Neural Articulated Radiance Field Supplemental Material

Atsuhiro Noguchi[1] Xiao Sun[2] Stephen Lin[2] Tatsuya Harada[1,3]

[1]The University of Tokyo [2]Microsoft Research Asia [3]RIKEN

A. Ablation Studies for AutoEncoder

In Table 1 of the main paper, we quantitatively evaluate our model in the case of a single articulated 3D object, while in Section 5 in the main paper, we only show qualitative results in the case of Autoencoder. Here, we supplement those results with the quantitative ablation study for Autoencoder in Table A. The same test data settings (namely "same pose/same view", "novel pose/same view", "same pose/novel view", and "novel pose/novel view"), metrics (namely PSNR, SSIM, and Mask), and baseline methods (namely NARF$_P$, NARF$_H$ and NARF$_D$, ***CNN-based***, ***P-NeRF*** and ***D-NARF***) are used for comparison. The same dataset as in Section 5 of the main paper is used for experimentation.

At testing phase, when extracting the latent shape and appearance vectors ($\mathbf{z}_s$ and $\mathbf{z}_a$) using the encoder, we use images under the same viewpoint distribution as in the training images as input. Then, images from novel views and poses are rendered by combining $\mathbf{z}_s$ and $\mathbf{z}_a$ with unseen views and poses in the training data.

Results Quantitative comparison results are given in Table A. Qualitative results under the novel pose/novel view setting are shown in Fig. B. Consistent with the case of a single object, our method NARF$_D$ outperforms the others under all the evaluation metrics and test data settings (best results shown in **bold**). High quality depth and segmentation images are jointly generated as shown in Fig. B (rightmost column). CNN based models cannot represent 3D structure effectively, so the performance drops significantly in the "novel view" settings. Fig. B (left-top) shows that in the novel view testing, the CNN-based method produces fuzzy images. Meanwhile, it cannot generate depth and segmentation images. P-NeRF and D-NARF fail in almost all settings due to implicit transformation and part dependency problems. NARF$_P$ generally performs well, but the computational cost is too high. NARF$_H$ has poor performance on "novel pose" due to part dependency issues. The performance drop on novel pose in the Autoencoder case is not as significant as in the single object case (shown in Table 1 of the main paper) since the pose diversity in the training data is much larger in the Autoencoder case.

B. Ablation Studies in RT-NeRF

In Section 3.3 of the main paper, we introduced the rigidly transformed neural radiance field (RT-NeRF) to effectively model a rigidly transformed object part. Here, we evaluate the effectiveness of the two most critical design elements in RT-NeRF. The first is the ***explicit transformation*** that converts a global 3D location into the *local* coordinate system and the local 3D location is then used to estimate the density using Eqs. 9 and 10 of the main paper. The second is the ***pose-dependent color*** estimation defined in Eq. 11 of the main paper. It takes the 6D vector $\mathfrak{se}(3)$ representation $\boldsymbol{\xi}$ of transformation l as a network input to estimate the RGB color c. To this end, two more baseline methods are introduced accordingly to compare to ***RT-NeRF***. The first is the rigid pose conditioned NeRF (***RP-NeRF***) that takes the global 3D location and the rigid transformation $\boldsymbol{\xi}$ as network inputs, similar to the P-NeRF defined in Eq. 8 of the main paper. The second is ***RT-NeRF w/o*** $\boldsymbol{\xi}$ that estimates the RGB color c without using the transformation $\boldsymbol{\xi}$ as input in Eq. 11 of the main paper.

Dataset We create a synthetic rigid object dataset of a rendered bulldozer using Blender (a software for rendering) for experimentation. In the dataset, the object (a bulldozer) can rigidly transform in the world coordinate system. For each rendered image, both rigid transformation and the camera viewpoint are randomly set. The camera will be translated to point to the center of the object so that the object will appear in the center of the rendered image. The resolution for all rendered images is set to 200 $\times$ 200. In total, 480 images are used for training and another 20 images are used for testing. The loss function is the same as in Eq. 25 of the main paper.

