Deformable Mesh Transformer for 3D Human Mesh Recovery Supplementary materials

Yusuke Yoshiyasu
National Institute of Advanced Industrial Science and Technology (AIST)
1-1-1 Umezono, Tsukuba, Japan
yusuke-yoshiyasu@aist.go.jp

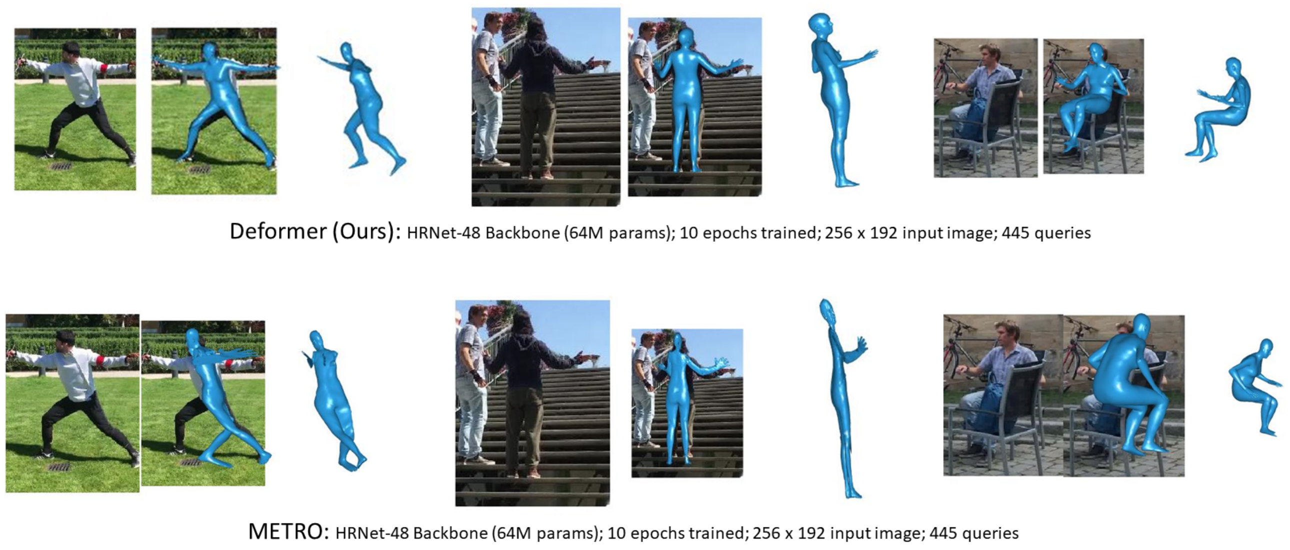
Deformer (Ours): HRNet-48 Backbone (64M params); 10 epochs trained; 256 x 192 input image; 445 queries
METRO: HRNet-48 Backbone (64M params); 10 epochs trained; 256 x 192 input image; 445 queries

Figure 1. Qualitative comparison with METRO [5].

1. Additional evaluation results

1.1. Qualitative comparison against METRO [5]

We qualitatively compare DeFormer and METRO [5] in Figs. 1 and 8-10. Here we constructed the METRO model using the HRNet-w48 backbone model. We trained it on multiple datasets for 50 epochs and fine-tuned for 10 epochs following the instruction provided in METRO's repository. We obtained similar 3DPW scores as reported in their repository. METRO produces the results with large errors on some of the difficult cases, such as large shape distortions, large image-to-mesh miss-alignments, orientation flips and confusions in left/right legs, whereas DeFormer fits a mesh model more stably for these cases (Figs. 1 and 9). DeFormer is also better at recovering subtle poses such as head, hand and foot poses than METRO (Figs. 8 and 10.These results show that DeFormer performs better in 3D human mesh recovery than METRO by exploiting multi-scale feature maps that contain global and local spatial contexts.

1.2. 2D keypoint detection on COCO dataset

We evaluated the mesh-to-image alignment ability of DeFormer based on the 2D keypoint detection task on COCO dataset. We follow the same evaluation protocol provided by PyMAF [11]. As shown in Table 2, DeFormer performs the best among the human mesh recovery approaches [3–5, 11].

1.3. Generalization to hand mesh recovery

To evaluate the generalization ability of DeFormer to other mesh recovery tasks, we trained and tested on the FreiHAND dataset [2]. DeFormer achieves the best performance among the transformer approaches [1, 5, 6]. In the

Table 1. Comparisons of transformer-based mesh recovery approaches on FreiHAND. Test time augmentation is not used for these results.

Method	PA-MPJPE ↓	PA-MPVPE ↓	F@5mm ↑	F@15mm ↑
METRO-HRNetW64 [5]	6.7	6.8	0.717	0.981
FastMETRO-L-HRNetW64 [1]	6.5	—	—	0.982
MeshGraphomer-HRNetW64 [6]	6.3	6.5	0.738	0.983
DeFormer-TransformerSA-HRNetW48	**6.2**	**6.4**	**0.743**	**0.984**
DeFormer-BodySparseSA-HRNetW48	6.3	6.6	0.729	0.984

Table 2. Comparison of 2D keypoint detection performance on COCO dataset. Numbers for [3, 4] are taken from [11].

Method	AP	AP50	AP75	APM	APL
GraphCMR [4]	9.3	26.9	4.2	11.3	8.1
SPIN [3]	17.3	39.1	13.5	19.0	16.6
PyMAF [11]	24.6	48.9	22.6	25.7	24.1
METRO [5]	20.0	48.2	13.3	22.1	18.8
DeFormer	**28.6**	**59.9**	**24.8**	**30.7**	**27.3**

hand mesh recovery case, the full transformer self-attention performs better than BodySparse-SA possibly because the hand shape and pose are more non-locally correlated to each other than the body.

1.4. Accuracy across iterations

Figure 2 shows that our mesh alignment feedback loop in the decoder improves the quality of image-mesh alignment, such as making the head orientation correct. We also measured accuracy of DeFormer across iterations (Table 3). The results show DeFormer's feedback loop indeed improves its accuracy through iterations.

