row of Fig. 6, we observe that none of the models produced any broken limbs. But we notice that an IntegratedPIFu that is given neither depth nor human parsing maps (i.e. Vanilla IntegratedPIFu) produced an overly large left arm and left hand. The slim wrist of the human subject is also not captured by this IntegratedPIFu. In the second row, both (b) the Vanilla IntegratedPIFu and (d) an IntegratedPIFu that uses only the human parsing maps erroneously produced a floating right fist. These two models also produced noisy artefacts in the third row of the figure. In the fourth row, the Vanilla IntegratedPIFu is unable to reconstructed the right arm correctly as half of the forearm sank into the shirt of the human mesh. Thus, as aforementioned, most of the errors in Fig. 5 in the **main paper** are replicated in Fig. 6.

Also, we like to highlight that although the reconstructed outputs produced by an (c) IntegratedPIFu with only depth maps can appear similar to that produced by an (e) IntegratedPIFu with both depth and human parsing maps, there can actually be subtle differences. For example, if we look at the third row of Fig. 6, we see that left arm produced by (c) is unnaturally curvy. In contrast, the left arm reconstructed by (e) looks more natural and human-like. There are other subtle differences such as a small floating artefact near the left elbow reconstructed by (c) in the second row of the figure. Hence, we believe that incorporating both depth maps and human parsing in an IntegratedPIFu is a more robust strategy than using only depth maps.

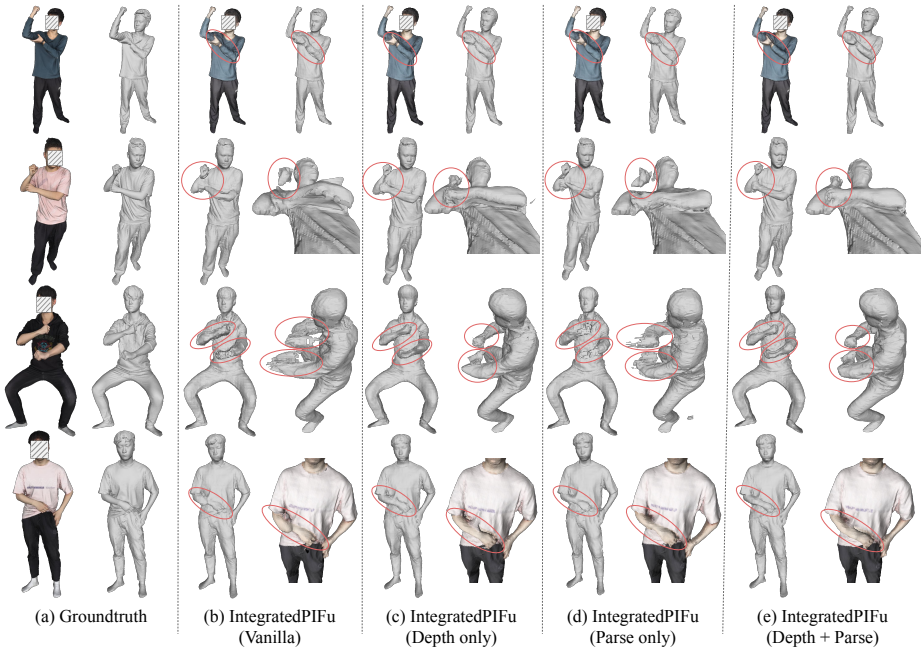

Fig. 6. Comparison of the final outputs produced by IntegratedPIFu when it is given (b) neither depth nor human parse maps, (c) depth maps only, (d) human parse maps only, (e) both depth and human parse maps. The human subjects used in this figure are deliberately chosen to align with the human subjects used in Fig. 5 in the **main paper**

4 Depth Prediction

Here, we will discuss techniques which have helped to improve the accuracy of our depth predictor.

4.1 Using Predicted Normal maps in Relative Depth Prediction

Similar to [2], our IntegratedPIFu uses a pix2pixHD network [3] to predict a frontal normal map from a RGB image.

We find that using this predicted frontal normal map as an additional input to the depth predictor helps to significantly improves the accuracy of the depth predictor's outputs. Using Tab. 2, we can compare the first row with the third row to see that not using the frontal normal map as an additional input would increase the average L1 error in test set by 19.8%.

Predicted frontal normal map would help in depth prediction because the normal map contains structural information of a human mesh body. For example, at the area around a male human chest, a normal map would indicate that the area is mostly flat (normal vectors are facing the camera direction), and this is

an important hint to the depth predictor to predict similar relative depth values for all pixels in this area.