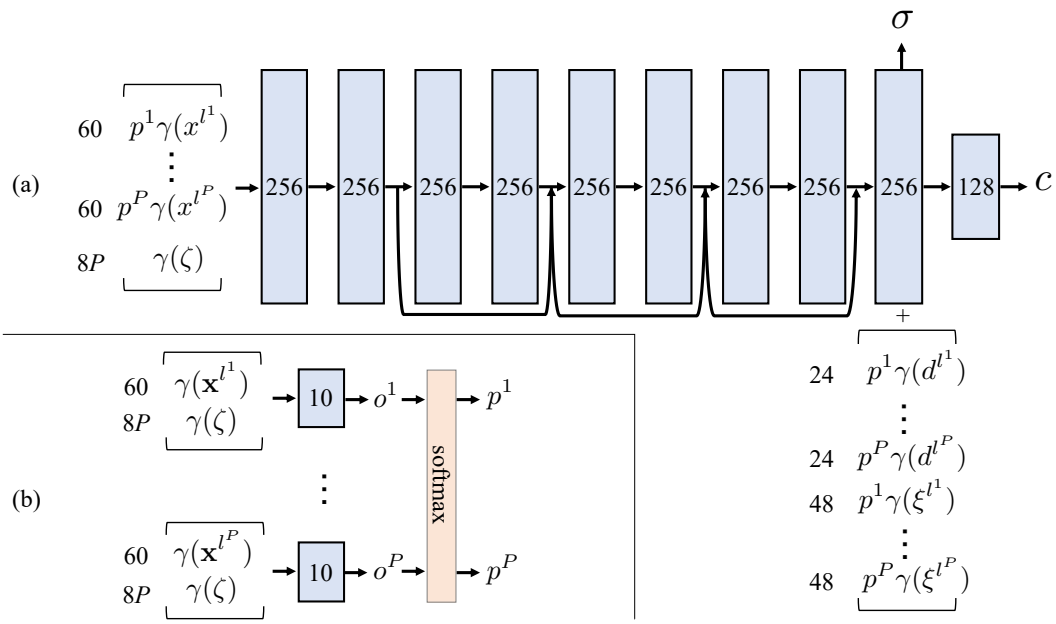

Figure A. Network architecture of Disentangled NARF in our experiments. (a) Network architecture of $F^{l,\zeta}_{\Theta_\sigma}$ and $F^{l,\zeta}_{\Theta_c}$ in Eqs. 22 and 23 in the main paper, modified from the original NeRF architecture diagram. '+' represents concatenation operation. (b) Network architecture of the selector $\mathcal{S}$.

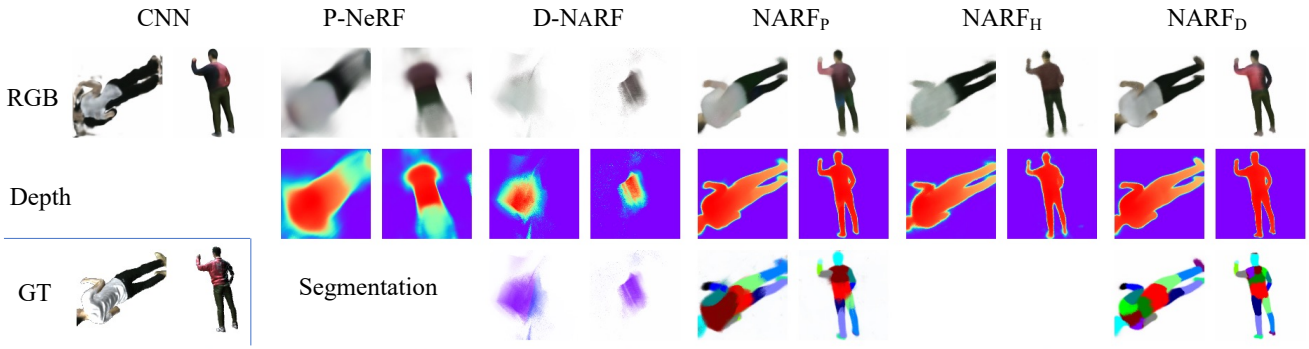

Figure B. Generative results comparison for AutoEncoder

Results The quantitative results are shown in Table B and the qualitative results are shown in Fig. C. The experimental results show that RP-NeRF is unable to learn a good 3D representation and fails to generalize to novel poses and views. In contrast, RT-NeRF effectively models the rigidly transformed object by *explicitly* transforming the global 3D location into the local coordinate system. In addition, the color estimation without the transformation input is less effective. This is concluded by comparing the results of "RT-NeRF w/o ξ" with the results of "RT-NeRF". Quantitatively, the performance of "RT-NeRF w/o ξ" drops significantly under the Mask metric in Table B. Qualitatively, the rendered images from "RT-NeRF w/o ξ" look blurry compared to "RT-NeRF" in Fig. C.

