VideoRF: Rendering Dynamic Radiance Fields as 2D Feature Video Streams

Liao Wang[1,3*] Kaixin Yao[1,3*] Chengcheng Guo[1] Zhirui Zhang[1] Qiang Hu[4#]
Jingyi Yu[1] Lan Xu[1†] Minye Wu[2†]
[1] ShanghaiTech University [2] KU Leuven [3] NeuDim [4] Shanghai Jiao Tong University

Figure 1. Our proposed VideoRF views dynamic radiance field as 2D feature video streams combined with deferred rendering. This technique facilitates hardware video codec and shader-based rendering, enabling smooth high-quality rendering across diverse devices.

Abstract

Neural Radiance Fields (NeRFs) excel in photorealistically rendering static scenes. However, rendering dynamic, long-duration radiance fields on ubiquitous devices remains challenging, due to data storage and computational constraints. In this paper, we introduce VideoRF, the first approach to enable real-time streaming and rendering of dynamic human-centric radiance fields on mobile platforms. At the core is a serialized 2D feature image stream representing the 4D radiance field all in one. We introduce a tailored training scheme directly applied to this 2D domain to impose the temporal and spatial redundancy of the feature image stream. By leveraging the redundancy, we show that the feature image stream can be efficiently compressed by 2D video codecs, which allows us to exploit video hardware accelerators to achieve real-time decoding. On the other hand, based on the feature image stream, we propose a novel rendering pipeline for VideoRF, which has specialized space mappings to query radiance properties efficiently. Paired with a deferred shading model, VideoRF has the capability of real-time rendering on mobile devices thanks to its efficiency. We have developed a real-time interactive player that enables online streaming and rendering of dynamic scenes, offering a seamless and immersive free-viewpoint experience across a range of devices, from desktops to mobile phones. Our project page is available at https://aoliao12138.github.io/VideoRF/.

* Authors contributed equally to this work. † The corresponding authors are Minye Wu (minye.wu@kuleuven.be) and Lan Xu (xulan1@shanghaitech.edu.cn). # Work done while at ShanghaiTech University.

1. Introduction

Photorealistic Free-Viewpoint Video (FVV) of dynamic scenes offers an immersive experience in virtual reality and telepresence. Work involving Neural Radiance Fields (NeRFs) has shown great potential in creating photorealistic Free-Viewpoint Videos (FVVs). However, there are still challenges in smoothly delivering and rendering FVVs using NeRFs on commonly used devices, similar to the ease of watching online videos. The difficulty lies in reducing the data capacity for transmitting and storing long sequences and ensuring a low, mobile-compatible computational load.

Neural Radiance Field (NeRF) [39] surpasses traditional 3D reconstruction methods in photorealistic novel view synthesis. Several works extend NeRF to dynamic scenes by maintaining a canonical space and matching it implicitly [10, 42, 48] or explicitly [34] to align with each frame's live space. However, their dependence on canonical space limits its effectiveness in sequences with large motions or topo-

logical changes. Other methods introduce new representations like 4D feature grids [18] or temporal voxel features [11], achieving impressive results on the scenes with topological transformations. Yet, [11] struggles with representing longer sequences owing to model capacity constraints, while [18] encounters streaming challenges due to large storage needs. Recent efforts [26, 54, 65] focus on compressing dynamic frames for streaming, but their computational intensity limits mobile applicability. Concurrently, on mobile devices, real-time rendering of static scenes has been achieved by baking NeRF into mesh templates [1, 52, 75] or texture assets [50]. However, these techniques fall short for dynamic scenes as their per-frame representation becomes too bulky for real-time loading. Moreover, while NeRF compression methods [51, 56, 65] can be employed, they introduce decoding or rendering overheads unsuitable for mobile platforms. Despite the existence of NeRF solutions for dynamic scenes and mobile optimization, a cohesive approach that effectively addresses both still poses a difficulty.

In this paper, we propose *VideoRF* – a novel neural modeling approach that enables real-time streaming and rendering of human-centric dynamic radiance fields on common mobile devices (see Fig. 1). Our key idea is to view the 4D feature volumes reconstructed from a dynamic scene as a 2D feature image stream, which is friendly to video codec. Each feature image records the densities and appearance features of one frame. We hence propose a rendering pipeline for this representation. In this rendering pipeline, VideoRF uses a set of mapping tables to connect the 3D space with the 2D feature images. This allows for O(1) density and feature retrieval from the images for every 3D sample point in the space. To reduce the computational complexity of rendering, we adopt the deferred shading model [50] with a global tiny MLP to decode the integrated feature vectors into pixel colors. All rendering operations here are low-cost and compatible with the fragment shader model, making it possible to implement this rendering pipeline on various devices that have GPUs.

Second, we present a sequential training scheme to effectively generate the 2D feature image stream. Specifically, VideoRF adaptively groups frames by analyzing motion from the sequential data to ensure temporal stability for mapping table generation. We deploy 3D and 2D Morton sorting techniques to improve spatial consistency in the mapping. VideoRF contains spatial and temporal continuity regularizers that are applied on the mapped 2D feature images using the mapping table. This training strategy enforces the temporal and spatial redundancy to the 2D feature image stream. We show that our feature image stream with spatiotemporal sparsity can be efficiently compressed by off-the-shelf video codecs and reaches high compression rates.

Moreover, based on these findings, we build a cross-platform player based on VideoRF that can play FVVs in real-time, supported by video hardware accelerators. In this way, users can interactively drag, rotate, pause, play, fast-forward, rewind, and jump to specific frames, providing a viewing experience as seamless as watching online videos on various devices, including smartphones, tablets, laptops, and desktops which was unseen before.

To summarize, our contributions include:

- We propose VideoRF, a novel approach to enable real-time dynamic radiance field decoding, streaming and rendering on mobile devices.
- We present an efficient and compact representation, which represents 4D radiance field into 2D feature stream with low rendering complexity to support hardware video codec and shader rendering.
- We introduce a training scheme to directly impose spatial-temporal consistency on our 2D feature stream for efficient compression.

2. Related work

Novel View Synthesis for Dynamic Scenes. Dynamic scenes pose a greater challenge in achieving realistic view synthesis results due to moving objects. One approach is to reconstruct the dynamic scene and render the geometry from new viewpoints. The conventional RGB [7, 23, 30, 36, 37, 49, 80] or RGB-D [9, 19–21, 41, 70, 71] solutions have been extensively investigated. Mesh-based neural network [5, 61, 68] techniques are effective for compact data storage and can record view-dependent texture [5, 68], but such methods heavily rely on geometry, especially perform poorly for topologically complex scenarios.

Many methods extend NeRF into the dynamic view synthesis settings. Some methods [10, 15, 29, 31–33, 43, 48, 63, 69, 77, 79] handle spatial change directly conditions on time and [14, 45, 46, 73, 82] use feature latent code to represent time information. However, they do not support the streaming of long sequences due to the limited representation. Others learn spatial offsets from the live scene to the canonical radiance field by using explicit voxel matching [34], skeletal poses [25, 35, 44], deformed hashtable [24] and deformed volume [22, 59, 60, 66, 72, 81]. While these methods can successfully render dynamic scenes with photo-realistic quality, their heavy reliance on the canonical space makes these methods vulnerable to long sequences, large motions, and topological alterations. Recently, several methods using 4D planes [3, 13, 18, 53, 74], voxel grid [11], Fourier representation [64], residual layers [38] or dynamic MLP maps [47] to represent dynamic scene. However, the inference computational complexity makes it difficult for them to implement streaming and decoding on mobile devices. Additionally, several methods such as [26, 54, 65] have managed to facilitate the streaming of dynamic radi-

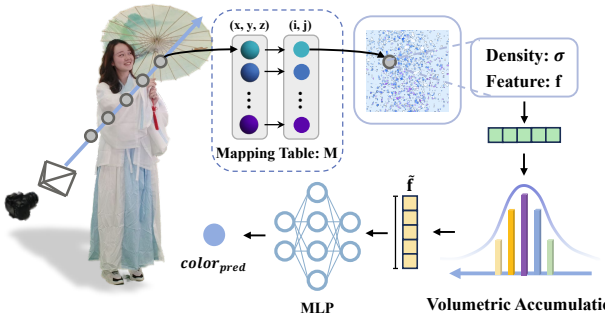

Figure 2. Demonstration of our rendering. For each 3D sample point, its density σ and feature $\mathbf{f}$ are fetched from the 2D feature image through the mapping table $\mathbf{M}$. Each point feature is first volumetrically accumulated to get the ray feature $\tilde{\mathbf{f}}$ and pass MLP Φ to decode the ray color.

ance fields by reducing the capacity of each frame, but the lack of general hardware acceleration makes it challenging to represent dynamic scenes in real-time on mobile phones.

