RPBG: Towards Robust Neural Point-based Graphics in the Wild

Qingtian Zhu[1⋆], Zizhuang Wei[2,3], Zhongtian Zheng[3], Yifan Zhan[1], Zhuyu Yao[4], Jiawang Zhang[4], Kejian Wu[4], and Yinqiang Zheng[1✉]

[1] The University of Tokyo
[2] Huawei Technologies
[3] Peking University
[4] XREAL

Abstract. Point-based representations have recently gained popularity in novel view synthesis, for their unique advantages, *e.g.*, intuitive geometric representation, simple manipulation, and faster convergence. However, based on our observation, these point-based neural re-rendering methods are only expected to perform well under ideal conditions and suffer from noisy, patchy points and unbounded scenes, which are challenging to handle but *defacto* common in real applications. To this end, we revisit one such influential method, known as Neural Point-based Graphics (NPBG), as our baseline, and propose Robust Point-based Graphics (RPBG). We in-depth analyze the factors that prevent NPBG from achieving satisfactory renderings on generic datasets, and accordingly reform the pipeline to make it more robust to varying datasets in-the-wild. Inspired by the practices in image restoration, we greatly enhance the neural renderer to enable the attention-based correction of point visibility and the inpainting of incomplete rasterization, with only acceptable overheads. We also seek for a simple and lightweight alternative for environment modeling and an iterative method to alleviate the problem of poor geometry. By thorough evaluation on a wide range of datasets with different shooting conditions and camera trajectories, RPBG stably outperforms the baseline by a large margin, and exhibits its great robustness over state-of-the-art NeRF-based variants. Code available at `https://github.com/QT-Zhu/RPBG`.

Keywords: Point-based graphics · Novel view synthesis · Neural rendering

1 Introduction

Novel view synthesis (NVS) aims to synthesize novel views under given camera poses by aid of a series of already posed images, which is a fundamental task in both computer vision and computer graphics and has been studied for decades.

Among the feasible solutions, NeRF (Neural Radiance Field) [26] approximates an implicit scene representation, *i.e.*, a radiance field (RF), encoded by a neural network mapping 3D coordinates and view directions to colors and densities. When rendering, the fitted RF is queried multiple times along a ray to volume-render the corresponding pixel for a novel view. Subsequent variants of NeRF follow a case-by-case design and

⋆ Work done during Q. Zhu's internship at XREAL.

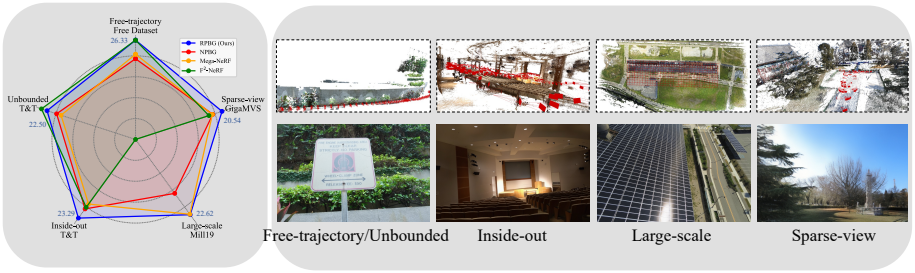

Fig. 1: Left: RPBG manages to achieve **all-round** good re-renderings (PSNR plotted) across generic datasets over the baseline [1] as well as state-of-the-art RF-based methods [42, 44]. **Right:** We demonstrate the point clouds (with camera trajectories visualized) and corresponding re-rendered novel views of the representative scenes, revealing the great robustness and scalability of RPBG. Zoom in for best view.

adopt different parameterization techniques specifically for different scene types, *e.g.*, object-centric [3, 54], free-trajectory [44], and large-scale scenes [38, 42]. Therefore, putting forward one robust method that can work stably across varying scene types with a unified parameterization technique is considered as challenging.

Recently, point-based alternatives [1, 18, 30, 33] have gained substantial attention for their unique advantages over implicit representations, *e.g.*, ease of manipulation and faster training. The practice of employing point-based parameterization for rendering can be traced all the way back to [16, 20], and is intuitive especially under the context of NVS. Analogue to NeRF and its variants (collectively referred to as RF-based methods), whose optimization and convergence heavily rely on the inherent co-visibility among images, the 3D points triangulated from a series of 2D images according to epipolar geometry have already contain all the verified co-visibility information. In this way, by enforcing the triangulated points as strong prior knowledge for NVS, the expected freedom of optimization is greatly constrained, leading to faster training as well as better robustness. Recent attempts have achieved promising results in terms of fast optimization and rendering [18, 30] and accurate geometric representation [56]. Concretely, we focus on the specific category of methods that manage to integrate CNN-based neural renderers to yield re-renderings with featuremetric neural buffers [22, 30], for their concise pipeline and the potential capability of obtaining visually pleasing re-renderings.

However, by evaluating the representative baseline, NPBG (Neural Point-based Graphics) [1], with generic datasets, we realize that the original design only intends for ideal conditions, *e.g.*, synthetic data [26] and well-captured human heads [31]. When synthesizing novel views under more general conditions, the performance of NPBG degrades severely. In this paper, we strive to boost the robustness of NPBG-like point-based neural re-rendering pipelines and reveal the true potential to achieve state-of-the-art performance across varying datasets in-the-wild, by analyzing the reasons for performance degradation and seeking for remedies.

Generally speaking, the major difficulties that the vanilla NPBG is faced up with include handling the background, handling patchy point clouds, and identifying correct

point visibility. We reform the CNN-based neural renderer, with an inspiration from image restoration algorithms [6, 8, 52] that are able to identify downgraded patterns and restore the corresponding high-quality images. To make sure the neural renderer can capture sufficient and valid context information with extremely sparse rasterizations, we particularly leverages a Downgrade-aware Convolution (DAC) module to determine the correct point visibility with regard to a given camera pose and performs a pseudo point-wise back-face culling operation with visual self-attention. The background is also modeled in a lightweight manner. Instead of incorporating a massive environment map [33], a simple default trainable feature vector can reach similar quantitative results when working with the stronger neural renderer. We also discover the pseudo density calculated from the neural textures can roughly verify the existence of a given 3D position, which can be used to augment the poorly triangulated point clouds. In addition, we also simplify the phased training paradigm in [1, 30] and optimize the parameters of both the neural textures and the renderer end-to-end collaboratively. We term our version of point-based re-rendering as **RPBG** (Robust Point-based Graphics), to emphasize its great robustness across different generic datasets.

