Free-Viewpoint Video of Outdoor Sports Using a Drone

Zhengdong Hong

Zhejiang University

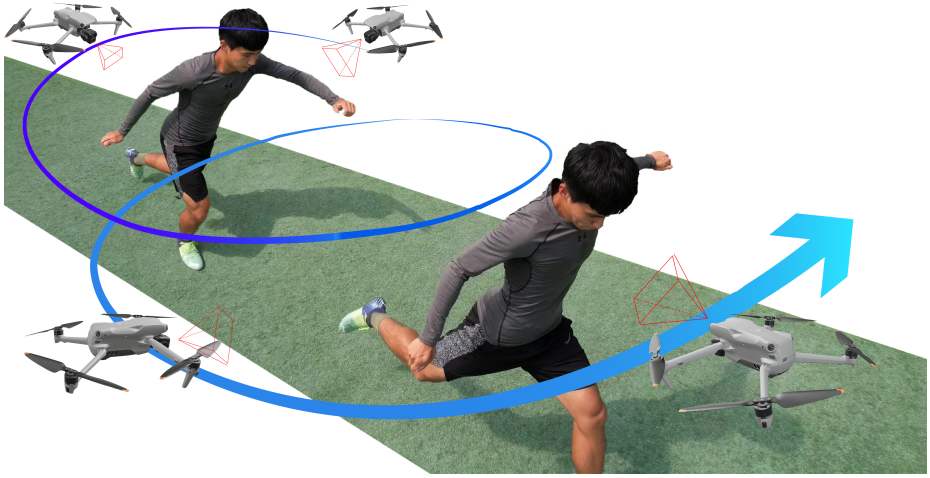

Fig. 1: System illustration: during a penalty kick, the drone can autonomously track and swiftly orbit around the athlete, offering diverse observation angles for reconstructing and rendering the entire sports scene.

1 Comparison with Other Dynamic Reconstruction Methods

We tested other methods in the scope of general 4D scene reconstruction and found that they exhibit performance degradation on our AerialRecon dataset, to be specific, having difficulties in accurately modeling fast-moving and changing human motions. We reported novel view rendering metrics in Table. 1. Note that all methods use the same training data from Drone No.1, and test on the same views from Drone No.2. For fair comparison, all methods use the same camera parameters from our calibration module.

NeRF-based methods which simply condition NeRF on a time variable or employ a deformation field fail to handle the dramatic action variations, and the rendered human could be mostly incomplete. Also, without special design on unbounded scene like [1, 24], it is hard for most of them to model the large-scale outdoor scenes in our datasets, resulting in blurring rendering on

the background. Monocular 4D Gaussian splatting based methods[21, 23] improve the unbounded scene rendering quality by a large margin, achieving better perfromance in metrics. However, all of them are struggling with fast, intricate, complex human motion in real-world outdoor sports. Blurring due to dramatic scene changes such as human motions remain unsolved for the SOTA methods[3, 4, 6, 17, 18, 20, 21, 22, 23].

Table 1: Comparison of monocular dynamic reconstruction methods on AerialRecon

	Climb			Jogging			Kungfu			Football		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D2-NeRF[22]	17.51	0.553	0.472	16.80	0.509	0.440	18.03	0.598	0.353	16.98	0.516	0.466
Nerfies[17]	15.63	0.406	0.576	16.77	0.455	0.542	16.72	0.460	0.533	17.55	0.434	0.487
HyperNeRF[18]	16.39	0.448	0.545	17.52	0.461	0.469	17.40	0.445	0.515	18.03	0.549	0.357
NeRF-Player	16.74	0.437	0.511	16.46	0.439	0.564	17.83	0.502	0.439	17.28	0.431	0.514
K-Planes[4]	15.50	0.383	0.586	16.03	0.451	0.556	15.38	0.379	0.590	16.35	0.457	0.560
TiNeuVox[3]	17.42	0.473	0.479	16.90	0.462	0.519	18.29	0.579	0.383	18.50	0.617	0.332
FFDNeRF[6]	17.58	0.491	0.462	18.28	0.587	0.348	18.15	0.556	0.391	17.74	0.523	0.425
Deformable-3DGS[23]	19.23	0.642	0.338	19.54	0.653	0.324	19.75	0.628	0.326	19.28	0.635	0.314
4D-GS[21]	18.92	0.613	0.372	19.61	0.628	0.310	19.32	0.643	0.335	18.95	0.642	0.309
Ours	**24.24**	**0.891**	**0.203**	**22.94**	**0.841**	**0.243**	**23.29**	**0.874**	**0.237**	**22.41**	**0.793**	**0.268**