Cross Device Neural Radiance Field Rendering. Recent works have demonstrated that static neural radiance fields can be rendered on mobile devices. They achieve this by directly converting the neural radiance field to mesh [58] or using traditional texture to represent the scene. [1, 75, 78] have endeavored to augment the capabilities of neural rendering by employing mesh templates and feature texture, [2, 52] facilitating the rendering process through fragment shader to achieve surface-like rendering with similar ideas. [62] represents neural radiance features encoded on a two-layer duplex mesh for better rendering quality.

NeRF Acceleration and Compression. NeRF demonstrates exceptional performance in generating images from arbitrary viewpoints, but it suffers from slow rendering efficiency. Some methods focus on integrating a compact structure with a simplified MLP, which reduces the complexity of MLP calculations in traditional NeRFs. Key strategies have been explored, including the employment of voxel grid [26, 55], octrees [12, 64, 76], tri-planes [4], hashing encoding [40], codebook [27, 56], tensor decomposition [6, 57] to accomplish this. Using explicit structure makes training and rendering faster, but it also means that the 3D structure takes up more storage space.

Therefore, while methods such as CP-decomposition [6], rank reduction [57], and vector quantization [56] manage to attain modest levels of data compression, their application remains confined to static scenes. Additionally, some approaches [8, 51, 65] opt for post-processing techniques, initially utilizing baseline methods to train a decent NeRF representation, then special encoding and decoding schemes are employed to reduce the storage space. However, these kinds of methods are not hardware-friendly, the inference process is computationally demanding and cannot support real-time streaming on mobile devices. In contrast, our VideoRF uses codec tailored training, hardware-friendly decoding and shader rendering, which enables real-time dynamic radiance field for long sequences with large motion.

3. VideoRF Representation

To facilitate dynamic radiance field rendering on mobile devices, we adopt a representation that aligns with the video codec format and maintains low computational complexity for shader-based rendering. We propose turning the 3D volume representation into a 2D formulation, coupled with a highly efficient rendering pipeline.

In our approach, each frame of the radiance field is represented as a feature image $\mathbf{I}$ where the first channel stores density and the remaining h channels store feature. As depicted in Fig. 2, given a 3D vertex position $\mathbf{x}$, we retrieve its density σ and feature $\mathbf{f}$ using the equation:

$$\sigma, \mathbf{f} = \mathbf{I}[\mathbf{M}(\mathbf{x})], \tag{1}$$

where $\mathbf{M}$ is the 3D-to-2D mapping table from Sec. 4.1. This mapping table not only effectively excludes empty space to reduce storage but also specifics mapping from each non-empty 3D vertex to a corresponding 2D pixel. The lookup operation is highly efficient, with a time complexity of O(1), facilitating rapid and convenient queries. It's worth noting that our sequence of feature images is in a 2D format, which is friendly to the video codec hardware.

For rendering, inspired by [17, 50], we use a deferred rendering model. We first accumulate the features along the ray:

$$\begin{aligned} \tilde{\mathbf{f}}(\mathbf{r}) &= \sum_{k=1}^{n_s} T_k (1 - \exp(-\sigma_k \delta_k)) \mathbf{f}_k, \\ T_k &= \exp\left(- \sum_{j=1}^{k-1} \sigma_j \delta_j \right), \end{aligned} \tag{2}$$

where n_s is the number of sample points along the ray $\mathbf{r}$. σ_k, $\mathbf{f}_k$, δ_k denotes the density, feature of the samples, and the interval between adjacent samples respectively. The view-dependent color of the ray is then computed using a tiny global MLP Φ shared across the frames as:

$$\tilde{C}(\mathbf{r}) = \text{sigmoid}\left(\Phi\left(\tilde{\mathbf{f}}(\mathbf{r}), \mathbf{d} \right) \right), \tag{3}$$

where $\mathbf{d}$ is the view direction of the ray after positional encoding [39]. In this way, the computational burden is significantly reduced as each ray requires only a single MLP decoding which can be implemented in a shader for real-time rendering on mobile devices.

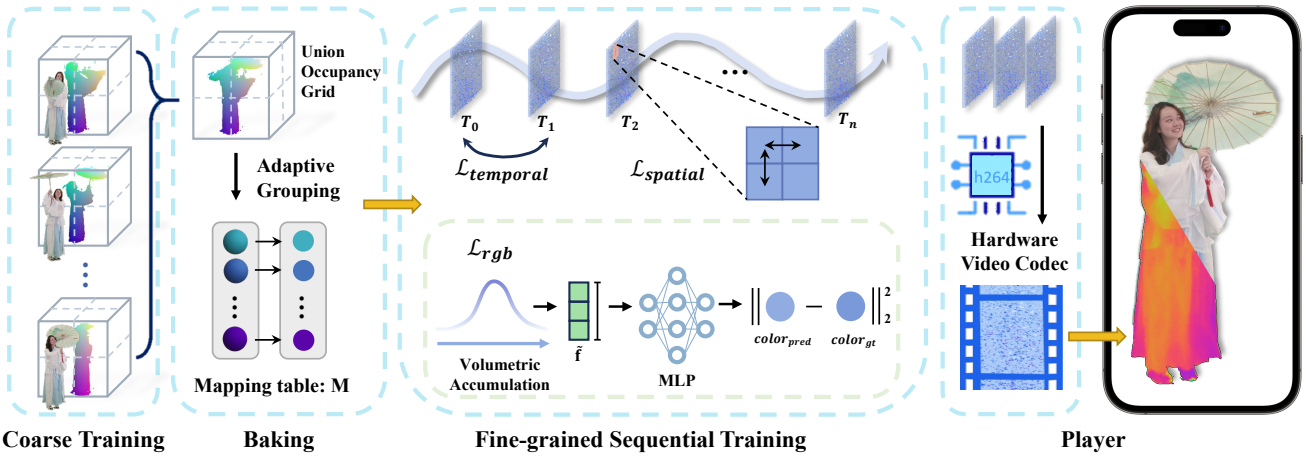

Figure 3. Overview of our video codec-friendly training. First, we apply our grid-based coarse training [55] to generate per-frame occupancy grid $\mathbf{O}^t$. Then, during baking, we adaptively group each frame and create a mapping table $\mathbf{M}$ for each group. Next, we sequentially train each feature image $\mathbf{I}^t$ through our spatial, temporal and photometric loss. Finally, feature images are compressed into the feature video streaming to the player.

4. Video Codec-friendly Training

In this section, we propose a training scheme, as depicted in Fig. 3, to achieve a high compression rate by maintaining spatial and temporal consistency. For spatial aspect, we incorporate 3D-2D Morton sorting to preserve 3D continuity and apply a spatial consistency loss directly on 2D feature images to enforce spatial coherence. On the temporal front, we employ adaptive grouping, which allows frames within a group to share mappings, thereby reducing temporal disruptions, and temporal consistency loss, which serves to further reinforce temporal coherence. These consistencies are crucial which provide a stable and predictable layout for video codecs.

4.1. Baking and Map Generation

Coarse stage pre-training. Given multiview images of each frame, we first adopt an off-the-shelf approach [55] to generate the explicit density grid $\mathbf{V}_\sigma^t$ for each frame t independently. We then create a per-frame occupancy grid $\mathbf{O}^t$ by masking in voxels whose opacity exceeds a certain threshold γ. This coarse stage sets the foundation for our subsequent adaptive grouping in the baking stage.