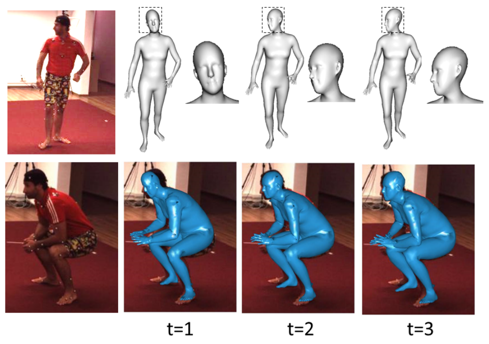

Figure 2. Visualization of reconstruction results across different iterations in the feedback loops within the transformer decoder.

Table 3. Quantitative evaluation across iterations

	$t = 1$	$t = 2$	$t = 3$
MPJPE (PA-MPJPE)	54.9 (41.2)	45.3 (33.3)	43.8 (31.2)

1.5. More comparisons on transformer approaches

We provide more comparison results on transformer-based approaches in Table. 6.

2. Additional ablation

2.1. Memory efficiency

The main contributions leading to memory efficiency of DeFomer come from: 1) the decoder-only architecture, 2) the use of deformable cross attention in DMA and 3) the learning of sparse connectivities in BodySparse-SA.

Firstly, unlike [1, 12] who used an encoder-decoder architecture, removing the encoder reduces memory costs without degrading its performance too much. In fact, when training with a batch size of 16, the memory consumption reduced approximately 7GB for a 256 $\times$ 192 input image setting and 16GB for 384 $\times$ 288 input image setting, while keeping performance drops by 0.2 PA-MPJPE only.

Secondly, with deformable cross attention in DMA, we can efficiently aggregate and exploit multi-scale image feature maps by attending only to a small set of key sampling points around reference points. For example, in the case of a 384 $\times$ 288 image and 6904 queries, attention computations can be reduced from $6904 \times (96^2 + 48^2 + 24^2 + 12^2)$ pixels with the standard transformer attention, which is prohibitive, to 6904×4 reference points with deformable cross attention.

Thirdly, memory efficiency is further improved using BodySparse-SA that exploits the sparse connectivity patterns extracted from a human body mesh model and its skeleton, which can restrict self-attention access patterns. Using 6904 queries, DeFormer with BodySparse-SA can be trained with a batch size of 30, whereas that with the standard transformer-SA can work with a batch size of up to 2. With the batch size of 2, the former uses 3.6GB and the latter uses 38GB memory. Furthermore, when training DeFormer with batch size of 16, using the approximate minimum degree (AMD) algorithm further reduces 0.6GB memory cost and the use of *sparse* v2j and j2v relations, instead of dense ones, leads to 0.2GB memory reduction.

2.2. Initial query encoding

Table 4 shows comparison between different types of query encoding. We tested the query encoding approaches using learned embedding [12] and using the 3D coordinates of a template human mesh as in METRO [5] but increasing their dimensions by MLPs or by sinusoidal functions [10] to get position embedding. For the approach using a template mesh, we also tested to construct the initial appearance feature from global 1×1 visual features from the encoder. As shown in Table 4 using learned embedding for both appearance feature and position embedding works the best.

Table 4. Ablation on initial query encoding

Query encode	MPJPE↓	PA-MPJPE ↓
Template (MLP)	45.6	31.9
Template (Sine)	46.5	32.4
Template + global feat.	45.2	31.8
Learned embed.	44.8	31.6

2.3. Query and reference point refinement

We tested the models with and without refinement of query and reference points. As shown in Table 5, refining reference points in decoder layers is effective in improving performance. On the other hand, the DAB-DETR like style query updates [7] in decoder layers do not contribute in performance improvements in our case. This is probably because in our case reference points already contain the knowledge about the deformed geometry. The queries work better by carrying information about the undeformed canonical pose, in order to promote non-local image-to-mesh attentions.

Table 5. Ablation on query and reference point refinement

Refinement		MPJPE ↓	PA-MPJPE↓
Query	Ref. points		
—	—	45.2	32.6
—	✓	44.8	31.6
✓	✓	45.3	31.9

2.4. Comparison of upsampler

We use an upsampling technique similar to the one based on a linear layer proposed in [5, 6]. However, using a linear layer to perform upsampling from 1.7K vertices to 6.9K vertices requires many network parameters. We thus use a two-layer MLPs having a bottleneck with 512 hidden-layer dimension. This reduces 7M model parameters from the linear upsampler, while achieving similar upsampling quality as shown in Fig. 3. Using sparse matrices [8] to perform upsampling as in [1, 4] produces non-smooth distorted results, which usually requires projections onto SMPL body models.

As with the linear layer based upsampler, one issue using the MLP-based upsampling method is that it produces spikes and noise on a surface. This can also be seen in the results obtained using METRO that uses the linear layer upsampling method. To solve this issue we incorporate Taubin smoothing [9] in the network. Taubin smoothing is performed using a uniform Laplacian for 10 iterations with its parameters set to $\lambda_{\text{taubin}} = 0.5$ and $\mu_{\text{taubin}} = -0.53$. As shown in Fig. 4, these spikes can be removed to an almost negligible range by applying Taubin smoothing.

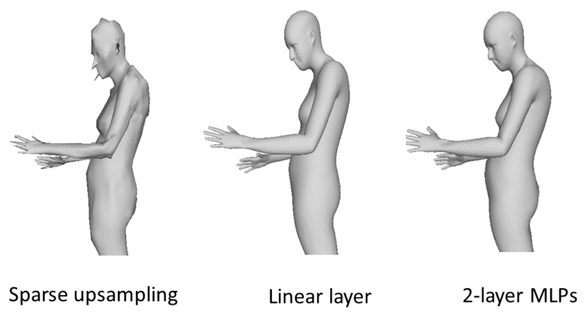

Figure 3. Comparison between upsampling layers.

2.5. Comparison between different mesh downsampling levels

Figure 5 compares the results using different number of queries (445, 1737 and 6904). Overall, the results obtained using different number of queries do not show large differences in recovered poses as shown in Fig. 5 top. On the other hand, using more queries, recovered surfaces are more favorable on some poses with large flexions and extensions of arms, which shows less shrinking around arms (Fig. 5 bottom).

