Learning Robust Image-Based Rendering on Sparse Scene Geometry via Depth Completion

Yuqi Sun, Shili Zhou, Ri Cheng, Weimin Tan, Bo Yan*, Lang Fu
School of Computer Science, Shanghai Key Laboratory of Intelligent Information Processing,
Shanghai Collaborative Innovation Center of Intelligent Visual Computing, Fudan University, Shanghai, China
{yqsun20, slzhou19, rcheng20, wmtan, byan*}@fudan.edu.cn, ful21@m.fudan.edu.cn

Abstract

Recent image-based rendering (IBR) methods usually adopt plenty of views to reconstruct dense scene geometry. However, the number of available views is limited in practice. When only few views are provided, the performance of these methods drops off significantly, as the scene geometry becomes sparse as well. Therefore, in this paper, we propose Sparse-IBRNet (SIBRNet) to perform robust IBR on sparse scene geometry by depth completion. The SIBRNet has two stages, geometry recovery (GR) stage and light blending (LB) stage. Specifically, GR stage takes sparse depth map and RGB as input to predict dense depth map by exploiting the correlation between two modals. As inaccuracy of the complete depth map may cause projection biases in the warping process, LB stage first uses a bias-corrected module (BCM) to rectify deviations, and then aggregates modified features from different views to render a novel view. Extensive experimental results demonstrate that our method performs best on sparse scene geometry than recent IBR methods, and it can generate better or comparable results as well when the geometric information is dense.[1]

1. Introduction

Image-based rendering (IBR), as one of the classic approaches of novel view synthesis, aims to synthesize novel view from real views. It has been widely used to enhance the visualization in various applications, such as virtual navigation [15], video stabilization [24], AR\VR [37, 38]. A novel view is generated in IBR by warping pixels from source view into target view with scene geometry and aggregating them with blending methods. The output quality depends on the accuracy and completeness of geometry information and the effectiveness of blending strategy.

[1]This work is supported by NSFC (Grant No.: U2001209, 61902076) and Natural Science Foundation of Shanghai (21ZR1406600).
* Corresponding author: Bo Yan.

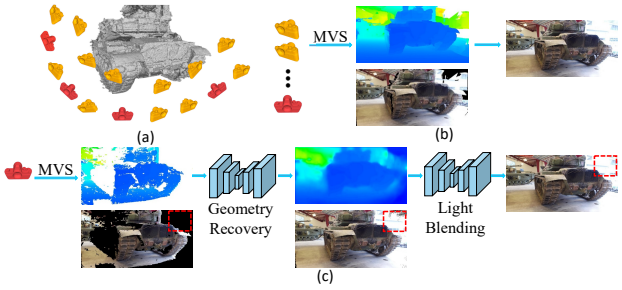

Figure 1. **Strategy Comparison.** (a) simulates dense view sampling on a tank model, while the red cameras represent sparse view sampling. (b) shows the strategy of recent IBR methods. They adopt dense views to generate a dense depth map for a better warped result (below the depth map) and then synthesize a novel view. (c) shows the paradigm of our SIBRNet. Given sparse views, a sparse depth map is generated by MVS before sent into a geometry recovery stage for completion. The complete depth map has a comparable warped result to the dense depth map. And then a light blending stage is applied to render the final result. Red boxes indicate the effect of two stages.

Recent IBR methods [1, 31, 32] ask for dense geometry to guarantee adequate correct projections. As shown in Figure 1.(b), they use plentiful views to reconstruct dense proxy geometry by multi-view stereo algorithm (MVS) [16,34,51]. This process is time-consuming and can only be conducted offline. Moreover, dense views are not available easily in practice. When input views are sparse, the scene geometry produced by MVS becomes sparse as well, leading to a rapid decline on performance for these methods.

In order to reduce the dependence on dense views, we propose SIBRNet to perform robust IBR on sparse scene geometry. It contains two stages, geometry recovery (GR) stage and light blending (LB) stage. Inspired by recent geometry recovery works like point cloud upsampling [21,54] and depth completion [25,46,56], the GR stage introduces a learning-based depth completion network into IBR to predict a complete depth map from the sparse one. As shown

Figure 2. **Visual comparisons with recent IBR methods on Tanks and Temples dataset**. FVS [31] and SVS [32] will produce a blur in the area where the input sparse depth map misses, while our method can still synthesize a photorealistic result.

in Figure 1.(c), given a sparse depth map, the GR stage can ensure a comparable warped result to the dense depth map. The LB stage is used to aggregate different source views and synthesize a novel view. In this way, our method can synthesize much better results than recent IBR methods on sparse scene geometry. In addition, although our method focuses on sparse geometry input, it can also achieve better or comparable results when the input geometry is dense.

Specifically, the GR stage takes a sparse depth map and the associated RGB image as input to predict a complete depth map. In order to take advantage of the correlation between depth and color, we design two subnetworks to extract different information. The final complete depth map is generated by fusing the results from two subnetworks. The LB stage utilizes the complete depth maps to warp source view feature maps into target view through a 3D warping. As the inaccurate depth value will produce projection biases, we design a bias-corrected module (BCM) to rectify these warped features. After that, a Bi-ConvLSTM is applied for aggregating information from different view points and synthesizing a candidate image and a confidence map for each view. Finally, a novel view is generated by blending all candidate images with a softmax.