Adaptive group. Generating an independent mapping table for each frame t could disrupt temporal continuity, leading to increased storage requirements after video codec compression. This issue arises because the same 3D point in adjacent frames might map to vastly different 2D positions on the feature image. On the other hand, uniformly applying a single mapping table across all frames could introduce significant spatial redundancy. Considering that a 3D point may only be occupied at sparse intervals, it could remain underutilized for most of the duration, resulting in inefficiency. To balance these factors, we divide the sequence into Groups of Frames (GOFs), maintaining a fixed resolution for our 2D feature image and adaptively determining the number of frames in each group. For a set of consecutive frames $\{i, i+1, \ldots, i+n\}$, we identify the maximum frame number α such that the number of occupied voxels in the union from i to α does not exceed our pixel limit θ:

$$\text{argmax}_\alpha \, \text{g}\left(\bigcup_{j=i}^{\alpha} \mathbf{O}^j\right) \leqslant \theta, \tag{4}$$

where $\text{g}()$ means the number of occupied grids in the union occupancy grid. Then, we will set the frame i to frame α as a GOF and frame $\alpha+1$ to be the start frame of a new GOF. In this way, all the frames in the group share the same mapping. The same 3D position within the group will be mapped to the same 2D pixel location which keeps temporal consistency and saves the storage.

Our grouping strategy, unlike the fixed feature grid grouping in ReRF [65], is adaptive, aligning with our objectives. This flexibility allows frames within a group to share a mapping table, optimized in size for a perfect balance between storage efficiency and maintaining temporal continuity. On the other hand, the primary aim of ReRF is to facilitate fast-seeking. Furthermore, our approach diverges from the per-frame occupancy grid of MLP-Maps [47]. We employ a union occupancy grid to calculate the effective number of pixels for adaptive grouping, rather than for speeding up rendering.

3D Morton sorting. After the union occupancy grid is achieved, we apply Morton sorting to record its 3D spatial continuity. Morton ordering [67], or Z-ordering, interleaves the binary representations of spatial coordinates, ensuring that spatially proximate entities remain adjacent in linear space. As shown in Fig. 4(a), we apply 3D Morton sort-

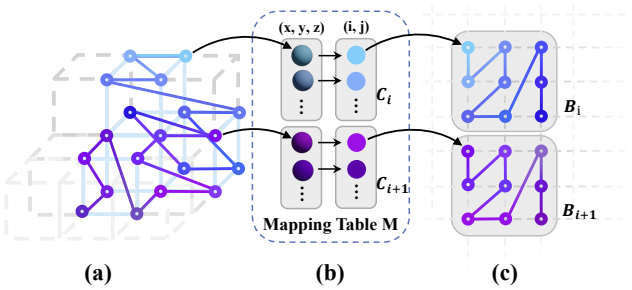

Figure 4. Illustration of our mapping table generation. We first perform 3D Morton sorting **(a)** on each nonempty vertice and group it into chunks C_i **(b)**. Next, we lay out each chunk into each block B_i of the feature image, arranged in 2D Morton order **(c)** within it.

ing to the vertices based on their position coordinates of the union occupancy grid. Vertices with a density below the threshold γ are excluded. This process effectively maintains 3D spatial consistency in the ordering.

2D block partitioning and 2D Morton sorting. To preserve the 3D spatial continuity within a 2D framework, we employ 2D Morton sorting and partition the feature images into blocks. This approach aligns with the block-wise compression of frames in video codecs, where blocks with local smoothness lead to more efficient storage. Specifically, we first divide the feature image into N 8×8 blocks, denoted as B_i, and correspondingly group the sorted vertices into N 8×8 chunks, denoted as C_i, as illustrated in Fig. 4(b). For each pixel $\mathbf{p}$ in a block B_i, we sort its relative position (u, v) in 2D Morton order. As shown in Fig. 4(c), each chunk C_i is then mapped to a block B_i, arranged in 2D Morton order within the block, to form the mapping table $\mathbf{M}$. This 2D Morton ordering ensures that sorted values in the 3D Morton ordering are positioned closely within each feature image block, facilitating efficient compression during the transformation process.

4.2. Fine-grained Sequential Training

With the aid of our mapping table, we can sequentially train our feature images within each group through spatial consistency loss applied to 2D feature images and temporal consistency loss between frames.

Spatial consistency loss. In video encoding, regions with homogeneous characteristics demonstrate high compression efficiency. This efficiency stems from the minimal variation in pixel values within these areas, leading to a significant reduction in high-frequency components post Discrete Cosine Transform (DCT) within the video codec. Consequently, these regions retain a higher proportion of low-frequency components. Furthermore, the quantized coefficients in such homogeneous regions are more likely to contain a greater number of zero values, facilitating a more compact data representation and reducing the overall data volume. In order to enhance the homogeneity of our 2D feature image, we introduce a total variance loss, $\mathcal{L}_{\text{spatial}}$, during our fine-grained training stage. For each channel of the feature image $\mathbf{I}$, we enforce its local smoothness by:

$$\mathcal{L}_{\text{spatial}} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{V}} \left(\Delta_u(\mathbf{p}) + \Delta_v(\mathbf{p}) \right), \tag{5}$$

where $\mathcal{P}$ is the pixels of the feature image, $\Delta_u(\mathbf{p})$ shorthand for Manhattan distance between the feature value at pixel $\mathbf{p} := (u, v)$ and the feature value at pixel $(u + 1, v)$ normalized by the resolution, and analogously for $\Delta_v(\mathbf{p})$. By increasing the spatial sparsity, the storage of feature videos after video encoding is decreased at the same quality.

Temporal consistency loss. A naive per-frame training scheme will disrupt temporal continuities by failing to incorporate inter-frame information and resulting in a high bitrate. This is because the residuals between frames are stored after entropy encoding. To optimize storage efficiency, we focus on minimizing the differences in the feature space between the adjacent frames to reduce the entropy. During our sequential training, we enhance inter-frame similarities by regularizing the current feature image with its predecessor, except for the initial frame of each adaptive group. This is achieved by applying

$$\mathcal{L}_{\text{temporal}} = \|\mathbf{I}^t - \mathbf{I}^{t-1}\|_1, \tag{6}$$

for each frame t in the group, ensuring small residuals between consecutive feature images. By employing temporal smoothness in this manner, we can further mitigate entropy throughout the video codec process, thereby facilitating storage conservation.

Training objective. Our total loss function is formulated as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{rgb}} + \lambda_s \mathcal{L}_{\text{spatial}} + \lambda_t \mathcal{L}_{\text{temporal}}, \tag{7}$$

where λ_s and λ_t are the weights for our regular terms and $\mathcal{L}_{\text{rgb}}$ is the photometric loss,

$$\mathcal{L}_{\text{rgb}} = \sum_{\mathbf{r} \in \mathcal{R}} \|\mathbf{c}(\mathbf{r}) - \hat{\mathbf{c}}(\mathbf{r})\|^2, \tag{8}$$

where $\mathcal{R}$ is the set of training pixel rays; $\mathbf{c}(\mathbf{r})$ and $\hat{\mathbf{c}}(\mathbf{r})$ are the ground truth color and predicted color of a ray $\mathbf{r}$ respectively.

4.3. VideoRF Player

Finally, we implement a companion VideoRF player to stream and render dynamic radiance fields on mobile devices. Given our feature streams, we quantize them into uint8 format and use H.264 for video codec. Different from [50], we save the mapping table in a 2D-to-3D format as an

Figure 5. Our VideoRF method generates results for inward-facing, 360° video sequences featuring human-object interactions with large motion. The images in the last row illustrate our ability to implement variable bitrate in these sequences.

RGB image to conserve bitrate during streaming. Then, to enable rapid rendering, we first use this mapping table to recover a 3D volume. We employ a compute shader for efficient processing, segmenting the 512×512 mapping table into 16×16 workgroups, each handling a 32×32 pixel section. The highly parallel architecture of compute shaders enables us to efficiently convert 2D features back into a 3D volume. For the rendering part, we implement it via a fragment shader. For fast raymarching, we employ a multi-resolution hierarchy of occupancy grids for each group to skip empty space at different levels. We leverage matrix multiplication in the shader to simulate our tiny MLP calculation. These techniques increase the overall speed while ensuring compact storage.

