RealLiFe: Real-Time Light Field Reconstruction via Hierarchical Sparse Gradient Descent

Yijie Deng*, Lei Han*, Tianpeng Lin, Lin Li, Jinzhi Zhang, and Lu Fang§

Abstract—With the rise of Extended Reality (XR) technology, there is a growing need for real-time light field reconstruction from sparse view inputs. Existing methods can be classified into offline techniques, which can generate high-quality novel views but at the cost of long inference/training time, and online methods, which either lack generalizability or produce unsatisfactory results. However, we have observed that the intrinsic sparse manifold of Multi-plane Images (MPI) enables a significant acceleration of light field reconstruction while maintaining rendering quality. Based on this insight, we introduce **RealLiFe**, a novel light field optimization method, which leverages the proposed Hierarchical Sparse Gradient Descent (HSGD) to produce high-quality light fields from sparse input images in real time. Technically, the coarse MPI of a scene is first generated using a 3D CNN, and it is further optimized leveraging only the scene content aligned sparse MPI gradients in a few iterations. Extensive experiments demonstrate that our method achieves comparable visual quality while being **100x** faster on average than state-of-the-art offline methods and delivers better performance (about **2 dB** higher in PSNR) compared to other online approaches.

Index Terms—Light field, Multi-plane Image, Hierarchical Sparse Gradient Descent.

1 Introduction

With the rapid development of XR / naked eye 3D display devices, there is an increasing demand for live light field video rendering techniques that can record and stream light field video in real time, allowing viewers to interactively change the viewpoint and focus of the video. Even with the latest breakthroughs in light field reconstruction [1], [2] and neural radiance fields [3], [4], it remains a challenging task.

To enable live light field reconstruction, the supporting algorithms and representations should be capable of producing high-quality view synthesis results in real time and generalize well to unseen scenes. However, current approaches face difficulties in achieving a balance between visual performance and real-time effectiveness.

- Methods relying on implicit scene representation, such as NeRF [3] and its derivatives [10], [11], [12], [13] can produce view synthesis results with high visual quality, but require long-time per-scene optimization and dense view inputs. While some generalization techniques [5], [6], [14], [15], [16], [17] can generate novel views that are generalizable with the help of multi-view stereo (MVS) geometry rules, it is still not fast enough for them to achieve real-time performance due to high memory usage and complex computation of intermediate cost volumes.
- Methods that rely on explicit scene representation, such as Plenoxels [18], PlenOctrees [19] and DVGO [20] enclose a 3D scene in a voxel grid and optimize the radiance fields within it. While such methods can reproduce intricate geometric and texture details at high grid resolutions, they can not generalize to new scenes without extensive per-scene optimization. Although some methods with learned features [1], [9], [21], [22] can generalize well to unseen scenes from sparse views, they still require a significant amount of time to infer a scene due to the heavy computational burden of processing large multi-plane image (MPI) and plane sweep volume (PSV).
- Methods relying on surrogate geometry are com-

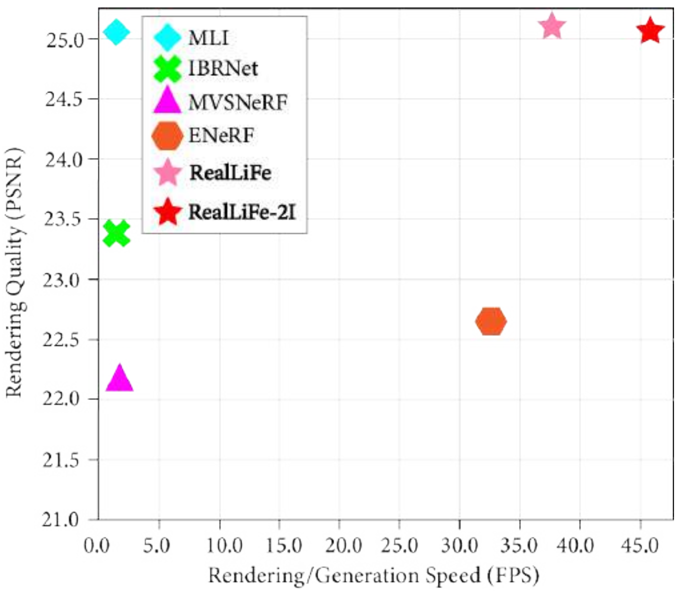

Fig. 1. Rendering quality and efficiency comparison with state-of-the-art novel view synthesis methods [5], [6], [7] and light field reconstruction methods [8] on Real Forward-Facing [9] of image size 512×384. **RealLiFe** is our default model with 3 iterations of gradient descent, and **RealLiFe-2I** is one with 2 iterations of gradient descent.

Yijie Deng, Jinzhi Zhang and Lu FANG are with Dept. of Electrical Engineering at Tsinghua University and also Beijing National Research Center for Information Science and Technology (BNRist), Beijing 100084. Yijie Deng and Jinzhi Zhang are also with Tsinghua Shenzhen International Graduate School. Lei Han, Tianpeng Lin and Lin Li are with Huawei Technology.
∗: Equal Contribution.
§: Correspondence Author (fanglu@tsinghua.edu.cn, http://luvision.net).

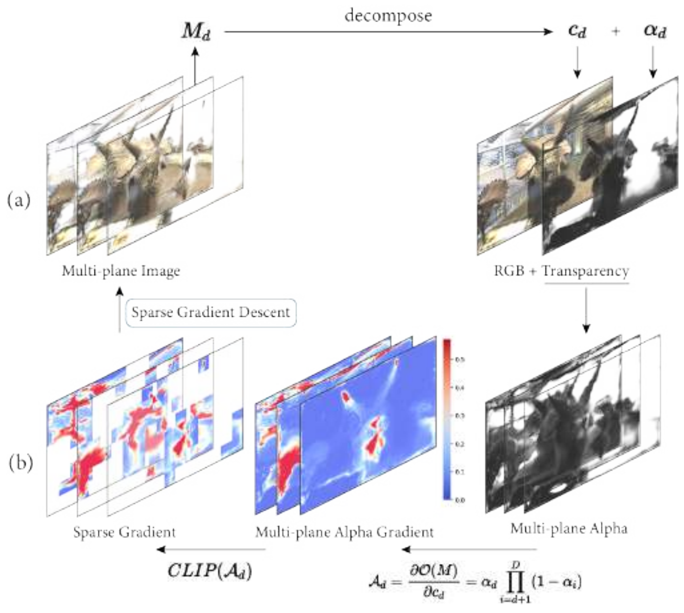

Fig. 2. The derivation of MPI sparse gradients and the optimization process. (a) An MPI plane M_d at depth d can be decomposed into an RGB plane c_d and an alpha (transparency) plane α_d. (b) The multi-plane alpha gradients $\mathcal{A}_d$ are computed from multi-plane alpha planes by calculating the gradient of M corresponding to c_d, and $\mathcal{A}_d$ is then sparsified by discarding small gradients, which is used to refine the MPI using our proposed sparse gradient descent strategy.

putationally efficient by sampling only around the estimated geometry scaffolds, such as depth maps [23] and surface mesh. However, these methods may fail if the precomputed geometry scaffolds are incorrectly estimated for semi-transparent surfaces or thin plates; therefore, they are not very robust for complex scenarios.

Based on the above observations, we conclude that live light field reconstruction remains a challenging task for the following reasons:

1) Recovering a 3D light field from 2D images is a difficult ill-posed inverse problem that necessitates either dense input views or geometric priors learned from data;

2) Implicit representations save memory but are computationally complex, resulting in a long inference time. Explicit representations offer faster rendering speed but require significant memory and time for constructing the light field;

3) High-quality light field reconstruction necessitates sophisticated optimization to handle intricate geometry details such as semi-transparent surfaces and thin structures.

To overcome the challenges, we propose **RealLiFe**, a novel method that employs Hierarchical Sparse Gradient Descent (HSGD) to produce high-quality light fields in real time from sparsely sampled views.

Unlike previous methods [1], [9], [21] that directly process and generate a full MPI, our method adopts a hierarchical approach, progressively refining it from low resolution to high resolution. This approach draws inspiration from the principles of Multi-View Stereo (MVS), where intricate geometric details can be iteratively enhanced with high-resolution features and well-established low-resolution geometric structures [24], [25], [26], [27], [28], [29]. We extend this concept from geometry to appearance, claiming and proving that the visual details within a light field can also be iteratively improved, given access to high-resolution captures and well-structured low-resolution representations.

By employing this hierarchical optimization pipeline, the computational efficiency can be greatly improved. The coarse stage not only reduces computation itself, it can also provide a sparse acceleration structure for the fine stage to further speed up the pipeline. This advantage aligns perfectly with the intrinsic nature of a light field, which is "dense in representation but sparse in content". A light field, represented as an MPI in this paper, is a set of densely arranged RGBα planes in shape $H \times W \times D \times 4$. Such a dense structure is a heavy computational burden that DeepView [1] has to infer patch by patch to get the complete light field. However, the real-world scenes are sparsely scattered, which presents a contradiction to the densely arranged multiplane images and also highlights the potential and validity of optimizing the light field in a hierarchical and sparse manner.

Before the design for efficiency, it is imperative to address the fundamental optimization process for deriving an MPI from sparse views. This task poses a significant challenge, as it involves solving an inverse problem. Conventionally, this is accomplished by iteratively minimizing the discrepancies between the predicted and observed measurements using analytic gradient descent, which can be a time-consuming process. However, DeepView [1] introduces a novel approach by incorporating the learned gradient descent strategy [30], [31]. This strategy enables the rapid generation of an MPI in only a few iterations by explicitly computing MPI gradients and relying on the neural network to perform adaptive gradient descent. We also apply this strategy for rapid gradient descent, but, in contrast to DeepView [1], which utilizes full-sized and full-channel MPI gradients, our approach only employs sparsely gradients that align with the scene contents. This choice arises from the observation that gradients within vacuous regions either insignificantly affect or negligibly contribute to the final rendering quality.

Based on our observations, we present the fundamental stages of the proposed Hierarchical Sparse Gradient Descent (HSGD), as depicted in Fig. 2. For example, in one stage, the multi-plane gradients $\mathcal{A}_d$ are initially obtained by computing the gradients of the MPI with respect to its RGB channel, as outlined in DeepView [1]. Subsequently, $\mathcal{A}_d$ undergoes sparsification by removing small-value gradients. Finally, the sparse gradient descent module updates the MPI based on the sparse gradients.

Experiments on public datasets demonstrate that compared to previous methods [1], [9], [21], our optimization strategy based on the sparsity of gradients significantly reduces computational complexity without loss of accuracy. Also, in comparison with methods [7] relying on surrogate geometry, our method achieves superior visual performance while maintaining high temporal efficiency.

Overall, the technical contributions are summarized as follows.