We also propose a new dataset called Surround to evaluate IBR methods in a surround setting. In addition, in order to train and evaluate our method on sparse scene geometry, we preprocess two public datasets, Tanks and Temples, Free View Synthesis, and our Surround dataset to generate depth maps at different sparsity levels. Quantitative and qualitative experiments on the three datasets show that our method is robust on different depth sparsity. It can generate more realistic results, especially on sparse scene geometry, as shown in Figure 2. Our method has obvious advantages in the depth missing area. Our code and dataset will be released here.

In summary, this paper has the following contributions:

- In order to reduce the dependence of existing methods on dense input views, we propose a two-stage model named SIBRNet to perform robust IBR on sparse scene geometry by introducing a learning-based depth completion network for the first time. It is robust for scene geometry on different sparsity levels.
- The inaccurate complete depth value will result in projection biases during the warping process, which may cause image distortions in the final result. Therefore, we design a bias-corrected module (BCM) using deformable convolution to rectify these deviations.
- A new dataset called Surround is proposed for evaluating. It contains a 360-degree panorama for each scene and is useful to evaluate IBR methods in a surround setting.

2. Related Work

Traditional image-based rendering methods. IBR has a long history in the field of computer vision and graphics. Early classical works [8,14,20,22,26,42] focus on light field and view interpolation, which require regularly-distributed camera array and small-baseline view. For wide-baseline situation, most methods [2,3,9,15,19] reconstruct dense 3D proxy geometry in different ways to guarantee good results. Some methods [3,9] compensate for poor 3D geometry with depth synthesis algorithm or optical flow, which share the same idea with us. However, our method conducts it in a deep-learning manner.

Leaning-based image-based rendering methods. In recent years, deep learning methods has achieved an appealing effect in IBR. Flynn *et al*. [13] firstly apply convolutional neural network (CNN) within plane-sweep volumes in view interpolation. Some methods [30, 43, 58] use implicit geometry and image generation methods to synthesize novel views. These methods only focus on single object or narrow-baseline imagery.

Scene representations are widely used to understand scene geometry explicitly or implicitly in novel view synthesis methods lately. Explicit methods use discrete representation to describe scene geometry, such as multi-plane images (MPI) [12, 28, 41, 57], layered depth images (LDI) [39,41,45], point cloud [49]. They usually require extensive memory and computational costs. Implicit methods consider deep networks as an implicit function of scene geometry to achieve a continuous representation. NeRF [29] and its variants [17,27,55] achieve impressive result by applying a 5D radiance field to estimate both geometry and appear-

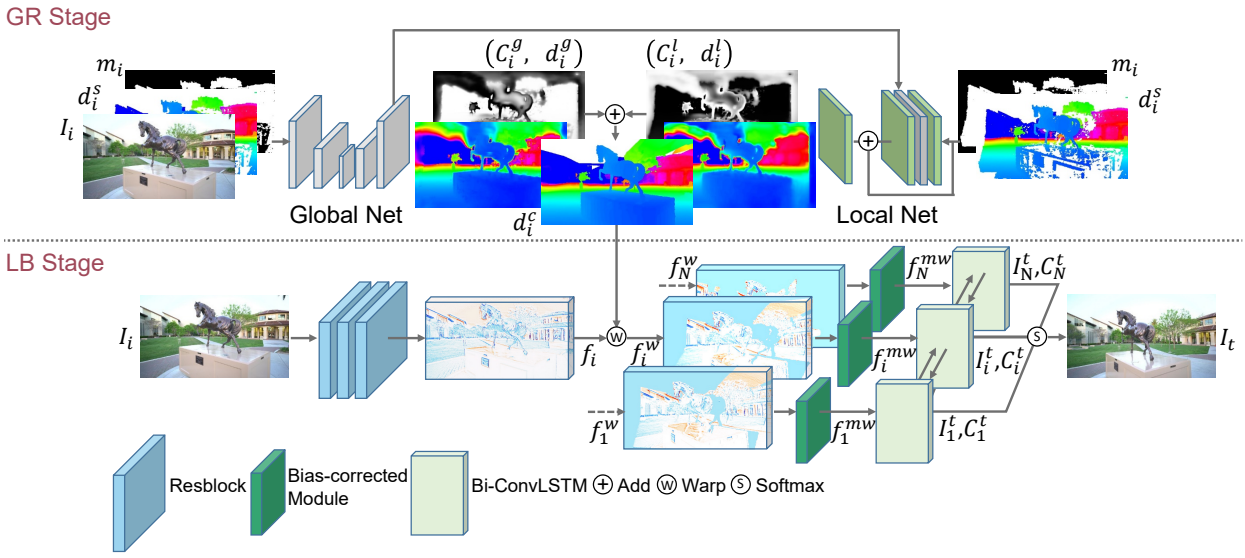

Figure 3. **Pipeline of SIBRNet**. The GR stage generates a complete depth map d_i^c for each view I_i by fusing the results from two branches, the global and local nets. The LB stage first extracts feature maps $\{f_i\}_{i=1}^N$ from N source views, and then warped them into target view $\{f_i^w\}_{i=1}^N$ with per-view complete depth maps $\{d_i^c\}_{i=1}^N$. As the inaccurate depth value will cause projection biases, all features are modified $\{f_i^{mw}\}_{i=1}^N$ by a bias-corrected module (BCM) and blended in a Bi-ConvLSTM to synthesize a candidate view I_i^t and a confidence map C_i^t for each view. Finally, we use a softmax fusing each view to render the final result I_t.

