Supplementary Materials: R4D-planes: Remapping Planes For Novel View Synthesis and Self-Supervised Decoupling of Monocular Videos

Anonymous Authors

1 IMPLEMENTATION DETAILS

Code is implemented by pytorch and all experiments are run on an RTX 3090 GPU. The experiments consisted of two tasks: novel view synthesis and dynamic and static decoupling. For both tasks, the number of iterations is set to 25000, with a batch size of 4096. The learning rate for the feature plane is 0.02, while for the neural networks it is 0.001, except for the remapping net, which is set to 0.0007. All learning rates decay exponentially by 0.1 every 20000 steps. Network is trained using Adam optimizer with β_1 is 0.9 and β_2 is 0.99. L in position encoding $\gamma(x, y, z)$ is 4, L in $\gamma(t)$ is 8,L in $\gamma(d)$ is 4, L for the appearance feature encoding is 2.

1.1 Monocular Novel View Synthesis

Settings in D-NeRF dataset[6]: Our experimental setup is consistent with Hexplane: step size of samples is 0.5, the resolution of r_x, r_y, r_z is initialized to 64 and τ is 0.25*frames, they are upsampled at 3000, 6000 and 9000 to end up with 200 × 200 × 200 × 0.5 × frames. We set F = 48 for the appearance plane and F = 24 for the density plane. We apply different TVloss weights to r_x, r_y, r_z and τ, the weight of r_x, r_y, r_z is set to 1, while the tvloss weight of τ is set to 10. We use MLP as the decoder of color and density.

Settings in HyperNeRF dataset[5]: the resolution of r_x, r_y, r_z, τ will be upsampled at 2000, 4000, 6000 and 8000 to end up with 240*240*240*frames. We set F = 24 for the appearance plane and F = 8 for the density plane. To reduce the training time, we increase the step size to 0.8 and use summation to obtain the density values. The weights of TVloss are set as in the D-NeRF experiment. Since our method is not based on deformation fields, we do not use the supervision of deformations provided in the dataset, whereas it is used in NDVG, V4D, Tineuvox, HyperNeRF[1–3, 5].

1.2 Static and Dynamic Decoupling

The dataset consists of eight validation rig scenes: chicken, banana, broom, balloon, water, cookie, duck and pick. Three of them are derived from HyperNeRF dataset[5]. The relevant parameters are the same as in the HyperNeRF experiment above. For the loss in this task:

$$\mathcal{L} = \mathcal{L}_c + \lambda_{tv}(\mathcal{L}_{tv}^{st} + \mathcal{L}_{tv}^{dy}) + \lambda_s \mathcal{L}_s + \lambda_d \mathcal{L}_d \tag{1}$$

$$\mathcal{L}_s(\mathbf{r}) = \frac{1}{N}\sum_{i=1}^{N} H\left(\frac{\sigma_i^{dy}}{\sigma_i^{dy} + \sigma_i^{st}}^{k}\right) \tag{2}$$

where $H(x) = -(x \cdot \log(x) + (1-x) \cdot \log(1-x))$

$$\mathcal{L}_d(\mathbf{r}) = \max\left(\frac{\sigma^{dy}}{\sigma^{st} + \sigma^{dy}}\right) \tag{3}$$

λ_s is set as 1e-4 first,and will increase exponentially to 1e-2 by step 10,000.

In addition, we utilized Windowed positional encoding, we set m to 10000.

We also apply different low-density initializations to the static and dynamic parts. Specifically, the calculated density values are activated at the end by the softplus function:$\sigma = \text{softplus}(\sigma + b)$, b is the shift, which is calculated by $log(1/(1-\alpha) - 1)$[7]. We apply different shift to the static and dynamic parts, for the dynamic part, α was set to 0.001 and b to -6.9, and for the static part we set b to 0. This prevents parts of the static from being incorrectly decoupled into the dynamic parts, increasing robustness to different scenes. Thus, although in a self-supervised manner, our method does not need to adjust the hyperparameter k in the loss when facing different scenes. In contrast, D^{2}NeRF[9] requires tuning of the hyperparameters in the loss for different scenes We set k uniformly to 1.5 in all experiments.