For thorough qualitative and quantitative evaluation, we collect 4 typical challenging scene types as the benchmark for robust NVS, *i.e.*, 360° unbounded scenes (with free trajectories), inside-out scenes, large-scale scenes (at the scale of a block or campus), and sparse-view scenes. As a result, RPBG exhibits great robustness with perceptually satisfactory synthesis across the aforementioned typical scenes types as showcased in Fig. 1, where the high-quality re-renderings are obtained with an exactly identical parameterization strategy without any manual configuration, relieving the practitioners from the exhaustive per-scene search of hyper-parameters. Its stable superiority over state-of-the-art NVS methods, we believe, is of great significance especially for real applications.

The main contributions are three-fold as follows:

- We put forward RPBG, as a more robust and practical alternative for re-rendering high-quality images from triangulated 3D points.
- According to our in-depth analysis, we enhance the neural re-rendering pipeline regarding the neural renderer, environment modeling, point cloud augmentation, and the training paradigm.
- RPBG manages to greatly boost the performance of neural point re-renderer by a large margin, and exhibits stably greater robustness and generalizability over RF-based methods with even better perceptual rendering quality.

2 Related Work

Radiance Fields for NVS Along with the proposed NeRF [26], the scene representation of RF is becoming popular for its ease of optimization by the differentiable volume rendering. In a RF, each position is assigned with an anisotropic color and a density, and to render a pixel of a novel view, one needs to sample the trained field and conduct ray marching. The original NeRF [26], together with its variants [2, 3, 39, 54], employs an MLP to represent the functional mapping from coordinates and view directions to the radiance values. Several attempts have been made to encode RFs with different

data structures for faster training and inference, *e.g*., grid voxels [12, 36], decomposed tensors [5], hash grids [27].

In practice, the parameterization strategies have a great impact on the rendering quality when reconstructing different types of scenes, *e.g*., forward-facing ones, 360° unbounded ones, and large-scale ones. For unbounded scenes, NeRF++ [54] applies separate networks and different parameterization to model near and distant objects; mip-NeRF 360 [3] designs a smooth contraction operator to parameterize the whole unbounded scene into a ball; Mega-NeRF [42] and Block-NeRF [38] partition 3D scenes explicitly and use different networks to represent each scene partition; F^2-NeRF [44] proposes to use perspective warping to handle sequential data with arbitrary trajectories. The varying case-by-case parameterization strategies of RF-based NVS greatly constraint the generalizability of methods.

Point-based Graphics A point cloud is a collection of 3D coordinates that is usually used as a topology-agnostic coarse shape representation of the 3D geometry and favored for its flexibility and sparsity for storage. The development of techniques of employing points as modeling primitives for rendering (referred as point-based graphics [15]) can be traced back to [16, 20], the best practice of which is to replace each point with an oriented circular disk (a surfel) and reply on splatting to blend overlapping surfels [28].

In contrast to conventional physically based rendering (PBR) techniques, neural rendering [40] learns to render high-quality images in a data-driven manner. Particularly, we focus on the methods that conduct neural re-rendering with point clouds. Bui *et al*. [4] propose to enhance the coarse point-based rendering by a GAN for image super-resolution. NRW [25] and InvSFM [29] attempt to re-render the reconstructed point cloud with respective auxiliary buffers, *e.g*., latent appearance vectors, semantic masks and SIFT descriptors [23]. NPBG [1,30] adopts a U-Net-like [32] CNN to render neural point textures as RGB images, and exhibits better flexibility over mesh-based proxies [41]. The practical applications in the context of autonomous driving, *e.g*., scene editing and stitching, and large-scale training, are further explored in READ [21]. Recently, ADOP [33] and Gaussian Splatting [18] have demonstrated notable accomplishments. However, the differentiable rendering/splatting scheme employed requires an extensive amount of memory to maintain the computational graph and gradients during training. This limitation hinders their application on large-scale scenes.

3 Neural Point-based Graphics Revisited

In this section, we revisit the pipeline of NPBG [1], analyze the existing drawbacks when attempting to re-render generic scenes, and attempt to explain why NPBG finds it difficult to handle the background and patchy points, and identify correct point visibility. We hope these discussions be of sufficient insights to support our modifications.

3.1 Preliminaries

Inputs As the common practice in NVS, the inputs are a series of images with respective camera parameters (both intrinsics and extrinsics), and specifically for point-based

(a) Rasterized Points (b) Ground Truth (c) NPBG (d) Ours

Fig. 2: Typically challenging scenes in T&T dataset [19] for NPBG. **Top:** *Auditorium*, where the walls and ceilings are extremely sparse. **Bottom:** *Museum*, where the point sparsity makes the occlusion and visibility complicated.

methods [1, 18, 30], the very first step is supposed to be the triangulation of 3D points from 2D observations. The original NPBG-like pipelines [1, 30] mainly consider this as a pre-processing without much attention.

Point Rasterization NPBG [1] and NPBG++ [30] apply a non-differentiable hard point z-buffering operation as an approximated back-face culling when rasterizing points as 2D fragments, where the fragment is updated when and only when the newly projected point has a smaller z-depth than the current one. After a traversal of all the points, the fragment keeps the record of the indices of rasterized points. Then tensor scattering is performed to index the neural texture at the corresponding positions. Compared to differentiable rendering/splatting alternatives [18, 33], we consider the most significant advantage is its memory efficiency and thus better scalability, since the computation graph required to keep is much smaller for NPBG. We also want to keep this scalability without interfering the elegant rasterization paradigm.

Neural Renderer The renderer $\mathcal{R}_\Theta : \mathbb{R}^{H\times W\times C} \rightarrow \mathbb{R}^{H\times W\times 3}$ of NPBG [1] follows a vanilla U-Net-like architecture [32]. Since the target point clouds for rendering are supposed to be well-constructed, the vanilla U-Net is expressive enough to complete the assigned task of mapping higher dimensional features from the neural texture $\mathbf{T}$ to RGB values. Empirically, we find the expected properties of $\mathcal{R}_\Theta$ have much in common with a network for low-level vision tasks, *e.g.*, image restoration.