- We present an optimization strategy: Hierarchical Sparse Gradient Descent (HSGD), which efficiently

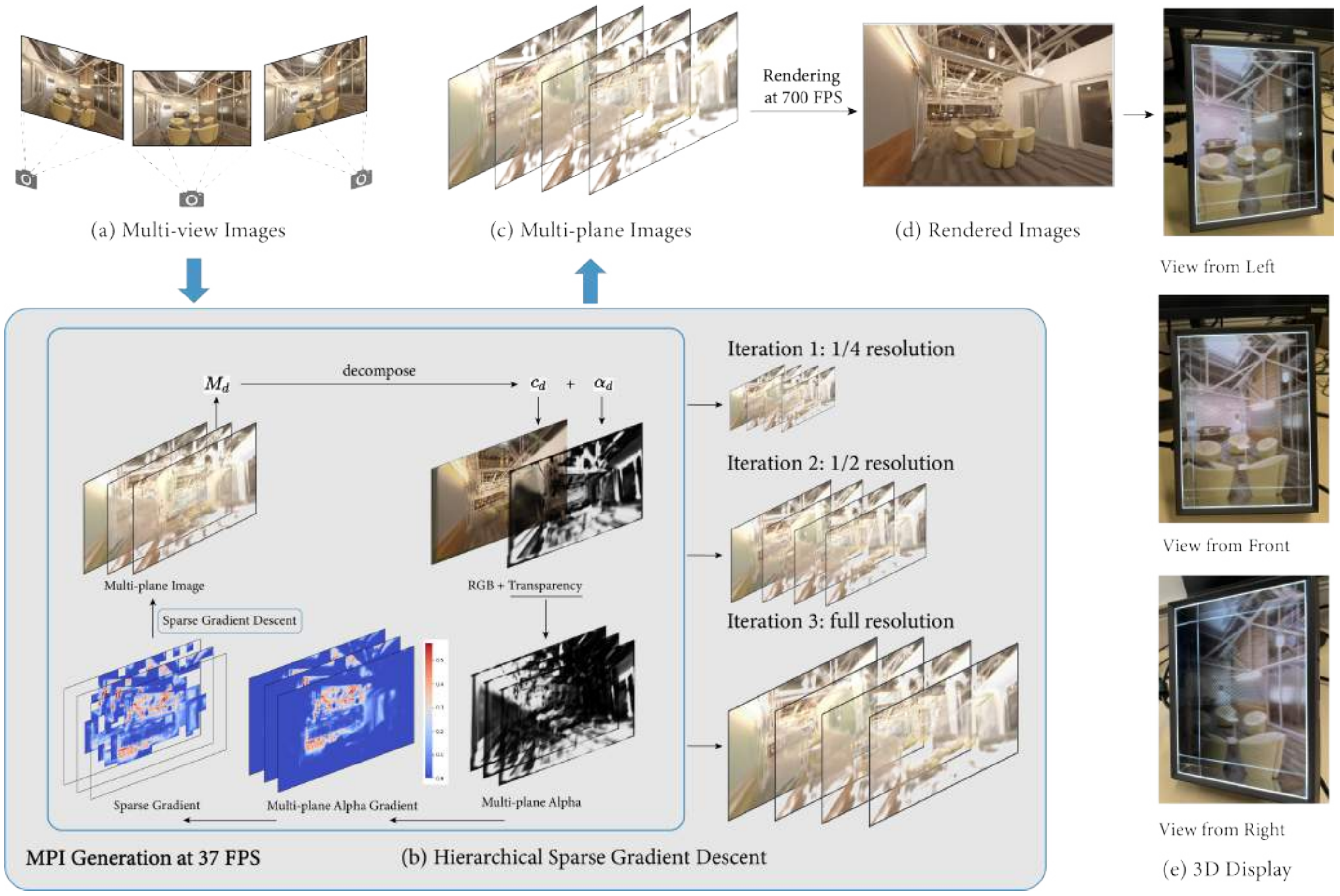

Fig. 3. Application of our method to support a real-time 3D display. (a) Sparse multi-view images that serve as the input to our model. (b) Our Hierarchical Sparse Gradient Descent (HSGD) is capable of generating multi-plane images (c) online at around 35 FPS. (d) Novel views can then be rendered offline from a Multi-plane Image at approximately 700 FPS. (e) Several novel views rendered from the MPI provide enough light field information to a 3D display, specifically a looking glass that supports naked-eye 3D effects from a wide range of views. (GD is short for gradient descent.)

recovers high-quality light fields while effectively reducing inference time and GPU memory usage.

- Extensive experiments on public datasets demonstrate that **RealLiFe** achieves a PSNR improvement of approximately **2db** compared to other online approaches, or comparable performance while being on average **100x** faster than offline approaches.
- As a 3D display application demonstrated in Fig. 3, our solution can generate MPIs at above 35 FPS, and novel views can be rendered from a built-up MPI at about 700 FPS for the resolution of 378×512. Novel views of different angles are then provided to a 3D display, which presents 3D effects in the naked eye through multilayer light field decomposition and directional backlighting [32].

2 Related Work

We focus on live light field reconstruction in this paper, which requires novel view synthesis that generalizes to arbitrary scenes with high rendering quality and can be processed in real time. However, existing approaches can hardly achieve all these requirements.

2.1 High-quality Novel View Synthesis

The rendering quality of novel view synthesis has seen significant improvement since the proposal of NeRF [3]. NeRF encodes a scene in a 3D radiance volume and optimizes the density and color of the volume using a simple MLP. Novel views are generated by integrating the volume's density and color along each viewing ray. Based on the success of NeRF, several methods such as Mip-NeRF [4], RawNeRF [34], Ref-NeRF [35], and Mip-NeRF360 [36] have further improved the rendering quality of the novel view synthesis. For example, Mip-NeRF reduces objectionable aliasing artifacts by casting cones and prefiltering the positional encoding function, while RawNeRF trains the NeRF model on raw camera data to achieve novel high dynamic range view synthesis. Ref-NeRF produces realistic shiny surfaces by explicitly modeling surface reflections, and Mip-NeRF360 solves the problem of unbounded scene novel view synthesis with a non-linear scene parameterization, online distillation, and a novel regularizer. Additionally, another MPI-based method [21] parameterizes each pixel as a linear combination of basis neural functions, successfully reproducing realistic view-dependent effects.

Although these methods have improved the rendering quality from different perspectives, they all require a long

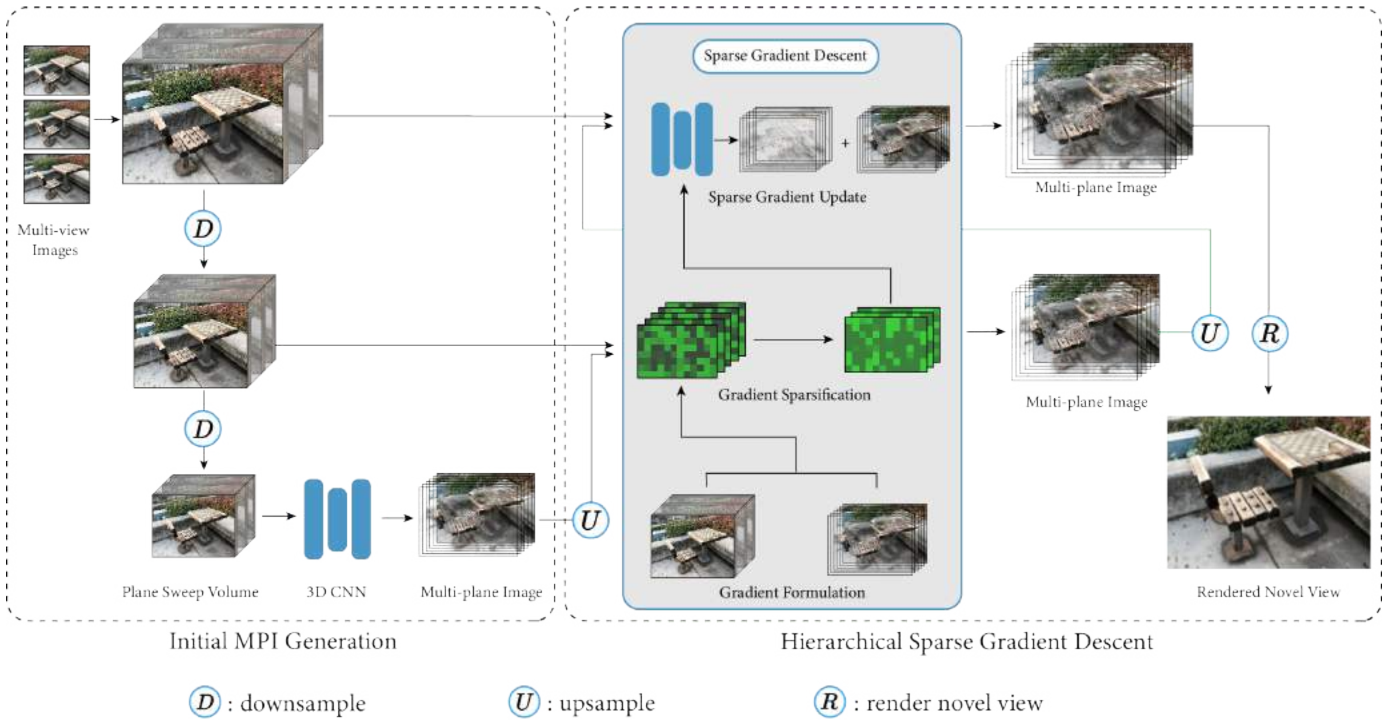

Fig. 4. The overview of **RealLiFe**. First, the PSV is constructed using multiview images by homographic warping, and the PSV is then downsampled hierarchically at multiple resolutions. The output MPI is then generated in several iterations. Initially, the lowest-resolution PSV is fed to a 3D CNN to extract a coarse-level MPI. The PSV and the upsampled MPI both go through the Sparse Gradient Descent module for a refined higher-resolution MPI. This loop is repeated until the resolution of the output MPI reaches the same resolution as that of the input images. Finally, a novel view can be easily rendered from the MPI using the repeated over operator [33].

per-scene optimization time because they train their models to encode the content of a particular scene, making them not generalizable to new scenes.

2.2 Generalizable Novel View Synthesis

To achieve generalizable novel view synthesis, several methods have been proposed that construct radiance fields from image features. PixelNeRF [16] encodes the input images into pixel-aligned feature grids and then renders points along a target ray by projecting them on the input image. For better performance, IBRNet [5] aggregates more information(image colors, features and viewing directions) and designs a ray transformer for decoding colors and densities. GRF [15] and NeuRay [14] achieve occlusion-aware novel view synthesis by incorporating the prior that ”The features of a point in 3D space are the same when being viewing from different angles”. StereoMag [21], LLFF [9], DeepView [1] and MLI [8] achieve generalization by modeling the scene as a set of fronto-parallel planes, in which multi-view consistency are subtly encoded. However, MLI [8] conducts an extra step to convert the planes to deformable layers and achieves high visual metrics. MVSNeRF [6] also leverages the concept of multi-view geometry and generalizes to new scenes with a neural cost volume. Instead of introducing image features and geometric priors for generalization, [37] applies meta-learning algorithms to learn the initial weight parameters for a NeRF model, enabling faster convergence.

Though generalizable, it still takes them seconds or minutes to generate a novel view. In addition, methods like PixelNeRF [16] and MVSNeRF [6] are not robust enough that they usually require extra finetuning to produce satisfying results.