2 NETWORK STRUCTURE

We show the network structure in Figures 1, 2 and 3. Pink and blue boxes denote MLPs with and without ReLU activation.γ denotes the positional encoding. $\mathcal{A}$ and $\mathcal{D}$ are the density and color features extracted from R4D-plane, respectively. d is the view direction.

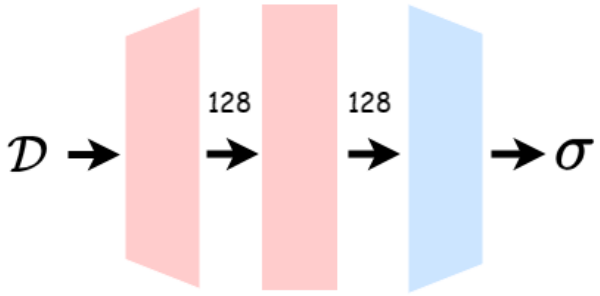

Figure 1: The structure of the density decoder (only used in D-NeRF dataset)

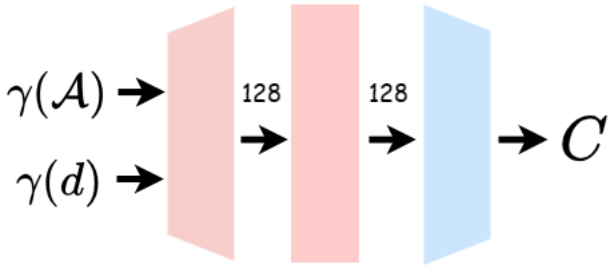

Figure 2: The structure of the color decoder

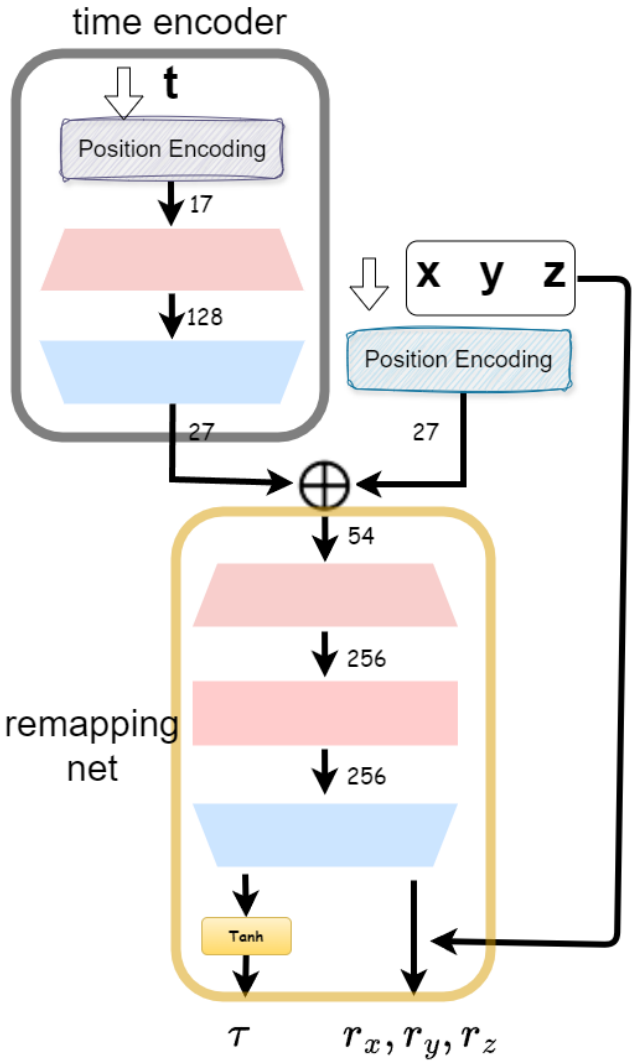

Figure 3: The structure of the remapping net and time encoder

3 EXPERIMENT RESULTS

The detailed metrics and renderings are presented in the corresponding figures and tables. In addition, we note the shortcomings of the current 3DGS-based dynamic scene reconstruction methods and show these cases.