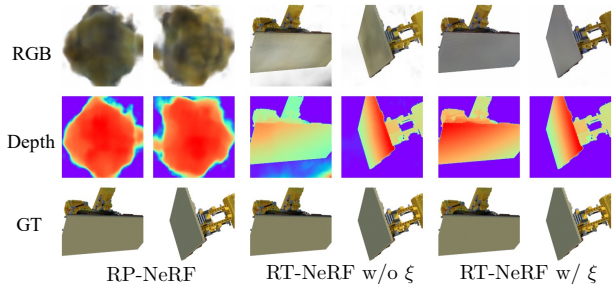

Figure C. Qualitative results for RT-NeRF comparison

	Cost			Same pose, same view			Novel pose, same view			Same pose, novel view			Novel pose, novel view		
Method	#Params	#FLPOS	#Memory	Mask↓	PSNR ↑	SSIM ↑	Mask ↓	PSNR ↑	SSIM ↑	Mask ↓	PSNR ↑	SSIM ↑	Mask ↓	PSNR ↑	SSIM ↑
CNN	15.6M	-	-	**89.0**	**25.59**	**0.8966**	**157.2**	**24.70**	**0.8757**	385.0	**22.70**	0.8213	381.5	**22.98**	0.8243
P-NeRF	0.85M	156M	356K	1526.6	19.24	0.5362	1572.1	19.60	0.5346	1911.3	18.13	0.4666	1963.2	19.08	0.4746
D-NaRF	0.66M	121M	382K	2067.2	18.27	0.6733	2060.2	18.81	0.6706	2026.7	18.00	0.5983	2049.5	18.82	0.6121
NARF$_P$	11.8M	2140M	6544K	159.5	22.57	0.8250	182.3	22.58	0.8211	196.4	21.32	0.8056	205.0	21.80	0.8088
NARF$_H$	1.06M	197M	344K	201.1	22.89	0.8244	225.1	23.14	0.8205	265.1	21.48	0.7891	275.1	22.36	0.7961
NARF$_D$	1.10M	205M	382K	123.4	23.84	0.8568	**163.5**	23.55	0.8435	**166.2**	22.25	**0.8313**	**186.3**	22.81	**0.8294**

Table A. Quantitative comparison for autoencoders. Best results in **bold**.

Method	Mask↓	PSNR ↑	SSIM ↑
RP-NeRF	4511.0	11.25	0.3103
RT-NeRF w/o ξ	29.2	19.83	0.8255
RT-NeRF	**16.9**	**20.05**	**0.8388**

Table B. RT-NeRF quantitative comparison. Best results in **bold**.

C. Additional Ablation Studies in NARF

In this section, we provide more ablation studies on the effects of the mask loss (the second item in Eq. 25 of the main paper), temperature parameter for NARF$_P$ (in Eq. 15 of the main paper), and softmax activation function for NARF$_D$ (in Eq. 21 of the main paper).

W/ and w/o mask loss We conduct experiments to evaluate the effect of the mask loss (the second term of Eq. 25 of the main paper) added to the rendered mask image. While the color loss optimizes the final RGB color of the rendered pixels, the gradient from the mask loss directly optimizes the densities of the 3D locations on the camera rays. Therefore, the additional mask loss is helpful in learning 3D shapes efficiently. The quantitative and qualitative results of w/ and w/o mask loss are shown in Table C and Fig. D respectively. In Table C, it can be seen that performances of all the three variants of NARF drop significantly without the mask loss, especially under the novel pose/novel view setting. Particularly, in Fig. D, the NARF$_H$ model is not able to converge at all without the mask loss. The NARF$_P$ and NARF$_D$ models still work without the mask loss but the rendered images get very blurry, especially on the background regions around the object.