2.3 Real-time Novel View Synthesis

To achieve real-time inference speed of a trained scene, structured representations and efficient computation skills are employed. KiloNeRF [12] replaces the original large MLP of NeRF with thousands of smaller faster-to-evaluate MLPs, enabling up to 40 FPS of rendering speed. Plenoctrees [19] render novel views by representing the scene in a structured octree-based 3D grid. FastNeRF [13] is capable of rendering novel views at 200 FPS by caching a deep radiance map and efficiently querying it. Except for NeRF-like representations, a built-up MPI from StereoMag/LLFF/DeepView [1], [9], [21] can achieve a rendering speed of 60 FPS. (The rendering speeds of above methods are all evaluated on 800x800 images of Synthetic-NeRF [3] dataset.) Despite the real-time inference speed of novel view synthesis, real-time building-up of a new scene is hard to be reached. The cutting-edge optimization-based method: InstantNGP [38] reduces training time of NeRF from hours to several seconds by using a hash grid and hardware acceleration techniques. DeepView [1] is now the fastest MPI-based method, however, it still requires about 50 seconds to generate an MPI. ENeRF [7] is the state-of-the-art generalizable novel view synthesis method and can render new views at 30 FPS on the ZJU-MoCap [39] dataset by sampling few points around a computed depth map; however, the rendering speed and image quality can still be improved.

2.4 Light Field Reconstruction

Instead of producing a single image at a time, there are methods that output the entire light field of the scene as multi-plane images [1], [9], [21] or layered mesh representations [2], [8], [40]. With a generated scene light field, novel views can be rendered easily with nearly no computation budget by simply ray querying. [2] encodes the scene in a set of multi-sphere images and compresses them into an atlas of scene geometry, allowing light field videos to be streamed over a gigabit connection. Though this method is real-time in rendering and streaming, it still requires a long time to reconstruct a new scene light field. [41], [42], [43] optimized for specific neural architectures for improved reconstruction quality of light fields. Their sophisticated designs for the light field capture systems and neural networks result in high metrics in rendering quality but also make them offline light field reconstruction methods. Starline [44] and VirtualCube [45] are another two methods that make live light field streaming possible. They both build up their systems using several RGBD cameras, and the additional depth channel enables fast scene geometry extraction, thus saving much time for computation. However, in this paper, we only compare RGB-based novel view synthesis methods to explore their potential for live light field reconstruction.

3 Method

The proposed method, **RealLiFe**, aims to facilitate real-time reconstruction of light fields using a set of sparse posed source view images. To accomplish this goal, we introduce Hierarchical Sparse Gradient Descent (HSGD), an efficient and high-quality optimization technique for MPIs.

Fig. 4 presents an overview of the proposed method, which comprises two primary processes: initial MPI generation and hierarchical sparse gradient descent. Sec. 3.2 introduces the initial MPI generation stage, which encompasses the construction of the plane sweep volume and the initial generation of the MPI. Sec. 3.3 introduces the hierarchical sparse gradient descent stage, as depicted in Fig. 5. This module comprises three major operations: gradient formulation, gradient sparsification, and sparse gradient update. Prior to discussing our method in detail, we provide a concise introduction of the MPI, and our fundamental optimization method, Learned Gradient Descent, in Sec. 3.1. Finally, we provide an introduction of our training approach, with a primary focus on how we construct the loss function, as outlined in Sec. 3.4.

3.1 Preliminaries

Multi-plane Image (MPI). An MPI [21] M is composed of D fronto-parallel $RGB\alpha$ planes in the view frustum of a camera. We denote the RGB channels of an MPI plane as c_d and the alpha channel as α_d. The target view $\tilde{I}_t$ is rendered by compositing the MPI in the back-to-front order utilizing the repeated over operator O as defined in [33]:

$$\tilde{I}_t = \mathcal{O}(\omega_t(M_r)), \tag{1}$$

where ω_t warps the MPI M_r from reference view to target view via homowarping. And the over operator $\mathcal{O}$ is defined as the compact form:

$$\mathcal{O}(M) = \sum_{d=1}^{D} c_d \alpha_d \prod_{i=d+1}^{D} (1-\alpha_i). \tag{2}$$

Learned Gradient Descent. Learned gradient descent is firstly proposed by [30], [31] to solve ill-posed inverse problems for CT reconstruction. Here, we follow DeepView [1] to apply learned gradient descent to the MPI generation problem. Inverse problems refer to problems where one seeks to reconstruct parameters characterizing the system under investigation from indirect observations. Mathematically, the problems can be formulated as reconstructing a signal $f_{true} \in X$ from data $g \in Y$:

$$g = \mathcal{T}(f_{true}) + \delta g, \tag{3}$$

where $\mathcal{T} : X \to Y$ is the forward operator that models how a signal generates data without noise, and $\delta g \in Y$ is a single sample of a Y-valued random variable that represents the noise component of data.

Typically, such problems are resolved by iteratively updating the reconstructed $\hat{f}$ with the analytical gradient descent method:

$$\hat{f}_{n+1} = \hat{f}_n + \lambda[\frac{\partial \mathcal{L}(\hat{f}_n)}{\partial \hat{f}_n} + \frac{\partial \phi(\hat{f}_n)}{\partial \hat{f}_n}], \tag{4}$$

where $\mathcal{L}$ is a loss function measuring the difference between the true signal f_{true} and the reconstructed $\hat{f}$, λ is the step size for each iteration, and ϕ is a prior on f_{true}.

However, it requires too many iterations to converge and is very likely to fall into local optima. Therefore, [30], [31] propose to use learned gradient descent instead of analytical gradient descent, where the update rule is defined by a deep neural network $\mathcal{N}_\omega$:

$$\hat{f}_{n+1} = \hat{f}_n + \mathcal{N}_\omega[\frac{\partial \mathcal{L}(\hat{f}_n)}{\partial \hat{f}_n} + \hat{f}_n]. \tag{5}$$

The learned gradient descent approach is already proved in various works to achieve better results with fewer iterations.

3.2 Initial MPI Generation

In this section, we delve into generation of the Initial MPI, a pivotal component in our optimization process, Hierarchical Sparse Gradient Descent. This process involves both the construction of the PSV and a straightforward network forwarding.

Hierarchical PSV Construction. Given N source view images $I_{i=1}^{N}$ of size $H \times W \times 3$ and their corresponding camera poses $[K_i, R_i|t_i]$, as well as for the reference view I_r, we construct the plane sweep volume P leveraging inverse homographic warping:

$$\mathcal{H}_i^{-1}(d) = K_i R_i (\mathcal{I} + \frac{(R_i^{-1}t_i - R_r^{-1}t_r)n^T R_r}{d}) R_r^{-1} K_r^{-1}, \tag{6}$$

where n denotes the plane normal and $\mathcal{I}$ is an identity matrix. The inverse homography matrix $\mathcal{H}_i^{-1}(d)$ warps the reference view meshgrid, defined by pixel positions (u, v) at depth d, to each source view I_i. And the inverse warped

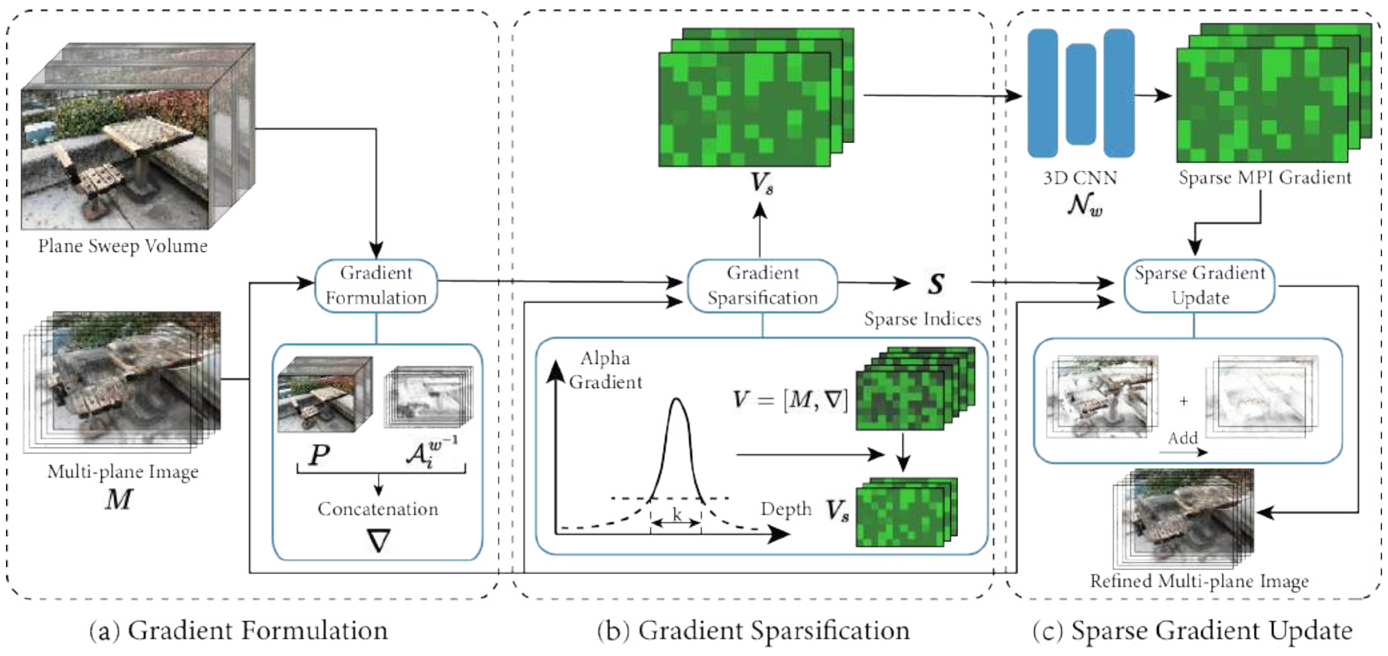

Fig. 5. The sparse gradient descent module. (a) The MPI gradients ∇ comprises the input plane sweep volume P and the warped alpha gradients $\mathcal{A}_i^w$ for each source view i. (b) The volume V is composed of the MPI M and the MPI gradients ∇. It is sparsified by selecting the top k voxels along the depth axis based on alpha gradients of the reference view. Simultaneously, the output sparse indices S store the positions of the selected k voxels along the depth axis. (c) The sparsified volume V_s is fed to a 3D CNN (learned gradient descent) for a refiend sparse MPI residual. Finally, the sparse gradient update module utilizes the sparse indices S to add the sparse MPI residual to the input multi-plane image M, resulting in a refined multi-plane image.

source view $I_i^{w^{-1}}$ is grid-sampled by the inverse warped meshgrid:

$$I_i^{w^{-1}}(u,v,d) = grid\_sample(I_i, \mathcal{H}_i^{-1}(d)[u,v,1]^T), \tag{7}$$

By concatenating all the inverse warped source view images $I_i^{w^{-1}}$ along the depth dimension, the plane sweep volume P of size $N \times D \times H \times W \times 3$ is constructed.

$$P = [I_0^{w^{-1}}, I_1^{w^{-1}}, ..., I_N^{w^{-1}}] \tag{8}$$

To enable optimization at different scales, we downsample it hierarchically at multiple resolutions. Specifically, we create the plane sweep volumes $\{P_l | 1 <= l <= L\}$ of size $N \times D \times H/2^l \times W/2^l \times 3$ for L iterations.

Network Forwarding. The coarsest-level PSV is reshaped into $3N \times D \times H/2^l \times W/2^l$ and fed to a 3D neural network to produce the initial MPI. The neural network is trained to aggregate information across views and generate a coarse MPI for subsequent optimization.