These methods do not work well for modeling fast-moving objects in dynamic scenes, especially when the camera viewpoint also changes significantly. In fact these methods for monocular video use static scene initialization, i.e., the scene is modeled as a static scene first for initialization of Gaussian attributes, and then the dynamics are encoded by a deformation field afterwards. For objects that move more regularly and gently, static initialization can capture a portion of them and model them completely in the subsequent densification process. Since the corresponding object can never be modeled, the densification process will be performed frequently due to the gradient and radius, and the number of gaussians keeps increasing, leading to slower training. For example, in our experiments, Deformable 3DGS took more than two hours to train in the 3d-printer scene. In the banana scene, it's even more than four hours.

4 ABLATION STUDY

We performed ablation experiments on the D-NeRF[6] dataset for the number of channels in the remapping net.

Table 1: ablation study of the remapping net

	PSNR↑	SSIM↑	LPIPS↓
w/o remapping	31.20	0.97	0.04
remapping w/ 64 channels	33.54	0.97	0.03
remapping w/ 128 channels	34.28	0.98	0.02
remapping w/ 256 channels	34.96	0.98	0.02

REFERENCES

[1] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. 2022. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*. 1–9.

[2] Wanshui Gan, Hongbin Xu, Yi Huang, Shifeng Chen, and Naoto Yokoya. 2022. V4d: Voxel for 4d novel view synthesis. *arXiv preprint arXiv:2205.14332* (2022).

[3] Xiang Guo, Guanying Chen, Yuchao Dai, Xiaoqing Ye, Jiadai Sun, Xiao Tan, and Errui Ding. 2022. Neural deformable voxel grid for fast optimization of dynamic view synthesis. In *Proceedings of the Asian Conference on Computer Vision*. 3757–3775.

[4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics* 42, 4 (July 2023). https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

[5] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. 2021. HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields. *ACM Trans. Graph.* 40, 6, Article 238 (dec 2021).

[6] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. 2021. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10318–10327.

[7] Cheng Sun, Min Sun, and Hwann-Tzong Chen. 2022. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5459–5469.

[8] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Wang Xinggang. 2023. 4D Gaussian Splatting for Real-Time Dynamic Scene Rendering. *arXiv preprint arXiv:2310.08528* (2023).

[9] Tianhao Wu, Fangcheng Zhong, Andrea Tagliasacchi, Forrester Cole, and Cengiz Oztireli. 2022. Dˆ 2NeRF: Self-Supervised Decoupling of Dynamic and Static Objects from a Monocular Video. *Advances in Neural Information Processing Systems* 35 (2022), 32653–32666.

[10] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. 2023. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101* (2023).

Table 2: Quantitative Results on D-NeRF dataset[6]

metric	hook	stand up	T-rex	bouncing ball	jumping jack	hell warrior	mutant	lego
PSNR ↑	34.25	37.45	36.49	42.90	36.23	29.80	37.42	25.18
SSIM ↑	0.985	0.990	0.991	0.995	0.989	0.977	0.991	0.942
LPIPS ↓	0.020	0.014	0.012	0.024	0.018	0.033	0.013	0.040

Table 3: Quantitative Results on D^2NeRF dataset[9]

metric	peel banana	broom	chicken	balloon	water	cookie	duck	pick
PSNR ↑	24.23	22.04	27.59	24.61	30.67	31.80	26.22	27.16
MS-SSIM ↑	0.90	0.78	0.96	0.94	0.98	0.99	0.96	0.97

Table 4: Quantitative Results on HyperNeRF dataset[5]

metric	3d printer	peel banana	broom	chicken
PSNR ↑	23.50	25.96	22.79	28.87
MS-SSIM ↑	0.86	0.932	0.73	0.96