The VideoRF player marks a significant milestone, as it first enables users to experience real-time rendering of dynamic radiance fields of any length. Within this player, users can drag, rotate, pause, play, fast forward/backward, seek dynamic scenes, or switch between different resolutions like watching online video, offering an extraordinary high-quality free-viewpoint viewing experience. This capability also extends across a wide range of devices, from smartphones and tablets to laptops and desktops, broadening the accessibility and applicability of dynamic radiance fields.

5. Experimental Results

In this section, we evaluate our VideoRF on a variety of challenging scenarios. We use the PyTorch Framework to train the model on a single NVIDIA GeForce RTX3090. Our new captured dynamic datasets contain around 80 views at the resolution of 1920 × 1080 at 30 fps. To ensure the robustness of the algorithm, we also use the ReRF dataset and the HumanRF dataset for result demonstration in Fig. 5. We can render images for immersive, 360° video sequences that capture human interactions with objects, especially for large motion and long duration sequences. Please refer to the supplementary video for more video results.

5.1. Comparison

To the best of our knowledge, our approach is the first real-time dynamic radiance field approach that can decode and render on mobile devices. Therefore, we compare to existing dynamic neural rendering methods, including ReRF [65], HumanRF [18], TiNeuVox [11] and per-frame static reconstruction methods like MeRF [50]. We use ReRF [65] dataset and HumanRF [18] Actor3 sequence 1 for a fair comparison, both in storage memory and image quality. As shown in Fig. 6, though TiNeuVox [11] has small storage,

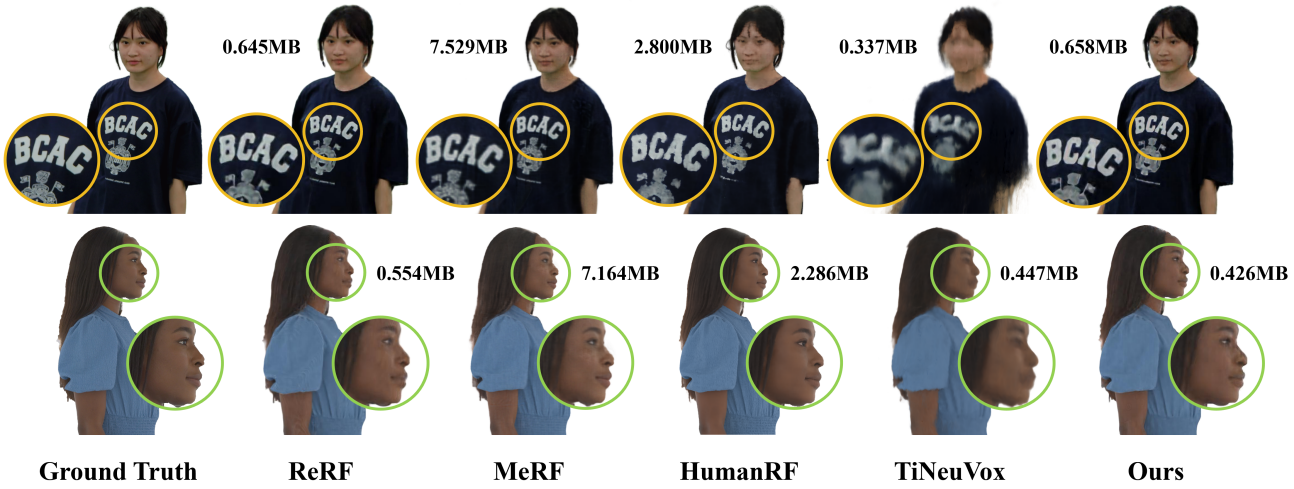

Figure 6. Qualitative comparison against dynamic scene reconstruction methods and per frame static reconstruction methods.

best second-best

Dataset	Method	PSNR↑	SSIM↑	LPIPS ↓	Size(MB)↓
ReRF	ReRF [65]	31.84	0.974	0.042	0.645
	MeRF [50]	31.12	0.975	0.030	7.529
	HumanRF [18]	28.82	0.900	0.069	2.800
	TiNeuVox[11]	22.70	0.923	0.083	0.337
	Ours	32.01	0.976	0.023	0.658
Actors-HQ	ReRF [65]	28.33	0.836	0.296	0.554
	MeRF [50]	27.22	0.807	0.271	7.164
	HumanRF [18]	28.98	0.888	0.151	2.286
	TiNeuVox [11]	22.98	0.752	0.430	0.447
	Ours	28.46	0.838	0.278	0.426

Table 1. Qualitative comparison against dynamic scene reconstruction methods and per frame static reconstruction methods.

it faces the challenge of intensifying blurring effects as the frame count rises. MeRF [50] suffers from the large storage which is not suitable for dynamic scenes when streaming and decoding for playing. HumanRF [18] though capable of representing dynamic scenes, also faces difficulties in streaming and rendering on mobiles due to its computation and storage load.

We also conduct a quantitative comparison using metrics such as the peak signal-to-noise ratio (PSNR), structural similarity index (SSIM), and the Learned Perceptual Image Patch Similarity (LPIPS) as metrics shown in Tab. 1. For a fair comparison in our experiments, in the ReRF dataset, we test the first 200 frames on the scene Kpop. We use 6 and 39 as test views and the others as training views. As for the HumanRF dataset, we apply the test methods as referenced in HumanRF on the first 250 frames on Actor3, sequence 1. We utilize the values reported for HumanRF and TiNeuVox methods directly from the HumanRF study. In the ReRF dataset, which includes large motion scenarios, we match the rendering quality of the standard ReRF, while also maintaining a compact data storage size. Meanwhile, in the HumanRF dataset with relatively small motion, we achieve a rendering quality that is second only

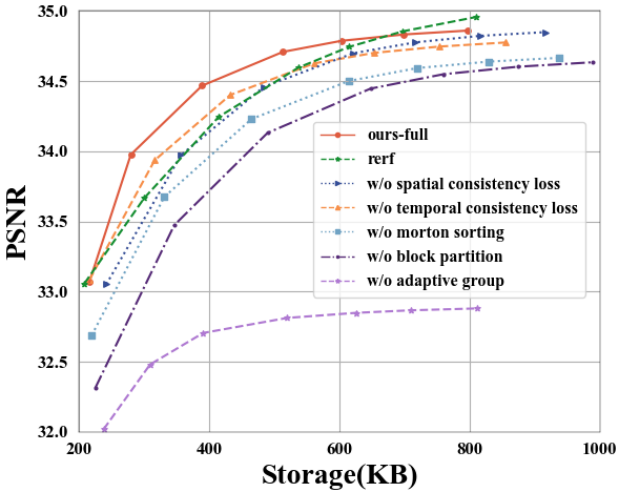

Figure 7. **Rate distortion curve.** The rate distortion curve illustrates the efficiency of various components within our system. We use different quantization factors to obtain the average rendering quality at different capacities. Our full model stands out as the most compact, allowing for flexible bitrate adjustments to meet diverse storage needs.

to HumanRF, accomplished with the most efficient storage utilization. Moreover, as indicated in Tab. 2, our method significantly surpasses ReRF in rendering speed when rendering the Kpop scene at an i7-12700F CPU and NVIDIA RTX3090 GPU at the resolution of 1920×1080, even on a tablet (iPad Pro with an M2 chip). It's worth noting that, our method is the only method that can provide both mobile and dynamic rendering tasks. The existing dynamic reconstruction methods like TiNeuVox [11], HumanRF [18] and ReRF [65] all cannot effectively utilize hardware encoding and decoding techniques, preventing them from being displayed on mobile devices, especially for the dynamic scene of a long sequence.

Figure 8. Qualitative evaluation of different variations in our method at 600KB.