3.2 Problems

Patchy Triangulation Consider the inputs, for a NVS system aiming to synthesize images for generic scenes, the triangulation of points is not trivial. The point triangulation step to perform is similar to the step of multi-view stereo (MVS) reconstruction [14, 48], the key of which is to identify co-visible pixels across images and lift the 2D pixels to a 3D points according to epipolar geometry. Such algorithms suffer from non-Lambertian surfaces and textureless regions, both of which are very common and lead to poor triangulated and patchy points. We attach two typical scenes that NPBG fails to yield good renderings in Fig. 2.

Wrong Point Visibility As can be imagined, such point visibility can be erroneous due to the poor quality of triangulated points, and also due to the inherent sparsity of points as a 3D representation. In this way, the points belonging to back faces, which should be considered as occluded, are rasterized as the fragment for further processing. Manually setting a depth threshold for bound check could not be helpful either, since being far from the camera does not necessarily indicate they should be occluded — they could be parts of the environment. While NHR [47] proposes to take the depth buffer as an additional rendering condition, we find its effectiveness not significant for generic scenes other than well-masked human captures.

Lack of Context for Re-rendering Similarly, when re-rendering an incomplete point cloud, where the neural buffers are usually with significant flaws, the whole receptive field of a kernel at a given position may only capture the downgraded regions, leading to failure to yield reasonable restoration, especially for a high target resolution. It is worth noting that, though the relationship between the number of network layers and rendering quality has been discussed in [30, 33], there is still huge room for the improvement of the renderer.

Failure to Effectively Model the Environment Unbounded scenes are a typical challenging scene type for NVS. RF-based methods that are designed on purpose to handle such scenes [3, 54] typically employ different parameterization to encode the areas outside a certain bounding sphere. For point-based methods, ADOP [33] proposes to apply an environment map of $H \times W \times C$ to model the environment, which is equivalent to wrapping the triangulated points with a sphere with $H \times W$ points, resulting in more than 5×10^5 points as overheads, according to the default configuration.

Summary To summarize, we attribute the observed problems to two main causes: poor geometry and weak, local renderer. The two causes are to some extent coupled for a better geometry will relieve the difficulty of the renderer and *vice versa*.

4 Robust Point-based Graphics

We are strongly convinced that the pipeline of point-based neural re-rendering has the great potential to outperform popular RF-based solutions, for its unified point-as-parameterization fashion and the incorporation of neural networks to yield visual details. Therefore in this section, we introduce Robust Point-based Graphics (RPBG), as an enhanced version of NPBG, with a particular focus on robustness under generic scenes. We will elaborate the modifications made and shed light on the underlying insights and motivation. The overall pipeline of RPBG as well as the training paradigm is briefly illustrated in Fig. 3.

4.1 Downgrade-aware Neural Renderer

As is attributed as one key problem in NPBG, the U-Net-based neural renderer is considered as too naive to handle the challenging situations in generic scenes. The

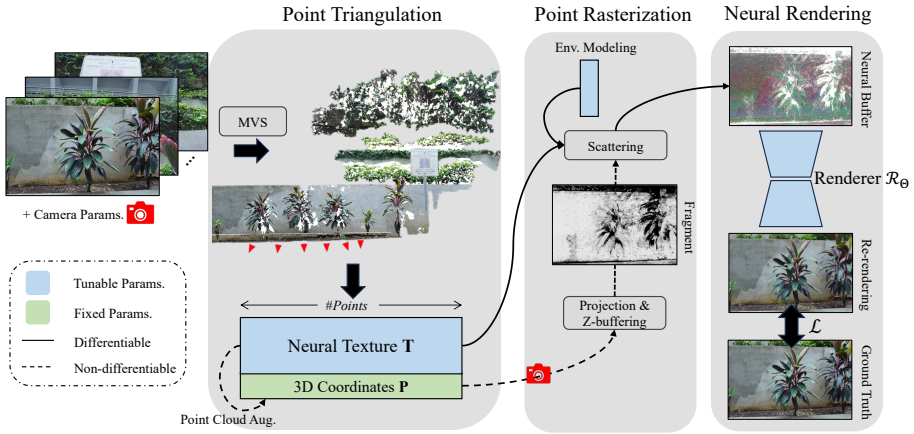

Fig. 3: The overall pipeline of RPBG. **Point Triangulation:** We first triangulate a 3D proxy for re-rendering with posed images, with its geometry-bounded neural texture initialized. **Point Rasterization:** The points are raterized to the given camera in a non-differentiable manner. By indexing the texture with the fragment, we obtain the neural buffer. A learnable point-size neural texture $\mathbf{T}_{\text{env}}$ is also optimized. **Neural Rendering:** The restoration from downgraded neural buffer to photo-realistic images is performed by a CNN. The network and the neural texture are optimized end-to-end by image-level losses. An offline point cloud augmentation strategy is introduced to alleviate the problem of patchy triangulation under challenging conditions.

modifications made are deeply inspired by low-level vision tasks, where networks can adaptively determine downgraded parts, *e.g.*, the deblurring/deraining networks are able to identify the blurring/raining pixels from the whole image, and correct them accordingly. We expect the neural renderer $\mathcal{R}_\Theta$ can benefit from relevant restoration-targeting techniques, and become able to decode high-quality visual information from patchy buffers.

After evaluating three state-of-the-art fundamental architectures for image restoration, *i.e.*, multi-scale fusion [8], multi-stage [52], and U-Net [6], following the taxonomy in [6]. According to the reported experiments in Sec. 5.4, we opt the paradigm in [8] for it achieves the best balance between performance and time/memory efficiency. However, the convolution layers in such paradigm is still with a fixed receptive field, lacking in robustness against point sparsity. We would like to further enlarge the receptive field to a global scale and attempt to explicitly model the visual attention to weigh the observed points.

Transformer-based architectures for image restoration, *e.g.*, Restormer [51], brings unacceptable memory overheads, so we rely on the frequency-domain alternative, Fast Fourier Convolution (FFC) [7], which can theoretically capture global contexts, to determine the correct point visibility adaptively. FFC performs channel-wise real 2D FFT (Fast Fourier Transform) and inverse real 2D FFT on 2D tensors. Real FFT uses only half of the spectrum and by convolving the transformed frequency-domain tensors. In this way, a receptive field covering the entire image is considered.