ance. Some latest works [44, 47, 53] extend NeRF to unseen data and sparse inputs. However, they only work well on synthetic objects or small sliding view of real scenes, while our method focuses on more complex and open scenario.

Following the traditional manner, some recent IBR methods reconstruct 3D proxy geometry by MVS algorithms [16, 51] and then blend pixels in deep networks. Riegler and Koltun [31, 32] use hundreds of images to generate dense depth maps by COLMAP [34]. Their methods can produce high-quality images, but require dense views for reconstructing dense proxy geometry. In another way, Choi *et al*. [6] estimate depth probability for source views by DeepMVS [16] and Shi *et al*. [36] introduce MVS module directly into the network for end-to-end training. Although their methods allow sparse views, they require huge memory costs due to the complexity and are not suitable for high-resolution (HR) input. In contrast, our method can achieve realistic results on sparse scene geometry by introducing a depth completion network. It reduces the number of required views and allows HR images at the same time.

Depth completion. The main purpose of depth completion task is to compensate poor depth captured by low-cost LiDAR and commercial RGB-D camera. Depending on whether there is an RGB as input, depth completion methods can be divided into two categories, depth-only methods [5, 10, 23] and image-guided methods [11, 25, 46, 56]. Since an RGB can provide strong prior on semantic and edge information, image-guided methods usually have a better result. Inspired by these works, our method applies depth completion to sparse depth maps reconstructed from sparse views. The complete depth maps can produce comparable warped results to dense depth maps.

3. Method

In this section, we present our method, SIBRNet. We begin with a dataset preprocessing step that prepares depth maps at different sparsity levels in Section 3.1. Then in Section 3.2 and Section 3.3, the GR stage and LB stage in SIBRNet are described in details. At last, loss functions and implementation details are shown in Section 3.4 and Section 3.5. Full pipeline of SIBRNet is shown in Figure 3.

3.1. Preprocessing

In order to train and evaluate our method on sparse scene geometry, we use COLMAP to generate depth maps at different sparsity levels. We first follow Riegler and Koltun [31] to estimate camera poses and dense depth maps from all source views by the structure-from-motion (SfM) and MVS method implemented in COLMAP. Since the generated depth maps become sparse when the number of input images reduces, we use the input number K to divide the depth sparsity levels. After setting K, we sample uniformly in source images and sent them into COLMAP to generate a sparse depth map at level K for each view. Specifically, we set K as 4 and 8. The dense depth maps obtained from all source views are considered as ground-truth, with K = all.

We show the visualization of different depth maps in Figure 4.(a) when K=4, 8 and all.

3.2. Geometry-Recovery Stage

Early work [46, 56] in image-guided depth completion shows that RGB and depth include different cues for completion. RGB can provide image structure and semantics, while depth can provide details like edges. Inspired by this idea, we design two subnetworks to extract different information separately. The global net takes an RGB image I_i, its sparse depth map d_i^s and valid depth mask m_i as input to generate a complete depth map d_i^g and a confidence map C_i^g. In order to extract color information, it follows a U-Net [33] structure with several skip connections. Moreover, we take a deep feature from the global net to guide the local net. We express the global net as:

$$C_i^g, d_i^g, f_{global} = GlobalNet(I_i, d_i^s, m_i) \tag{1}$$

The local net only requires a sparse depth map d_i^s and a valid depth mask m_i. It uses a residual-add design to preserve origin depth as much as possible, which is shown in Figure 3 "Local Net". The guidance from the global net is used to ensure a large receptive field. The local net also generates a complete depth map d_i^g and a confidence map C_i^g:

$$C_i^l, d_i^l = LocalNet(d_i^s, m_i, f_{global}) \tag{2}$$

The final complete depth map d_i^c is generated by adding d_i^g and d_i^l with weights, which is obtained by a softmax from C_i^g and C_i^l. The process can be described as:

$$d_i^c = w_i^g * d_i^g + w_i^l * d_i^l. \tag{3}$$

where $w_i^g, w_i^l = softmax(C_i^g, C_i^l)$. Illustrations for two confidence maps C_i^g and C_i^l in Figure 3 "GR Stage" show that the global net and local net focus on different areas in depth completion. The global net pays more attention to scene structure like the trunk area of the horse, while the local net concerns high-frequency details like the limbs of the horse.