5.2. Evaluation

Ablation study. We analyze the impact of spatial consistency, temporal consistency, Morton sorting, block storage, and adaptive grouping on compression and rendering. In the case of models that do not employ 3D and 2D Morton sorting, spatial point data is sorted sequentially in row-major order while employing 2D block-wise storage. For models not utilizing block storage, a 3D spatial Morton sorting is applied, followed by storage in a row-major format within the 2D feature space. For models without adaptive grouping, the mapping table for each frame is calculated only based on the occupancy grid of the current frame. As illustrated in Fig. 7, our full model exhibits the best performance in terms of rendering quality and storage efficiency. By utilizing our modules, the quantization of residual frequency coefficients becomes more precise, retaining more information at the same bitrate. This results in less performance loss when compressed to the same size compared to the variations without them. While ReRF [65] exhibits slightly better quality in the storage range of 700-800KB, our method consistently demonstrates higher PSNR across most capacity ranges and supports mobile rendering. Fig. 8 shows that under a 600KB storage limit, our complete model yields more realistic results with less compression blur.

Cross device runtime analysis. We evaluate the runtime breakdown analysis of VideoRF as detailed in Tab. 2. Our experimental setup includes a Desktop with an i7-12700F CPU and NVIDIA RTX3090 GPU, a Laptop with an i5-1135G7 CPU and Integrated GPU, a Tablet (iPad Pro) with an M2 chip, and a phone (iPhone 14 Pro) with an A16 Bionic chip. For Desktops and Laptops, we employ a Python code base along with ModernGL. Meanwhile, for Tablet and Smartphone, the VideoRF player is developed using Swift and Metal. It's worth noting that the video decoding part primarily relies on CPU performance, while the rendering part mainly depends on GPU performance. These two parts operate asynchronously and simultaneously. The other part mainly covers operations such as data conversion between the GPU and CPU. Our results demonstrate that VideoRF allows users to enjoy free-view videos at high frame rates on multiple devices, providing an experience that rivals the smoothness of watching 2D videos on platforms such as YouTube.

Method	Device	FPS	Decoding	Rendering	Others
Ours	Desktop	116	5.548 ms	3.602 ms	3.072 ms
Ours	Laptop	25	10.54 ms	29.31 ms	11.69 ms
Ours	Tablet	40	3.822 ms	23.86 ms	1.150 ms
Ours	Phone	23	13.38 ms	40.82 ms	3.003 ms
ReRF	Desktop	4	45.74 ms	194.7 ms	—

Table 2. Runtime analysis across different devices when processing the same HD image with a resolution of 1920×1080.

6. Conclusion

We have presented VideoRF, a novel approach enabling real-time streaming and rendering of dynamic radiance fields on mobile devices. Our VideoRF innovatively processes feature volumes as 2D feature streams and adopts deferred rendering to effectively leverage classical video codecs and rasterization pipelines. Our video codec friendly training scheme is implemented on 2D feature space to enhance spatial-temporal consistency for compactness. Additionally, our tailored player supports seamless streaming and rendering of dynamic radiance fields across a range of devices, from desktops to smartphones. Our experiments demonstrate its capability for compact and effective dynamic scene modeling. With the unique ability of real-time rendering of dynamic radiance fields on mobile devices, we believe that our approach marks a significant step forward in neural scene modeling and immersive VR/AR applications.

7. Acknowledgements

This work was supported by National Key R&D Program of China (2022YFF0902301), Shanghai Local college capacity building program (22010502800). We also acknowledge support from Shanghai Frontiers Science Center of Human-centered Artificial Intelligence (ShangHAI). This work was supported by the programs of NSFC (61976138 and 61977047) and STCSM (2015F0203-000-06). We thank Siyan Zhuo for her assistance in the figures.

References

[1] MobileNeRF: Exploiting the Polygon Rasterization Pipeline for Efficient Neural Field Rendering on Mobile Architectures, author=Zhiqin Chen and Thomas Funkhouser and Peter Hedman and Andrea Tagliasacchi. In *The Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 3

[2] Aljaž Božič, Denis Gladkov, Luke Doukakis, and Christoph Lassner. Neural assets: Volumetric object capture and rendering for interactive environments, 2022. 3

[3] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. *CVPR*, 2023. 2

[4] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16123–16133, 2022. 3

[5] Anpei Chen, Minye Wu, Yingliang Zhang, Nianyi Li, Jie Lu, Shenghua Gao, and Jingyi Yu. Deep surface light fields. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 1(1):1–17, 2018. 2

[6] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, 2022. 3

[7] Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. High-quality streamable free-viewpoint video. *ACM Transactions on Graphics (TOG)*, 34(4):69, 2015. 2

[8] Chenxi Lola Deng and Enzo Tartaglione. Compressing Explicit Voxel Grid Representations: Fast NeRFs Become Also Small. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1236–1245, 2023. 3

[9] Mingsong Dou, Philip Davidson, Sean Ryan Fanello, Sameh Khamis, Adarsh Kowdle, Christoph Rhemann, Vladimir Tankovich, and Shahram Izadi. Motion2fusion: Real-time volumetric performance capture. *ACM Trans. Graph.*, 36(6): 246:1–246:16, 2017. 2

[10] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B. Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 1, 2

[11] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, 2022. 2, 6, 7

[12] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022. 3

[13] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *CVPR*, 2023. 2

[14] Guy Gafni, Justus Thies, Michael Zollhöfer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8649–8658, 2021. 2

[15] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 2

[16] Xiang Guo, Guanying Chen, Yuchao Dai, Xiaoqing Ye, Jiadai Sun, Xiao Tan, and Errui Ding. Neural deformable voxel grid for fast optimization of dynamic view synthesis. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2022. 2

[17] Peter Hedman, Pratul P. Srinivasan, Ben Mildenhall, Jonathan T. Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5855–5864, 2021. 3

[18] Mustafa Işık, Martin Rünz, Markos Georgopoulos, Taras Khakhulin, Jonathan Starck, Lourdes Agapito, and Matthias Nießner. HumanRF: High-fidelity neural radiance fields for humans in motion. *ACM Transactions on Graphics (TOG)*, 42(4):1–12, 2023. 2, 6, 7

[19] Yuheng Jiang, Suyi Jiang, Guoxing Sun, Zhuo Su, Kaiwen Guo, Minye Wu, Jingyi Yu, and Lan Xu. Neuralhofusion: Neural volumetric rendering under human-object interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6155–6165, 2022. 2

[20] Yuheng Jiang, Kaixin Yao, Zhuo Su, Zhehao Shen, Haimin Luo, and Lan Xu. Instant-nvr: Instant neural volumetric rendering for human-object interactions from monocular rgbd stream. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 595–605, 2023.

[21] Hanbyul Joo, Tomas Simon, and Yaser Sheikh. Total capture: A 3d deformation model for tracking faces, hands, and bodies. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8320–8329, 2018. 2

[22] Moritz Kappel, Vladislav Golyanik, Susana Castillo, Christian Theobalt, and Marcus Magnor. Fast non-rigid radiance fields from monocularized data, 2023. 2

[23] Hansung Kim, Jean-Yves Guillemaut, Takeshi Takai, Muhammad Sarim, and Adrian Hilton. Outdoor dynamic 3-d scene reconstruction. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(11):1611–1622, 2012. 2

[24] Tobias Kirschstein, Shenhan Qian, Simon Giebenhain, Tim Walter, and Matthias Nießner. Nersemble: Multi-view radiance field reconstruction of human heads. *ACM Trans. Graph.*, 42(4), 2023. 2

[25] Youngjoong Kwon, Lingjie Liu, Henry Fuchs, Marc Habermann, and Christian Theobalt. Deliffas: Deformable light fields for fast avatar synthesis, 2023. 2

[26] Lingzhi Li, Zhen Shen, Zhongshu Wang, Li Shen, and Ping Tan. Streaming radiance fields for 3d video synthe-

sis. *Advances in Neural Information Processing Systems*, 35: 13485–13498, 2022. 2, 3

[27] Lingzhi Li, Zhongshu Wang, Zhen Shen, Li Shen, and Ping Tan. Compact real-time radiance fields with neural codebook, 2023. 3

[28] Ruilong Li, Julian Tanke, Minh Vo, Michael Zollhofer, Jurgen Gall, Angjoo Kanazawa, and Christoph Lassner. Tava: Template-free animatable volumetric actors. In *European Conference on Computer Vision (ECCV)*, 2022. 2

[29] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5521–5531, 2022. 2

[30] Zhengqi Li, Tali Dekel, Forrester Cole, Richard Tucker, Noah Snavely, Ce Liu, and William T. Freeman. Learning the depths of moving people by watching frozen people. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4516–4525, 2019. 2

[31] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. 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)*, pages 6498–6508, 2021. 2

[32] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering, 2023.