2 Implementation Details

We choose NeRF-T[10], NeuMan[8] and HOSNeRF[12] for comparison on three datasets: the dataset from NeuMan[8], the dataset from HOSNeRF[12], and our AerialRecon dataset. We also test several SOTA method[3, 4, 6, 17, 18, 20, 21, 22, 23] in the scope of general 4D reconstruction on our AerialRecon dataset. The implementation details are as follows:

HyperNeRF We set the learning rate which starts from 1e-3 and exponentially decays to 1e-4. Following [18], our implementation involves a deformable slicing surface modeled as a MLP with a depth of 6 and width of 64, incorporating a skip connection at the 5th layer. While leveraging the official implementation of [18], we enhance training by including densely sampled frames from the videos. We adopt a sampling rate of 256 points per ray for input resolution of 1920×1080 and conduct training on four NVIDIA GeForce RTX 3090 GPUs, iterating for approximately 1.2 million steps to achieve convergence on our dataset.

NeRF-T Utilizing a learning rate of 0.0005 as suggested in [10], we employ a sampling rate of 128 points per ray for input resolution of 1920×1080. Our model training occurs across four NVIDIA GeForce RTX 3090 GPUs, requiring approximately 3 days for each scene within our dataset.

NeuMan We have directly utilized the official implementation of NeuMan [8] and conducted testing on our dataset. Training the model on four NVIDIA GeForce RTX 3090 GPUs required approximately 7 days for each scene in our

dataset. Initially, during the training's first stage, the closest point-finding operation relied on the RANSAC algorithm. However, challenges arose when working with datasets like Rock Climbing or grassland due to potential multiple contact points between humans and the scene. The instability of the RANSAC algorithm resulted in significant errors across different instances, leading to failures in our dataset. To address this issue, we introduced benchmark scales calculated by a calibration board for scenes like football and rock climbing. Despite these adjustments, NeuMan's limitation in considering unbounded scene reconstruction design led to rendering issues such as noticeable blurring in the background and ghosting effects in large-scale outdoor settings when compared to the original NeRF outputs.

HOSNeRF Based on the official implementation of HOSNeRF [12], we follow the three-stage training procedure outlined in their codebase. Training the model on eight NVIDIA GeForce RTX 3090 GPUs took approximately 6 days to complete. In our utilization of the AerialRecon dataset, we leveraged the scale factor determined by the calibration board positioned within the scene. This approach enhances the precision and robustness of coordinate system alignment, ensuring accurate human-scene alignment. Notably, this calibration information is readily available across all methods within the AerialRecon dataset, facilitating alignment adjustments as needed.

Monocular 4D Gaussian Splatting We use the point cloud attained in our calibration model for gaussian in [21, 23] as initialization. To be specific, we combine the first-frame human point could converted by SMPL[13] and the background point cloud attained during our calibration process for point cloud initialization. Then we use the official implementation from [21, 23].

Nerfies We directly adopt the official implementation of Nerfies[17] by employing the Adam optimizer with a learning rate that exponentially decays by a factor of 0.1 until reaching the maximum number of iterations. The training cost eight NVIDIA GeForce RTX 3090 GPUs for 24hours.

NeRFPlayer We use the nerfstudio version code for test. We select the version using Instant-NGP[16] as backbone. The method utilizing the TensoRF backbone is not documented due to the excessive GPU memory demands during training.

K-Planes The official implementation using hybrid version with modules like proposal sampling are used in experiments. It is important to note that importance sampling is not applicable to monocular videos or datasets involving moving cameras.

TiNeuVox We use the official PyTorch implementation of TiNeuVox-B. We initialize the neural voxels with zero-values and double the voxel resolution after 4K and 8k iterations, respectively.

FFDNeRF We implement the method according to the paper[6]. The average splatting and coarse-to-fine training strategy are utilized. We train the model on each scene using one NVIDIA GeForce RTX 3090 GPU for around 30 hours.

