Method	↑ PSNR	↑ SSIM	↓ LPIPS	Inference Time
w/o progressive	**26.15**	**0.918**	**0.098**	56min
full model	**26.15**	**0.918**	**0.098**	**17min**

Table 10. **Effectiveness of progressive rendering strategy.** The progressive rendering strategy can reduce the inference time by around 70% while without influencing the performance.

Method	↑ PSNR	↑ SSIM	↓ LPIPS
NHP	25.65	0.917	0.148
GP-NeRF	26.46	0.918	0.158
Ours	**28.08**	**0.939**	**0.087**

Table 11. **Fitting performance on training frames.** Our method shows the best fitting ability compared with previous methods.

Method	↑ PSNR	↑ SSIM	↓ LPIPS
w/o $\mathcal{L}_{PER}$	**26.16**	0.916	0.146
full model	26.15	**0.918**	**0.098**

Table 12. **Influence of perceptual loss.** Perceptual loss mainly improves the LPIPS with less effect on PSNR and SSIM.

A. Progressive Rendering

GP-NeRF [6] has proposed a progressive rendering strategy using the coarse geometry provided by the output 3D feature volume of SPC to reduce the number of rendering points. Although there is no SPC in our framework, we find that simply using the fitted SMPL as the alternative works pretty well. Specifically, after sampling points on marched rays from the target view, we only render the points whose euclidean distance to the SMPL template is smaller than $0.1m$. Then, for these close points, we first get the density values for all of them, and then only send part of them whose density value is larger than 0 for the following color inference, which is in line with [6]. The effectiveness of this strategy is illustrated in Table 10. While without decreasing the performance, the inference time is reduced by around 70%. Notably, even without using such accelerating strategy, the inference is still over 2 times faster than NHP [19] (56min *vs*. 1h55min, Table 9). This strongly proves the efficiency of our method.

B. Performance on Training Frames

Following previous methods [19, 6], we report the fitting performance on the training set in Table 11. We achieve the best fitting performance among the generalizable methods, which shows the superior capacity of our method.

C. Additional Ablation Studies

We provide more detailed ablation studies in this section.

C.1. Influence of Perceptual Loss

In Table 12, we demonstrate the influence of perceptual loss. Obviously, perceptual loss can largely improve the LPIPS, *i.e*., make the results visually pleasing, while shows less effect on PSNR and SSIM. Without perceptual loss, we still outperform previous methods by consistent margins in PSRN and SSIM.

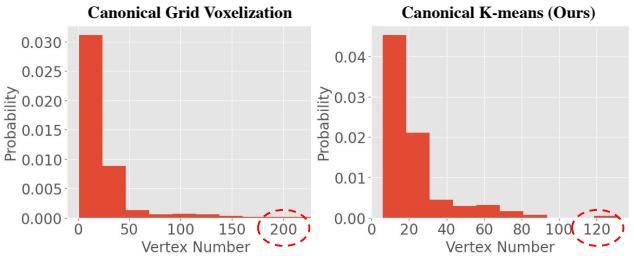

Figure 8. **Comparisons of vertex number distributions between canonical grid voxelization and canonical k-means.** Canonical k-means gives more uniform split with smaller variance.

Method	↑ PSNR	↑ SSIM	↓ LPIPS
can. grid voxelization	26.01	0.917	0.100
can. k-means (ours)	**26.15**	**0.918**	**0.098**

Table 13. **Comparisons between using k-means and grid voxelization in canonical body grouping.**

C.2. Canonical K-means *vs*. Canonical Grid Voxelization

In canonical body grouping, we employ the k-means clustering to get the grouping dictionary. Actually, using grid voxelization under the canonical space is also feasible. However, the uniform grid leads to the large variance of vertex number in each voxel considering the shape of human body, as shown in Fig. 8. Therefore, we use k-means instead for a more uniform split. As illustrated in Table 13, canonical k-means performs better than canonical grid voxelization.

D. Additional Visualization Examples

We provide more comparison examples with previous state-of-the-art methods in Fig. 9.

E. Human Split

We list the detailed human split information in Table 14. We hope that it can serve as a standard split for the following researchers. The code will also be available upon acceptance.