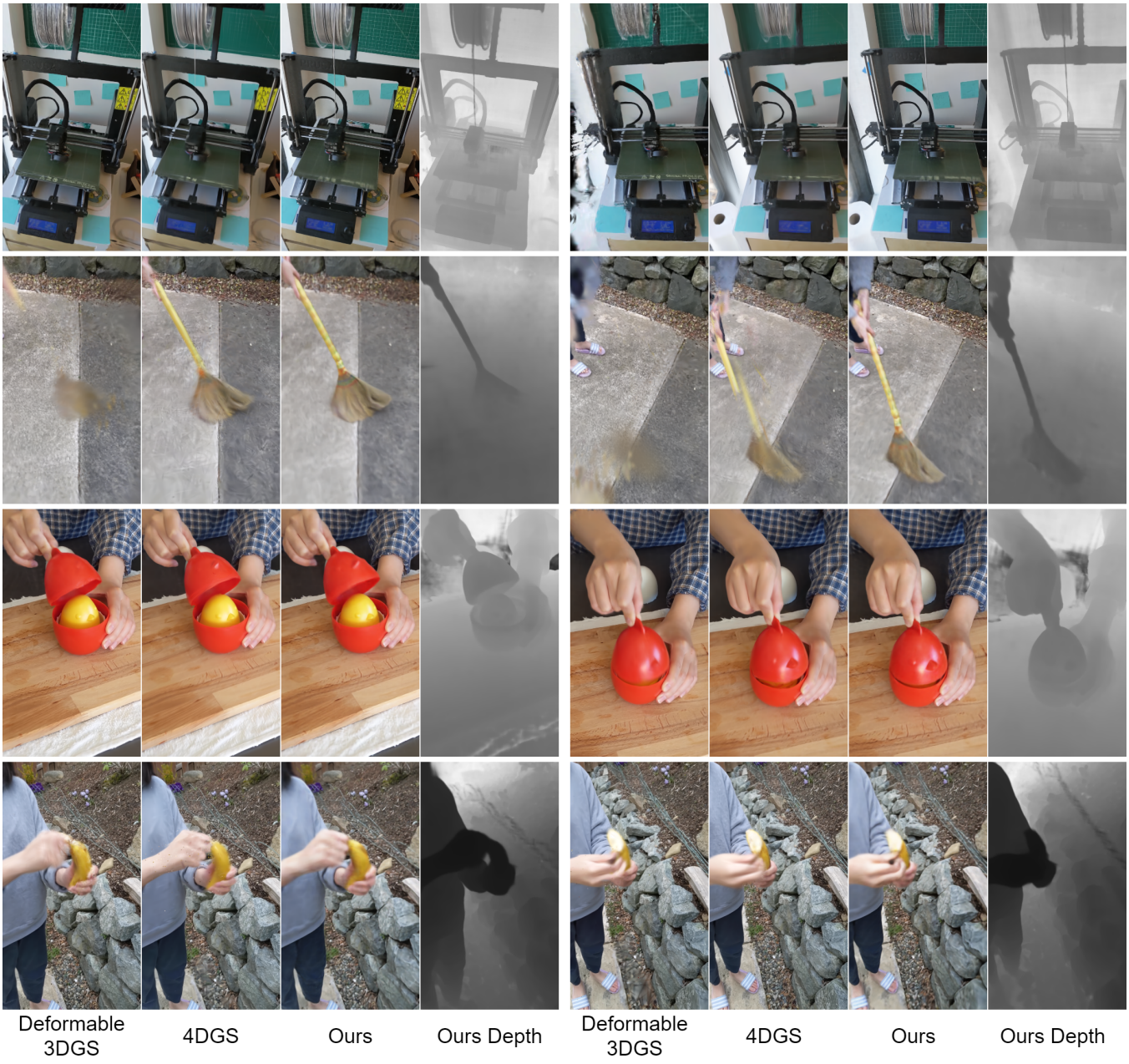

Figure 4: View Synthesis Results on HyperNeRF Dataset[5]. We show rendering results for Deformable 3DGS[10] and 4DGS[8], which are representative methods for dynamic expansion of 3DGS[4]. We can see that these methods are not able to reconstruct fast-moving objects in the scene, such as broom and material columns of 3d printer, but they render well for static backgrounds. Our approach can keep geometric consistency across different viewpoints. From top to bottom: 3D-printer, broom, chicken, peel banana