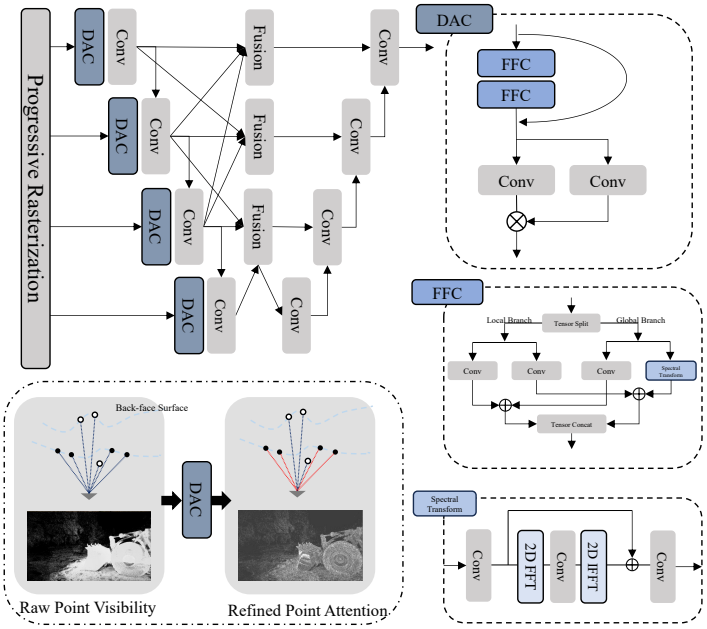

Fig. 4: The architecture of the downgrade-aware neural renderer in RPBG, with some conventional modules omitted. From the visualized attention map, DAC manages to adaptively handle the severely erroneous point visibility.

Inspired by [37], we apply FFT (as the global branch) in parallel to conventional convolution layers (as the local branch) and rely on the fused features of both branches to determine the downgraded regions for the gated convolution [50] to filter. Based on the common practice in image inpainting, we leverage gated convolutions at the early stage of the renderer, to help locate the downgrade by wrong point visibility. We name such customized gated convolution module as the Downgrade-aware Convolution (DAC) module, and as can be inferred from the point attention maps before and after DAC in Fig. 4, DAC manages to determine complicated point visibility with patchy triangulation.

4.2 Point Triangulation

Though there are advances in MVS empowered by deep learning [43, 45, 46], recovering a complete, accurate, and dense point cloud for certain regions remains a challenging. For RPBG, we adopt an off-the-shelf MVS method [46] to perform point triangulation for its memory efficiency to allow large-batch inference.

As for problem the poor triangulation, we partially leave it to the neural renderer $\mathcal{R}_\Theta$, relying on a stronger renderer to recover the re-rendering from downgraded buffers. Inspired by the point cloud extraction method of RF-based implementations [39], where the estimated radiance density σ can be a rough indicator for surface, we propose a point cloud augmentation technique to densify the initial triangulation.

The point cloud augmentation follows a trial-and-error paradigm by first assuming the existence of newly sampled points and then verify them by estimated pseudo densities $\sigma_i = \sum |\mathbf{T}_i|$. It is empirically observed that the absolute activation of the point-wise neural texture can roughly represent the reliability of the 3D position — it is reasonable since an outlier will be less distinguishable by the neural renderer and thus less visual attention will be given.

Note that this strategy is optional, and we only apply such strategy to scenes with extremely poor triangulation. Please refer to the Supplementary Material for more discussions regarding the self-pruning.

4.3 Environment Modeling

As a result of the incorporation of a stronger neural renderer, we find that given a neural texture with $C = 8$, the dense environment map suggested in [33] is redundant. Instead, we shrink the overheads for environment modeling from $H \times W \times C$ to $1 \times C$, and relying on the stronger neural renderer to decode the background.

When rasterizing, we employ a tunable feature vector $\mathbf{T}_{\text{env}}$ aside the neural point texture $\mathbf{T}$, as the default value for vacant pixels in the fragment, which is also involved in the end-to-end training. By experiments in Sec. 5.4, we demonstrate that the lightweight modeling strategy reaches the quantitative performance equivalent to applying an environment map, yet with a negligible overhead.

4.4 Collaborative Optimization

Recall that the point rasterization procedure is completely parameter-free. The overall collaborative optimization scheme can be thus formulated as

$$\mathbf{T}^*, \Theta^* = \arg\min_{\mathbf{T},\Theta} \sum_k \mathcal{L}(\mathbf{I}_k, \hat{\mathbf{I}}_k) = \arg\min_{\mathbf{T},\Theta} \sum_k \mathcal{L}(\mathbf{I}_k, \mathbf{K}, \mathbf{R}_k, \mathbf{t}_k | \mathbf{T}, \mathcal{R}_\Theta), \quad (1)$$

where $\mathbf{T}$ and Θ stand for the tunable parameters, *i.e.*, the neural point texture (alongside the globally shared $\mathbf{T}_{\text{env}}$) and the parameters of the renderer $\mathcal{R}_\Theta$ for neural re-rendering, while $\mathbf{I}_k$ is the target image whose calibration is $\mathbf{K}$ and $[\mathbf{R}_k|\mathbf{t}_k]$ and $\hat{\mathbf{I}}_k$ is the re-rendering.

Note that, we also discard all the typical tossing sampling [21] and optimization steps [30, 33] of point-based neural re-rendering, and manage to tune all the parameters involved in a simple but effective collaborative end-to-end way. The neural texture is initialized with all zeros while the rendering CNN is trained from scratch for each scene.

Loss Function Since the rendering scheme in both NPBG and RPBG is convolutional, where images are rendered in patches, we are able to apply patch-aware losses to enforce the involvement of neighboring pixels to ensure patch-to-patch consistency, offering perceptually good renderings. To this end, in addition to the pixel-wise Huber norm $\mathcal{L}_{\text{huber}}$ providing the basic supervision and numerical stability, we apply two patch-aware losses to the collaborative optimization of RPBG, namely the perceptual VGG loss [10, 17] $\mathcal{L}_{\text{vgg}}$, and the FFT loss [8, 13] $\mathcal{L}_{\text{fft}}$. VGG loss compares the rendered image and the target ground-truth image in a high-dimensional feature space by a pre-trained VGG-19

network [35]. It reveals the perceptual similarity between images that pixel-wise metrics fail to measure. FFT loss measures the image-to-image distance in the frequency domain by carrying out 2D FFT towards images. The frequency components are considered to be crucial in terms of the perceptual quality [13].