Our Method Our comprehensive joint training process takes approximately 3 days, involving the simultaneous training of our complete model, which encompasses both the scene model Ψ_{s} and the human model Ψ_{h}, across four NVIDIA RTX 3090 GPUs. Before that, the preprocessing stage, which includes tasks like COLMAP and MoCap, requires around 3 hours when processing input images of roughly 600 frames. When applying our method to other datasets such as NeuMan and HOSNeRF, we adhere to their implementations as detailed in [8, 12], manually calculating scales and aligning scenes accordingly. Notably, we utilize Mip-NeRF360 as the backbone for our scenes. Transitioning to a faster background backbone like the Gaussian-Splatting-based method [9] holds the potential to significantly expedite training processes.

3 Dataset and Future Work

Our proposed AerialRecon dataset features 25 real sports scenes captured using 4 synchronized drone cameras. Comprising 400 clips of drone videos and over 120,000 images, it includes multi-view images accurately aligned within the same point cloud under a unified world coordinate system for each specific scene. This dataset paves a road for future research endeavors.

Dynamic Objects Reconstructing dynamic objects from a monocular RGB video captured in real-world settings remains as a challenge. The inherent ambiguity in monocular depth estimation complicates tasks such as monocular 3D object tracking and 6DOF pose estimation, especially in unpredictable environments. Leveraging multi-view data from 4 timely-synchronized drones in AerialRecon could be helpful in those tasks.

Gaussian Splatting Existing datasets of dynamic scene reconstruction are either synthetic or contain limited human movements. There is an urgent need for a dataset collected in real-world scenarios to fill the gap. AerialRecon is a dataset consists of real-world outdoor dynamic scenarios. It will facilitate research in 4D reconstruction, especially in the emerging field of 4D Gaussian Splatting.

Drone Formations In AerialRecon, each sports scene aligns images from four distinct viewpoints to a unified world coordinate during calibration. This alignment sets the stage for developing multi-view systems using the AerialRecon dataset. Subsequent research could harness the synchronized data from the four drones to devise algorithms centered around drone formations.

4 Layered Representation

Following [7, 19], a neural layered representation is utilized in joint training 4.3 and compositional volume rendering4.2.

4.1 Layered Ray Sampling Strategy

In Fig. 2, we illustrate the partitioning of 3D space into two regions using a 3D bounding box. During human motion capture, we acquire per-frame human SMPL meshes [13] in the 3D world coordinate system, accompanied by their respective 3D bounding boxes. This 3D bounding box serves as a reference point for segmenting the 3D scene into interior and exterior spaces.

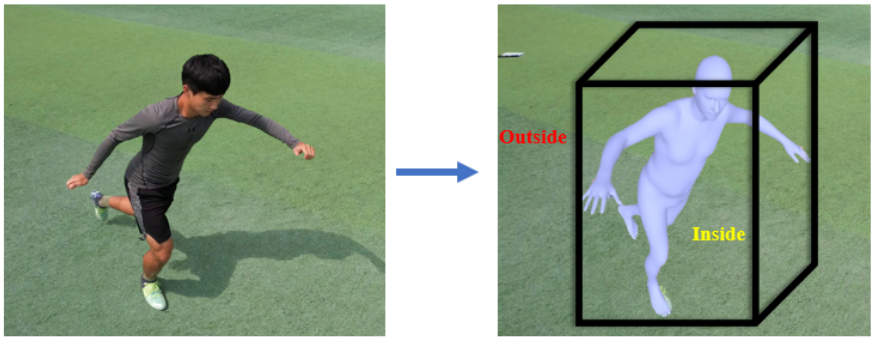

Fig. 2: 3D Bounding from SMPL[13] for 3D parsing scenes

Subsequently, we implement a layered ray sampling method shown in Fig. 3. More specifically, when a camera ray intersects with the human, we select points within the bounding box for training our human model Ψ_{h} and points outside the bounding box for training our background model Ψ_{s}. In cases where the ray does not pass through the bounding box, all sampled points are utilized for training the background model Ψ_{s}.