3.3. Light-Blending Stage

Each pixel in a 2D image can be regarded as a light ray captured from the observed scene's light field. Our LB stage takes N source views $\{I_i\}_{i=1}^N$ as input to blend lights from different views and synthesize a novel view I_t. Three operations, warp, bias-correction and blending, are executed in sequence.

Warp. We first use a feature extractor with several residual blocks to extract feature maps $\{f_i\}_{i=1}^N$ for each source view. And then we take the per-view feature and the complete depth map obtained from GR stage to perform a 3D warping. 3D warping implements transformation of 2D pixel coordinates by projecting them into identical locations in 3D space. Taking depth D, camera intrinsic K and extrinsic $[R|t]$ (R and t refer to rotation and translation matrices) as input, the 3D warping is conducted following the equation:

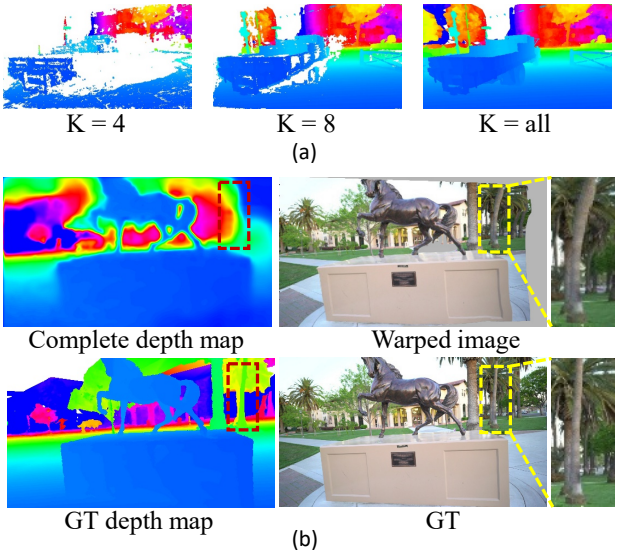

Figure 4. **Illustration for depth maps and image distortions**. (a) shows depth maps at different sparsity levels when K = 4, 8 and all. (b) shows image distortions caused by inaccurate complete depth. The red box in the complete depth map shows an inaccurate area. It causes projection biases during the 3D warping process, leading to distortions in the yellow box of the warped image.

$$K_s^{-1} D_s(p_s) * p_s = R_r K_t^{-1} D_t(p_t) * p_t + t_r,$$
$$\text{where } R_r = R_s R_t^T, t_r = t_s - R_r t_t. \tag{4}$$

p is the homogeneous coordinate of a 2D pixel, and s, t represent source view and target view. For each source view I_i, we use the complete depth map d_i^c to project its feature f_i into the target view directly by forward warping. Through this process, we obtain warped features f_i^w. We visualize the feature maps before and after warping, as shown in Figure 3 "LB Stage". It is clear to see that $\{f_i\}_i^N$ and $\{f_i^w\}_i^N$ are aligned with the source views and the target view respectively.

Bias-corrected module. Ideally, the warped features $\{f_i^w\}_i^N$ are strictly aligned with the pixels in target view. However, the inaccuracy of predicted depth values will cause projection biases in the warped features. It is critical to remove these biases because the misaligned feature may lead to content distortions shown in Figure 4.(b). Since the tree in the background is missing in the complete depth map, it bends incorrectly in the warped image. To address

this problem, we adopt a bias-corrected module (BCM). As the input sparse depth map d_i^s has accurate values in the valid area, we utilize it to obtain a accurate sparse-area feature f_i^{sw} and a corresponding bias-potential mask M_i^{sw} by 3D warping. After that, we calculate an offset Δp_i from the source feature f_i and the warped feature f_i^w in the bias-potential area with several convolution layers:

$$\Delta p_i = Conv(f_i, f_i^w, M_i^{sw}) \tag{5}$$

The offset and warped features $\{f_i^w\}_i^N$ are then input into deformable convolutions [7,59] to generate a revised feature f_i^{dw}.

$$f_i^{dw} = DConv(f_i^w, \Delta p_i), \tag{6}$$

Finally, we aggregate the sparse accurate warped feature f_i^{sw} and the revised feature f_i^{dw} with the sparse warped mask M_i^{sw} to create the final modified warped feature f_i^{mw}:

$$f_i^{mw} = M_i^{sw} * f_i^{sw} + (1 - M_i^{sw}) * f_i^{dw}. \tag{7}$$

Blending. Following [31], we try to use a recurrent neural network (RNN) to establish connections between N source views. However, a simple single-directional RNN is not appropriate for IBR because there is no strict order in source input views. Therefore, we apply a bidirectional ConvLSTM (Bi-ConvLSTM) [35, 50] for blending. Empirically, it is more robust for arbitrary input orders. The Bi-ConvLSTM takes N modified warped features $\{f_i^{mw}\}_i^N$ as input to aggregate information from different source views, and then generates a candidate image I_i^t and a confidence map C_i^t for each view:

$$\{I_i^t, C_i^t\}_{i=1}^N = Bi\text{-}ConvLSTM(f_1^{mw}, ..., f_N^{mw}) \tag{8}$$

And then we calculate weights $\{w_i\}_{i=1}^N$ from confidence maps $\{C_i^t\}_{i=1}^N$ through softmax. The final result I_t is produced by fusing all candidate images with a weighted add $I_t = \sum_{i=1}^N w_i * I_i^t$.