3. Visualization of attention patterns

3.1. Block sparse self-attention

Figure 7 shows sparse attention patterns and their block sparsity patterns in different mesh resolutions. By combining joint-to-joint, joint-to-vertex, vertex-to-joint and vertex-to-vertex connectivity patterns, nonzero block densities can be reduced up to 4.2% (the third column). Using the approximate minimum degree algorithm (AMD) further reduces the density of nonzero blocks by 40-50% from when without AMD (the fourth column).

3.2. MS deformable cross attention

In Fig. 6, we plot the sampling points and attention weights from feature maps of different resolutions in one picture corresponding to each joint query. The sampling

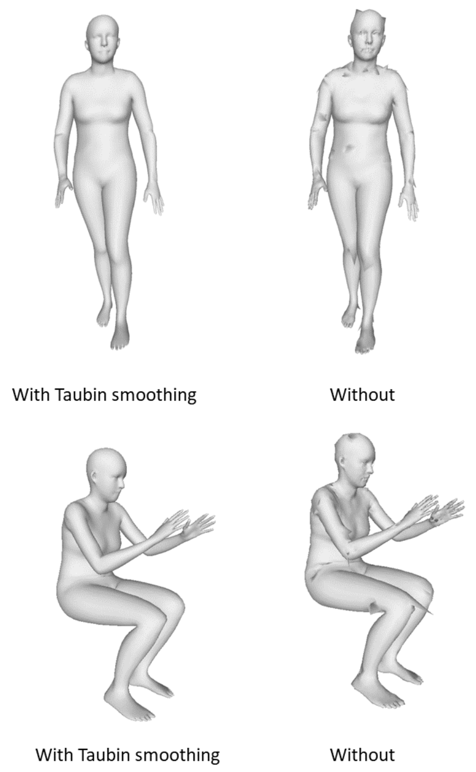

Figure 4. Taubin smoothing can remove surface spikes and noise.

points are relatively concentrated around the joint corresponding to the query but are also somewhat scattered around it. This allows us to capture non-local interactions between image points and mesh points efficiently.

4. More qualitative results

More qualitative results are shown in Fig. 11.

5. Limitations

One of the limitations of our work is that, as with other human mesh recovery techniques, it is difficult for our method to reconstruct body shapes for the body styles that are extremely different from SMPL labels, such as the cases of children in Fig. 12.

Acknowledgement

The author would like to thank Louise Allain and Chun-Kwang Tan for proofreading.

References

[1] Junhyeong Cho, Kim Youwang, and Tae-Hyun Oh. Cross-attention of disentangled modalities for 3d human mesh recovery with transformers. In *ECCV*, 2022. 1, 2, 3, 5

[2] Jimei Yang Bryan Russel Max Argus Christian Zimmermann, Duygu Ceylan and Thomas Brox. Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 1

[3] Nikos Kolotouros, Georgios Pavlakos, Michael J Black, and Kostas Daniilidis. Learning to reconstruct 3d human pose and shape via model-fitting in the loop. In *ICCV*, 2019. 1, 2

[4] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. Convolutional mesh regression for single-image human shape reconstruction. In *CVPR*, 2019. 1, 2, 3

[5] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. In *CVPR*, 2021. 1, 2, 3, 5

[6] Kevin Lin, Lijuan Wang, and Zicheng Liu. Mesh graphormer. In *ICCV*, 2021. 1, 2, 3, 5

[7] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. DAB-DETR: Dynamic anchor boxes are better queries for DETR. In *International Conference on Learning Representations*, 2022. 3

[8] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J Black. Generating 3d faces using convolutional mesh autoencoders. In *ECCV*, pages 704–720, 2018. 3

[9] Gabriel Taubin, Tong Zhang, and Gene Golub. *Optimal surface smoothing as filter design*, volume 1064 of *LNCS*, pages 283–292. 1996. 3

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NeurIPS*, volume 30, 2017. 3

[11] Hongwen Zhang, Yating Tian, Xinchi Zhou, Wanli Ouyang, Yebin Liu, Limin Wang, and Zhenan Sun. Pymaf: 3d human pose and shape regression with pyramidal mesh alignment feedback loop. In *ICCV*, 2021. 1, 2

[12] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 2, 3

Table 6. Comparison of mesh transformer approaches using different backbones. Num. of vertex/joint queries are 431 and 14, respectively.

Method	Epochs	Input size	Feat. resolution	Backbone	Backbone Pre-train dataset	H36M MPJPE $\downarrow$ (PA-MPJPE $\downarrow$)	#Params Total / Backbone	Inference FPS
METRO [5]	200	224×224	1	ResNet50	ImageNet	56.5 (40.6)	125.8M / 23.5M	32
	200	224×224	1	HRNet-w40	ImageNet	55.9 (38.5)	159.8M / 57.5M	—
	200	224×224	1	HRNet-w64	ImageNet	54.0 (36.7)	230.4M / 128.1M	14
	50	224×224	1	ResNet50	ImageNet	58.3 (42.2)	125.8M / 23.5M	32
	50	224×224	1	ResNet50	MPII	58.5 (42.8)	125.8M / 23.5M	32
	50	256×192	1	HRNet-w48	COCO	48.9 (35.8)	165.2M / 63.6M	16
Graphormer [6]	200	224×224	$\{7, 1\}$	HRNet-w64	ImageNet	51.2 (34.5)	226.5M / 128.1M	14
FastMETRO [1]	60	224×224	7	ResNet50	ImageNet	53.9 (37.3)	48.4M / 23.5M	35
	60	224×224	7	HRNet-w64	ImageNet	52.2 (33.7)	153.0M / 128.1M	14
DeFormer	50	256×256	$\{64,64\}$	Hourglass$\times$4	MPII	54.7 (39.2)	32.7M / 2.7M	25
	50	224×224	$\{28,14,7,1\}$	ResNet50	ImageNet	57.5 (41.1)	54.5M / 23.5M	41
	50	224×224	$\{56,28,14,7,1\}$	ResNet50	MPII	54.5 (39.1)	54.5M / 23.5M	41
	50	224×224	$\{56,28,14,7,1\}$	ResNet50	COCO	50.7 (36.3)	54.5M / 23.5M	41
	50	224×224	$\{28,14,7,1\}$	HRNet-w32	MPII	49.9 (35.7)	58.3M / 28.5M	21
	50	224×224	$\{56, 28,14,7,1\}$	HRNet-w32	MPII	49.5 (34.8)	58.3M / 28.5M	21
	50	256×192	$\{32,16,8,1\}$	HRNet-w48	COCO	46.6 (33.1)	93.5M / 63.6M	18
	50	256×192	$\{64,32,16,8,1\}$	HRNet-w48	COCO	**44.8 (31.6)**	93.5M / 63.6M	18
	50	384×288	$\{48,24,12,1\}$	HRNet-w48	COCO	51.4 (36.4)	93.5M / 63.6M	15
	50	384×288	$\{64,48,24,12,1\}$	HRNet-w48	COCO	43.7 (30.7)	93.5M / 63.6M	15