The final loss function applied in RPBG is composed as

$$\mathcal{L} = \lambda_{\text{huber}}\mathcal{L}_{\text{huber}} + \lambda_{\text{vgg}}\mathcal{L}_{\text{vgg}} + \lambda_{\text{fft}}\mathcal{L}_{\text{fft}}. \tag{2}$$

5 Experiments

5.1 Datasets

We conduct experiments covering a diverse range of scene types to quantitatively examine the generalizability and robustness of NVS methods, including the Free dataset [44] (7 free-trajectory scans with long camera trajectories), Tanks and Temples (T&T) dataset [19] (4 unbounded scans and 5 inside-out scans, without salient dynamic objects), Mill19 dataset [42] (2 large-scale aerial scans with over 1600 images in each scan), and GigaMVS dataset [53] (8 sparse-view outdoor scans). Note that for T&T, we use the raw undistorted images provided by the benchmark without any masking [22]. We follow the common split protocol, that 1 frame out of every 8 frames is evaluated, for the Free dataset, T&T dataset, and GigaMVS dataset. For Mill19 dataset, we apply the officially recommended split to align with the experiments in [42].

We also include some other challenging datasets for a thorough evaluation, *e.g.*, ScanNet++ dataset [49], a challenging indoor benchmark, OMMO dataset [24], a multi-modal aerial NVS dataset and ETH-MS dataset [11], a super-large-scale dataset (around $5k$ images). Please refer to the Supplementary Material for more quantitative and qualitative results.

5.2 Implementation Details

Data Preparation To obtain the per-scene point cloud representation for re-rendering, we follow the mapping procedure of COLMAP [34] for sparse triangulation and the MVS reconstruction with a trained network [46] for dense triangulation. Note that the reconstruction only takes a small proportion of time over the whole training phase and we conduct ablation experiments to show that RPBG is robust against different triangulation configurations. Please refer to Sec. 5.4 as well as the Supplementary Material for more information.

Training Settings We randomly crop images to square patches of 256×256, with a batch size of 8. The learning rate for the neural point textures is 10^{-1}, and 10^{-4} for the rendering networks, which will decay by a factor of 0.5 if 5 consequent epochs witness no drop in the loss function. A dimension of 8 is applied for any neural texture regardless of the scene scale or complexity. For the weights of loss functions, we globally set $\lambda_{\text{huber}} = 10^3$, $\lambda_{\text{vgg}} = 1$, and $\lambda_{\text{fft}} = 1$. The training is performed on one NVIDIA GeForce RTX 3090, with a GPU memory consumption of up to 23 GB. It takes around 8 to 30 GPU hours for training, depending on the data scale. We would like to emphasize that, RPBG does not require case-by-case scene parameterization or grid search of training hyper-parameters for all the scenes covered in the experiments.

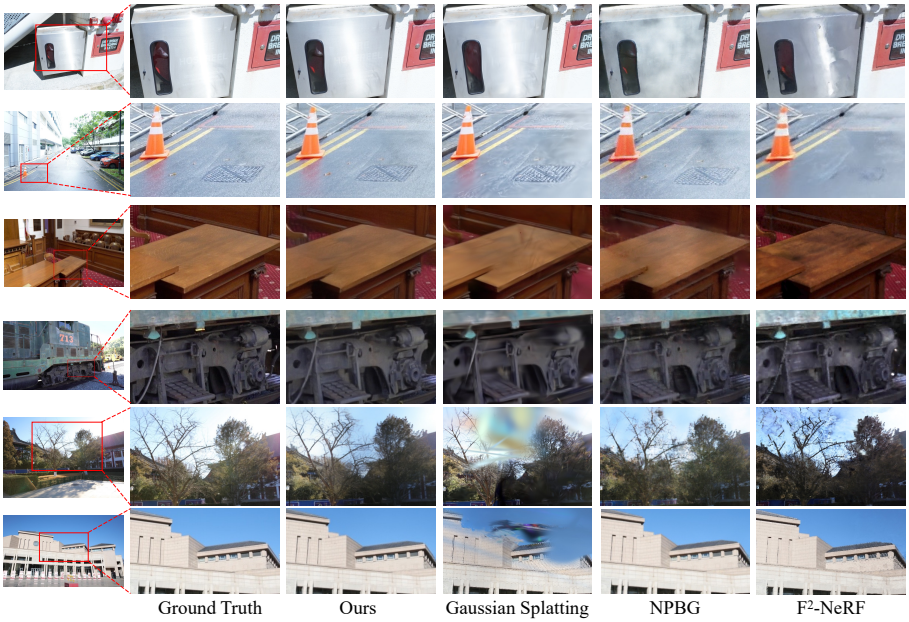

Fig. 5: Visualized comparisons over varying scenes. **From top to bottom:** *sky* and *hydrant* of the Free dataset [44], *Courtroom* and *Train* of T&T dataset [19], and *DayaTemple* and *MemorialHall* of GigaMVS dataset [53]. We include the results of RPBG (Ours), Gaussian Splatting [18], NPBG [1], and F^2-NeRF [44] for comparison. Zoom in for best view.

Evaluation Metrics We adopt the metrics of PSNR, SSIM and LPIPS (VGG) [55] for the evaluation between the synthesized and the target images. According to [55], PSNR does not faithfully measure image sharpness and so cannot properly account for the nuances of human visual perception.

5.3 Results

In Fig. 5, we demonstrate several typical groups of visual comparisons of NVS results by RPBG, two representative point-based methods, *i.e.*, Gaussian Splatting [18] and NPBG [1], and a state-of-the-art NeRF variant for unbounded scenes, F^2-NeRF [44]. We group the datasets by category and report the corresponding scene-averaged quantitative scores in Tab. 1. Note that RPBG achieves the best SSIM and LPIPS, which are more relevant to the high-frequency components of images, across all datasets.