Temperature parameter τ in NARF$_P$ We study the effect of the temperature parameter $\tau \in (0, \infty)$ in NARF$_P$ (in Eq. 15 of the main paper). The temperature parameter determines how soft the selection is among the multiple RT-NeRFs. When τ is close to 0, hard selection is performed. Though the ***Part Dependency*** prior is strictly satisfied in this case, convergence in the training is difficult since the highest current estimate will completely block the gradient from back-propagating to the others. It is especially worse in the early stage of the training when the highest estimate is almost random. In turn, when τ is close to ∞, averaging is performed. In this case, the gradient is back-propagated to all RT-NeRFs, but the ***Part Dependency*** issue arises again, which will harm the generalization ability to novel poses. The quantitative and qualitative results are shown in Table C and Fig. E, respectively. We empirically use the best-performing $\tau = 100$ setting as shown in Table C.

Softmax vs. sigmoid activation in NARF$_D$ In Eq. 21 of the main paper, we use softmax activation, which is also motivated by the ***Part Dependency*** prior. Here, we provide the results of using the sigmoid activation as an alternative. Formally, Eq. 21 of the main paper is replaced with Eq. A:

$$O^i_\Gamma : (\gamma(\mathbf{x}^{l^i}), \gamma(\boldsymbol{\zeta})) \rightarrow (o^i), \quad p^i = \frac{1}{1 + \exp(-o^i)} \tag{A}$$

The quantitative results are shown in Table C. In conclusion, softmax outperforms sigmoid, especially in the "novel pose/novel view" setting.

D. Implementation Details in P-NeRF

In Eq. 8 of the main paper, P-NeRF takes a global 3D location $\mathbf{x}$ and the part transformations $\{l^i | i = 1, ..., P\}$ as input. For implementation, we use the 6D vector $\mathfrak{se}(3)$ representation $\boldsymbol{\xi}^i$ of transformation l^i, and concatenate them together with $\mathbf{x}$ and the bone length ζ as the network input. Positional encoding is performed before the concatenation. Formally,

$$F^{\mathcal{P}}_\Theta : (\gamma(\mathbf{x}), \{\gamma(\xi^i) | i = 1, ..., P\}, \gamma(\zeta), \gamma(\mathbf{d})) \rightarrow (\sigma, c). \tag{B}$$

The density and color sub-networks are defined as

$$F^{\mathcal{P}}_{\Theta_\sigma} : (\gamma(\mathbf{x}), \{\gamma(\xi^i) | i = 1, ..., P\}, \gamma(\zeta)) \rightarrow (\sigma, \mathbf{h}), \tag{C}$$

$$F^{\mathcal{P}}_{\Theta_c} : (\mathbf{h}, \{\gamma(\xi^i) | i = 1, ..., P\}, \gamma(\mathbf{d})) \rightarrow (c). \tag{D}$$

E. Implementation Details in D-NaRF

D-NeRF [4] uses a canonical template and learns the observation-to-canonical deformation

$$\Psi : (\mathbf{x}, \omega) \rightarrow \mathbf{x}' \tag{E}$$

where ω is a deformation latent code.