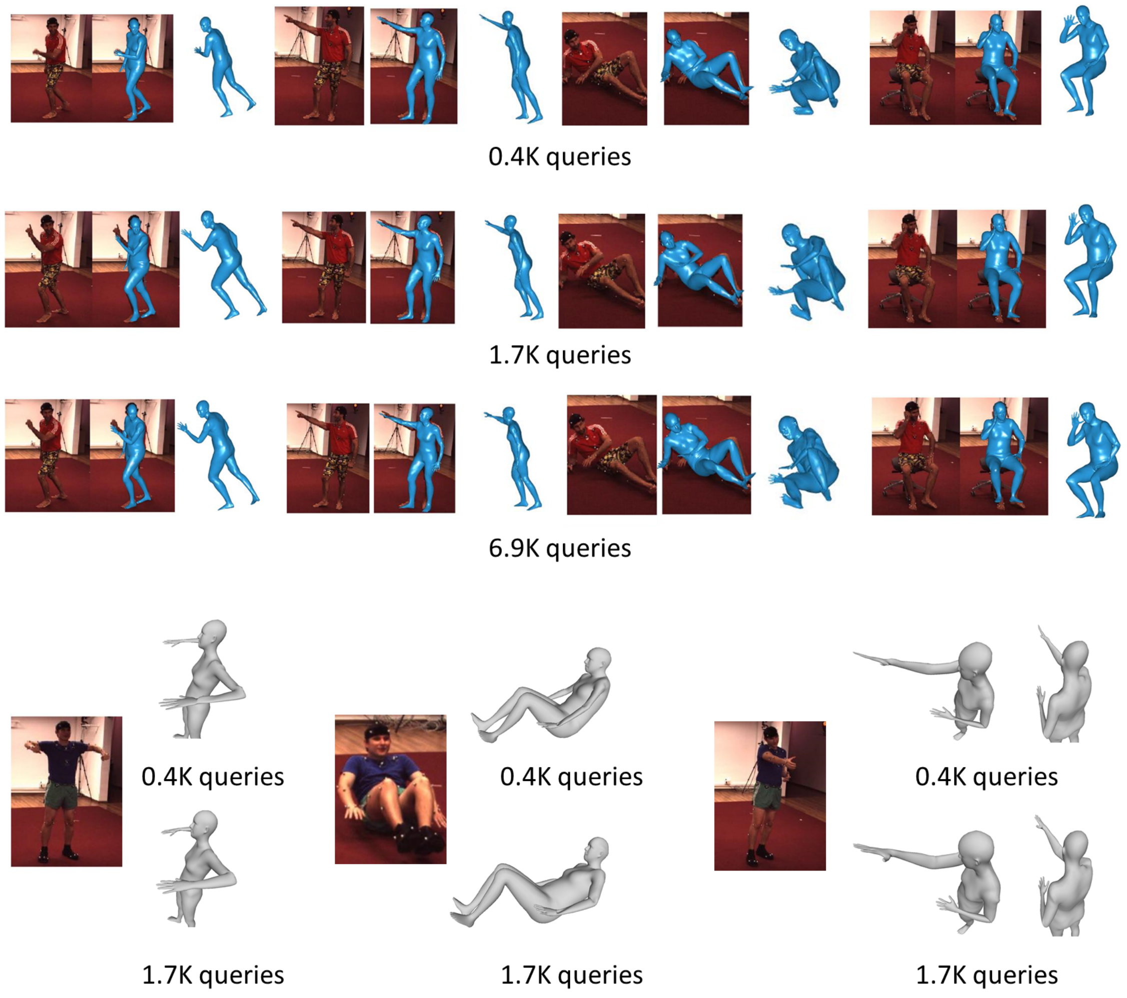

Figure 5. Comparison between results with different mesh down-sampling levels (number of queries).