3.4. Loss Functions

We train GR stage and LB stage with different losses for depth and color.

Loss in GR stage. The global net and local net in GR stage will produce two different depth maps, d_i^g and d_i^l. And then the final complete depth map d_i^c is generated by fusing them together. We use the same depth L2 Loss $\mathcal{L}_d$ for the three depth maps above. As the range of depth in different scenes vary greatly, we normalize the L2 loss by dividing the ground-truth depth d_i^t, described as:

$$\mathcal{L}_d(d_i) = \frac{\|d_i - d_i^t\|_2}{d_i^t} \tag{9}$$

Moreover, inspired by recent monocular depth estimation work [52], we add an edge-aware depth smooth loss to the final result d_i^c using the RGB image I_i to preserve sharp edges.

$$\mathcal{L}_s(d_i) = \nabla|d_i| \cdot exp(\text{-}|\nabla I_i|) \tag{10}$$

where $|\cdot|$ means absolute value and ∇ means differential operator. The total loss in GR stage is:

$$\mathcal{L}_{GR} = \mathcal{L}_d(d_i^g) + \mathcal{L}_d(d_i^l) + \mathcal{L}_d(d_i^c) + \mathcal{L}_s(d_i^c) \tag{11}$$

Loss in LB stage. The LB stage uses N source views $\{I_i\}_{i=1}^N$ to synthesize a novel view I_t. The image reconstruction loss consists of a L1 loss and a perceptual loss [4]. They are used for pixel-level and feature-level supervision separately. Given the predicted image I_t and ground-truth image I_t^g, the loss is:

$$\mathcal{L}_{LB} = \|I_t - I_t^g\|_1 + \sum_l \lambda_l \|\Phi_l(I_t) - \Phi_l(I_t^g)\|_1 \tag{12}$$

Φ_l denotes the outputs of middle layers of a pretrained VGG-19 network [40]. The weights λ_l are set as in [31].

3.5. Implementation Details

The GR stage and LB stage in our SIBRNet are trained separately. We train the GR stage firstly. The global net in GR stage is a U-Net structure with four stages and each stage has two convolution layers followed by an average pooling. The local net has two down-sample layers consists of four residual blocks and two transposed convolutions for up-sampling. We train GR stage with 60 epochs and batch size is set as 10. After that, we fix the GR stage and train the LB stage. The feature extractor consists of 16 residual blocks and our BCM module adopts a pyramid cascading deformable (PCD) module [48]. We train LB stage with 40 epochs with batch size as 1. The number of input source views N is set as 5. In both stages, we use Adamax optimizer and set learning rate as 1e-4, patch size as 256×256. We train all the networks on an NVIDIA RTX 3090.

4. Experiments and Analysis

4.1. Experimental Settings

Public Datasets. Two public datasets, Tanks and Temples [18] and Free View Synthesis [31], are used for training or evaluation. We preprocess them as described in Section 3.1. We train our method on Tanks and Temples, following Riegler and Koltun [31]. It has 21 scenes. 17 scenes are used as training dataset and 4 scenes are selected out as testing dataset. Free View Synthesis dataset is a testing dataset. It contains 6 scenes, and each scene provides a source image sequence and a target image sequence. For simplicity, we only use the target image sequence for evaluation.

Surround dataset. Recent applications for IBR focus on circle-around scenarios, such as basketball and soccer stadiums, they wish to achieve a smooth circular view movement. Therefore, we propose a new dataset called Surround

Method	Input	K	Train			Playground			M60			Truck		
			↑ PSNR	↑ SSIM	↓ LPIPS	↑ PSNR	↑ SSIM	↓ LPIPS	↑ PSNR	↑ SSIM	↓ LPIPS	↑ PSNR	↑ SSIM	↓ LPIPS
FVS [31]	5	4	18.96	0.6688	0.3001	21.54	0.6711	0.2758	19.25	0.7132	0.2917	20.18	0.7013	0.2416
SVS [32]			17.34	0.6638	0.3769	19.70	0.6683	0.3530	17.22	0.6917	0.3983	19.90	0.7279	0.2858
Ours			**22.54**	**0.7549**	**0.1374**	**25.00**	**0.7681**	**0.1287**	**23.92**	**0.8162**	**0.1210**	**22.99**	**0.7799**	**0.1194**
FVS [31]	5	8	21.25	0.7433	0.1899	24.88	0.7875	0.1270	23.61	0.8259	0.1419	22.38	0.7700	0.1355
SVS [32]			20.28	0.7850	0.2228	25.31	0.8782	0.1094	22.86	0.8544	0.1734	23.64	0.8479	0.1437
Ours			**23.44**	**0.7905**	**0.1161**	**26.50**	**0.8234**	**0.0922**	**26.08**	**0.8704**	**0.0846**	**23.75**	**0.8092**	**0.0992**
EVS [6]	5	-	20.53	0.6795	0.1585	23.87	0.7558	0.1018	22.68	0.7884	0.1168	19.97	0.6419	0.1891
SVNVS [36]	6	-	20.43	0.6512	0.2125	22.43	0.6968	0.1451	22.36	0.7817	0.1346	21.42	0.7142	0.1439
Our	5	4	**22.54**	**0.7549**	**0.1374**	**25.00**	**0.7681**	**0.1287**	**23.92**	**0.8162**	**0.1210**	**22.99**	**0.7799**	**0.1194**