[33] Haotong Lin, Sida Peng, Zhen Xu, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Efficient neural radiance fields for interactive free-viewpoint video. In *SIGGRAPH Asia Conference Proceedings*, 2022. 2

[34] Jia-Wei Liu, Yan-Pei Cao, Weijia Mao, Wenqiao Zhang, David Junhao Zhang, Jussi Keppo, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Devrf: Fast deformable voxel radiance fields for dynamic scenes. In *Advances in Neural Information Processing Systems*, 2022. 1, 2

[35] Haimin Luo, Teng Xu, Yuheng Jiang, Chenglin Zhou, Qiwei Qiu, Yingliang Zhang, Wei Yang, Lan Xu, and Jingyi Yu. Artemis: Articulated neural pets with appearance and motion synthesis. *ACM Trans. Graph.*, 41(4), 2022. 2

[36] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM Transactions on Graphics (TOG)*, 39(4):71–1, 2020. 2

[37] Zhaoyang Lv, Kihwan Kim, Alejandro Troccoli, Deqing Sun, James M. Rehg, and Jan Kautz. Learning rigidity in dynamic scenes with a moving camera for 3d motion field estimation. In *Computer Vision – ECCV 2018*, pages 484–501, Cham, 2018. Springer International Publishing. 2

[38] Marko Mihajlovic, Sergey Prokudin, Marc Pollefeys, and Siyu Tang. ResFields: Residual neural fields for spatiotemporal signals. *arXiv preprint arXiv:2309.03160*, 2023. 2

[39] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer, 2020. 1, 3

[40] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022. 3

[41] Richard A. Newcombe, Dieter Fox, and Steven M. Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 343–352, 2015. 2

[42] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5865–5874, 2021. 1

[43] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields. *ACM Trans. Graph.*, 40(6), 2021. 2

[44] Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. Animatable neural radiance fields for modeling dynamic human bodies. In *ICCV*, 2021. 2

[45] 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. 2

[46] Sida Peng, Chen Geng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. Implicit neural representations with structured latent codes for human body modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(8):9895–9907, 2023. 2

[47] Sida Peng, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Representing volumetric videos as dynamic mlp maps. In *CVPR*, 2023. 2, 4

[48] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 1, 2

[49] René Ranftl, Vibhav Vineet, Qifeng Chen, and Vladlen Koltun. Dense monocular depth estimation in complex dynamic scenes. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4058–4066, 2016. 2

[50] Christian Reiser, Richard Szeliski, Dor Verbin, Pratul P. Srinivasan, Ben Mildenhall, Andreas Geiger, Jonathan T. Barron, and Peter Hedman. Merf: Memory-efficient radiance fields for real-time view synthesis in unbounded scenes. *SIGGRAPH*, 2023. 2, 3, 5, 6, 7

[51] Daniel Rho, Byeonghyeon Lee, Seungtae Nam, Joo Chan Lee, Jong Hwan Ko, and Eunbyung Park. Masked wavelet representation for compact neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20680–20690, 2023. 2, 3

[52] Sara Rojas, Jesus Zarzar, Juan C. Pérez, Artsiom Sanakoyeu, Ali Thabet, Albert Pumarola, and Bernard Ghanem. Re-

ReND: Real-Time Rendering of NeRFs across Devices. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3632–3641, 2023. 2, 3

[53] Ruizhi Shao, Zerong Zheng, Hanzhang Tu, Boning Liu, Hongwen Zhang, and Yebin Liu. Tensor4d: Efficient neural 4d decomposition for high-fidelity dynamic reconstruction and rendering, 2023. 2

[54] Liangchen Song, Anpei Chen, Zhong Li, Zhang Chen, Lele Chen, Junsong Yuan, Yi Xu, and Andreas Geiger. NeRFPlayer: A Streamable Dynamic Scene Representation with Decomposed Neural Radiance Fields, year=2023. *IEEE Transactions on Visualization and Computer Graphics*, 29 (5):2732–2742. 2

[55] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5449–5459, 2022. 3, 4, 1

[56] Towaki Takikawa, Alex Evans, Jonathan Tremblay, Thomas Müller, Morgan McGuire, Alec Jacobson, and Sanja Fidler. Variable bitrate neural fields. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–9, 2022. 2, 3

[57] Jiaxiang Tang, Xiaokang Chen, Jingbo Wang, and Gang Zeng. Compressible-composable NeRF via Rank-residual Decomposition. In *Advances in Neural Information Processing Systems*, 2022. 3

[58] Jiaxiang Tang, Hang Zhou, Xiaokang Chen, Tianshu Hu, Errui Ding, Jingdong Wang, and Gang Zeng. Delicate Textured Mesh Recovery from NeRF via Adaptive Surface Refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 17739–17749, 2023. 3

[59] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Nonrigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12959–12970, 2021. 2

[60] Edith Tretschk, Vladislav Golyanik, Michael Zollhöfer, Aljaz Bozic, Christoph Lassner, and Christian Theobalt. SceNeRFlow: Time-Consistent Reconstruction of General Dynamic . In *International Conference on 3D Vision (3DV)*, 2024. 2

[61] Michael Waechter, Nils Moehrle, and Michael Goesele. Let there be color! large-scale texturing of 3d reconstructions. In *European conference on computer vision*, pages 836–850. Springer, 2014. 2

[62] Ziyu Wan, Christian Richardt, Aljaž Božič, Chao Li, Vijay Rengarajan, Seonghyeon Nam, Xiaoyu Xiang, Tuotuo Li, Bo Zhu, Rakesh Ranjan, and Jing Liao. Learning neural duplex radiance fields for real-time view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8307–8316, 2023. 3

[63] Liao Wang, Ziyu Wang, Pei Lin, Yuheng Jiang, Xin Suo, Minye Wu, Lan Xu, and Jingyi Yu. ibutter: Neural interactive bullet time generator for human free-viewpoint rendering. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 4641–4650, 2021. 2

[64] Liao Wang, Jiakai Zhang, Xinhang Liu, Fuqiang Zhao, Yanshun Zhang, Yingliang Zhang, Minye Wu, Jingyi Yu, and Lan Xu. Fourier plenoctrees for dynamic radiance field rendering in real-time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13524–13534, 2022. 2, 3

[65] Liao Wang, Qiang Hu, Qihan He, Ziyu Wang, Jingyi Yu, Tinne Tuytelaars, Lan Xu, and Minye Wu. Neural residual radiance fields for streamably free-viewpoint videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 76–87, 2023. 2, 3, 4, 6, 7, 8, 1

[66] Yiming Wang, Qin Han, Marc Habermann, Kostas Daniilidis, Christian Theobalt, and Lingjie Liu. Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 2

[67] Wikipedia. Z-order curve, 2024. 4

[68] Daniel N Wood, Daniel I Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David H Salesin, and Werner Stuetzle. Surface light fields for 3d photography. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 287–296, 2000. 2

[69] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9421–9431, 2021. 2

[70] Lan Xu, Zhuo Su, Lei Han, Tao Yu, Yebin Liu, and Lu Fang. Unstructuredfusion: Realtime 4d geometry and texture reconstruction using commercial rgbd cameras. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(10):2508–2522, 2020. 2

[71] Lan Xu, Wei Cheng, Kaiwen Guo, Lei Han, Yebin Liu, and Lu Fang. Flyfusion: Realtime dynamic scene reconstruction using a flying depth camera. *IEEE Transactions on Visualization and Computer Graphics*, 27(1):68–82, 2021. 2

[72] Yuelang Xu, Lizhen Wang, Xiaochen Zhao, Hongwen Zhang, and Yebin Liu. Avatarmav: Fast 3d head avatar reconstruction using motion-aware neural voxels. In *ACM SIGGRAPH 2023 Conference Proceedings*, 2023. 2