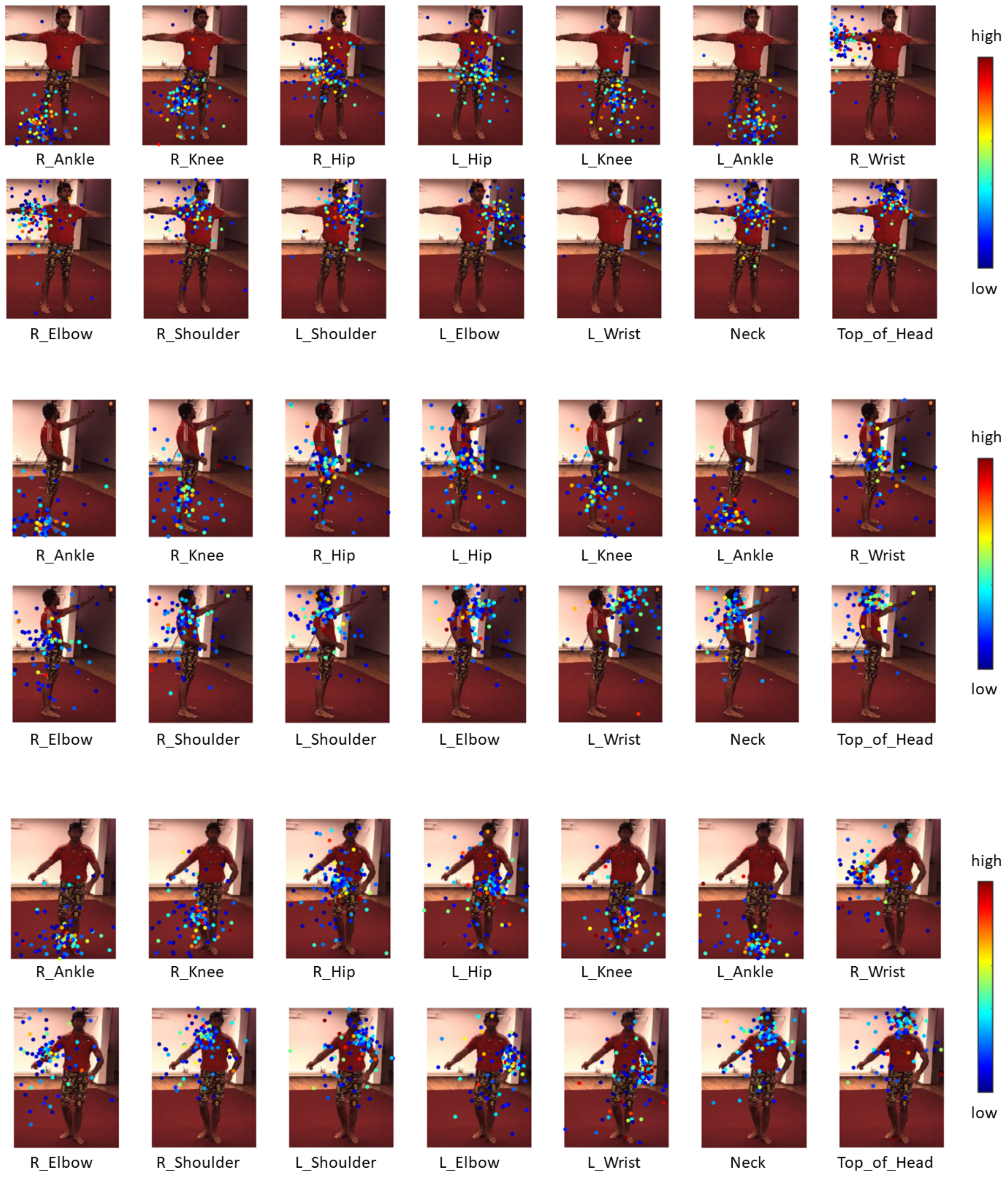

Figure 6. Visualization of multi-scale deformable cross attentions. For each joint query, we plot the sampling points and attention weights from feature maps of different resolutions in one picture. Each sampling point is marked as a filled circle with its color indicating attention weight for that point.

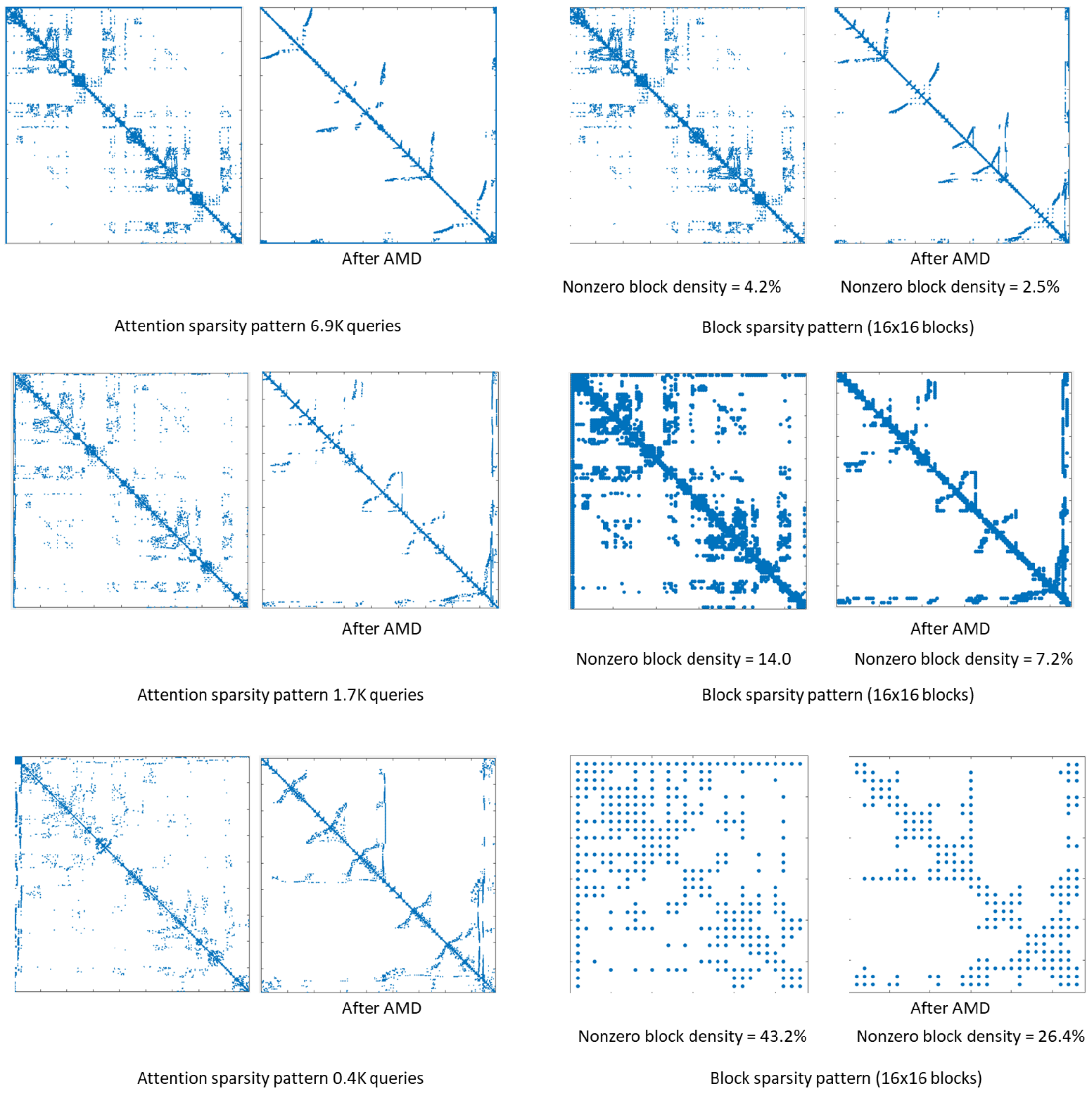

Figure 7. Visualization of block sparse attention patterns for different vertex query resolution.