Table 1. **Quantitative comparisons on Tanks and Temples dataset**. "Input" and "K" denotes the number of input source views and depth sparsity levels, respectively. We show the best results in bold.

Figure 5. **Qualitative comparisons on Tanks and Temples dataset when** $K = 4$. FVS [31] and SVS [32] perform badly on sparse scene geometry. EVS [6] misses image content, and SVNVS [36] cause obvious color changes. Our method achieves the best realistic results.

Method	Input	K	Total		
			↑ PSNR	↑ SSIM	↓ LPIPS
FVS [31]	5	4	26.46	0.8454	0.0924
SVS [32]			26.52	0.8798	0.1273
Ours			**29.19**	**0.8880**	**0.0645**
FVS [31]	5	8	26.91	0.8510	0.0857
SVS [32]			27.55	**0.8980**	0.1097
Ours			**29.23**	0.8891	**0.0636**
EVS [6]	5	-	27.31	0.8600	0.0686
SVNVS [36]	6	-	24.94	0.8175	0.1151
Ours	5	4	**29.19**	**0.8880**	**0.0645**

Table 2. **Quantitative comparisons on Free View Synthesis**.

for evaluating IBR methods in surround setting. By taking a handheld camera to shoot a scene in a circle, we can capture a 360-degree video around the scene. And then we uniformly sample this video to extract source views. Surround contains 6 scenes, *Basketball*, *Meetingroom*, *Park*, *Philosopher*, *Soccer* and *Statue*. The *Meetingroom* is an indoor scene, while the others are outdoor scenes, and each scene has 150 to 300 images. We use COLMAP to estimate camera poses, depth maps and 3D point clouds. We described this dataset in our supplementary materials in detail.

View selection. In both training and testing, we choose one image as target and select N nearby images as source views.

4.2. Comparisons with State-of-the-Art

We compare our method with four recent state-of-the-art (SOTA) IBR methods, FVS [31], SVS [32], EVS [6] and SVNVS [36]. FVS [31] and SVS [32] use hundreds of views to estimate dense depth maps. When depth maps become sparse, their methods result in a blur in the region where depth misses. EVS [6] and SVNVS [36] can be used in sparse input views. However, they require huge computation and memory costs due to complexity and are not suitable for HR images. We reduce the resolution of input images to about 250×500 for evaluating them. For fair comparison, we retrain FVS [31] and SVS [32] in the same pre-processed Tanks and Temples dataset. For EVS [6] and SVNVS [36], we use the provided pretrained model.

Quantitative and qualitative comparisons on Tanks and Temples dataset are shown Table 1 and Figure 5, where "Input" denotes the input number of each method, and "K" represents the number of images used to reconstruct proxy geometry as described in Section 3.1. Our method achieves significant superiority on sparse scene geometry.

We conduct more careful experiments on Tanks and Temples dataset. Instead of using K, we directly calculate a valid depth ratio to represent the depth sparsity level. The valid depth ratio is the percentage of number of pixels

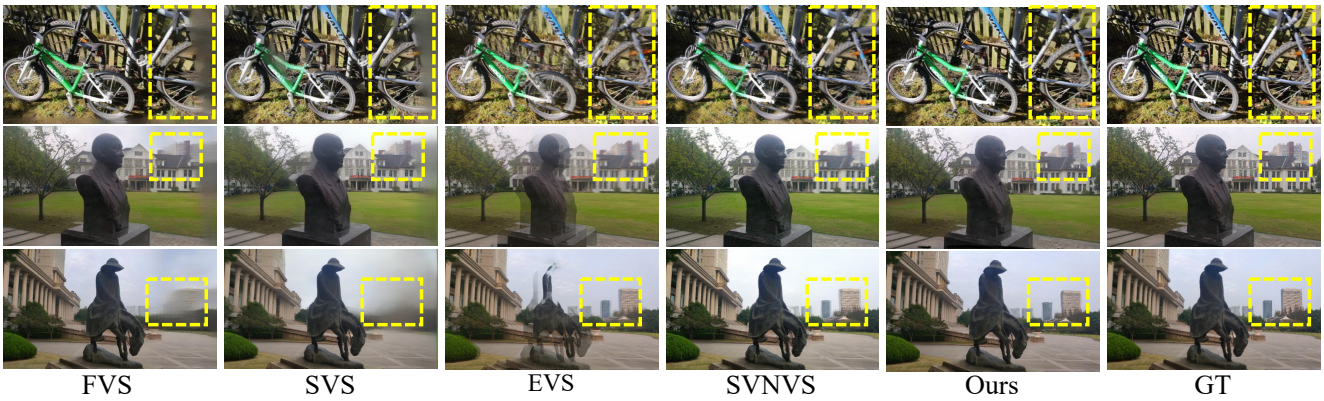

Figure 6. **Qualitative results on Free View Sythesis and Surround dataset when K = 4**. The first row show results on Free View Synthesis dataset, while the others are on Surround dataset. FVS [31], SVS [32] will produce blur. EVS [6] and SVNVS [36] produce artifacts and color change. Our method generates results close to ground-truth.