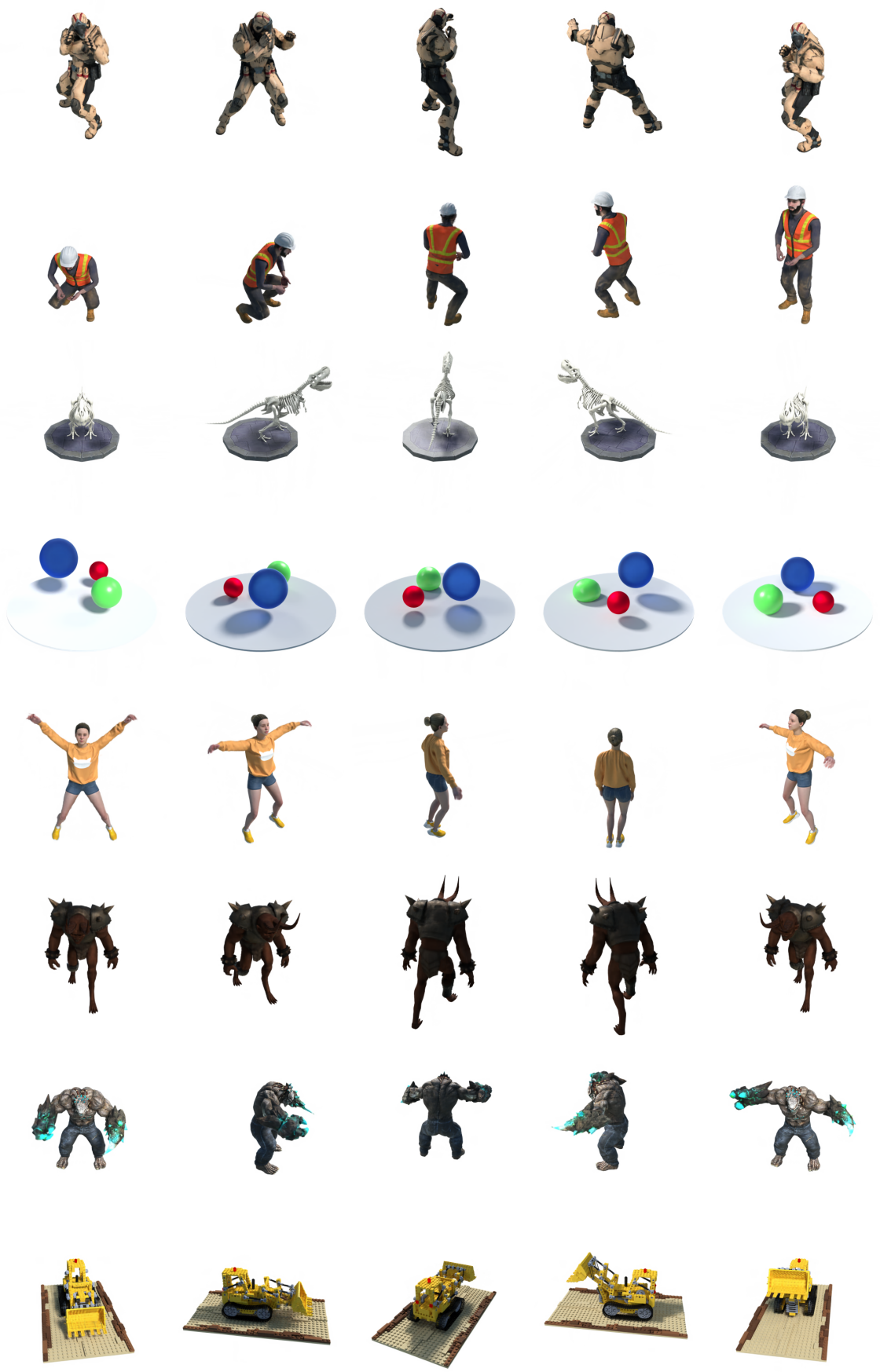

Figure 5: View Synthesis Results on D-NeRF Dataset[6]. From top to bottom: hook, stand up, T-rex, bouncing ball, jumping jacks, hell warrior, mutant, lego

Figure 6: Decoupling Results on D^{2}NeRF Dataset. From top to bottom: chicken, peel banana, broom, balloon, cookie, pick, duck, water pour.