	Cost			Same pose, same view			Novel pose, same view			Same pose, novel view			Novel pose, novel view		
Method	#Params	#FLPOS	#Memory	Mask L2 ↓	PSNR ↑	SSIM ↑	Mask L2 ↓	PSNR ↑	SSIM ↑	Mask L2 ↓	PSNR ↑	SSIM ↑	Mask L2 ↓	PSNR ↑	SSIM ↑
CNN	15.6M	-	-	76.9	29.12	0.9429	134.8	27.30	0.9211	365.9	25.19	0.8532	392.2	24.53	0.8470
P-NeRF	0.85M	156M	356K	778.7	21.42	0.8006	1077.0	20.42	0.7696	844.9	21.19	0.7897	1110.1	20.27	0.7648
D-NARF	0.66M	121M	382K	2182.6	18.90	0.1143	2308.2	18.81	0.1140	2137.3	19.09	0.1144	2241.3	18.88	0.1133
NARF$_P$	11.8M	2149M	6544K	92.0	28.56	0.9258	116.2	26.83	0.9052	101.5	27.54	0.9144	125.8	26.50	0.9104
NARF$_H$	1.06M	197M	344K	55.6	29.91	0.9470	376.8	24.09	0.8665	70.5	28.81	0.9370	374.6	23.98	0.8646
NARF$_D$	1.10M	205M	382K	**50.5**	**30.86**	**0.9586**	**114.4**	**27.93**	**0.9317**	**64.1**	**29.44**	**0.9466**	**123.8**	**27.24**	**0.9230**
NARF$_D$*sigmoid*	1.10M	205M	382K	50.8	30.74	0.9578	117.7	27.83	0.9304	64.6	29.35	0.9459	125.4	26.17	0.9129
NARF$_P$32	0.32M	59M	824K	114.8	27.85	0.9191	139.9	26.86	0.9068	122.9	27.14	0.9105	147.3	25.74	0.8933
NARF$_P$64	0.98M	178M	1641K	105.5	27.86	0.9212	126.9	26.78	0.9094	113.8	27.21	0.9127	133.1	26.28	0.9041
NARF$_P$128	3.28M	596M	3275K	93.8	28.46	0.9284	116.1	27.15	0.9135	102.9	27.56	0.9180	124.2	26.23	0.9068
NARF$_P$256	11.8M	2149M	6544K	92.0	28.56	0.9258	116.2	26.83	0.9052	101.5	27.54	0.9144	125.8	26.50	0.9104
NARF$_P, \tau = 0.01$	0.32M	59M	824K	240.1	25.81	0.8730	282.0	24.92	0.8541	256.0	25.32	0.8624	287.9	23.72	0.8333
NARF$_P, \tau = 0.1$	0.32M	59M	824K	168.1	26.79	0.8995	196.7	25.92	0.8863	176.1	26.26	0.8911	204.8	25.10	0.8756
NARF$_P, \tau = 1$	0.32M	59M	824K	141.5	27.24	0.9077	169.9	26.29	0.8949	150.9	26.56	0.8989	178.2	25.16	0.8809
NARF$_P, \tau = 10$	0.32M	59M	824K	114.8	27.85	0.9191	139.9	26.86	0.9068	122.9	27.14	0.9105	147.3	25.74	0.8933
NARF$_P, \tau = 100$	0.32M	59M	824K	107.9	27.83	0.9190	130.2	26.96	0.9059	115.9	27.23	0.9117	137.5	26.24	0.8983
NARF$_P, \tau = 1000$	0.32M	59M	824K	109.0	27.44	0.9137	130.8	26.68	0.9014	117.0	26.91	0.9072	138.3	26.08	0.8953
NARF$_P$ w/o L_{mask}	0.32M	59M	824K	193.9	28.01	0.9131	220.2	26.72	0.8994	208.7	27.33	0.9028	229.8	25.54	0.8792
NARF$_H$ w/o L_{mask}	1.06M	197M	344K	6816.2	24.25	0.7004	6893.1	21.10	0.6636	6876.3	23.40	0.7029	6957.2	20.53	0.5244
NARF$_D$ w/o L_{mask}	1.10M	205M	382K	113.4	**31.39**	**0.9630**	165.7	**28.09**	**0.9346**	132.8	**29.90**	**0.9508**	263.7	25.71	0.8889

Table C. Quantitative results of ablation studies.