Free-trajectory/Unbounded Scenes Compared with NVS methods designed in particular for unbounded scenes, NeRF++ [54] and F^2-NeRF [44], RPBG achieves the best SSIM and LPIPS and comparable PSNR. For visual effects showcased in Fig. 5, since RPBG involves a neighborhood for rendering, its results appear more visually harmonious, especially than Gaussian Splatting [18] and F^2-NeRF [44]. It is necessary to mention

Table 1: Quantitative evaluation of state-of-the-art NVS methods [1, 5, 18, 26, 30, 42, 44, 54] and RPBG across diverse scenes grouped by category, including free trajectory/unbounded scenes, inside-out scenes, large-scale scenes, and sparse-view scenes. The figure following the dataset name stands for the number of scenes the dataset contains.

Method	Free-trajectory/Unbounded						Inside-out			Large-scale			Sparse-view		
	Free Dataset-7			T&T-4			T&T-5			Mill19-2			GigaMVS-8		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
NeRF [26]	17.75	0.405	0.597	16.84	0.396	0.731	19.82	0.615	0.526	20.34	0.524	0.529	18.29	0.632	0.499
TensoRF [5]	21.74	0.549	0.600	20.73	0.643	0.566	19.43	0.634	0.570	20.07	0.497	0.597	18.70	0.653	0.506
NeRF++ [54]	23.47	0.603	0.499	21.66	0.658	0.529	19.25	0.610	0.585	20.19	0.520	0.531	18.38	0.632	0.495
F^2-NeRF [44]	26.32	0.779	0.276	23.66	0.764	0.303	20.12	0.706	0.394	N/A	N/A	N/A	17.44	0.540	0.470
Mega-NeRF [42]	22.60	0.570	0.562	18.73	0.578	0.478	19.16	0.617	0.451	22.49	0.550	0.510	18.25	0.581	0.394
NPBG [1]	21.40	0.639	0.340	19.85	0.698	0.376	20.57	0.696	0.371	16.21	0.357	0.644	18.34	0.620	0.405
NPBG++ [30]	20.06	0.592	0.445	17.23	0.653	0.474	18.30	0.684	0.411	17.04	0.400	0.648	19.30	0.663	0.443
Gaussian Splatting [18]	25.23	0.740	0.290	23.51	0.782	0.293	23.46	0.783	0.277	N/A	N/A	N/A	16.84	0.530	0.391
Ours	26.33	0.832	0.177	22.50	0.782	0.276	23.29	0.804	0.242	22.62	0.596	0.368	20.54	0.686	0.317

that, for RF-based methods [5, 26, 42, 44], we have manually adjusted the scene-specific hyper-parameters to achieve better results.

Inside-out Scenes The inside-out indoor scenes are strictly bounded, but lack high-quality ray intersections required for optimization, which will lead to under-fitting of the RF (*Courtroom* in Fig. 5). Similar to free-trajectory/unbounded scenes, RPBG outperforms state-of-the-art methods in SSIM and LPIPS.

Large-scale Scenes We mainly evaluate RPBG against Mega-NeRF [42] on the massive imagery of Mill19 dataset [42]. Our method outperforms Mega-NeRF [42] in every metric. It is worth noting that Mega-NeRF takes 240 GPU hours for training while RPBG only takes 29 GPU hours. Notably, our system with 256 GB RAM and RTX 3090 fails to afford the training of F^2-NeRF [44] and Gaussian Splatting [18] (marked as N/A in the table). In contrast, RPBG proves to be viable for large-scale scenes under the same hardware constraint.

Sparse-view Scenes Sparse-view inputs are considered to be extremely challenging for NVS methods. Since RPBG, NPBG [1], and NPBG++ [30] incorporate a point-based 3D proxy for re-rendering, they showcase better robustness over RF-based methods. As revealed in Fig. 5, Gaussian Splatting [18] fails to regularize the strong approximation power of point-wise Gaussians at training views, and yields obvious needle-like artifacts at novel views.

5.4 Ablation Study

Environment Modeling We study different strategies of environment modeling for NVS in-the-wild, namely leaving the blank pixels with zeros, filling the blanks with a default learnable feature (as is done in RPBG), and

(a) *w/o* Env. Modeling (b) *w/* Env. Modeling

Fig. 6: The impact of leveraging the default feature vector for environment modeling in an unbounded scene.

Table 2: Ablation on network designs on T&T dataset [19]. The reported inference time and memory consumption is tested with a target resolution of 1920×1080.

Method	Time(s)	Mem.(GB)	PSNR↑	SSIM↑	LPIPS↓
Baseline	1.03	5.49	20.01	0.666	0.390
+Multi-scale fusion	1.62	12.67	21.88	0.701	0.337
+DAC	1.67	13.01	22.92	0.769	0.320
+FFT loss	–	–	22.94	0.794	0.257
+Multi-stage	1.77	17.14	21.75	0.713	0.338
+U-Net	1.58	10.32	20.85	0.692	0.364

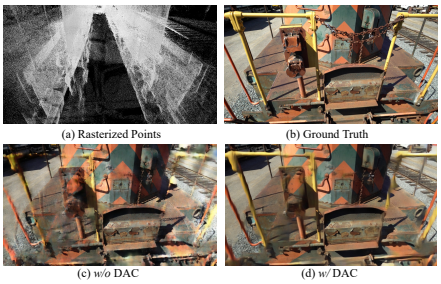

Fig. 7: A challenging case in *Train*, which is a zoomed-in view with complex occlusion and thin objects (*e.g.*, the handrails).

Table 3: Ablation on environment modeling on two unbounded scenes, namely *Caterpillar* of T&T dataset [19] and *sky* of Free dataset [44]. *#Points* ($\times 10^6$) represent the equivalent overheads when applying different strategies.

Env.	***Caterpillar***				***sky***			
	#Points	PSNR↑	SSIM↑	LPIPS↓	*#Points*	PSNR↑	SSIM↑	LPIPS↓
Zeros	7.95	19.89	0.456	0.403	17.49	21.77	0.598	0.315
Learnable	7.95	21.78	0.687	0.289	17.49	24.81	0.866	0.199
Sphere	8.95	21.68	0.686	0.283	18.49	25.15	0.869	0.179

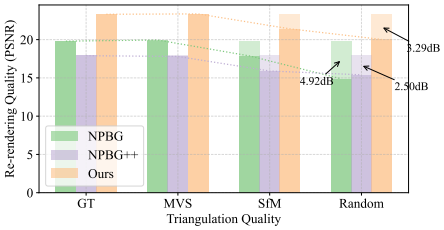

Fig. 8: Re-rendering quality (PSNR) with points of different quality on *Church*. RPBG manages to maintain its robustness against different levels of point sparsity with a 14% drop in PSNR (NPBG [1]: 25%; NPBG++ [30]: 14%).

wrapping the points with a sphere of 10^6 points (equivalent to an environment map in [33]). The quantitative results are in Sec. 5.4, where the strategy incorporated by RPBG boosts the performance by a large margin, with almost no additional overhead.