3.3 Hierarchical Sparse Gradient Descent

Hierarchical sparse gradient descent is a coarse-to-fine approach that progressively refines the MPI using sparse gradients. This design is supported by two key insights: Coarse-to-fine Appearance Refinement and Sparse Gradients Suffice for Light Field Reconstruction. The coarse-to-fine geometry refinement has previously demonstrated its efficacy in various MVS works [24], [25], [26], [27], [28], [29]. In this context, a low-resolution geometric scaffold can be rapidly derived and employed as a prior to guide the subsequent high-resolution detail refinement process. Similarly, the task of deriving an MPI involves allocating scene contents to their respective depth planes. So the hierarchical pipeline is well-suited for MPI generation, offering a structured foundation upon which the learned gradient descent algorithm can iteratively refine appearance details. And the choice of sparse gradients stems from the empirical observation in Fig. 6 that after only one iteration of network forwarding, above half of the voxels in an MPI become empty. Hence, any subsequent optimization targeting these empty regions does not lead to appreciable enhancements in rendering quality but instead incurs unnecessary computational overhead.

So we propose **Hierarchical Sparse Gradient Descent**, an adaptive and iterative approach that only leverages sparse gradients for efficient optimization. Specifically, this method consists of three steps: gradient formulation, gradient sparsification, and sparse gradient update. The detailed process is shown in Fig. 5.

Gradient Formulation. To obtain the sparse MPI gradient, we first formulate the components of input gradients. They are defined as the concatenation of the input PSV P and the inverse warped alpha gradient $\mathcal{A}_i^{w^{-1}}$ of each source view i. Theoretically, the full components of MPI gradients [1] should be in the format of Eq. 9 (The detailed derivation can be referred from [1].):

$$\nabla = [P, \widetilde{P}, \mathcal{A}_1^{w^{-1}}, ..., \mathcal{A}_N^{w^{-1}}, \mathcal{T}_1^{w^{-1}}, ..., \mathcal{T}_N^{w^{-1}}], \tag{9}$$

where $\widetilde{P}$ is the PSV constructed with rendered images at each source view from the generated MPI, and $\mathcal{T}_i^{w^{-1}}$ is the warped MPI gradient respective to MPI's α channel. But we simplify the gradients to be in the format of 10:

$$\widetilde{\nabla} = [P, \mathcal{A}_1^{w^{-1}}, ..., \mathcal{A}_N^{w^{-1}}], \tag{10}$$

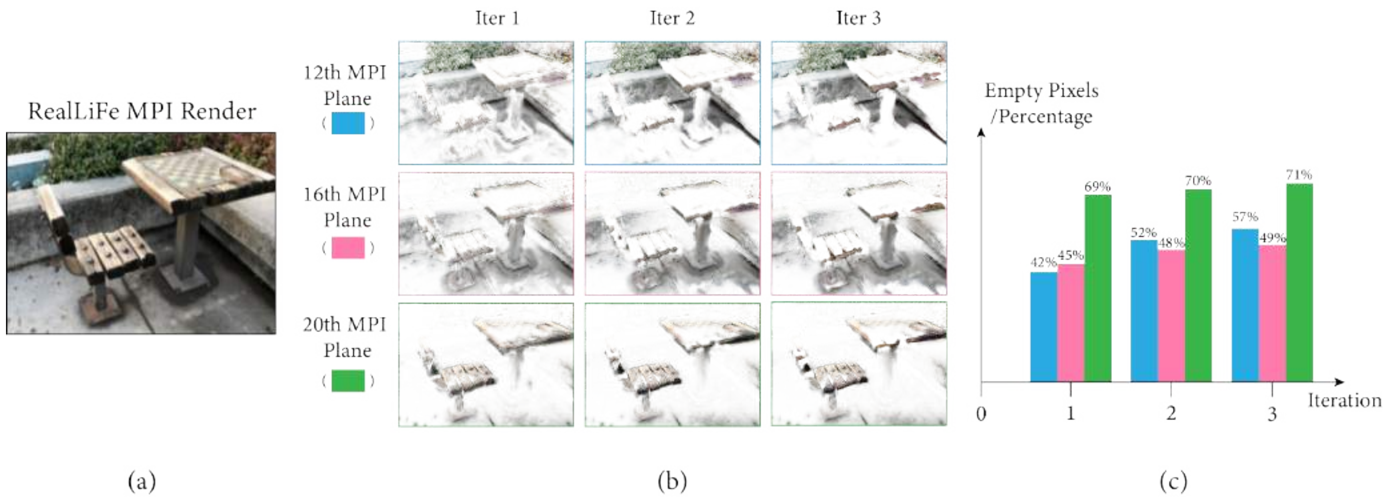

Fig. 6. Proportions of empty pixels of individual MPI planes throughout optimization iterations. (a) A rendered image of **RealLife** (without using HSGD) with 40 MPI planes. (b) A subset of MPI planes (i.e., the 12th, 16th, and 20th MPI planes) of three iterations. (c) Proportions of empty pixels in individual MPI planes. (pixels characterized by $\alpha < 0.1$ are considered empty, as their contribution to rendering quality is marginal.)

for several reasons:

1) The iterative optimization resembles a residual refinement network, where the PSV P is intuitively effective stable and precise residual information directly derived from the input data.

2) The MPI generation process involves the allocation of scene contents to their respective depth planes, with $\mathcal{A}$ being in perfect alignment with the scene contents. This alignment constitutes a crucial gradient component for the accurate construction of MPIs.

3) In the quest for an optimal trade-off between rendering quality and computational efficiency, it is proved in DeepView [1] that $\widetilde{P}$ and $\mathcal{T}$ can only marginally enhance the ultimate rendering quality, but their large spatial sizes can potentially hinder real-time efficiency.

To compute $\mathcal{A}^{w_i^{-1}}$ for a single source view i, the MPI of the reference view is initially warped to source view i to obtain M_i. Then the alpha gradient $\mathcal{A}_{i_d}$ corresponding to depth d is calculated by the following equation:

$$\mathcal{A}_{i_d} = \frac{\partial \mathcal{O}(M_i)}{\partial c_d} = \alpha_d \prod_{j=d+1}^{D} (1 - \alpha_j), \tag{11}$$

After this, the inverse warped alpha gradient $\mathcal{A}^{w_i^{-1}}$ is derived by applying inverse homographic warping Eq. 6 from the source view to the reference view.

After formulating the gradients, the full 3D volume V to be sparsified and optimized is the concatenation of $\widetilde{\nabla}$ and M:

$$V = [\widetilde{\nabla}, M], \tag{12}$$

Gradient Sparsification. To safely sparsify gradient components without adversely affecting the final rendering quality, it becomes imperative to establish criteria for the exclusion of voxels that impart minimal contributions. A commonly adopted strategy in various Multiview Stereo (MVS) studies [24], [25], [26] is to reduce the number of depth layers in the full cost volume from D to k, where k is usually an odd number, composed of the plane with the highest probability and its adjacent $2 \times (k-1)/2$ layers. However, this *MVS-sampling* strategy primarily relies on surface-centric geometric considerations, which might be too aggressive to capture some light field details like semi-transparent objects and thin plates. If we revisit Eq. 2, it can be rewritten as:

$$\mathcal{O}(M) = \sum_{d=1}^{D} c_d \mathcal{A}_d, \tag{13}$$

from which we can easily find that the influence of the color component c_d of each MPI plane on the ultimate color is determined by the alpha gradient derived from the reference view $\mathcal{A}_d$. The alpha gradient aligns more closely with the scene contents, as it takes into consideration not only solid geometry but also semi-transparent surfaces and thin plates, making it a more comprehensive and effective choice. In the ablation study of Sec. 5.5, we demonstrate that our sparsification strategy consistently outperforms the *MVS-sampling* approach in terms of overall performance. Consequently, the criterion for gradient sparsification is determined by $\mathcal{A}_d$.

The remaining challenge is to decide the optimal k for gradient sparsification. Fig. 7 provides insights into the determination of the optimal value for k. For instance, when $k = 7$, the rendered color almost recovers more than 80% of the original color, indicating a trade-off between computational efficiency and rendering quality. Generally, it appears from Fig. 7 that values of k in the range of 5 to 7 represent favorable choices, achieving a balance between time efficiency and rendering quality. Further ablation experiments on different values of k will be presented in Sec. 5.5.

After sparsification, only k voxels with the highest alpha gradient along the depth axis are kept, resulting in the compression of the input volume V, which is the concatenation of the original gradient volume $\widetilde{\nabla}$ and the input MPI M, from $C \times D \times H \times W$ to $C \times k \times H \times W$. The sparse indices S, which store the positions of the k selected voxels in the original depth axis, are also retained to enable the generated

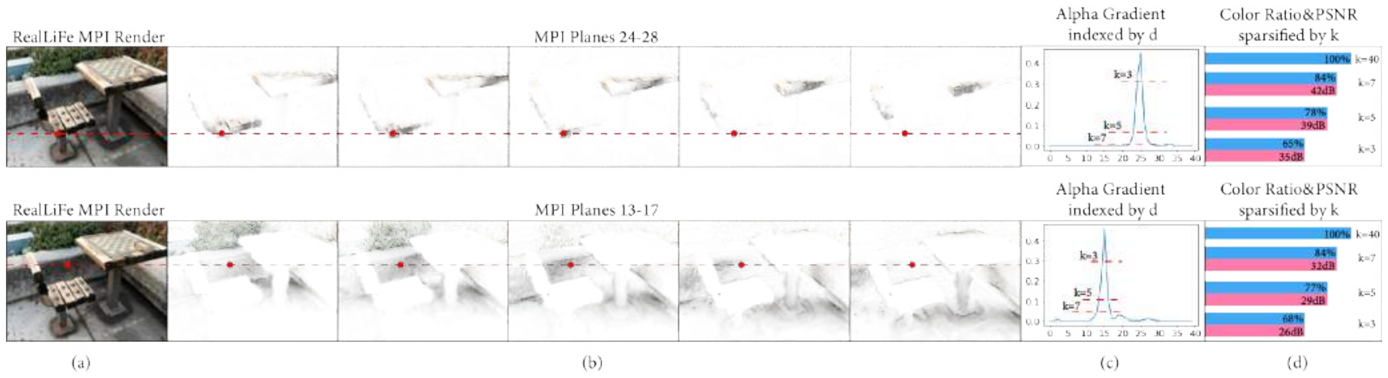

Fig. 7. The influence of k on the rendering quality. (a) A rendered image of **RealLiFe** (without using HSGD) with 40 MPI planes. (b) Top 5 MPI planes with the highest alpha gradients $\mathcal{A}$ for the red pixel. (The MPI is multiplied by $\mathcal{A}$ to better visualize its contribution to the final rendering result.) (c) The alpha gradients of the red pixel across 40 MPI planes, with the red dotted lines partitioning d based on whether its alpha gradient falls within the top k. (d) The color ratio ($RGB_{k=3,5,7}/RGB_{k=40}$) and PSNR of the rendered red pixel in comparison to when $k = 40$.

spasified MPI residual to be added to the input complete MPI for an update. It is evident that while the complete spatial structure of the light field is sacrificed after sparsification, the sparsified layers are retained in their original spatial order, preserving the potential surface structure of the light field for subsequent optimization.