[73] Yuelang Xu, Hongwen Zhang, Lizhen Wang, Xiaochen Zhao, Huang Han, Qi Guojun, and Yebin Liu. Latentavatar: Learning latent expression code for expressive neural head avatar. In *ACM SIGGRAPH 2023 Conference Proceedings*, 2023. 2

[74] Zhen Xu, Sida Peng, Haotong Lin, Guangzhao He, Jiaming Sun, Yujun Shen, Hujun Bao, and Xiaowei Zhou. 4k4d: Real-time 4d view synthesis at 4k resolution. 2023. 2

[75] Lior Yariv, Peter Hedman, Christian Reiser, Dor Verbin, Pratul P. Srinivasan, Richard Szeliski, Jonathan T. Barron, and Ben Mildenhall. Bakedsdf: Meshing neural sdfs for real-time view synthesis. In *ACM SIGGRAPH 2023 Conference Proceedings*, New York, NY, USA, 2023. Association for Computing Machinery. 2, 3

[76] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF*

International Conference on Computer Vision, pages 5752–5761, 2021. 3

[77] Heng Yu, Joel Julin, Zoltán Á. Milacski, Koichiro Niinuma, and László A. Jeni. Dylin: Making light field networks dynamic. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12397–12406, 2023. 2

[78] Anlan Zhang, Chendong Wang, Bo Han, and Feng qian. Yuzu:neural-enhanced volumetric video streaming. 2022. 3

[79] Jiakai Zhang, Xinhang Liu, Xinyi Ye, Fuqiang Zhao, Yanshun Zhang, Minye Wu, Yingliang Zhang, Lan Xu, and Jingyi Yu. Editable free-viewpoint video using a layered neural representation. *ACM Transactions on Graphics (TOG)*, 40(4):1–18, 2021. 2

[80] Fuqiang Zhao, Yuheng Jiang, Kaixin Yao, Jiakai Zhang, Liao Wang, Haizhao Dai, Yuhui Zhong, Yingliang Zhang, Minye Wu, Lan Xu, and Jingyi Yu. Human performance modeling and rendering via neural animated mesh. *ACM Trans. Graph.*, 41(6), 2022. 2

[81] Xiaochen Zhao, Lizhen Wang, Jingxiang Sun, Hongwen Zhang, Jinli Suo, and Yebin Liu. Havatar: High-fidelity head avatar via facial model conditioned neural radiance field. *ACM Trans. Graph.*, 2023. 2

[82] Zerong Zheng, Xiaochen Zhao, Hongwen Zhang, Boning Liu, and Yebin Liu. Avatarrex: Real-time expressive full-body avatars. *ACM Transactions on Graphics (TOG)*, 42(4), 2023. 2

VideoRF: Rendering Dynamic Radiance Fields as 2D Feature Video Streams

Supplementary Material

8. Training Details for VideoRF

8.1. Coarse Stage Pre-training and Baking

Given a long-duration multi-view sequence, we initially adopt the approach from DVGO [55] to generate an explicit density volume grid $\mathbf{V}_\sigma$ and color feature grids $\mathbf{V}_c$ representation for each frame t. Following ReRF [65], we employ a global MLP Φ_c during this coarse stage training. This MLP comprises a single hidden layer with 129 channels, and we set the color feature dimension at $h = 12$. Throughout the training, we incrementally upscale the volume grid, from $(125 \times 125 \times 125) \rightarrow (150 \times 150 \times 150) \rightarrow (200 \times 200 \times 200) \rightarrow (250 \times 250 \times 250)$, after reaching the training step 2000, 4000 and 6000, respectively. For loss calculation, we utilize both the photometric MSE loss and the total variation loss on $\mathbf{V}_\sigma$, expressed as:

$$\mathcal{L}_{\text{rgb}_{\text{coarse}}} = \sum_{\mathbf{r}\in\mathcal{R}} \|\mathbf{c}(\mathbf{r}) - \hat{\mathbf{c}}(\mathbf{r})\|^2, \tag{9}$$

$$\mathcal{L}_{\text{TV}_{\text{coarse}}} = \frac{1}{|\mathbf{V}_\sigma|} \sum_{\mathbf{v}\in\mathbf{V}_\sigma} \sqrt{\Delta_x^2\mathbf{v} + \Delta_y^2\mathbf{v} + \Delta_z^2\mathbf{v}}, \tag{10}$$

$$\mathcal{L}_{\text{coarse}} = \mathcal{L}_{\text{rgb}_{\text{coarse}}} + \lambda_{\text{TV}}\mathcal{L}_{\text{TV}_{\text{coarse}}}, \tag{11}$$

where $\lambda_{\text{TV}} = 0.000016$. Here, $\mathcal{R}$ represents the set of training pixel rays, with $\mathbf{c}(\mathbf{r})$ and $\hat{\mathbf{c}}(\mathbf{r})$ denoting the actual and predicted colors of a ray $\mathbf{r}$, respectively. $\Delta^2_{x,y,z}\mathbf{v}$ signifies the squared difference in the voxel's density value. Notably, the total variation loss is activated only during the training iterations from 1000 to 12000. For optimization, we utilize the Adam optimizer for training 16000 iterations with a batch size of 10192 rays. The learning rates for $\mathbf{V}_\sigma$, $\mathbf{V}_c$ and global MLP are 0.1, 0.11 and 0.002, respectively.

Once we obtain the density grid $\mathbf{V}^t_\sigma$ for each frame t in the coarse training phase, we generate a per-frame occupancy grid $\mathbf{O}^t$ by retaining voxels with a density above the threshold $\gamma = 0.003$. During our adaptive grouping stage, we set the pixel limit to $\theta = 512 \times 512$.

8.2. Fine-grained Sequential Training

After creating the mapping tables, we proceed to fine-grained sequential training within each group. At this stage, we also introduce a global tiny MLP Φ_f designed for efficient rendering on mobile platforms. This minimal MLP Φ_f consists of only one hidden layer with 16 channels, and we maintain the color feature dimension h at 12. Similar to the coarse stage, we progressively upscale the volume grid during training, moving from $(125 \times 125 \times 125)$ to $(150 \times 150 \times 150)$, then to $(200 \times 200 \times 200)$, and finally to $(250 \times 250 \times 250)$, corresponding to the training steps at 2000, 4000, and 6000, respectively. For loss calculations, we employ both the photometric MSE loss and the total variation loss on the density volume $\mathbf{V}_\sigma$, as well as spatial consistency loss and temporal consistency loss on the feature image $\mathbf{I}$:

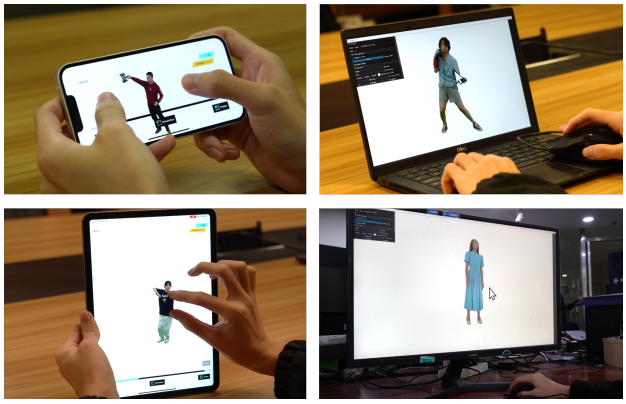

Figure 9. Our VideoRF facilitates dynamic radiance field rendering on ubiquitous devices, from desktops to mobile phones.

$$\mathcal{L}_{\text{rgb}_{\text{fine}}} = \sum_{\mathbf{r}\in\mathcal{R}} \|\mathbf{c}(\mathbf{r}) - \hat{\mathbf{c}}(\mathbf{r})\|^2, \tag{12}$$

$$\mathcal{L}_{\text{TV}_{\text{fine}}} = \frac{1}{|\mathbf{V}_\sigma|} \sum_{\mathbf{v}\in\mathbf{V}_\sigma} \sqrt{\Delta_x^2\mathbf{v} + \Delta_y^2\mathbf{v} + \Delta_z^2\mathbf{v}} \tag{13}$$

$$\mathcal{L}_{\text{spatial}} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p}\in\mathcal{V}} \left(\Delta_u(\mathbf{p}) + \Delta_v(\mathbf{p})\right), \tag{14}$$

$$\mathcal{L}_{\text{temporal}} = \|\mathbf{I}^t - \mathbf{I}^{t-1}\|_1, \tag{15}$$

$$\mathcal{L}_{\text{fine}} = \mathcal{L}_{\text{rgb}_{\text{fine}}} + \lambda_{\text{TV}}\mathcal{L}_{\text{TV}_{\text{fine}}} + \lambda_{\text{s}}\mathcal{L}_{\text{spatial}} + \lambda_{\text{t}}\mathcal{L}_{\text{temporal}}, \tag{16}$$

where $\lambda_{\text{TV}} = 0.000016$, $\lambda_{\text{s}} = 0.0001$ and $\lambda_{\text{t}} = 0.0001$. The total variation loss is specifically activated during training iterations 1000 to 12000. We continue to use the Adam optimizer for 16000 iterations with a batch size of 10192 rays. The learning rates for $\mathbf{V}_\sigma$, $\mathbf{V}_c$, and the global MLP are set to 0.1, 0.11, and 0.002, respectively.