Network Architecture and Loss We study the key components in the rendering network in Sec. 5.4. The baseline refers to NPBG [1] trained per-scene from scratch with $\mathcal{L}_1$ and VGG loss. We compare three fundamental architectures for image restoration (with necessary modifications on the first several layers), *i.e.*, multi-scale fusion [8], multi-stage [52], and U-Net [6] following the taxonomy in [6]. Compared to the baseline, all modern restoration-oriented architectures bring remarkable improvement to the overall performance, and [8] is opted for a better balance between performance and time/memory efficiency. The gain by DAC is also considerable while the FFT loss mainly improves the perceptual quality. By the case in Fig. 7, we show the necessity of the DAC module when re-rendering challenging cases without in particular handling the erroneous point visibility.

Triangulation Configurations We also evaluate RPBG with different triangulation configurations. Indicated by the results in Fig. 8, RPBG witnesses a total drop of 3.29 dB in re-rendering PSNR when switching the triangulation from ground truth (GT) to randomly initialized points (Random). Note that even with sparse points (SfM),

which are usually yielded as a by-product when computing camera parameters, RPBG (21.41 dB) still outperforms F^2-NeRF (20.57 dB).

6 Discussion

Effectiveness of RPBG We would like to shed some light on the effectiveness of RPBG. The first factor should be the point representation. The triangulated 3D points have already contained all the verified co-visibility that is required for NVS. Compared to RF-based methods, which often struggle at local minima, such triangulation-as-parameterization paradigm greatly lower the complexity of searching, and is unified across different scene types. Besides, the patch-wise rendering scheme mentioned in Sec. 4.4 also improves the perceptual quality of re-rendering while RF-based methods apply pixel/ray-wise volume rendering, where the correlation between pixels are not modeled explicitly. For more discussions on the insights, we strongly refer the readers to the Supplementary Material.

Limitations An obvious problem with point-based re-rendering (RPBG and NPBG [1]) is that compared to lightweight RF-based variants [5, 27], they take more space to store the CNN parameters and the neural texture. Although it makes the scaling-up easier, the number of parameters grows linearly with the point cloud scale. Besides, due to the non-local neural renderer we employ, each point is encoded with visual context of a larger range, hindering the editability of RPBG. We have also noticed that, the CNN-based rendering scheme can lead to unsatisfactory temporal consistency especially when the triangulation is patchy so that we mainly count on the inpainting ability of the neural render for yielding re-renderings. Such flicker issue, as well as its potential solutions, has been discussed in [9].

7 Conclusion

In this paper, we present RPBG as a robust and practical alternative to NPBG, a baseline of point-based NVS methods, performing neural re-rendering on triangulated points. We analyze the key problems in NPBG, when attempting to generalize to more generic scenes other than the well-captured scans, and reform the pipeline to reveal the real potential of point-based graphics. Respectively motivated and inspired by RF-based methods and low-level image restoration methods, we reform the pipeline according to our analysis. By extensive experiments on diverse datasets, RPBG achieves stably superior results over state-of-the-art RF-based and point-based NVS methods, especially on the metrics with more attention paid to the nuances of human visual perception, without case-by-case parameterization across all scenes, indicating its robustness and generalizability.

Acknowledgments

This research was supported in part by JSPS KAKENHI Grant Numbers 24K22318, 22H00529, 20H05951, and JST-Mirai Program JPMJMI23G1.

References

1. Aliev, K.A., Sevastopolsky, A., Kolos, M., Ulyanov, D., Lempitsky, V.: Neural point-based graphics. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16. pp. 696–712. Springer (2020)
2. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5855–5864 (2021)
3. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5470–5479 (2022)
4. Bui, G., Le, T., Morago, B., Duan, Y.: Point-based rendering enhancement via deep learning. The Visual Computer **34**, 829–841 (2018)
5. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: Tensorf: Tensorial radiance fields. In: European Conference on Computer Vision. pp. 333–350. Springer (2022)
6. Chen, L., Chu, X., Zhang, X., Sun, J.: Simple baselines for image restoration. In: European Conference on Computer Vision. pp. 17–33. Springer (2022)
7. Chi, L., Jiang, B., Mu, Y.: Fast fourier convolution. Advances in Neural Information Processing Systems **33**, 4479–4488 (2020)
8. Cho, S.J., Ji, S.W., Hong, J.P., Jung, S.W., Ko, S.J.: Rethinking coarse-to-fine approach in single image deblurring. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4641–4650 (2021)
9. Dai, P., Zhang, Y., Li, Z., Liu, S., Zeng, B.: Neural point cloud rendering via multi-plane projection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7830–7839 (2020)
10. Dosovitskiy, A., Brox, T.: Generating images with perceptual similarity metrics based on deep networks. Advances in neural information processing systems **29** (2016)
11. ETH Zurich Computer Vision Group and Microsoft Mixed Reality & AI Lab Zurich: The ETH-Microsoft Localization Dataset. `https://github.com/cvg/visloc-iccv2021` (2021)
12. Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5501–5510 (2022)
13. Fuoli, D., Van Gool, L., Timofte, R.: Fourier space losses for efficient perceptual image super-resolution. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2360–2369 (2021)
14. Furukawa, Y., Ponce, J.: Accurate, dense, and robust multiview stereopsis. IEEE transactions on pattern analysis and machine intelligence **32**(8), 1362–1376 (2009)
15. Gross, M., Pfister, H.: Point-based graphics. Elsevier (2011)
16. Grossman, J.P., Dally, W.J.: Point sample rendering. In: Rendering Techniques' 98: Proceedings of the Eurographics Workshop in Vienna, Austria, June 29—July 1, 1998 9. pp. 181–192. Springer (1998)
17. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14. pp. 694–711. Springer (2016)

18. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics (TOG) **42**(4), 1–14 (2023)
19. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics (ToG) **36**(4), 1–13 (2017)
20. Levoy, M., Whitted, T.: The use of points as a display primitive (1985)
21. Li, Z., Li, L., Zhu, J.: Read: Large-scale neural scene rendering for autonomous driving. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 1522–1529 (2023)
22. Liu, L., Gu, J., Zaw Lin, K., Chua, T.S., Theobalt, C.: Neural sparse voxel fields. Advances in Neural Information Processing Systems **33**, 15651–15663 (2020)
23. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision **60**, 91–110 (2004)
24. Lu, C., Yin, F., Chen, X., Chen, T., Yu, G., Fan, J.: A large-scale outdoor multi-modal dataset and benchmark for novel view synthesis and implicit scene reconstruction. arXiv preprint arXiv:2301.06782 (2023)
25. Meshry, M., Goldman, D.B., Khamis, S., Hoppe, H., Pandey, R., Snavely, N., Martin-Brualla, R.: Neural rerendering in the wild. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6878–6887 (2019)
26. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: European Conference on Computer Vision. pp. 405–421 (2020)
27. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Transactions on Graphics (ToG) **41**(4), 1–15 (2022)
28. Pfister, H., Zwicker, M., Van Baar, J., Gross, M.: Surfels: Surface elements as rendering primitives. In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques. pp. 335–342 (2000)
29. Pittaluga, F., Koppal, S.J., Kang, S.B., Sinha, S.N.: Revealing scenes by inverting structure from motion reconstructions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 145–154 (2019)
30. Rakhimov, R., Ardelean, A.T., Lempitsky, V., Burnaev, E.: Npbg++: Accelerating neural point-based graphics. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15969–15979 (2022)
31. Ramon, E., Triginer, G., Escur, J., Pumarola, A., Garcia, J., Giro-i Nieto, X., Moreno-Noguer, F.: H3d-net: Few-shot high-fidelity 3d head reconstruction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5620–5629 (2021)
32. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18. pp. 234–241. Springer (2015)
33. Rückert, D., Franke, L., Stamminger, M.: Adop: Approximate differentiable one-pixel point rendering. ACM Transactions on Graphics (ToG) **41**(4), 1–14 (2022)
34. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4104–4113 (2016)
35. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
36. Sun, C., Sun, M., Chen, H.T.: Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5459–5469 (2022)
37. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1–9 (2015)

38. Tancik, M., Casser, V., Yan, X., Pradhan, S., Mildenhall, B., Srinivasan, P.P., Barron, J.T., Kretzschmar, H.: Block-nerf: Scalable large scene neural view synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8248–8258 (2022)
39. Tancik, M., Weber, E., Ng, E., Li, R., Yi, B., Wang, T., Kristoffersen, A., Austin, J., Salahi, K., Ahuja, A., et al.: Nerfstudio: A modular framework for neural radiance field development. In: ACM SIGGRAPH 2023 Conference Proceedings. pp. 1–12 (2023)
40. Tewari, A., Fried, O., Thies, J., Sitzmann, V., Lombardi, S., Sunkavalli, K., Martin-Brualla, R., Simon, T., Saragih, J., Nießner, M., et al.: State of the art on neural rendering. In: Computer Graphics Forum. vol. 39, pp. 701–727. Wiley Online Library (2020)
41. Thies, J., Zollhöfer, M., Nießner, M.: Deferred neural rendering: Image synthesis using neural textures. Acm Transactions on Graphics (TOG) **38**(4), 1–12 (2019)
42. Turki, H., Ramanan, D., Satyanarayanan, M.: Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12922–12931 (2022)
43. Wang, F., Galliani, S., Vogel, C., Speciale, P., Pollefeys, M.: Patchmatchnet: Learned multi-view patchmatch stereo. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 14194–14203 (2021)
44. Wang, P., Liu, Y., Chen, Z., Liu, L., Liu, Z., Komura, T., Theobalt, C., Wang, W.: F2-nerf: Fast neural radiance field training with free camera trajectories. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4150–4159 (2023)
45. Wang, X., Zhu, Z., Huang, G., Qin, F., Ye, Y., He, Y., Chi, X., Wang, X.: Mvster: Epipolar transformer for efficient multi-view stereo. In: European Conference on Computer Vision. pp. 573–591. Springer (2022)
46. Wei, Z., Zhu, Q., Min, C., Chen, Y., Wang, G.: Aa-rmvsnet: Adaptive aggregation recurrent multi-view stereo network. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6187–6196 (2021)
47. Wu, M., Wang, Y., Hu, Q., Yu, J.: Multi-view neural human rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1682–1691 (2020)
48. Yao, Y., Luo, Z., Li, S., Fang, T., Quan, L.: Mvsnet: Depth inference for unstructured multi-view stereo. In: Proceedings of the European conference on computer vision (ECCV). pp. 767–783 (2018)
49. Yeshwanth, C., Liu, Y.C., Nießner, M., Dai, A.: Scannet++: A high-fidelity dataset of 3d indoor scenes. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12–22 (2023)
50. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Free-form image inpainting with gated convolution. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4471–4480 (2019)
51. Zamir, S.W., Arora, A., Khan, S., Hayat, M., Khan, F.S., Yang, M.H.: Restormer: Efficient transformer for high-resolution image restoration. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5728–5739 (2022)
52. Zamir, S.W., Arora, A., Khan, S., Hayat, M., Khan, F.S., Yang, M.H., Shao, L.: Multi-stage progressive image restoration. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 14821–14831 (2021)
53. Zhang, J., Zhang, J., Mao, S., Ji, M., Wang, G., Chen, Z., Zhang, T., Yuan, X., Dai, Q., Fang, L.: Gigamvs: a benchmark for ultra-large-scale gigapixel-level 3d reconstruction. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(11), 7534–7550 (2021)
54. Zhang, K., Riegler, G., Snavely, N., Koltun, V.: Nerf++: Analyzing and improving neural radiance fields. arXiv preprint arXiv:2010.07492 (2020)

55. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 586–595 (2018)
56. Zhang, Y., Peng, S., Moazeni, A., Li, K.: Papr: Proximity attention point rendering. Advances in Neural Information Processing Systems **36** (2024)