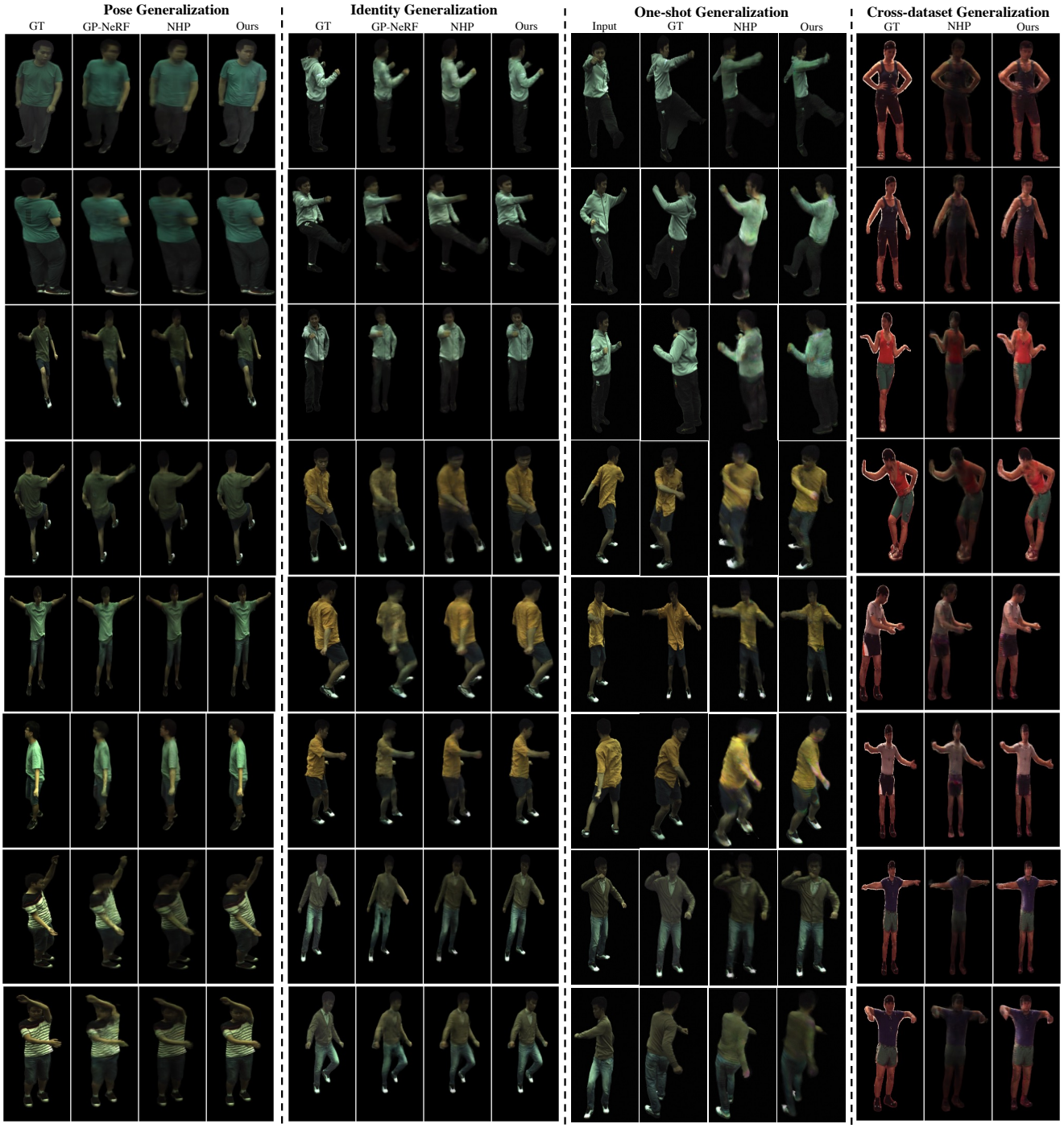

Figure 9. **Supplemented visualization examples on ZJU-MoCap [33] (pose generalization, identity generalization, one-shot generalization) and H36M [16] (cross-dataset generalization).**

Human ID	[Start, End)	Interval	Frame Number	Total Frames	Reference View	Target View
Training Frames						
313	[0, 60)	1	60×21		Rand 3	Rand 1
315	[0, 400)	6	67×21		Rand 3	Rand 1
377	[0, 300)	30	10×23		Rand 3	Rand 1
386	[0, 300)	6	50×23	7589	Rand 3	Rand 1
390	[700, 1000)	6	50×23		Rand 3	Rand 1
392	[0, 300)	6	50×23		Rand 3	Rand 1
396	[810, 1080)	5	54×23		Rand 3	Rand 1
Pose Generalization						
313	[60, 1060)	30	34×6		0, 7, 15	3, 5, 10, 12, 18, 20
315	[400, 1400)	30	34×6		0, 7, 15	3, 5, 10, 12, 18, 20
377	[300, 617)	30	11×6		0, 7, 15	3, 5, 10, 12, 18, 20
386	[300, 646)	30	12×6	798	0, 7, 15	3, 5, 10, 12, 18, 20
390	[0, 700)	30	24×6		0, 7, 15	3, 5, 10, 12, 18, 20
392	[300, 556)	30	9×6		0, 7, 15	3, 5, 10, 12, 18, 20
396	[1080, 1350)	30	9×6		0, 7, 15	3, 5, 10, 12, 18, 20
Identity Generalization						
387	[0, 654)	30	22×6		0, 7, 15	3, 5, 10, 12, 18, 20
393	[0, 658)	30	22×6	438	0, 7, 15	3, 5, 10, 12, 18, 20
394	[0, 859)	30	29×6		0, 7, 15	3, 5, 10, 12, 18, 20
One-shot Generalization						
387	[0, 654)	30	22×6		0	3, 5, 10, 12, 18, 20
393	[0, 658)	30	22×6	438	0	3, 5, 10, 12, 18, 20
394	[0, 859)	30	29×6		0	3, 5, 10, 12, 18, 20
Cross-dataset Generalization						
S1	[0, 750)	150	5×1		0, 1, 2	3
S5	[0, 1250)	150	9×1		0, 1, 2	3
S6	[0, 750)	150	5×1		0, 1, 2	3
S7	[0, 1500)	150	10×1	54	0, 1, 2	3
S8	[0, 1250)	150	9×1		0, 1, 2	3
S9	[0, 1300)	150	9×1		0, 1, 2	3
S11	[0, 1000)	150	7×1		0, 1, 2	3

Table 14. **Detailed human split.** For ZJU-MoCap [33] (training frames, pose, identity, and one-shot generalization), we follow the human split from the officially released code of NHP [19], while for H36M [16] (cross-dataset generalization), we follow the split from [31]. Note that in ZJU-MoCap, "313" and "315" only contains 21 camera views, while the left ones have 23 camera views.