Supplementary Materials for GM-NeRF: Learning Generalizable Model-based Neural Radiance Fields from Multi-view Images

Jianchuan Chen[1] Wentao Yi[1] Liqian Ma[2] Xu Jia[1] Huchuan Lu[1]

[1] Dalian University of Technology, China [2] ZMO AI Inc.

In our supplementary materials, we provide additional quantitative and qualitative results of novel view synthesis and novel pose synthesis as shown in Fig 2, Tab 2, and Tab 3. More video demos and code are available at https://github.com/JanaldoChen/GM-NeRF. We highly recommend watching the provided video demos to get a better understanding. Additionally, we provide extra details as follows: 1) network architecture details (Appendix A); 2) more explanation about normal regularization (Appendix B 3) additional results on Genebody [2] (Appendix C); 4) more ablation experiments (Appendix D).

A. Network Architecture

Image Feature Extraction Network. In order to adequately exploit the m calibrated multi-view images, we deploy a U-Net-like architecture as our image feature extraction network to extract their features with shared weights. Specifically, the encoder of our image extraction network is pretrained ResNet34 [5] on Imagenet [9]. We remove layer4 and max-pooling. As shown in Tab. 1, we take one input image of size $512 \times 512 \times 3$ as an example to describe the network details.

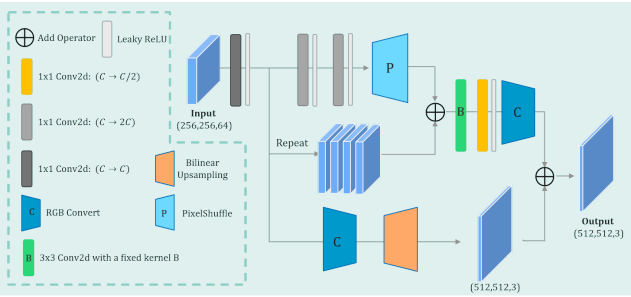

Figure 1. **2D Neural rendering architecture.**

2D Neural Rendering Network. It is memory intensive to render a whole image using volume rendering, therefore we take a hybrid rendering approach of 3D volume rendering and 2D neural rendering. Specifically, we first use volume rendering to get a low-resolution feature map and then upsample it to get the final high-resolution map by neural rendering. However, the typical upsampling approach will compromise the 3D multi-view consistency (e.g., the appearance of the subject will change when moving the camera). To alleviate this problem, we use the upsampling method adopted in [1, 4].

Layer	Input (**ID** : HWC)	Output (**ID** : HWC)
Conv, 64, k=7, s=2	**0** : $512 \times 512 \times 3$	**1** : $256 \times 256 \times 64$
Residual layer 1	**1** : $256 \times 256 \times 64$	**2** : $256 \times 256 \times 64$
Residual layer 2	**2** : $256 \times 256 \times 64$	**3** : $128 \times 128 \times 128$
Residual layer 3	**3** : $128 \times 128 \times 128$	**4** : $64 \times 64 \times 256$
Upconv, 128, k=3, f=2	**4** : $64 \times 64 \times 256$	**5** : $128 \times 128 \times 128$
iConv, 128, k=3, s=1	**3** © **5**: $128 \times 128 \times 256$	**6** : $128 \times 128 \times 128$
Upconv, 64, k=3, f=2	**6** : $128 \times 128 \times 128$	**7** : $256 \times 256 \times 64$
iConv, 64, k=3, s=1	**2** © **7**: $256 \times 256 \times 128$	**8** : $256 \times 256 \times 64$
iConv, 64, k=3, s=1	**1** © **8**: $256 \times 256 \times 128$	**9** : $256 \times 256 \times 64$
Conv, 64, k=1, s=1	**9** : $256 \times 256 \times 64$	**10** : $256 \times 256 \times 64$

Table 1. **Image feature extraction network**. 'Conv' stands for a sequence of operations: convolution (k is kernal size and s is stride), rectified linear units (ReLU) and Batch Normalization [6]. 'iConv' replace the Batch Normalization with Instance Normalization [11] compare with 'Conv'. 'Upconv' stands for a bilinear upsampling with specific factor (f), followed by a 'iConv' operation with stride=1. © represents channel-wise concatenation. 'Residual layer' is the residual blocks of the original ResNet34 [5] design, of two feature maps

B. Normal Regularization

We use numerical approximation to obtain the normal $\mathbf{n}(\mathbf{x}) = \frac{\nabla_{\mathbf{x}}\sigma(\mathbf{x})}{\|\nabla_{\mathbf{x}}\sigma(\mathbf{x})\|_2}$ for an arbitrary point $\mathbf{x} = (x, y, z)$.

$$\nabla_{\mathbf{x}}\sigma(\mathbf{x}) = (\frac{\sigma(x+\varepsilon, y, z) - \sigma(x-\varepsilon, y, z)}{2\varepsilon}, \cdots, \frac{\sigma(x, y, z+\varepsilon) - \sigma(x, y, z-\varepsilon)}{2\varepsilon}) \tag{1}$$

where $\varepsilon = 0.002$ is a minimal variable.