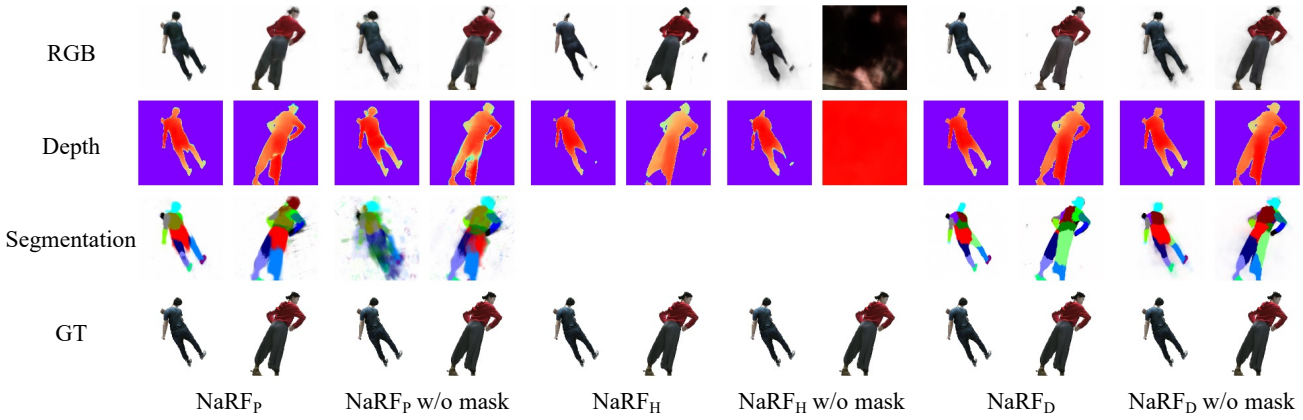

Figure D. Comparison of mask loss.

D-NERF is defined on the deformed position $\mathbf{x}'$ in the canonical template,

$$G : (\Psi(\mathbf{x}, \omega), \mathbf{d}, \psi) \rightarrow (\sigma, c) \tag{F}$$

where ψ is a latent appearance code.

In our case, ω and ψ correspond to the pose configuration $\mathcal{P}$ and the appearance latent vector $\mathbf{z}_a$ respectively. Ψ is implemented using MLP, which seems to suffer from the problem of implicit transformation. In our setup, the pose of each part is given, so it can be implemented more directly. In our implementation, we first use the occupancy network similar to the one defined in Eq. 21 of the main paper to decide which part the input point belongs to.

$$O^i_\Gamma : (\gamma(\mathbf{x}^{l^i}), \gamma(\boldsymbol{\zeta})) \rightarrow (o^i), \tag{G}$$

$$p^i = \frac{\exp(o^i)}{\sum_{k=1}^{P} \exp(o^k)}, \tag{H}$$

Then, we calculate the coordinates $\mathbf{x}'$ on the canonical shape as

$$\mathbf{x}' = \sum_i (\mathbf{x}^{l^i} + \mathbf{t}^i_{\text{canonical}}) * p_i \tag{I}$$

where $\mathbf{t}^i_{\text{canonical}}$ is the origin of a canonical pose's i^{th} part in the global coordinate system. View direction in the canonical space and the transformation vector are defined as

$$\mathbf{d} = \sum_i \mathbf{d}^{l^i} * p_i, \boldsymbol{\xi} = \sum_i \boldsymbol{\xi}^i * p_i \tag{J}$$

Then, the D-NARF we have implemented in the experiment is defined as

$$F^{l,\boldsymbol{\zeta}}_{\Theta_\sigma} : (\gamma(\mathbf{x}'), \gamma(\boldsymbol{\zeta})) \rightarrow (\sigma, \mathbf{h}), \tag{K}$$

$$F^{l,\boldsymbol{\zeta}}_{\Theta_c} : (\mathbf{h}, \gamma(\mathbf{d}), \gamma(\boldsymbol{\xi})) \rightarrow (c). \tag{L}$$

This implementation is similar to the implementation of NARF$_D$, differing only in how the coordinates are input to the model. The results of the experiments show that a