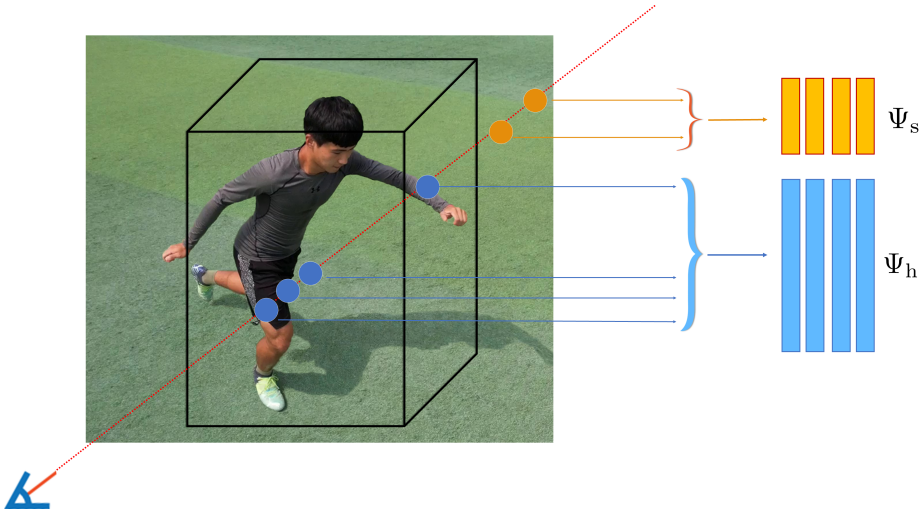

Fig. 3: Layered Ray Sampling Strategy

4.2 Compositional Volume Rendering

Vanilla-NeRF [15] uses conventional Volume Rendering Equations by Max's insights on the quadrature rule of volume rendering [14].

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} T_i \left(1 - \exp\left(-\sigma_i \delta_i\right)\right) \mathbf{c}_i \tag{1}$$

$$T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right) \tag{2}$$

Utilizing the layered Ray Sampling Strategy, we individually query Ψ_h and Ψ_s to obtain the composited rendering output. Here, $C^h(\mathbf{r})$ represents the accumulated color from samples within the 3D bounding box, while $C^s(\mathbf{r})$ denotes the accumulated color from samples outside the 3D bounding box. The rendered color of a pixel $\hat{C}(\mathrm{r})$ can be calculated as follows:

$$\hat{C}(\mathbf{r}) = C^h(\mathbf{r}) + \left(1 - \alpha^h(\mathbf{r})\right) C^s(\mathbf{r}) \tag{3}$$

Here the α^h is the alpha values of the human layer

$$\alpha_i^h = 1 - \exp\left(-\sigma_i^h \delta_i^h\right) \tag{4}$$

4.3 Joint Training without segmentation mask

NeRF[15] uses L2-norm as a loss function for supervised training:

$$\mathcal{L} = \sum_{\mathrm{r} \in \mathcal{R}} \left(\left\| C(\mathrm{r}) - \hat{C}(\mathrm{r}) \right\|_2^2 \right) \tag{5}$$

A conventional approach involves utilizing a 2D human segmentation mask to partition input images into distinct sections for separate network processing, leading to seperate training of the human and scene models. However, these methods typically necessitate a post-processing step involving α-composition during rendering, which can introduce significant artifacts in the final output. In contrast to these approaches, our method directly segments the scene using a 3D bounding box, eliminating the requirement for a 2D segmentation mask to differentiate between human and background pixels during training.

5 3D Pose Visualization

We also visualize the per-frame human 3D vertices onto the 3D background produced by MVS dense reconstruction [5].
Consecutive 3D pose estimation: The visualization result, as shown in Fig. 4, helps us to validate the temporal consistency of the estimated positions of the performer in the 3D space. In the illustration, varying colors on the human body signify the athlete's positions at different instances during a penalty kick. The results demonstrate that the estimated human motion maintains a plausible spatial alignment in 3D space while exhibiting temporal consistency across frames.