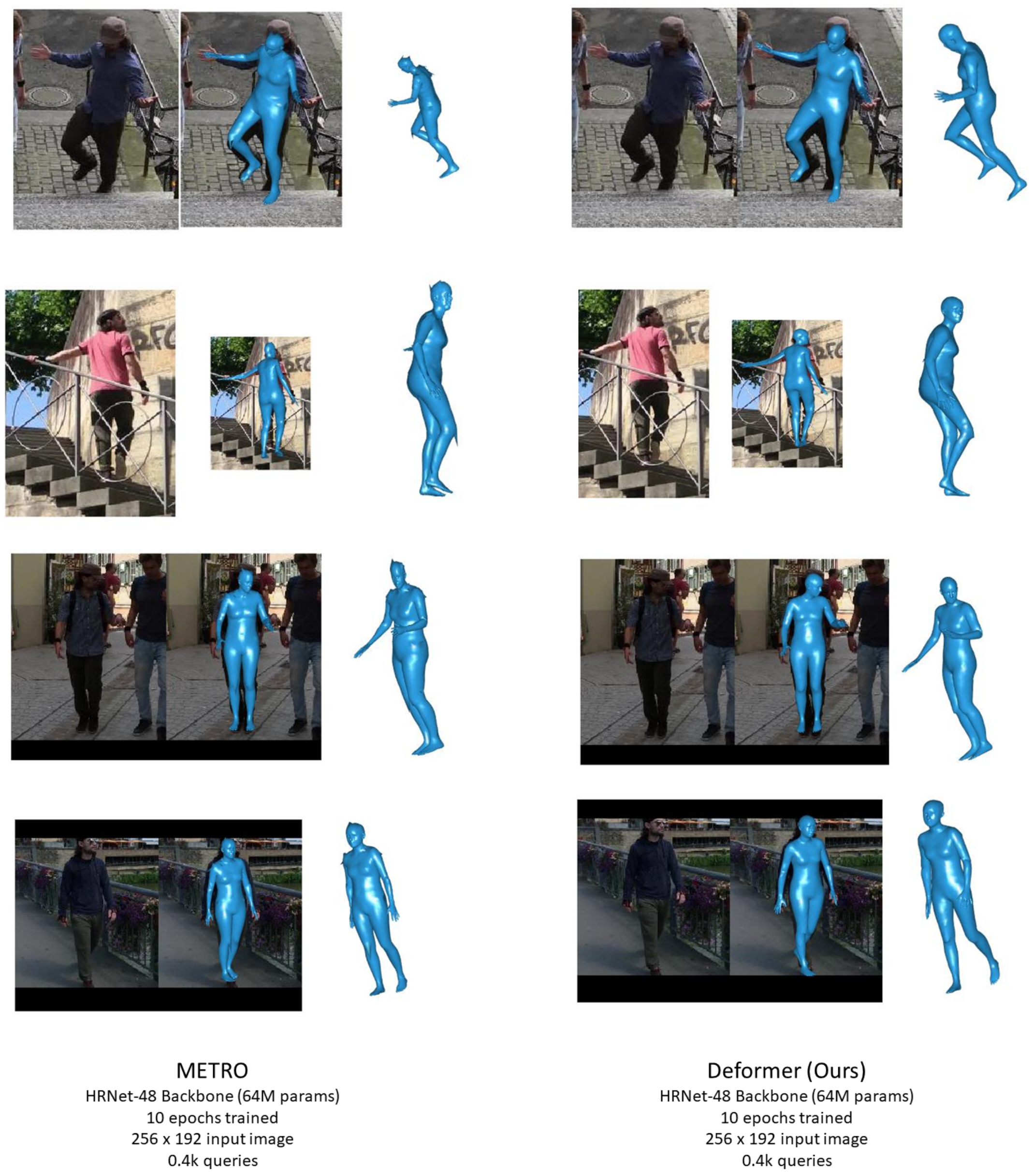

Figure 8. Qualitative comparison with METRO.