Sparse Gradient Update. After the gradient components of each voxel has been decided, a network backbone needs to be decided for the learned gradient descent module. A CNN with large perceptive fields is proved effective in DeepView [1]. However, after gradient sparsification, an issue arises where adjacent voxels may not necessarily belong to the same depth plane. This raises concerns about the suitability of convolutional operations on these voxels. So a straightforward decision for the backbone would be an MLP, which treats each voxel independently. However, this choice compromises the benefits of an extensive receptive field, resulting in suboptimal performance. Moreover, even though the depths of a selected subset of voxels may exhibit considerable disparities, convolutional processing enables the model to adapt to these variations by learning shared features and patterns across different regions. We further prove the adaptability of the 3D CNN in enhancing rendering quality without introducing conspicuous artifacts through qualitative analysis in Sec. 5.3 and quantitative evaluation in Sec. 5.2.

As shown in Fig. 5 (c), the sparsified input volume V_s goes through a 3D CNN, functioning as the learned gradient descent network, to derive the sparse MPI residual for an MPI refinement iteration. The voxels in the sparse MPI residual are then restored to their original positions using the sparse indices S, and subsequently added to the input MPI to obtain a refined version:

$$M_{n+1} = M_n + \mathcal{R}(\mathcal{N}_w(V_s), S), \tag{14}$$

where $\mathcal{N}_w$ is the 3D CNN that functions as the learend gradient descent neural network, and $\mathcal{R}$ is the operator that restored the voxels of the sparse MPI residual to their original positions.

3.4 Training

During training, the network parameters are progressively optimized by minimizing the difference between the synthesized views and the ground truth novel views. And we choose the commonly used deep feature matching loss L_{VGG} [46] as the basic loss function. In detail, our overall rendering loss L_r is a weighted average loss of all iterations, and the rendering loss for each iteration is also a weighted average one for both the synthesized reference and source views:

$$L_r = \sum_{i=1}^{l} \lambda_i (L_{VGG}(I_{r_i}, \tilde{I}_{r_i}) + \mu \sum_{j=1}^{N} L_{VGG}(I_{j_i}, \tilde{I_{j_i}})), \tag{15}$$

where I_{r_i} and I_{j_i} are the ground truth reference and source images at iteration i, $\tilde{I}_{r_i}$ and $\tilde{I}_{j_i}$ are the rendered reference and source images from the generated MPI at iteration i. And l is the number of iterations, N is the number of input source images, λ_i is a hyper-parameter weighing the importance of each iteration, and μ is also a hyper-parameter balancing reference and source image supervision.

In addition, we propose a sparsity loss L_s that regularizes the alpha values of the MPI to be close to 0 or 1:

$$L_s = \frac{\sum_{h=1}^{H} \sum_{w=1}^{W} \sum_{d=1}^{D} log(1.5 - |0.5 - \alpha_{h,w,d}|)}{H \times W \times D}, \tag{16}$$

where $\alpha_{h,w,d}$ is the alpha channel of the MPI, and H, W, D are the height, width and number of depth planes of the volume. This loss aims to reduce the entropy of the MPI with respect to its α channel. It is ideal for gradient sparsification, as it allows us to select more informative top k samples that lie around the surfaces of a scene. The overall training loss is:

$$L = L_r + \lambda_s L_s, \tag{17}$$

where λ_s is a hyper-parameter balancing color supervision and sparsity regularization.

4 Implementation Details

The generalized model was trained using an RTX 3090 GPU with the Adam optimizer [47]. The initial learning rate was configured at $1e-3$, with a reduction by half if the training loss consistently decreased for a continuous span of 10 epochs. The model underwent training for 1000k iterations, taking approximately 13 hours to complete. The trainable components within the entire pipeline included the 3D

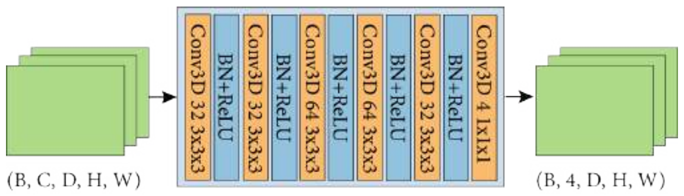

Fig. 8. The backbone 3D CNN of **RealLiFe**, which contains only three convolution layers.

CNNs responsible for converting the input plane sweep volume into a multi-plane image (as depicted in Fig. 4) and refining the sparse MPI gradient (as illustrated in Fig. 5). The structure of the 3D CNN, as shown in Fig. 8, comprised 5 basic convolutional blocks (a convolution layer followed by a batchnorm layer and a relu layer) and 1 1x1x1 3D convolution layer. To further improve the efficiency of our method, we optimize to run our pipeline with TensorRT[1].

To configure the loss function hyperparameters, a preference was given to increased color supervision from the reference view and the last iteration, while maintaining the impact of the sparsity loss. An example set of these parameters was as follows: $\lambda_1 = 0.2$, $\lambda_2 = 0.3$, $\lambda_3 = 0.5$, $\mu = 0.5$, $\lambda_s = 0.2$. Regarding experimental parameters, the model utilized 40 MPI planes, underwent 3 network iterations, and employed a sparsification factor k set to 5 as the default. These settings may vary in the context of ablation experiments.

5 Experiments

5.1 Baselines, datasets and metrics

Baselines. In order to evaluate the visual performance and real-time efficiency of our method, we compare it with four generalizable novel view synthesis methods: IBRNet [5], MVSNeRF [6], and ENeRF [7]. Additionally, we compare our method with five light field reconstruction methods, namely Stereo Magnification (StereoMag) [21], Soft3D [49], MLI [8], and DeepView [1]. Among the baseline methods, IBRNet, MVSNeRF and ENeRF conduct extra finetuning experiments to improve performance. However, finetuning takes so much time that it is incompatible with real-time light field reconstruction, thus we only compare the results without finetuning. LLFF [9] is excluded as a baseline method because the number of input views for the released model is set fixed to 5. Additionally, LLFF fuses multiple MPIs to render novel views, leveraging information of up to 10-20 input views, making it a method with dense view inputs. However, it is essential to acknowledge that LLFF [9] remains a robust approach for generating high-quality light fields offline.

Datasets. For training and evaluation, we select 4 datasets: Spaces [1], Real Forward-Facing [9], SWORD [40] and Shiny [48]. To compare results on the Real Forward-Facing evaluation set, comprising 8 scenes, the SWORD evaluation set, comprising 25 scenes (selected from the total evaluation scenes) and Shiny evaluation set, comprising 8 scenes, we trained our method using a combined training set in the same manner as IBRNet [5]. This combined training set consists of 90 scenes from the Spaces training set and 35 scenes from the Real Forward-Facing training set. To compare results on the Spaces evaluation set, which consists of 10 scenes with three settings of large, medium, and small baselines, we trained our method using the same training dataset as DeepView [1], comprising 90 scenes from the Spaces training set.

Metrics. We compared the visual performance of the proposed method using standard metrics, structural similarity (SSIM), peak signal-to-noise ratio (PSNR), and perceptual similarity (LPIPs). We evaluated the temporal efficiency of all methods based on *Generation Time*, the time required to generate a single representation (RGB images for IBRNet [5], MVSNeRF [6] and ENeRF [7], MPIs for Soft3D [49], StereoMag [21], DeepView [1] and our method, multi-layered meshes for MLI [8]). Furthermore, given that light field reconstruction methods yield light field representations suitable for offline rendering, we compared the offline rendering speed of light field reconstruction methods, also in FPS. (We use the OpenGL renderer provided by LLFF [9] to evaluate the offline rendering speed.) To assess the viability of each method for 3D display applications, we evaluated their *3D display efficiency* (measured in FPS), which we define as the reciprocal of the time required to provide sufficient light field information to support light field rendering on a 3D display. Specifically, a 3D display (e.g., a looking glass) requires n views (n is set to 18 in Tab. 1) at different viewing angles to reproduce a light field, providing a continuous, forward-facing viewing experience. Novel view synthesis methods necessitate n forward processes to generate n new views. In contrast, light field reconstruction methods require only a single forward process to generate n novel views because the built-up representations enable fast novel view rendering in the forward-facing viewing range.

5.2 Experiment Configurations

We compare our method with state-of-the-art generalizable novel view synthesis methods and light field reconstruction methods on public datasets including SWORD [40], Real Forward-Facing [9], Shiny [48] and Spaces [1].

SWORD, Real Forward-Facing and Shiny. To compare results on a sparse view setting, we set the number of input views to 3 for all methods. The evaluation image resolution for three datasets is 378×512, and the number of MPI planes for our method is set to 40. To compare with MLI [8], we choose their best-performance model SIMPLI-8L that uses 8 layered meshes for representations. We trained two versions of our models, RealLiFe and RealLiFe-2I. RealLiFe has 3 iterations of learned gradient descent, while RealLiFe-2I has 2 iterations, directly upsamping the 1/4-scale MPI to the original scale. In order to assess the interpolation and extrapolation capabilities, we evaluate all views from the three datasets. For each view of a given scene, we select the nearest available 3 views as input. The evaluation results for a single scene are calculated by averaging the results across all views. Likewise, the final evaluation results for a dataset are computed by averaging the results across all scenes.

Spaces. We follow the configuration of DeepView [1] to compare with their evaluation results. The number of input

1. https://developer.nvidia.com/tensorrt

TABLE 1
Quantitative comparison on SWORD [40] (25 scenes), Real Forward-Facing [9] (8 scenes) and Shiny [48] (8 scenes) evaluation dataset. The rank-1 method's metrics are highlighted in deep green, while those of the rank-2 method are highlighted in light green.

Category	Model	Generation Time, sec	Offline Rendering, fps	3D Display Efficiency, fps	SWORD			Real Forward-Facing			Shiny		
					PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Offline	IBRNet [5]	4.174	-	∼0.01	23.04	0.77	0.23	23.64	0.78	0.21	24.20	0.82	0.17
	MVSNeRF [6]	1.523	-	∼0.04	21.79	0.74	0.24	22.25	0.75	0.21	23.33	0.83	0.14
	SIMPLI-8L [8]	1.677	∼480.0	∼0.60	24.22	0.83	0.13	25.01	0.86	0.09	27.27	0.91	0.06
Online	ENeRF [7]	0.030	-	∼1.85	22.95	0.75	0.19	22.90	0.77	0.18	26.03	0.86	0.10
	RealLiFe	0.026	∼700.0	∼37.04	23.82	0.77	0.20	25.46	0.84	0.15	27.15	0.85	0.10
	RealLiFe-2I	0.022	∼700.0	∼45.45	23.80	0.77	0.20	25.33	0.84	0.16	27.03	0.85	0.10

TABLE 2
Quantitative comparison on Spaces evaluation dataset in terms of SSIM. The rank-1 method's metrics are highlighted in deep green, while those of the rank-2 method are highlighted in light green.

Configuration	Soft3D [49]	StereoMag$^+$ [21]	DeepView [1]	**RealLiFe**	**RealLiFe-2I**
Small baseline	0.9260	0.8884	0.9541	0.9396	0.9286
Medium baseline	0.9300	0.8874	0.9544	0.9317	0.9159
Large baseline	0.9312	0.8673	0.9485	0.9180	0.9052
Generation Time, sec	-	-	∼20.00	0.044	0.037

views is 4, the evaluation image resolution is 480×800, and the number of planes of an MPI is 40. We use the same input and evaluation views as DeepView [1] for three baseline settings.