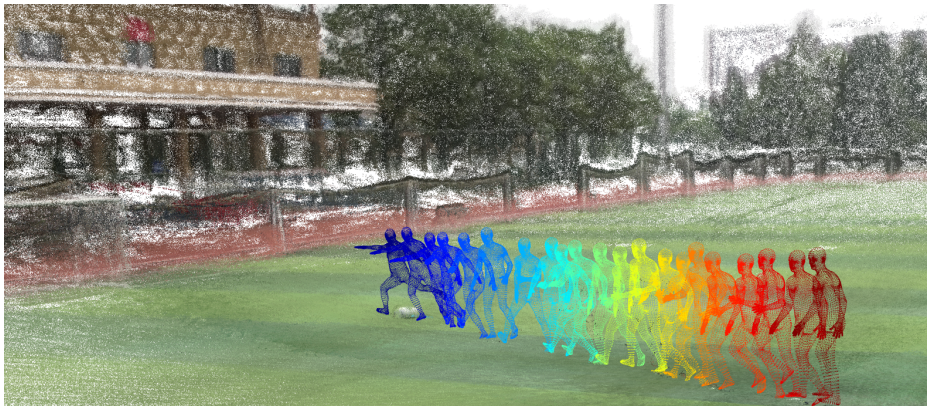

Fig. 4: Visualization of consecutive SMPL vertices on MVS dense point cloud, different color refers to the different moments in a kicking motion

6 Human Pose Alignment

We perform a quantitative evaluation of human pose estimation comparing drone and handheld methods. The metrics **AP**, $\mathbf{AP}^{50}$, and $\mathbf{AP}^{75}$ are defined in the COCO 2017 dataset [11]. While our SMPL estimation uses a 25-point keypoint

model from OpenPose [2], our quantitative analysis of 2D keypoints uniformly uses the 17-point model specified by COCO 2017 [11], as illustrated in Fig. 5. It is worth noting that 25-point keypoint can be converted into the 17-point model, as all the 17 points can be found in 25-point result.

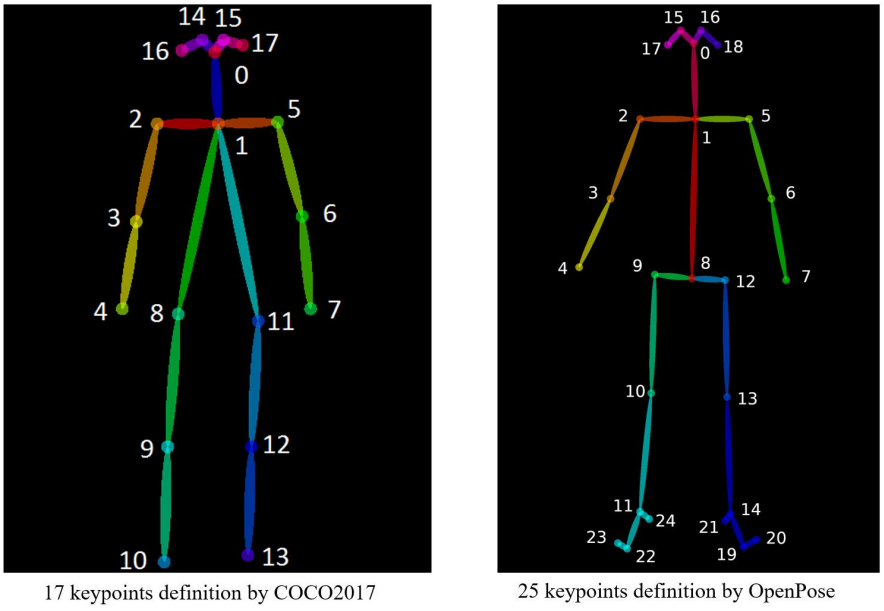

Fig. 5: 2D keypoints alignment between OpenPose[2] and COCO[11]

Bibliography

[1] Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5470–5479 (2022)

[2] Cao, Z., Hidalgo Martinez, G., Simon, T., Wei, S., Sheikh, Y.A.: Openpose: Realtime multi-person 2d pose estimation using part affinity fields. IEEE Transactions on Pattern Analysis and Machine Intelligence (2019)

[3] Fang, J., Yi, T., Wang, X., Xie, L., Zhang, X., Liu, W., Nießner, M., Tian, Q.: Fast dynamic radiance fields with time-aware neural voxels. In: SIGGRAPH Asia 2022 Conference Papers. pp. 1–9 (2022)