9. Implementation Details for VideoRF Player

During the codec process of our player, the 2D density map and each feature map channel are considered as single channel images. These images are normalized and quantized into 8-bit depth. The H.264 encoder converts these images into the yuvj444p color space for hardware compatibility. During decoding, the data is converted back from the

Method	Metric	20	50	100	250	500	1000
		best	second-best				
HumanRF	↓ LPIPS	0.120	0.138	0.135	0.151	0.155	0.160
	↑ PSNR	31.02	30.26	30.25	28.98	29.50	29.19
	↑ SSIM	0.893	0.888	0.896	0.888	0.885	0.881
TiNeuVox	↓ LPIPS	0.352	0.298	0.406	0.430	0.436	0.452
	↑ PSNR	27.51	26.62	24.13	22.98	22.30	21.28
	↑ SSIM	0.782	0.791	0.760	0.752	0.751	0.747
NDVG	↓ LPIPS	0.240	0.281	0.354	0.435	0.453	0.481
	↑ PSNR	28.76	25.83	23.13	21.17	20.05	17.83
	↑ SSIM	0.841	0.812	0.763	0.731	0.724	0.692
HyperNeRF	↓ LPIPS	0.233	0.250	0.275	0.322	0.374	0.388
	↑ PSNR	25.75	26.53	25.96	24.85	23.29	23.04
	↑ SSIM	0.827	0.818	0.800	0.777	0.758	0.761
NeuralBody	↓ LPIPS	0.288	0.333	0.354	0.368	0.396	0.429
	↑ PSNR	27.51	25.88	27.18	25.30	24.81	25.68
	↑ SSIM	0.804	0.777	0.739	0.762	0.745	0.668
TAVA	↓ LPIPS	0.261	0.303	0.341	0.410	0.467	0.504
	↑ PSNR	28.47	26.93	25.83	24.28	23.13	22.21
	↑ SSIM	0.820	0.801	0.782	0.749	0.721	0.704
MeRF	↓ LPIPS	0.278	0.276	0.259	0.271	0.272	0.263
	↑ PSNR	28.24	28.19	27.24	27.22	27.31	27.68
	↑ SSIM	0.783	0.791	0.815	0.807	0.805	0.814
ReRF	↓ LPIPS	0.297	0.296	0.297	0.296	0.292	0.294
	↑ PSNR	28.69	28.51	28.55	28.33	28.12	27.73
	↑ SSIM	0.834	0.828	0.827	0.836	0.836	0.841
Ours	↓ LPIPS	0.276	0.285	0.283	0.278	0.274	0.275
	↑ PSNR	29.14	28.79	28.81	28.46	28.50	28.32
	↑ SSIM	0.840	0.835	0.830	0.838	0.840	0.844

Table 3. Quantitative comparison on long-duration sequence. We evaluate on the Actor 3, Sequence 1 of the Actors-HQ Dataset.

yuvj444p color space to single-channel images with 8-bit depth. Meanwhile, we adopt a multi-resolution occupancy grid to bypass empty 3D spaces at various levels. This approach significantly reduces unnecessary network inferences during the ray marching process. The largest occupancy grid is derived from max-pooling the full-resolution binary mask. Each subsequent grid is designed to be half the resolution of its predecessor. For instance, considering our full-resolution binary mask is of size 288×288×288, our multi-resolution occupancy grids follow suit with sizes of 144×144×144, 72×72×72, 36×36×36, 18×18×18, and 9×9×9.

10. Additional Experiments

As illustrated in Fig. 9, our method can enable dynamic radiance field rendering on a wide range of devices, including desktops (an i7-12700F CPU and NVIDIA RTX3090 GPU), laptops (an i5-1135G7CPU and Integrated GPU), tablets (iPad Pro, an M2 chip) and mobile phones (iPhone 14 Pro, an A16 Bionic chip).

Long-duration dynamic scenes. Following the approach in HumanRF [18], we assess performance on a long-duration sequence (Actor3, sequence1, 1000 frames) from the Actors-HQ dataset. We compare our method with ReRF [65] and MeRF [50] through per-frame static reconstruction in Fig. 10. Our method keeps a small storage to enable streaming while maintaining a high rendering quality. We adopt the testing methods outlined in HumanRF. The performance metrics for HumanRF [18], TiNeuVox [11], NDVG [16], HyperNeRF [43], NeuralBody [45], TAVA [28] are directly sourced from the HumanRF publication. As shown in Fig. 10 and Tab. 3, our approach demonstrates its capability to sustain high photorealism which is only second to HumanRF throughout long-duration sequences. Note that, our VideoRF is the only method to enable rendering dynamic scenes on mobile platforms.

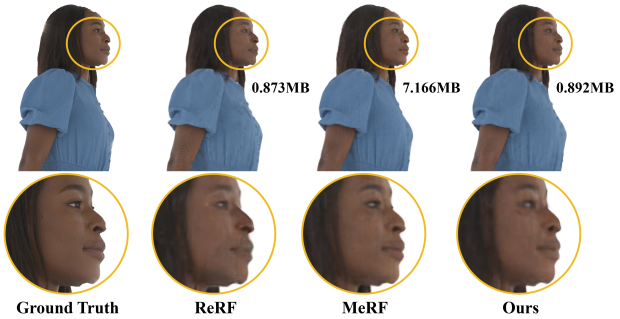

Figure 10. Qualitative comparison on the long-duration sequence against recent dynamic scene reconstruction methods and per frame static reconstruction methods.

Components	Size(KB)
Feature Images	661.62
3D to 2D Mapping Table	2.58
Occupancy Images	2.18
MLP Parameters	3.40
Total Size	669.78

Table 4. Storage of different components. The result is averaged over a sequence of Kpop scene from ReRF [65] dataset.

Storage of different components analysis. We present the storage requirements of each VideoRF component in Tab. 4. This encompasses the average file sizes for several key elements: the feature images for model detail, 3D-to-2D mapping table, occupancy images used to efficiently skip over empty spaces, and MLP parameters for the neural network. Note that our model's total average size is a mere 669.78KB. This compact representation facilitates rapid streaming across various devices.

11. Limitation and Future Work

As the first trial to enable a real-time dynamic radiance field approach capable of decoding and rendering on mobile devices, our approach presents certain limitations. On average, training each frame takes approximately 20 minutes using a single NVIDIA RTX3090 GPU. This includes about

5 minutes for coarse training, less than 1 second for the baking stage, and around 14 minutes for the fine-grained sequential training. Fortunately, our coarse training stage can be parallel for each frame, and fine-grained sequential training can be parallel across groups. Consequently, with a setup of 8 RTX3090 GPUs, the training time can be reduced to approximately 3 minutes per frame. It should be noted that once a sequence has been trained and encoded, users as content consumers can directly use it, and the training cost is transparent to the users. Therefore, on a practical application level, our method can provide a smooth experience on multiple platforms for users. Faster training speed is indeed important for content creators, and this remains an area for our future work. Additionally, our method currently lacks support for temporal interpolation, signifying another direction for future exploration.