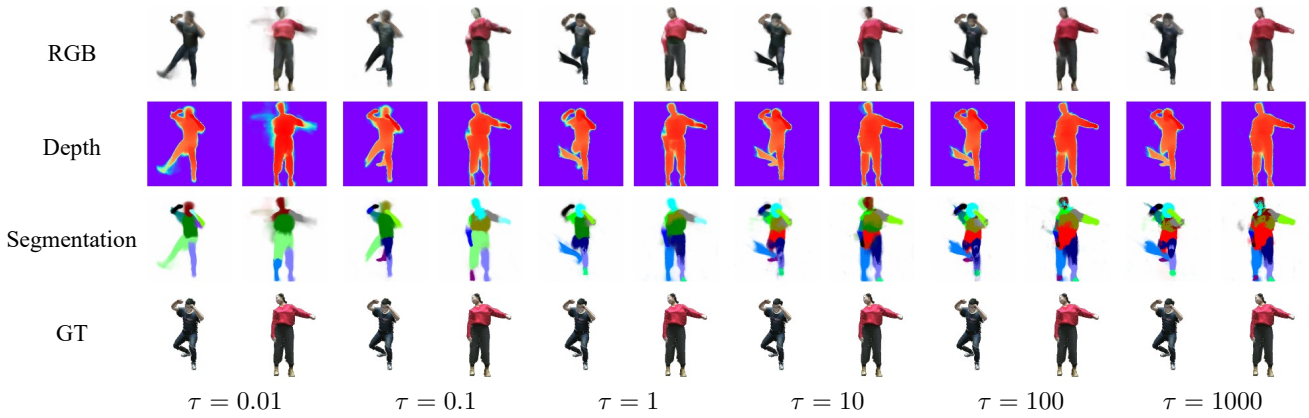

Figure E. Comparison of temperature τ of NARF$_P$.

concatenation-based NARF$_D$ model that retains the coordinates for all parts is more effective than a transformation on the input coordinates in D-NARF.

F. Training Details

We used the Adam [2] optimizer with an initial equalized learning rate [1] of 0.01. The learning rate is decayed to $0.99995\times$ of the previous iteration. Particularly, P-NeRF and NARF$_H$ based autoencoders are trained with an initial learning rate of 0.001 since the training will explode if a learning rate of 0.01 is used. The batch size is set to 16 for all experiments. We sample as many camera rays as can be fit in the GPU memory. The training converges at about 100,000 iterations. The training of our method NARF$_D$ takes 24 hours on 4 V100 GPUs. The code for creating our synthetic datasets is available at https://github.com/nogu-atsu/NARF.

G. Cross Dataset Evaluation on SURREAL Dataset

In order to verify the generalization ability of NARF autoencoder across different datasets, we use a cross dataset evaluation protocol that trains the model on THUman dataset [7], then tests it on SURREAL dataset [6]. There are several differences between the THUman and SURREAL datasets. First, even though the human samples in SURREAL contain rendered SMPL meshes and textures as in THUman, but unlike in THUman, the meshes do not contain clothing. Second, the camera and shape parameters in SURREAL, including the distance between camera and human, human pose distribution and the range of body size are quite different from that in THUman. Even so, we test images from the test set of SURREAL using the NARF autoencoder trained on THUman. The qualitative reconstruction results are shown in Fig. F. From left to right, we show the input SURREAL images, their reconstructed images,

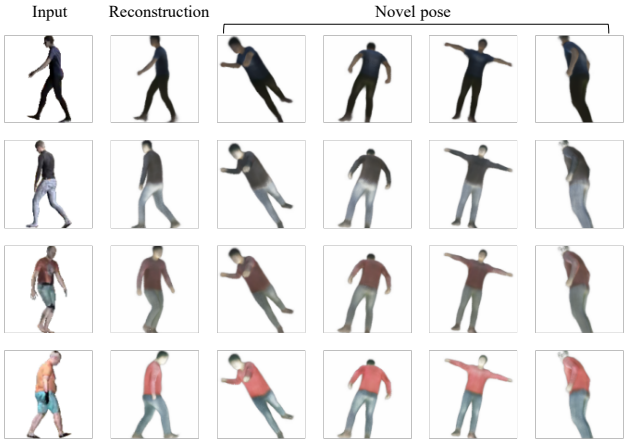

Figure F. Qualitative results on SURREAL dataset. The cross dataset protocol is used for evaluation.

and the images re-rendered under different pose configurations. Since the THUman dataset is not diverse enough in terms of clothing and body size, the reconstructions of short pants and fat people are less effective. But the novel view and pose rendering results look quite reasonable.

H. Experiment on Real Human Images

In this section, we test our approach on a real human dataset ZJU-MOCAP [5]. ZJU-MOCAP is a multi-view person video dataset. For each frame, SMPL parameters [3] are given. We use the first 1969 frames (90%, 2185 frames in total) of the Taichi class video for training and the remaining 216 frames (10%) for testing (novel pose). The resolution of the image is 512×512.