Our models are trained for 100,000 iterations for both configurations, and all methods are evaluated on an RTX 3090 GPU.

5.3 Quantitative Results

Tab. 1 presents the quantitative results on SWORD [40], Real Forward-Facing [9] and Shiny [48]. We classify all methods into offline and online according to *Generation time*.

Compared with offline methods, our model RealLiFe and RealLiFe-2I stands out in terms of visual metrics, achieving top-1 or top-2 rankings on three datasets. And our default model RealLiFe strikes a balance between the rendering quality and efficiency. Our models, on average, achieve a $100\times$ faster generation speed compared to other offline methods. Compared with the online method. Our models are a little bit faster than ENeRF [7] in generation time; however, our method exhibits superior visual metrics on three datasets. When considering the metric of 3D Display Efficiency, our method stands out as the only one capable of supporting real-time light field video display. This is due to its high temporal efficiency in both light field reconstruction and offline rendering speed.

Tab. 2 lists the quantitative results on the Spaces evaluation set, where we can see that DeepView generates the highest-quality rendering results overall, followed by our method and Soft3D [49]. The decrease in rendering quality may be attributed to the fact that our method depends on network connections to propagate visibility information. As we increase the baseline, there is a corresponding increase in the number of network connections required to effectively propagate visibility, as noted in [1]. However, in terms of time efficiency, our method is about 400 times faster than DeepView. The *Generation Time* of Soft3D and StereoMag$^+$ are left as "-" because Soft3D is not open-sourced and StereoMag$^+$ is a modified version by the authors of DeepView, so that we can not evaluate their time efficiency accurately. To summarize, we have the following observations:

- Compared with offline novel view synthesis methods [5], [6] and offline light field reconstruction methods [1], [8], [21], [49], our approach achieves around $100\times$ speedup ratio with superior or comparable visual performance.
- Compared with the online novel view synthesis method [7], our method achieves better visual performance, while maintaining comparable time efficiency.
- Novel views can be rendered at significantly higher FPS than other novel view synthesis methods [5], [6], [7] from our generated MPIs. This property makes our method particularly suitable for displaying on 3D displays and other light mobile devices.

5.4 Qualitative Results

We show qualitative comparisons on Real Forward-facing [9] (the first two rows), Shiny [48] (the third row) and SWORD (the last row) in Fig. 9. For a comprehensive comparison with previous state-of-the-art methods, we evaluate the rendering quality in the following aspects.

Occluded Regions. In Fig. 9 (a), ghosting effects are produced at the edges of flowers by ENeRF [7] and MVSNeRF [6]. In contrast, our method generates clear boundaries, comparable to IBRNet [5] and SIMPLI-8L [8]. In Fig. 9 (b), ENeRF [7] produces an incomplete horn, and MVSNeRF [6] blurs the background around the horn. Our results are comparable to IBRNet [5], which show a complete horn and clean background, but not as good as SIMPLI-8L [8].

Intricate Texture Details. Fig. 9 (d) shows the faces of two statues with intricate texture details. ENeRF [7], MVSNeRF [51], and IBRNet [5] produce varying degrees of blurriness, while our method and SIMPLI-8L [8] restore the complicated texture details very well.

View-dependent Effects. Fig. 9 (c) and (e) show view-dependent specularity effects on the cd and the black pillar, respectively. For the cd scene, MVSNeRF [6] fails to produce reasonable specular patterns, and ENeRF [7] and SIMPLI-8L [8] generate blurry edges of the white reflection. However, our method and IBRNet [5] produce more accurate view-dependent effects. For the black pillar, ENeRF [7], SIMPLI-8L [8], and MVSNeRF [6] fail to generate the complete shape of the pillar, and IBRNet [5] does not restore the specularity as accurately as our method.

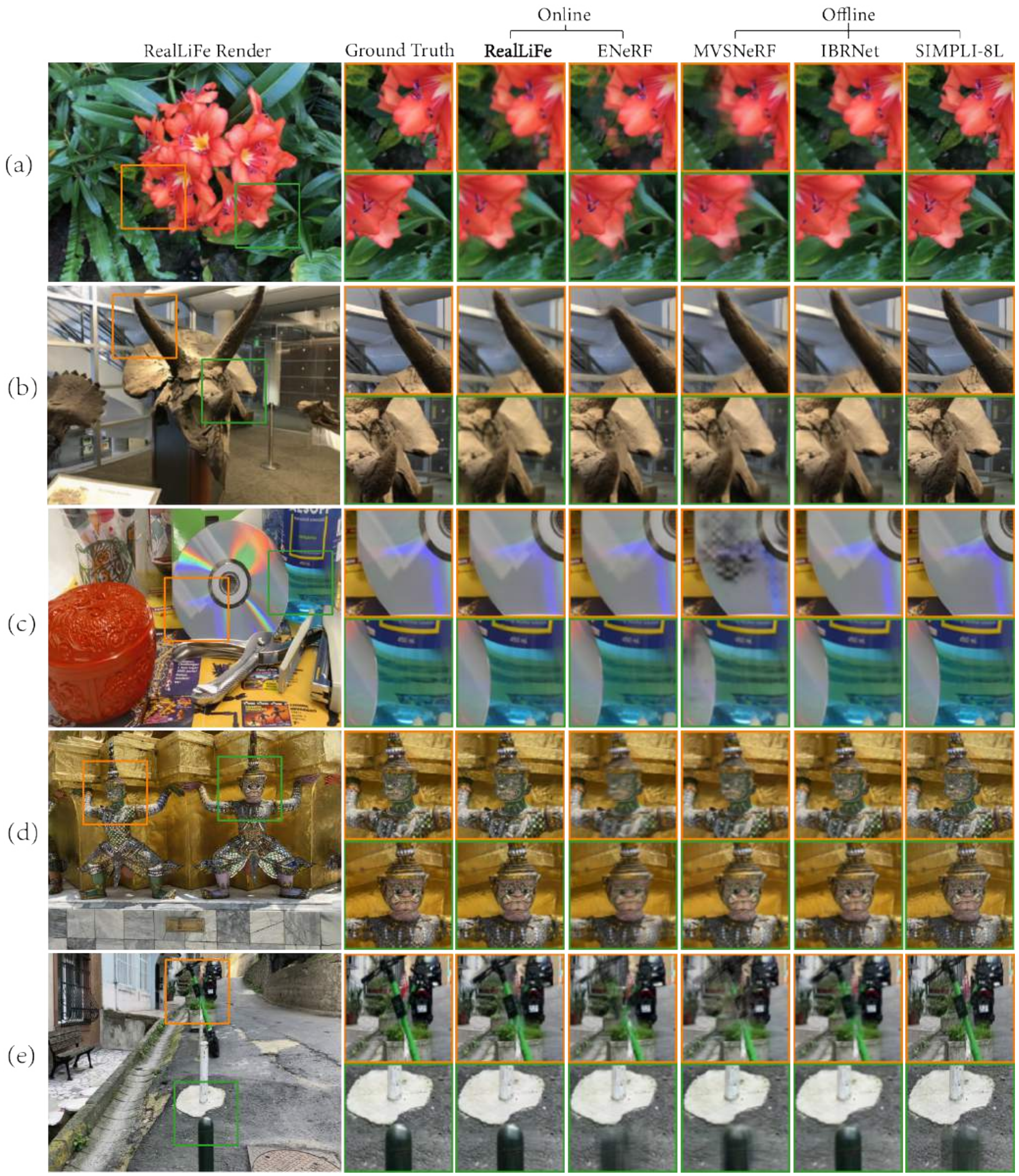

Fig. 9. Qualitative results on Real Forward-Facing [9] (a) and (b), SWORD [50] (c) and (d) and Shiny [48] (e) evaluation datasets.

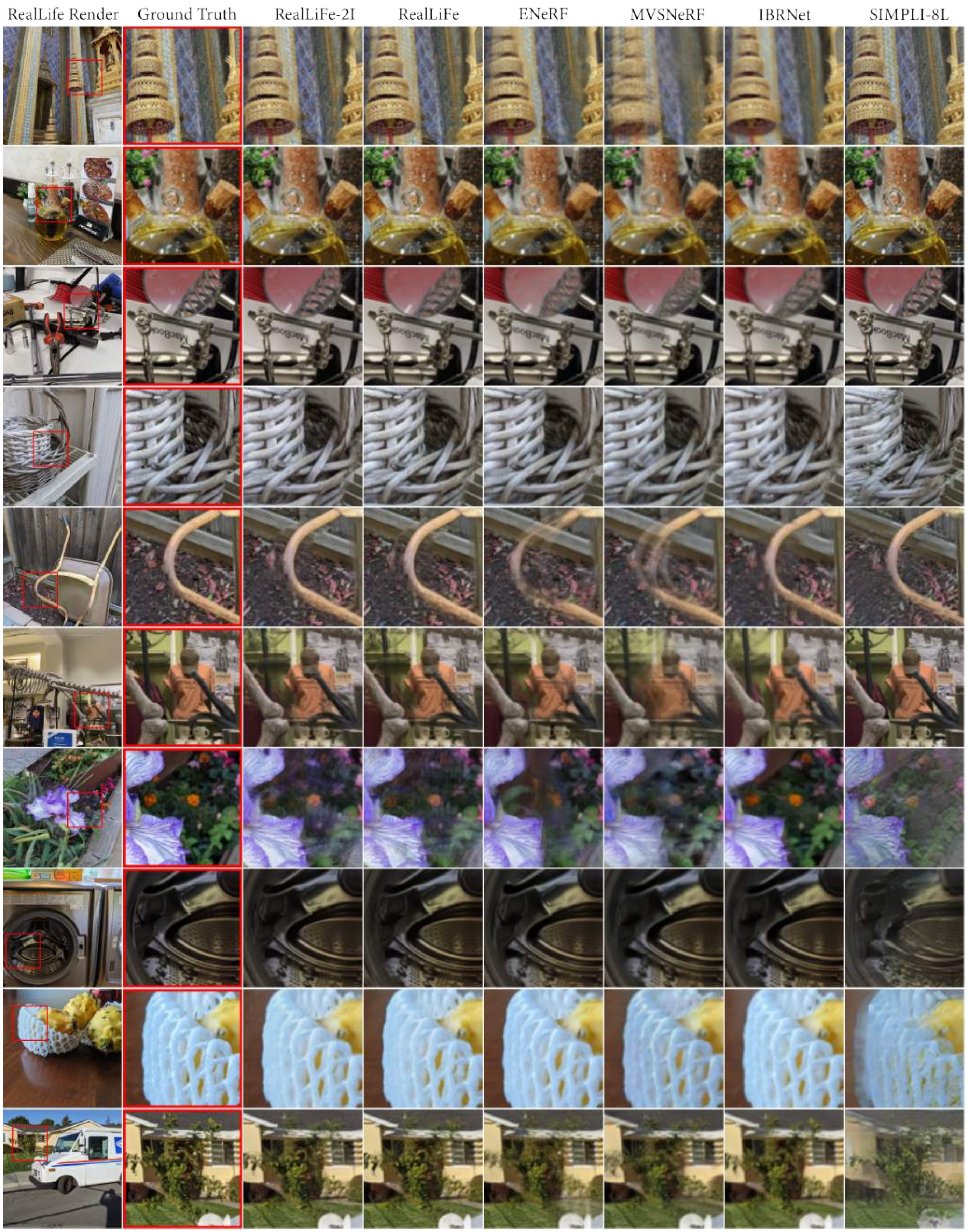

Fig. 10. Extra qualitative comparison on Shiny [8] and IBRNet collected [5].