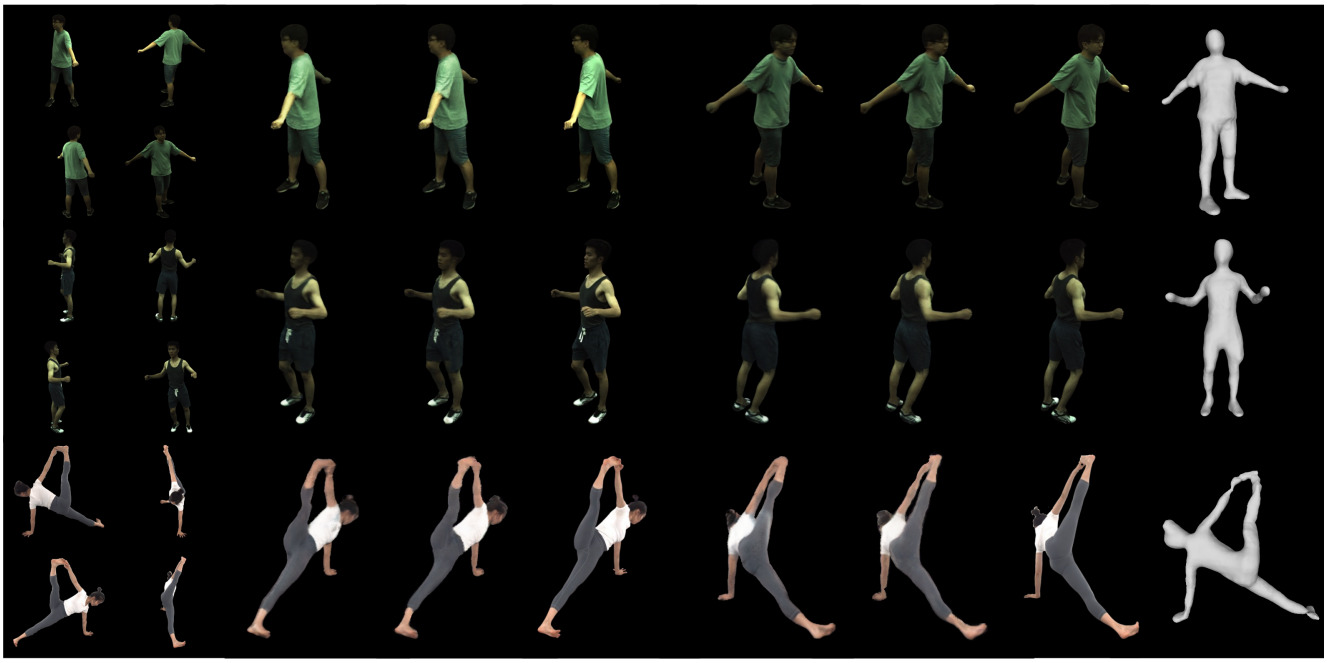

Figure 2. **Qualitative results of our method on ZJU Mocap [8] and GeneBody [2] datasets**. Fs denotes training from scratch, Ft indicates fine-tuning the model after pretraining on THuman2.0 [13] dataset.

	From Scratch			Finetune		
Model	PNSR↑	SSIM↑	LPIPS↓	PNSR↑	SSIM↑	LPIPS↓
313	30.56	0.965	0.0532	31.49	0.971	0.0444
315	26.78	0.952	0.0702	26.98	0.956	0.0607
377	27.36	0.943	0.0775	28.38	0.952	0.0641
386	27.50	0.914	0.1218	29.26	0.932	0.0900
387	25.56	0.920	0.0986	26.46	0.932	0.0804
390	26.91	0.915	0.1070	27.73	0.926	0.0880
392	28.54	0.929	0.0980	29.45	0.942	0.0776
393	26.81	0.920	0.0947	27.69	0.933	0.0785
394	28.05	0.924	0.0923	28.64	0.934	0.0760
Average	27.56	0.931	0.0904	28.45	0.942	0.0733

Table 2. **Quantitative results of novel view synthesis on ZJU-Mocap [8] dataset**.

	From Scratch			Finetune		
Model	PNSR↑	SSIM↑	LPIPS↓	PNSR↑	SSIM↑	LPIPS↓
313	27.65	0.943	0.0731	28.29	0.949	0.0622
315	25.03	0.935	0.0858	25.68	0.944	0.0717
377	26.49	0.930	0.0952	27.58	0.942	0.0778
386	27.63	0.911	0.1288	29.51	0.930	0.0983
387	24.61	0.910	0.1084	25.59	0.926	0.0863
390	26.86	0.916	0.1105	27.90	0.929	0.0897
392	28.07	0.931	0.0979	28.94	0.942	0.0788
393	26.72	0.926	0.0883	27.54	0.938	0.0734
394	27.04	0.916	0.0977	27.65	0.927	0.0801
Average	26.68	0.924	0.0984	27.63	0.936	0.0798

Table 3. **Quantitative results of novel pose synthesis on ZJU-Mocap [8] dataset**.