Method	Input	K	Basketball		Meetingroom		Park		Philosopher		Soccer		Statue	
			↑ PSNR	↓ LPIPS	↑ PSNR	↓ LPIPS	↑ PSNR	↓ LPIPS	↑ PSNR	↓ LPIPS	↑ PSNR	↓ LPIPS	↑ PSNR	↓ LPIPS
FVS [31]	5	4	26.19	0.0774	25.97	0.0650	27.29	0.0861	26.84	0.1274	24.61	0.1568	26.92	0.1152
SVS [32]			26.76	0.0687	24.57	0.1751	26.59	0.1337	26.00	0.1873	23.23	0.2549	26.37	0.1625
Ours			**28.46**	**0.0588**	**27.63**	**0.0500**	**28.27**	**0.0688**	**28.79**	**0.0946**	**26.11**	**0.1129**	**29.23**	**0.0770**
FVS [31]	5	8	26.47	0.0726	26.32	0.0611	27.68	0.0776	27.12	0.1238	25.06	0.1338	27.44	0.1000
SVS [32]			27.32	0.0612	25.78	0.1356	27.75	0.0917	26.59	0.1749	24.86	0.1922	27.70	0.1252
Ours			**28.37**	**0.0600**	**27.66**	**0.0502**	**28.32**	**0.0682**	**28.68**	**0.0951**	**26.15**	**0.1114**	**29.26**	**0.0763**
EVS [6]	5	-	25.64	0.0684	24.49	0.1175	27.57	0.0837	27.43	0.0912	24.41	0.1152	27.87	0.0722
SVNVS [36]	6	-	24.27	0.0890	24.55	0.1066	24.09	0.1334	24.37	0.1133	23.59	0.1425	24.78	0.1058
Ours	5	4	**28.46**	**0.0588**	**27.63**	**0.0500**	**28.27**	**0.0688**	**28.79**	**0.0946**	**26.11**	**0.1129**	**29.23**	**0.0770**

Table 3. **Quantitative comparisons on Surround dataset**. Our result performs best in all 6 scenes.

Method	Input	K	Tanks and Temples		Free View Synthesis		Surround	
			↑ PSNR	↓ LPIPS	↑ PSNR	↓ LPIPS	↑ PSNR	↓ LPIPS
FVS [31]	5	all	24.77	0.0907	27.71	0.0689	27.16	0.0862
SVS [32]			**26.25**	**0.0688**	29.07	0.0780	27.69	0.1054
Ours			25.48	0.0860	**29.33**	**0.0619**	**28.18**	**0.0768**

Table 4. **Quantitative comparisons when K = all**. When depth map is dense, our method also has better or comparable results.

with valid depth values to the total pixel number. We use more sampling strategies and different K to generate different valid depth ratios and show the result in Figure 7. When the valid depth ratio becomes small, the performance of FVS [31] and SVS [32] drops off rapidly, while our method keeps high performances with only a slight drop.

Table 2 shows comparison on Free View Synthesis dataset. For simplicity, we only show total results, and the detailed results on 6 scenes are provided in the supplementary material. We also evaluate these methods on our proposed Surround dataset in Table 3. Qualitative comparisons are in Figure 6. FVS [31] and SVS [32] lose image content and details, while EVS [6] and SVNVS [36] will cause artifacts and color changes. In contrast, our method can synthesize more realistic results.

Though our method focuses on sparse scene geometry, it can also achieve better or comparable results than FVS [31] and SVS [32] on dense scene geometry. We show the comparison results on three datasets in Table 4. When the input depth map is dense as K = all, our method performs best in Free View Synthesis and Surround datasets and only performs slightly worse than SVS [32] in Tanks and Temples.

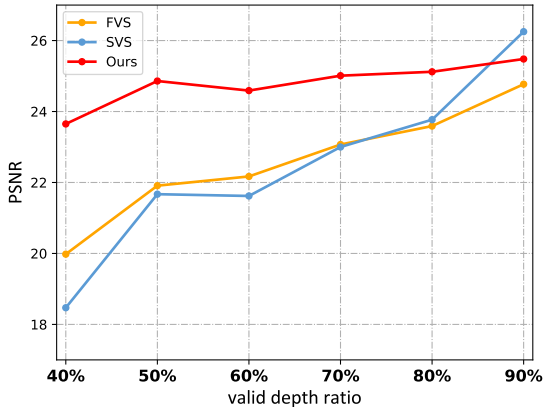

Figure 7. **PSNR comparison at different depth sparsity levels on Tanks and Temples dataset**. The performance of FVS [31] and SVS [32] drops off rapidly when input depth map becomes sparse, while our method is able to maintain PSNR at a high level.