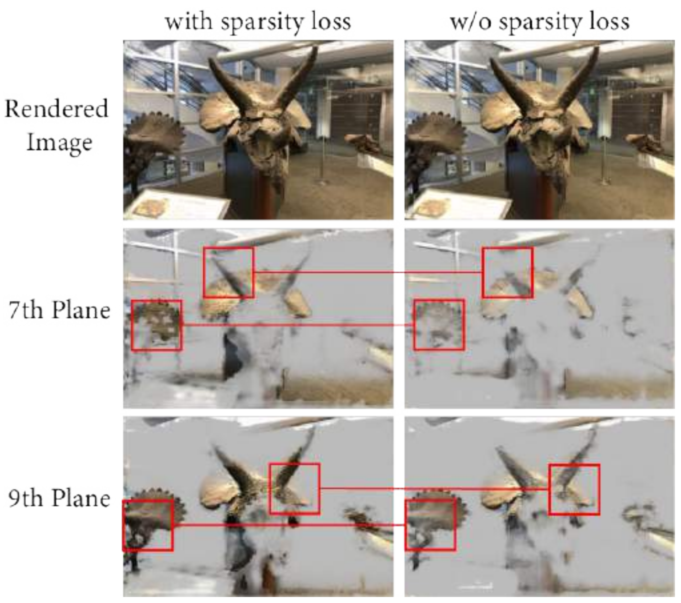

Fig. 11. Qualitative comparison on the ablation configuration of *with or without sparsity loss*. The left and right columns correspond to rendering results produced by the model trained with sparsity loss and without sparsity loss respectively. The PSNR for the left rendered image and the right rendered image are 28.51 and 27.42. The MPI planes trained with sparsity loss are clearer and less cloudy.

Thin Plates. In Fig. 9 (e), the green stick of the scooter is a very thin plate. ENeRF [7], MVSNeRF [6], and IBRNet [5] fail to restore the clear structure of the stick. Our method produces a relatively complete shape, comparable to SIMPLI-8L [8].

Among all the evaluation metrics, our approach achieves comparable or better visual quality with offline approaches, and performs superior to online approaches. More qualitative comparison results on Shiny [8] and IBRNet collected [5] are shown in Fig. 10. Our method produces clear results that accurately captures view-dependent effects, albeit with slightly less sharpness compared to SIMPLI-8L [8]. This trade-off is made in exchange for reduced time spent constructing the light field representations.

5.5 Ablation Study

In this section, we conduct a series of ablation experiments to assess the contribution of each component to the final rendering performance. The metrics of all experiments are evaluated on the Real Forward-facing [9] dataset, which contains typical challenging cases such as occlusion, complex scene geometry, and view-dependent effects. The rendered images for training and evaluation are downscaled to 378×512, and the number of input views is set to 3. The default number of planes in an MPI is 40. For detailed ablation settings and details, please refer to Tab. 3.

Sparsification factor k. It is evident that increasing the value of k results in enhanced rendering performance but at the expense of slower generation speed. Hence, it is crucial to identify an appropriate k that strikes a balance between rendering quality and efficiency. Notably, as k surpasses the value of 7, the demand for GPU memory during the training process exceeds the capacity of an RTX 3090 GPU. To circumvent this limitation, models with k exceeding 7 have to be trained and inferred in smaller patches, which may inevitably downgrades the rendering quality. So we conducted ablation on k with values 3, 5 and 7 in Tab. 3. The results confirm that an increasing number of k results in marginal increase in visual quality but obvious decrease in generation speed.

Number of iterations. Comparing the configurations of the *default model* with that of *L=2 (2 iterations)* in Tab. 3, it is evident that employing additional iterations of learned gradient descent yields only a marginal enhancement in rendering quality. However, this refinement results in a notable decrease of approximately 8 FPS in temporal efficiency. Therefore, to strike a balance between rendering quality and efficiency, employing a 2-iteration model suffices. This efficiency trade-off stems from the diminishing returns of increased iterations, where the incremental improvement in rendering quality does not adequately offset the cost in terms of temporal performance, hence motivating the preference for the 2-iteration model.

Number of planes. When comparing the configurations of the *default model*, *D=16 (fewer planes)*, and *D=64 (fewer planes)* in Tab. 3, it becomes evident that employing too few planes results in a substantial deterioration in rendering quality, albeit compensating for expedited generation speed. Conversely, an increase in the number of planes yields only marginal enhancements in rendering quality while significantly diminishing temporal efficiency. This observation suggests the presence of a potential saturation point for the number of MPI planes. Beyond this point, the incremental addition of planes may not yield linear improvements in performance.

Sparsity Loss. The *default model* demonstrates superior rendering quality compared to the *w/o sparsity loss* in Tab. 3. And Fig. 11 visually compares the differences in the rendered images and individual MPI planes from experiments conducted with and without the sparsity loss. Compared to the model trained with sparsity loss, the model trained without it generates rendered images that display more artifacts, particularly around sharp edges. A comparison of individual MPI planes shows that those produced by the model with sparsity loss contain clearer surfaces. This evidence suggests that sparsity loss aids the network in minimizing scene content layout ambiguity, resulting in cleaner and sharper images.

Backbone network. Unlike the most straightforward way to process the sparse MPI gradients with an *MLP*, we employ a 3D CNN for better performance as discussed in Sec. 3.3. We can see from Tab. 3 that the configuration of *MLP* backbone network results in a reduction in rendering quality when compared to the *default model*. The diminished quality is attributed to the limited receptive field of an MLP. Given that the default 3D CNN structure aims to strike a balance between rendering quality and efficiency, further exploration of a more robust and efficient architecture is warranted to enhance performance.

Spasification criterion. In Section 3.3, we referenced an MVS [24], [25], [26], [27], [28], [29]-style gradient sparsification approach that involves selecting the layer with the highest alpha gradient along the depth axis as well as its adjacent $(k-1)/2$ layers to form a sparsified volume of k layers of

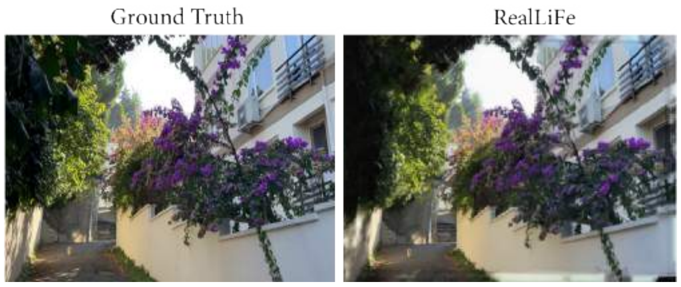

Fig. 12. The border artifacts of our approach. The borders of our rendered image contains obvious color difference.

gradient data. This strategy, however, only takes geometry into consideration and may not adequately capture certain light field intricacies, such as semi-transparent objects and thin structures. An overall comparison in Table 3 reveals that the *MVS-sampling* strategy exhibits lower rendering quality when compared with the *default model*.

TABLE 3
Quantitative ablation of the design choice on Real Forward-facing evaluation set.

Configuration	Generation FPS↑	PSNR ↑	SSIM↑	LPIPS↓
$k = 3$	43.29	24.95	0.83	0.17
$k = 7$	32.79	**25.51**	**0.84**	**0.15**
$D = 16$(fewer planes)	**54.64**	24.39	0.79	0.19
$D = 64$(more planes)	28.57	25.48	0.84	0.15
$L = 2$(2 iterartions)	45.45	25.33	0.84	0.16
w/o sparsity loss	31.73	25.20	0.81	0.16
MLP	48.54	24.86	0.82	0.17
MVS-sampling	37.04	24.34	0.79	0.19
default model	37.04	25.46	0.84	0.15

6 Conclusion

This paper presents a novel method for efficiently reconstructing light fields using hierarchical sparse gradient descent. Our proposed method achieves a resolution of 378×512 for light field representations (MPIs) at a frame rate of above 35 FPS, from which novel views can be rendered at around 700 FPS. The hierarchical sparse gradient descent module of our network focuses on scene-aligned sparse MPI gradients, resulting in significant improvements in temporal efficiency without compromising rendering quality. Our method has the potential to deliver high-quality and real-time light field videos for XR and Naked Eye 3D display devices.

Limitations. There are some limitations of our approach. Firstly, our current implementation can only be trained with a fixed number of views, and the order of input views may slightly affect the final rendering quality. Secondly, our model is unable to produce relatively good results at the borders of the image where not all source views are overlapped, shown in Fig. 12.

Future work. Although the proposed approach is able to generate light fields at above 35 FPS of resolution 378×512, its efficiency could still be improved with customized CUDA kernels, which is not emphasized in this paper. Furthermore, a robust and adaptive view-aggregating module is required to support an arbitrary number of input views. This enhancement will facilitate the use of our model for various camera configurations.

References

[1] J. Flynn, M. Broxton, P. E. Debevec, M. DuVall, G. Fyffe, R. S. Overbeck, N. Snavely, and R. Tucker, "Deepview: View synthesis with learned gradient descent," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2362–2371, 2019, doi: 10.1109/CVPR.2019.00247.

[2] M. Broxton, J. Flynn, R. S. Overbeck, D. Erickson, P. Hedman, M. DuVall, J. Dourgarian, J. Busch, M. Whalen, and P. E. Debevec, "Immersive light field video with a layered mesh representation," ACM Transactions on Graphics (TOG), vol. 39, pp. 86:1 – 86:15, 2020, doi: 10.1145/3386569.3392485.

[3] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in European Conference on Computer Vision, 2020, doi: 10.1145/3503250.

[4] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, "Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 5835–5844, 2021, doi: 10.1109/ICCV48922.2021.00580.

[5] Q. Wang, Z. Wang, K. Genova, P. P. Srinivasan, H. Zhou, J. T. Barron, R. Martin-Brualla, N. Snavely, and T. A. Funkhouser, "Ibrnet: Learning multi-view image-based rendering," 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4688–4697, 2021, doi: 10.1109/CVPR46437.2021.00466.

[6] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su, "Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 14 104–14 113, 2021, doi: 10.1109/ICCV48922.2021.01386.

[7] H. Lin, S. Peng, Z. Xu, Y. Yan, Q. Shuai, H. Bao, and X. Zhou, "Efficient neural radiance fields for interactive free-viewpoint video," SIGGRAPH Asia 2022 Conference Papers, 2021, doi: 10.1145/3550469.3555376.

[8] P. Solovev, T. Khakhulin, and D. Korzhenkov, "Self-improving multiplane-to-layer images for novel view synthesis," 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 4298–4307, 2022, doi: 10.1109/WACV56688.2023.00429.

[9] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar, "Local light field fusion: Practical view synthesis with prescriptive sampling guidelines," ACM Trans. Graph., vol. 38, no. 4, jul 2019, doi: 10.1145/3306346.3322980.

[10] D. B. Lindell, J. N. P. Martel, and G. Wetzstein, "Autoint: Automatic integration for fast neural volume rendering," 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 14 551–14 560, 2020, doi: 10.1109/CVPR46437.2021.01432.