The qualitative results of NARF$_D$ on this dataset are shown in Fig. G. The left part of Fig. G shows the pose used in the training, but rendered from novel viewpoints, and the right part of Fig. G shows the novel pose/novel view test-

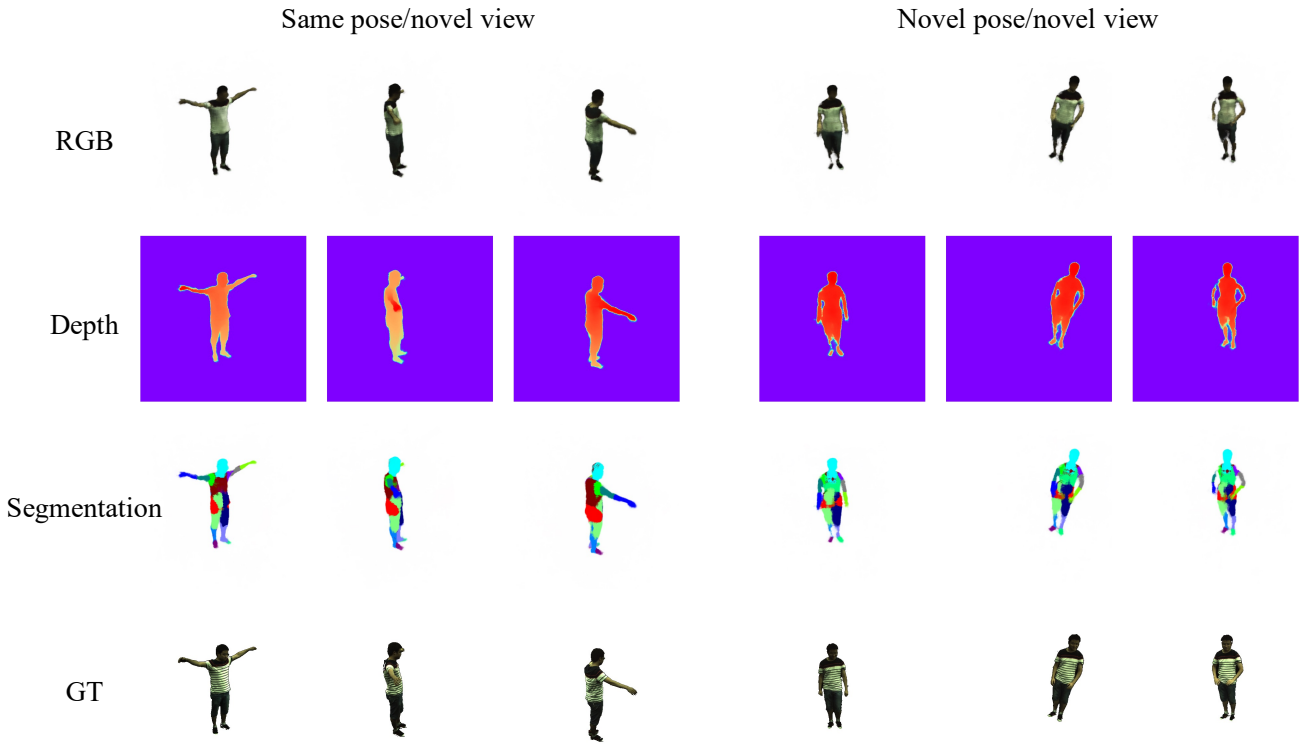

Figure G. Results of NARF$_D$ on real human images

ing results. The quality of the rendered images is not as good as testing on our synthetic datasets. This might be caused by the assumption that the parts are rigid objects, which may not be perfectly satisfied for real images. For example, loose clothes may move when a person makes a movement. This issue can be considered in future work, for example, by learning latent variables to account for both pose-dependent and pose-independent deformations similar to Neural Body [5].

Although the quality of the rendered images for a real person from our method still has room for improvement, we believe that the proposed explicitly controllable representation of viewpoint, pose, bone parameters, and appearance for the articulated object is an important contribution.

References

[1] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018. 5

[2] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5

[3] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):1–16, 2015. 5

[4] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo-Martin Brualla. Nerfies: Deformable neural radiance fields. In *ICCV*, 2021. 3

[5] 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*, 2021. 5, 6

[6] Gül Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *CVPR*, 2017. 5

[7] Zerong Zheng, Tao Yu, Yixuan Wei, Qionghai Dai, and Yebin Liu. Deephuman: 3d human reconstruction from a single image. In *ICCV*, 2019. 5