4.2 Using a Center indicator map in Relative Depth Prediction

In the groundtruth relative depth maps, the center pixel would always have a value of zero since it is where the reference vertex is. Ideally, we want our predicted relative depth maps to have a zero at its center pixel as well. Thus, we use what we called a center indicator map to indicate to the depth predictor where the reference vertex is. And, consequently, the center indicator map is simply a map of zero values except its center pixel (has value of one). Our center indicator map will have the same size as the RGB image, and it is an additional input to the depth predictor. If the RGB image is 1024x1024, then no single pixel would be the center, so we will treat the center as a 2x2 pixel region.

By comparing the second and third rows in Tab. 2, we can observe the impact of using the center indicator map as an additional input to the depth predictor. When center indicator map is used, both the average L1 error and the L1 error incurred by pixels near the center fall.

Table 2. Evaluation of using different configurations in the depth predictor. Results obtained using the test set of THuman2.0 Dataset. Center error is the sum of L1 error incurred by pixels near the center of the predicted depth map (We use a 4x4 region at the center of the predicted depth map to compute the Center error)

Methods	L1 error	Center error
One-Stage (Center)	10080	40.21
One-Stage (Normal)	8451	26.80
One-Stage (Normal + Center)	8417	21.95
Two-Stage (Normal + Center)	**8327**	**21.33**

4.3 Using a Two-staged depth prediction process

The depth predictor consists of two separate U-Nets and predict relative depth in a two-staged process. In the first stage, the first U-Net would take a RGB image, a predicted frontal normal map, and center indicator map as inputs, and it produces a coarse relative depth map. In the second stage, the second U-Net refines the coarse relative depth map and generates a refined relative depth map. The second U-Net is given three inputs, namely a RGB image, a predicted frontal normal map, and a coarse relative depth map.

In Tab. 2, we see the importance of having this two-staged process. By comparing the third and fourth rows, we see that the two-stage process is able to reduce the L1 error of the generated depth maps significantly. Moreover, it also has an effect of reducing the L1 error incurred by the center pixels.

5 Human Parsing Prediction

Here, we will discuss techniques which have helped to improve the accuracy of our human parsing predictor.

5.1 Using Predicted Normal maps in Human Parsing Prediction

The human parsing predictor is a U-Net that takes in a RGB image and a predicted frontal normal map as inputs, and it outputs a corresponding human parsing map. Similar to the depth predictor, we find that giving the human parsing predictor the predicted frontal normal map as an additional input helps to improve its accuracy (See Tab. 3).

The reason for this could be that the frontal normal maps provide the human parsing predictor with structural information of the human body (e.g. orientation of the different human body parts). We believe that this information could have made it easier for the human parsing predictor to perform its task.

Table 3. Evaluation of using different configurations in the human parsing predictor. Results are obtained using the test set of THuman2.0 Dataset. The error metric used in the table is the average number of pixels that are misclassified in each predicted human parsing map

Methods	Misclassified pixels
No Normal	13238.6
Normal	**12500.2**

6 Implications of using Rear Normal maps in a Pixel-aligned implicit model

Unlike PIFuHD [2], IntegratedPIFu does not use a rear normal map as an input. The reason for this is that we do not feel it is practical to expect a model (i.e. a Rear Normal Predictor) to predict an accurate rear normal map on unseen data when the model is only given a frontal RGB image as input. To illustrate this, we show a pair of groundtruth rear normal maps from test data and compare them with the rear normal maps predicted by a Rear Normal Predictor in Fig. 7. The Rear Normal Predictor is implemented the same way as in [2], and it is also used in PIFuHD. As seen in the Fig. 7, the Rear Normal Predictor would hallucinate creases and wrinkles in random locations, and most of these hallucinations are incorrect. This is expected because the Rear Normal Predictor has no means of knowing where the wrinkles and creases would be located just by looking at a frontal RGB image.

If a pixel-aligned implicit model, such as PIFuHD or IntegratedPIFu, uses a rear normal map as input, then the human meshes reconstructed by the model would usually have hallucinated creases and wrinkles on their rear as well. This

is illustrated in Fig. 8b. The figure (Fig. 8) is produced by training two separate PIFuHDs, where one of them is given rear normal maps, and the other one is not given rear normal maps. From this figure, we see that when a rear normal map is not given, the PIFuHD simply produces a smooth back/rear for the reconstructed human mesh. Strictly speaking, neither using rear normal maps nor not using rear normal maps would correctly reconstruct the rearside wrinkles and creases. The choice to include or exclude rear normal maps may depend on the specific application scenarios of the pixel-aligned implicit model.

7 Implementation details of IntegratedPIFu

Our code is available at https://github.com/kcyt/IntegratedPIFu.

The training of IntegratedPIFu is similar to that of PIFuHD. We first train a Frontal Normal Predictor to predict 1024x1024 frontal normal maps from 1024x1024 RGB images. The architecture of the Frontal Normal Predictor is the same as the one used in [2]. In addition, we also train our Depth Predictor and Human Parsing Predictor to predict 1024x1024 relative depth maps and 1024x1024 human parsing maps from 1024x1024 RGB images and 1024x1024 predicted frontal normal maps. In total, we will have predicted frontal normal maps, predicted relative depth maps, and predicted human parsing maps.