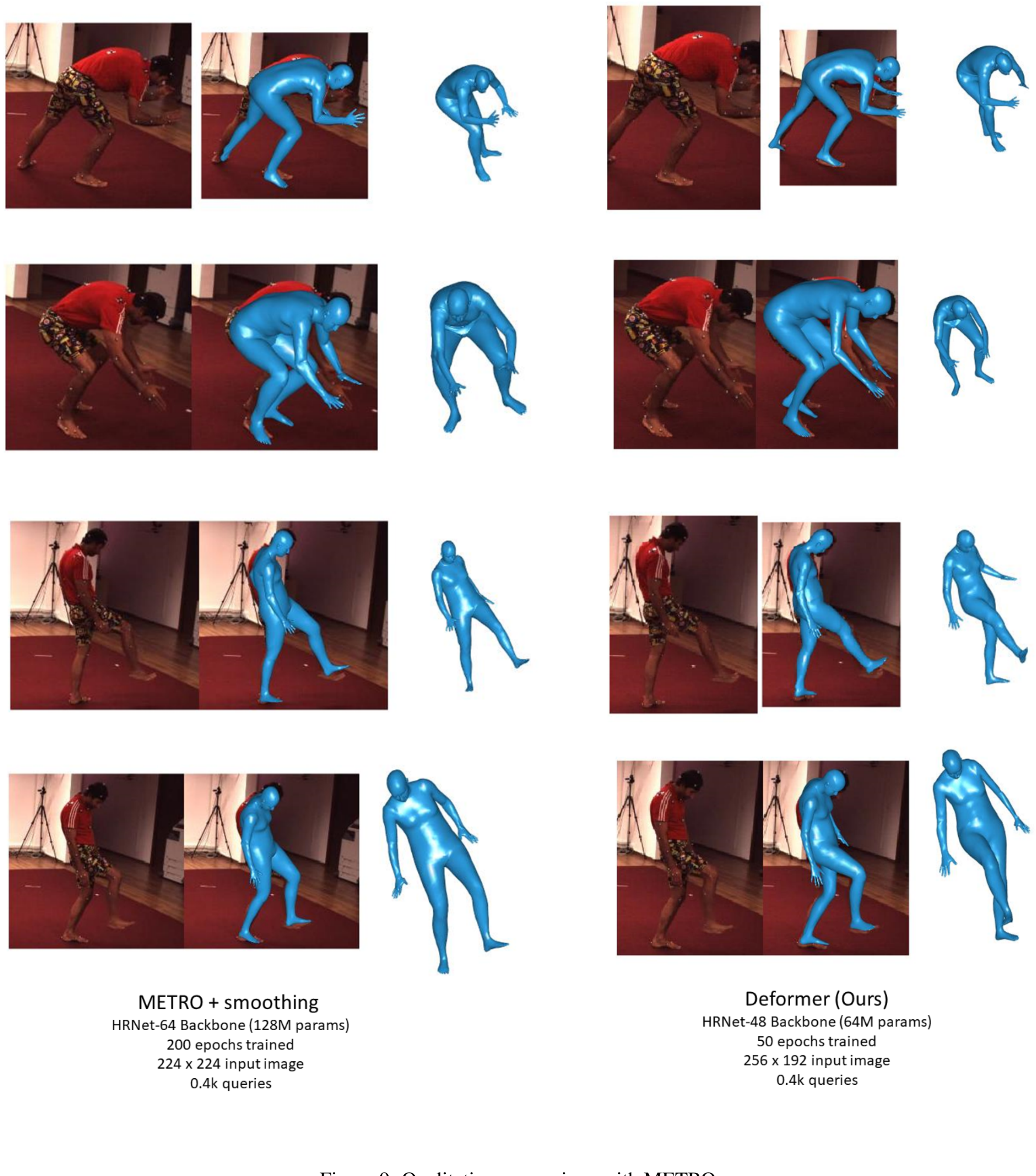

Figure 9. Qualitative comparison with METRO.

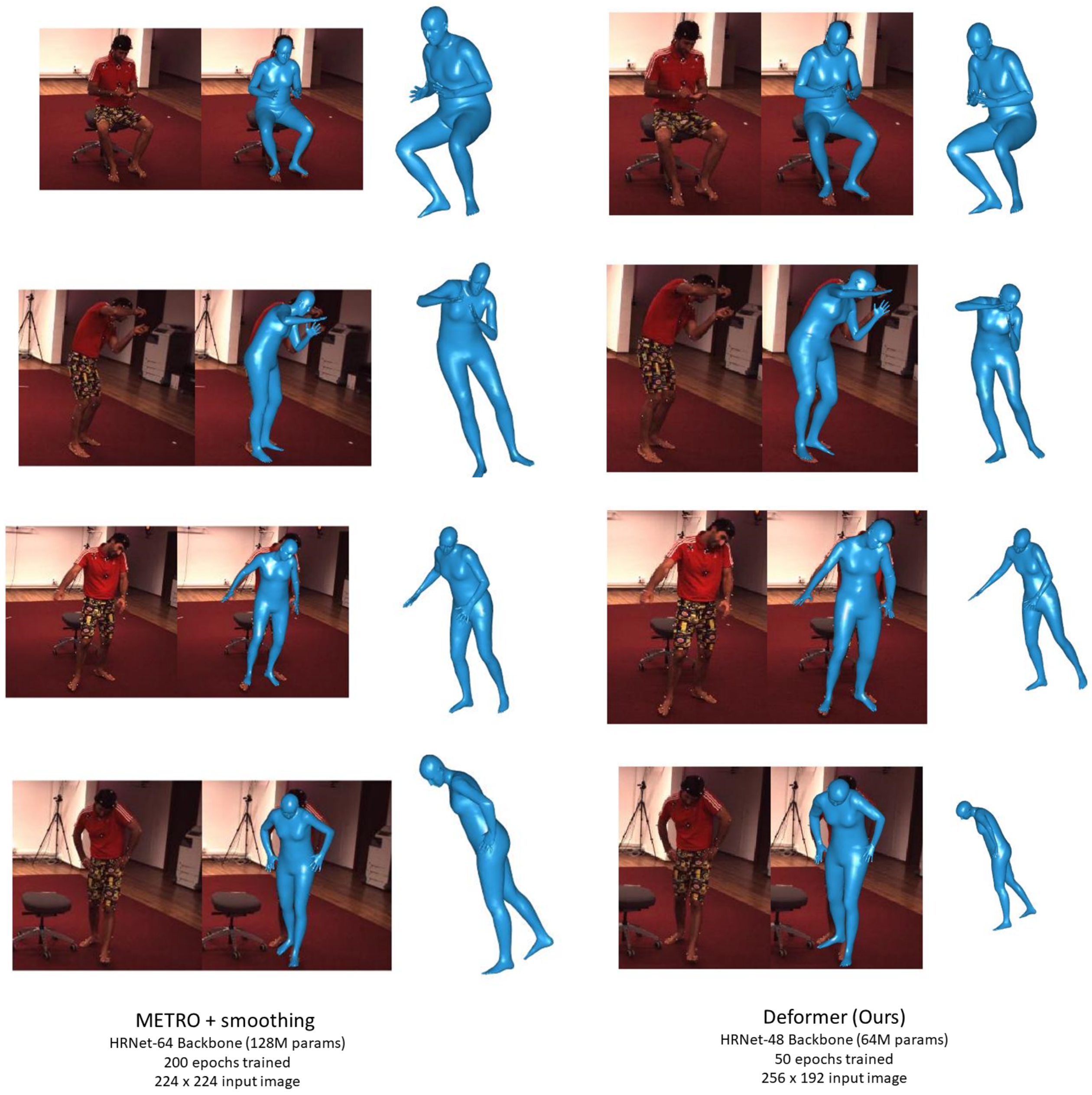

Figure 10. Qualitative comparison with METRO.