C. Experiments on the Genebody

One advantage of our approach is that our method can be pre-trained on a large dataset of various identities and then quickly fine-tuned on a specific identity. As shown in Tab. 4 and Fig. 2, such a strategy not only gives our model stronger generalizability compared with training from scratch but also requires only fine-tuning fewer steps to converge. In our experiments, we first pre-trained our model on the THuman2.0 [13] dataset, which contains a large number of different identities, and then fine-tuned $2,000$ steps on the first 100 frames for each sequence in the Genebody [2] test dataset, which cost about 20 minutes. For comparison,

Figure 3. **An extreme case from Genebody [2] dataset**. Our model usually synthesizes thinner results when the performer is wearing a loose dress.

we also train our model from scratch without any pretraining, Compared to per-scene optimization methods such as NT [10], NHR [12], NB [8], which often spend hours or even a day on training, our method can quickly produce

more realistic and detailed images. Genebody [2] has a broad distribution across different clothing styles and poses, even containing professional occasions such as traditional opera costumes, which is more challenging than ZJUMocap [8]. Due to the minimal-clothed topology of SMPL, our model struggles to express extremely loose clothes and accessories. As shown in Fig. 3, our model usually synthesizes thinner results when the performer is wearing a loose dress. To solve this problem, we plan to introduce extensions to SMPL (e.g. SMPLicit [3]) in future work.

	GeneBody [2]		
Model	PSNR↑	SSIM↑	LPIPS↓
NV [7]	19.86	0.774	0.267
NT [10]	21.68	0.881	0.152
NHR [12]	20.05	0.800	0.155
NB [8]	20.73	0.878	0.231
Ours(Fs)	25.38	0.912	0.106
Ours(Ft)	**26.15**	**0.921**	**0.081**

Table 4. **Quantitative comparisons on unseen pose with case-specific optimization methods**. We evaluate the novel view synthesis of unseen poses with case-specific optimization methods on GeneBody [2].

D. More Ablation Experiments

The impact of the number of input views. Benefiting from the multi-view feature fusion mechanism we designed, our method can theoretically accept different numbers of input views. In our experiments, we randomly select $m = 4$ view images as inputs during training. To verify the effect on the number of input views, we conducted an experiment that change the number of input views during testing. As shown in Tab. 5, our method achieves better performance as the number of input views increases. To balance performance and efficiency, we select $m = 4$ views (front, back, left and right) around the subject as inputs during testing in all our experiments.

Input views	PSNR↑	SSIM↑	LPIPS↓
1	27.36	0.9281	0.04854
2	29.25	0.9393	0.03689
4(Ours)	30.18	0.9472	0.03049
6	30.65	0.9530	0.02838
8	31.01	0.9565	0.02663
16	31.54	0.9605	0.02448

Table 5. **The Impact of the number of input views**.

The impact of different SMPL noise levels. We propose a framework for learning generalized neural radiance fields from sparse multi-view images, which introduces SMPL as a geometric prior. Theoretically, the introduction of SMPL should lead to better gains, but in practice, the imprecise estimation of SMPL leads to worse results as shown in Tab 6.

Noise τ(cm)	0	1	3	5	10
PSNR↑	30.12	29.14	28.58	28.19	27.09
SSIM↑	0.9365	0.9302	0.9243	0.9206	0.8959
LPIPS↓	0.0374	0.0462	0.0532	0.0566	0.0729

Table 6. **The impact of different SMPL noise levels.** By adding Gaussian noise (with variance τ) to the SMPL vertices, the performance increases as the noise decreases.

References

[1] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *CVPR*, pages 8628–8638. Computer Vision Foundation / IEEE, 2021. 1

[2] Wei Cheng, Su Xu, Jingtan Piao, Chen Qian, Wayne Wu, Kwan-Yee Lin, and Hongsheng Li. Generalizable neural performer: Learning robust radiance fields for human novel view synthesis. *CoRR*, abs/2204.11798, 2022. 1, 2, 3

[3] Enric Corona, Albert Pumarola, Guillem Alenyà, Gerard Pons-Moll, and Francesc Moreno-Noguer. Smplicit: Topology-aware generative model for clothed people. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 11875–11885. Computer Vision Foundation / IEEE, 2021. 3

[4] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d aware generator for high-resolution image synthesis. In *ICLR*. OpenReview.net, 2022. 1

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778. IEEE Computer Society, 2016. 1

[6] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015. 1

[7] Stephen Lombardi, Tomas Simon, Jason M. Saragih, Gabriel Schwartz, Andreas M. Lehrmann, and Yaser Sheikh. Neural volumes: learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, 2019. 3

[8] Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *CVPR*, pages 9054–9063, 2021. 2, 3

[9] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 115(3):211–252, 2015. 1

[10] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019. 2, 3

[11] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 1

[12] Minye Wu, Yuehao Wang, Qiang Hu, and Jingyi Yu. Multi-view neural human rendering. In *CVPR*, pages 1679–1688, 2020. 2, 3

[13] 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 *CVPR*, pages 5746–5756, 2021. 2