The IntegratedPIFu consists of the Low-Resolution PIFu (Low-PIFu) and the High-Resolution Integrator (HRI). The Low-PIFu is first trained with four downsampled inputs: 1. 512x512 RGB images 2. 512x512 predicted frontal normal maps 3. 512x512 predicted depth maps 4. 512x512 predicted human parsing maps. We use spatial sampling scheme to train Low-PIFu.

After the Low-PIFu is trained, we would start alternative training of the Low-PIFu and the HRI. HRI is given two inputs: 1. 1024x1024 RGB images 2. 1024x1024 predicted frontal normal maps. During the alternative training, we would use only depth-oriented sampling scheme to train both Low-PIFu and HRI. After the end of the alternative training, we would finetune the HRI by training only the HRI for five epochs.

For both the spatial sampling scheme and the depth-oriented sampling scheme, we use 16000 sample points for each RGB image. For depth-oriented sampling, we reserve 10% of the sample points to be extreme points (either deeply inside a mesh with respect to the camera direction or outside and very far away from the mesh surface in the camera direction). These extreme points will have a label of one if they inside the mesh, and a label of zero if they are outside.

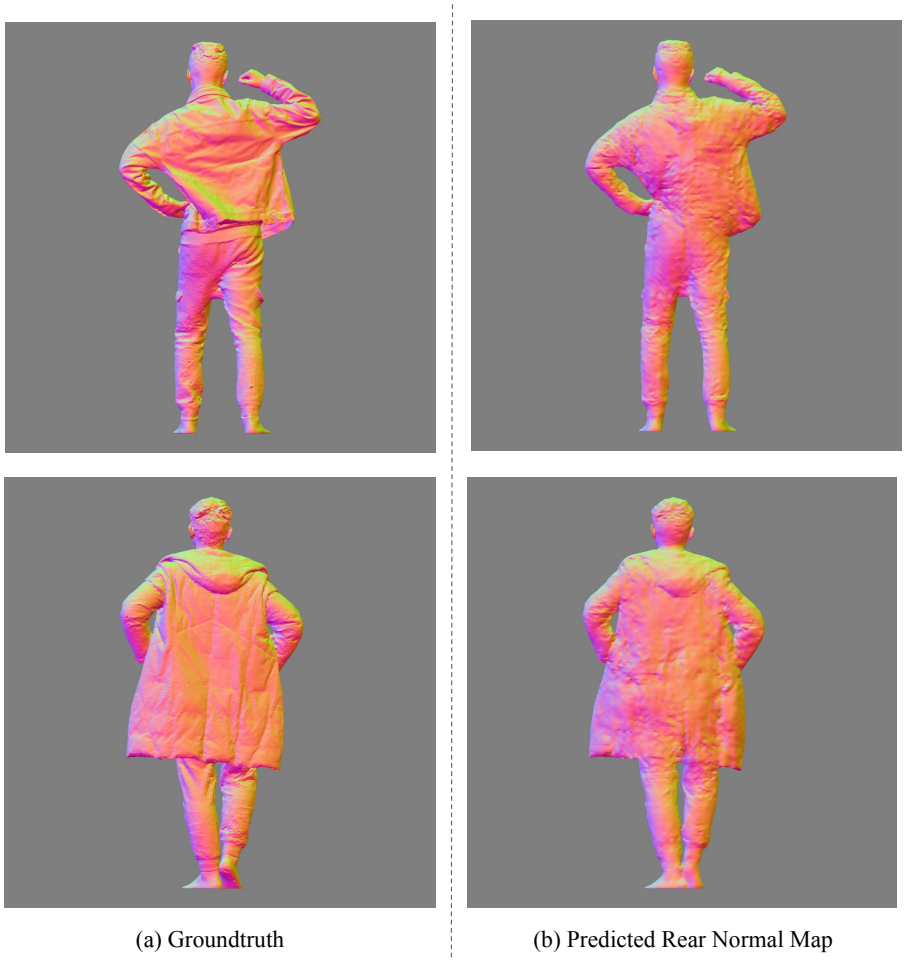

Fig. 7. Comparison of the groundtruth rear normal maps and the predicted rear normal maps in the test dataset. It is impractical to expect the Rear Normal Predictor to predict rear normal maps accurately with just the frontal RGB image as input

Fig. 8. Comparison of the outputs produced by a PIFuHD that is given predicted rear normal maps and a PIFuHD that is not given predicted rear normal maps. Results are from test data

References

1. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 165–174 (2019)
2. Saito, S., Simon, T., Saragih, J., Joo, H.: 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. pp. 84–93 (2020)
3. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8798–8807 (2018)