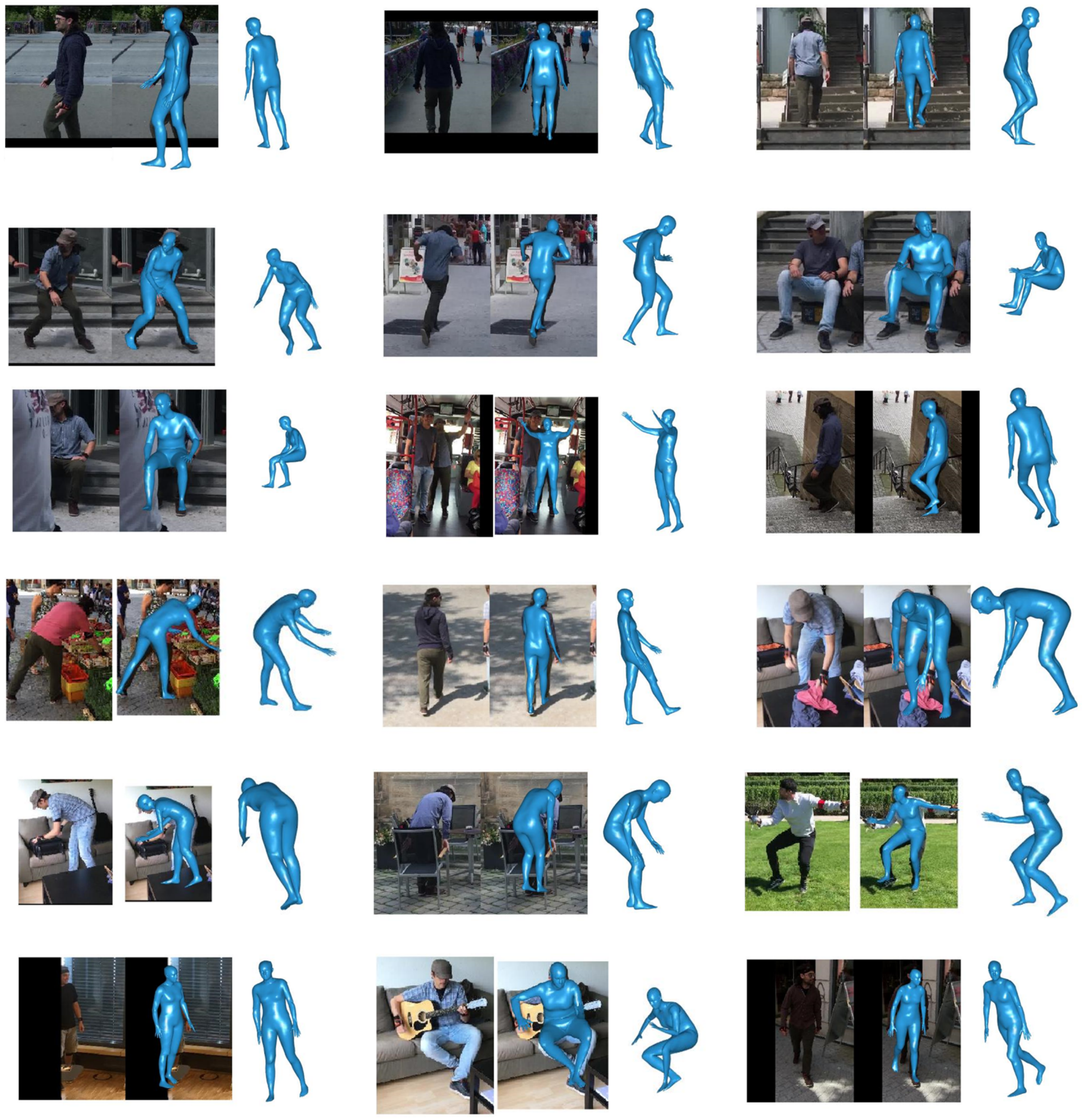

Figure 11. More results on 3DPW.

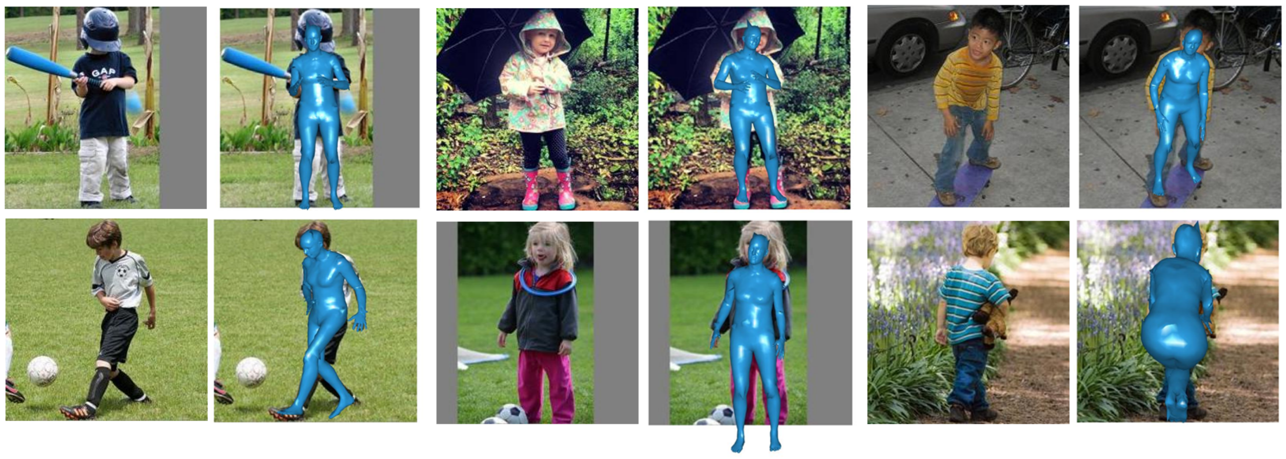

Figure 12. Results on COCO children images.