[4] Fridovich-Keil, S., Meanti, G., Warburg, F.R., Recht, B., Kanazawa, A.: K-planes: Explicit radiance fields in space, time, and appearance. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12479–12488 (2023)

[5] Furukawa, Y., Ponce, J.: Accurate, dense, and robust multiview stereopsis. IEEE transactions on pattern analysis and machine intelligence **32**(8), 1362–1376 (2009)

[6] Guo, X., Sun, J., Dai, Y., Chen, G., Ye, X., Tan, X., Ding, E., Zhang, Y., Wang, J.: Forward flow for novel view synthesis of dynamic scenes. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 16022–16033 (2023)

[7] Jiakai, Z., Xinhang, L., Xinyi, Y., Fuqiang, Z., Yanshun, Z., Minye, W., Yingliang, Z., Lan, X., Jingyi, Y.: Editable free-viewpoint video using a layered neural representation. In: ACM SIGGRAPH (2021)

[8] Jiang, W., Yi, K.M., Samei, G., Tuzel, O., Ranjan, A.: Neuman: Neural human radiance field from a single video. In: Proceedings of the European conference on computer vision (ECCV) (2022)

[9] Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (2023)

[10] Li, Z., Niklaus, S., Snavely, N., Wang, O.: Neural scene flow fields for space-time view synthesis of dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2021)

[11] Lin, T.Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C.L., Dollár, P.: Microsoft coco: Common objects in context (2014), `http://arxiv.org/abs/1405.0312`, cite arxiv:1405.0312Comment: 1) updated annotation pipeline description and figures; 2) added new section describing datasets splits; 3) updated author list

[12] Liu, J.W., Cao, Y.P., Yang, T., Xu, E.Z., Keppo, J., Shan, Y., Qie, X., Shou, M.Z.: Hosnerf: Dynamic human-object-scene neural radiance fields from a single video. arXiv preprint arXiv:2304.12281 (2023)

[13] Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.J.: Smpl: A skinned multi-person linear model. ACM transactions on graphics (TOG) **34**(6), 1–16 (2015)
[14] Max, N.: Optical models for direct volume rendering. IEEE Transactions on Visualization and Computer Graphics **1**(2), 99–108 (1995)
[15] Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM **65**(1), 99–106 (2021)
[16] Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. Graph. **41**(4), 102:1–102:15 (Jul 2022). https://doi.org/10.1145/3528223.3530127, https://doi.org/10.1145/3528223.3530127
[17] Park, K., Sinha, U., Barron, J.T., Bouaziz, S., Goldman, D.B., Seitz, S.M., Martin-Brualla, R.: Nerfies: Deformable neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5865–5874 (2021)
[18] Park, K., Sinha, U., Hedman, P., Barron, J.T., Bouaziz, S., Goldman, D.B., Martin-Brualla, R., Seitz, S.M.: Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. ACM Trans. Graph. **40**(6) (dec 2021)
[19] Qing, S., Chen, G., Qi, F., Sida, P., Wenhao, S., Xiaowei, Z., Hujun, B.: Novel view synthesis of human interactions from sparse multi-view videos. In: SIGGRAPH Conference Proceedings (2022)
[20] Song, L., Chen, A., Li, Z., Chen, Z., Chen, L., Yuan, J., Xu, Y., Geiger, A.: Nerfplayer: A streamable dynamic scene representation with decomposed neural radiance fields. IEEE Transactions on Visualization and Computer Graphics **29**(5), 2732–2742 (2023)
[21] Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., Xinggang, W.: 4d gaussian splatting for real-time dynamic scene rendering. arXiv preprint arXiv:2310.08528 (2023)
[22] Wu, T., Zhong, F., Tagliasacchi, A., Cole, F., Oztireli, C.: D^ 2nerf: Self-supervised decoupling of dynamic and static objects from a monocular video. Advances in Neural Information Processing Systems **35**, 32653–32666 (2022)
[23] Yang, Z., Gao, X., Zhou, W., Jiao, S., Zhang, Y., Jin, X.: Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. arXiv preprint arXiv:2309.13101 (2023)
[24] Zhang, K., Riegler, G., Snavely, N., Koltun, V.: Nerf++: Analyzing and improving neural radiance fields. arXiv preprint arXiv:2010.07492 (2020)