[11] D. Rebain, W. Jiang, S. Yazdani, K. Li, K. M. Yi, and A. Tagliasacchi, "Derf: Decomposed radiance fields," 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 14 148–14 156, 2020, doi: 10.1109/CVPR46437.2021.01393.

[12] C. Reiser, S. Peng, Y. Liao, and A. Geiger, "Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 14 315–14 325, 2021, doi: 10.1109/ICCV48922.2021.01407.

[13] S. J. Garbin, M. Kowalski, M. Johnson, J. Shotton, and J. P. C. Valentin, "Fastnerf: High-fidelity neural rendering at 200fps," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 14 326–14 335, 2021, doi: 10.1109/ICCV48922.2021.01408.

[14] Y. Liu, S. Peng, L. Liu, Q. Wang, P. Wang, C. Theobalt, X. Zhou, and W. Wang, "Neural rays for occlusion-aware image-based rendering," 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7814–7823, 2022, doi: 10.1109/CVPR52688.2022.00767.

[15] A. Trevithick and B. Yang, "Grf: Learning a general radiance field for 3d representation and rendering," in 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 15 162–15 172, doi: 10.1109/WACV56688.2023.00429.

[16] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, "pixelnerf: Neural radiance fields from one or few images," 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4576–4585, 2021, doi: 10.1109/CVPR46437.2021.00455.

[17] J. Chibane, A. Bansal, V. Lazova, and G. Pons-Moll, "Stereo radiance fields (srf): Learning view synthesis for sparse views of novel scenes," 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7907–7916, 2021, doi: 10.1109/CVPR46437.2021.00782.

[18] A. Yu, S. Fridovich-Keil, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance fields without neural networks," 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5491–5500, 2021, doi: 10.1109/CVPR52688.2022.00542.

[19] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa, "Plenoctrees for real-time rendering of neural radiance fields," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 5732–5741, 2021, doi: 10.1109/ICCV48922.2021.00570.

[20] C. Sun, M. Sun, and H.-T. Chen, "Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction," 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5449–5459, 2021, doi: 10.1109/CVPR52688.2022.00538.

[21] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely, "Stereo magnification: Learning view synthesis using multiplane images," ArXiv, vol. abs/1805.09817, 2018, doi: 10.1145/3197517.3201323.

[22] P. P. Srinivasan, R. Tucker, J. T. Barron, R. Ramamoorthi, R. Ng, and N. Snavely, "Pushing the boundaries of view extrapolation with multiplane images," in 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 175–184, doi:10.1109/CVPR.2019.00026.

[23] K. Deng, A. Liu, J.-Y. Zhu, and D. Ramanan, "Depth-supervised nerf: Fewer views and faster training for free," 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 12 872–12 881, 2021, doi: 10.1109/CVPR52688.2022.01254.

[24] J. Chen, Z. Yu, L. Ma, and K. Zhang, "Uncertainty awareness with adaptive propagation for multi-view stereo," Applied Intelligence, 2023, doi: 10.1007/s10489-023-04910-z. [Online]. Available: https://api.semanticscholar.org/CorpusID:261038143

[25] P. Jiang, X. Yang, Y.-R. Chen, W.-Z. Song, and Y. Li, "Adaptmvsnet: Efficient multi-view stereo with adaptive convolution and attention fusion," Computers & Graphics, 2023, doi: 10.1016/j.cag.2023.08.014. [Online]. Available: https://api.semanticscholar.org/CorpusID:260792500

[26] R. Weilharter and F. Fraundorfer, "Highres-mvsnet: A fast multi-view stereo network for dense 3d reconstruction from high-resolution images," IEEE Access, vol. 9, pp. 11 306–11 315, 2021, doi: 10.1109/ACCESS.2021.3050556.

[27] C.-Y. Chiu, Y.-T. Wu, I.-C. Shen, and Y.-Y. Chuang, "360mvsnet: Deep multi-view stereo network with 360° images for indoor scene reconstruction," in 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2023, pp. 3056–3065, doi: 10.1109/WACV56688.2023.00307.

[28] W. Su, Q. Xu, and W. Tao, "Uncertainty guided multi-view stereo network for depth estimation," IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 11, pp. 7796–7808, 2022, doi: 10.1109/TCSVT.2022.3183836.

[29] X. Guan, W. Tong, S. Jiang, P. Z. H. Sun, E. Q. Wu, and G. Chen, "Multistage pixel-visibility learning with cost regularization for multiview stereo," IEEE Transactions on Automation Science and Engineering, vol. 20, no. 2, pp. 751–762, 2023, doi: 10.1109/TASE.2022.3165944.

[30] J. Adler and O. Öktem, "Learned primal-dual reconstruction," IEEE Transactions on Medical Imaging, vol. 37, pp. 1322–1332, 2017, doi: 10.1109/TMI.2018.2799231.

[31] ——, "Solving ill-posed inverse problems using iterative deep neural networks," Inverse Problems, vol. 33, p. 124007, 2017, doi: 10.1088/1361-6420/aa9581.

[32] G. Wetzstein, D. Lanman, M. Hirsch, and R. Raskar, "Tensor displays," ACM Transactions on Graphics (TOG), vol. 31, pp. 1 – 11, 2012, doi: 10.1145/2343456.2343480.

[33] T. K. Porter and T. D. S. Duff, "Compositing digital images," international conference on computer graphics and interactive techniques, 1984, doi: 10.1145/964965.808606.

[34] B. Mildenhall, P. Hedman, R. Martin-Brualla, P. P. Srinivasan, and J. T. Barron, "Nerf in the dark: High dynamic range view synthesis from noisy raw images," 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 16 169–16 178, 2021, doi: 10.1109/CVPR52688.2022.01571.

[35] D. Verbin, P. Hedman, B. Mildenhall, T. E. Zickler, J. T. Barron, and P. P. Srinivasan, "Ref-nerf: Structured view-dependent appearance for neural radiance fields," 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5481–5490, 2021, doi: 10.1109/CVPR52688.2022.00541.

[36] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Mip-nerf 360: Unbounded anti-aliased neural radiance fields," 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5460–5469, 2021, doi: 10.1109/CVPR52688.2022.00539.

[37] M. Tancik, B. Mildenhall, T. Wang, D. Schmidt, P. P. Srinivasan, J. T. Barron, and R. Ng, "Learned initializations for optimizing coordinate-based neural representations," 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2845–2854, 2020, doi: 10.1109/CVPR46437.2021.00287.

[38] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," ACM Transactions on Graphics (TOG), vol. 41, pp. 1 – 15, 2022, doi: 10.1145/3528223.3530127.

[39] S. Peng, Y. Zhang, Y. Xu, Q. Wang, Q. Shuai, H. Bao, and X. Zhou, "Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans," 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 9050–9059, 2020, doi: 10.1109/CVPR46437.2021.00894.

[40] T. Khakhulin, D. Korzhenkov, P. Solovev, G. Sterkin, A.-T. Ardelean, and V. S. Lempitsky, "Stereo magnification with multi-layer images," 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8677–8686, 2022, doi: 10.1109/CVPR52688.2022.00849.

[41] J. Jin, M. Guo, J. Hou, H. Liu, and H. Xiong, "Light field reconstruction via deep adaptive fusion of hybrid lenses," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 10, pp. 12 050–12 067, 2023, doi: 10.1109/TPAMI.2023.3287603.

[42] G. Wu, Y. Wang, Y. Liu, L. Fang, and T. Chai, "Spatial-angular attention network for light field reconstruction," IEEE Transactions on Image Processing, vol. 30, pp. 8999–9013, 2021, doi: 10.1109/TIP.2021.3122089.

[43] J. Shi, X. Jiang, and C. Guillemot, "Learning fused pixel and feature-based view reconstructions for light fields," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 2552–2561, doi: 10.1109/CVPR42600.2020.00263.

[44] J. Lawrence, D. B. Goldman, S. Achar, G. M. Blascovich, J. G. Desloge, T. Fortes, E. M. Gomez, S. Häberling, H. Hoppe, A. Huibers et al., "Project starline: A high-fidelity telepresence system," 2021, doi: 10.1145/3478513.3480490.

[45] Y. Zhang, J. Yang, Z. Liu, R. Wang, G. Chen, X. Tong, and B. Guo, "Virtualcube: An immersive 3d video communication system," IEEE Transactions on Visualization and Computer Graphics, vol. PP, pp. 1–1, 2021, doi: 10.1109/TVCG.2022.3150512.

[46] Q. Chen and V. Koltun, "Photographic image synthesis with cascaded refinement networks," international conference on computer vision, 2017, doi: 10.1109/ICCV.2017.168.

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," CoRR, vol. abs/1412.6980, 2014. [Online]. Available: https://api.semanticscholar.org/CorpusID:6628106

[48] S. Wizadwongsa, P. Phongthawee, J. Yenphraphai, and S. Suwajanakorn, "Nex: Real-time view synthesis with neural basis expansion," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 8534–8543, doi: 10.1109/CVPR46437.2021.00843.

[49] E. Penner and L. Zhang, "Soft 3d reconstruction for view synthesis," ACM Transactions on Graphics (TOG), vol. 36, pp. 1 – 11, 2017, doi: 10.1145/3130800.3130855.

[50] R. Szeliski and P. Golland, "Stereo matching with transparency and matting," International Journal of Computer Vision, vol. 32, pp. 45–61, 1998, doi: 10.1109/ICCV.1998.710766.

[51] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan, "Mvsnet: Depth inference for unstructured multi-view stereo," in European Conference on Computer Vision, 2018, doi: 10.1007/978-3-030-01237-3_47.

Yijie Deng is currently a master student in Tsinghua-Berkeley Shenzhen Institute (TBSI), Tsinghua University. He received B.E. from Wuhan University in 2021. His research interest is 3D vision.

Lei Han is currently a researcher in Linx Lab of HiSilicon (subsidiary of Huawei Technologies). He studied Electrical Engineering at the Hong Kong University of Science and Technology and Tsinghua University. He received the B.S. degree in July 2013 and joined the Department of Electrical Computing Engineering at the Hong Kong University of Science and Technology in September 2016, where he is pursuing the PhD degree. His current research focuses on multi-view geometry and 3D computer vision.

Lin Li is currently an engineer in Huawei. She received the B.S. and the integrated M.S. and Ph. D. degrees from the Department of Electronic Information Technology and Instrument Institute of ZheJiang University in 2010 and 2015, respectively.

Tianpeng Lin is currently a researcher in Linx Lab of HiSilicon (subsidiary of Huawei Technologies). He received Ph.D in GeoInformatics in the Chinese University of Hong Kong in 2016, and B.E. from Tsinghua University in 2009. He works for Kirin chipsets and Linx Lab and focuses on the chipset algorithm development in SLAM and 3D reconstruction.

Jinzhi Zhang is currently a Ph.D student in Tsinghua-Berkeley Shenzhen Institute (TBSI), Tsinghua University. He received B.E. from Huazhong University of Science and Technology in 2019. His research interest is 3D vision.

Lu Fang is currently an Associate Professor in Tsinghua University. She received Ph.D from the Hong Kong Univ. of Science and Technology in 2011, and B.E. from Univ. of Science and Technology of China in 2007. Her research interests include computational imaging and visual intelligence. Dr. Fang is currently IEEE Senior Member, Associate Editor of IEEE TIP and TMM.