	Train			Playground			M60			Truck		
	↑ PSNR	↑ SSIM	↓ LPIPS	↑ PSNR	↑ SSIM	↓ LPIPS	↑ PSNR	↑ SSIM	↓ LPIPS	↑ PSNR	↑ SSIM	↓ LPIPS
SIBRNet w/o BCM	20.80	0.7427	0.1553	23.81	0.7608	0.1385	22.18	0.8066	0.1325	21.42	0.7709	0.1330
SIBRNet w/o GR stage	21.31	0.7330	0.1888	24.02	0.7511	0.1706	22.15	0.7847	0.1924	22.27	0.7627	0.1527
SIBRNet w/o Bi-ConvLSTM	21.74	0.7380	0.1545	24.48	0.7555	0.1426	23.56	0.8051	0.1332	22.60	0.7678	0.1330
SIBRNet	**22.54**	**0.7549**	**0.1374**	**25.00**	**0.7681**	**0.1287**	**23.92**	**0.8162**	**0.1210**	**22.99**	**0.7799**	**0.1194**

Table 5. **Quantitative results of ablation study**. By disabling each component of SIBRNet, we show its impact on the final result. GR stage and BCM has a great influence on LPIPS and PSNR respectively. Our full model achieves the best performance.

Figure 8. **Qualitative results of ablation study**. SIBRNet w/o GR stage will miss image content, resulting in a blur result, while the SIBRNet w/o BCM will cause distortions. SIBRNet w/o Bi-ConvLSTM will also lose some details. The full SIBRNet can synthesize the most realistic novel view.

Geometric / Photometric error	Train	Playground	M60	Truck
Sparse depth	27.01/0.0567	32.24/0.0574	31.05/0.0728	31.85/0.0531
GR stage w/o Local Net	14.05/0.0394	10.96/0.0319	10.66/**0.0380**	11.09/0.0337
GR stage w/o Global Net	13.34/0.0391	11.97/0.0319	11.51/0.0396	10.88/0.0341
GR stage	**12.83/0.0389**	**9.21/0.0316**	**10.54**/0.0383	**8.29/0.0332**

Table 6. **Ablation study on GR stage**. GR stage reduces the geometric and photometric error greatly compared to sparse depth. And the separate global and local net will drop off the performance.

4.3. Ablation Study

We conduct the ablation study on Tanks and Temples dataset with K as 4. Quantitative and qualitative results are shown in Table 5 and Figure 8. Removing any component will lead to a significant drop in performance.

GR stage. GR stage generates a dense depth map from the sparse one to ensure more warped pixels. In GR stage, the global net is used to learn semantics, while the local net focus on details. We remove each of them and show the geometric and photometric error changes in Table 6. The geometric error is a normalized L2 loss described in equation 9, while the photometric error is a L1 loss between the image warped by the complete depth and the ground-truth depth. In Table 5, the LPIPS increases a lot when we remove the GR stage. The reason is that an image warped by a sparse depth map has lots of invisible pixels, which will lead to a blur result, as shown in Figure 8"w/o GR stage". This has more severe impacts on LPIPS than on PSNR.

BCM and Bi-ConvLSTM. BCM module can rectify the projection biases caused by some inaccurate depth values predicted by GR stage. Without BCM, biases in the warped feature will produce content distortions, such as the pillar in Figure 8 "w/o BCM". These misaligned pixels preserve some semantic information, so the LPIPS does not change a lot, but they result in a great descent on PSNR, as shown in Table 5. Bi-ConvLSTM is used to aggregate information from different views, which can preserve image details. By removing Bi-ConvLSTM, the final result will lose clear edges, as shown in Figure 8 "w/o Bi-ConvLSTM".

5. Limitation

The GR stage plays an important role in our SIBRNet, it generates a complete depth map from the sparse one to guarantee more visible pixels in the LB stage. The accuracy and completeness of the complete depth map is critical for the final synthesized result. Our GR stage can recover dense depth information of global scene structure, but may lose some details, such as thin objects like trees or pillars. These objects are very small and hard to distinguish from the background, which makes it difficult to complete or predict the depth in these areas. It will be our future work to propose an improved depth completion network for this problem.

6. Conclusion

In this paper, we propose a two-stage model named SIBRNet to perform IBR on sparse scene geometry by introducing a learning-based depth completion network for the first time. It is robust for scene geometry at different sparsity levels and generate better or comparable results than recent IBR methods whether the input depth map is sparse or dense. As the inaccuracy of complete depth maps will cause projection biases, which may result in image distortions, we design a bias-corrected module (BCM) to remove these biases for ensuring realistic results. We also propose a new dataset Surround, which is useful to evaluate IBR methods in a surround setting.

References

[1] Peter , Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (TOG)*, 37(6):1–15, 2018. 1

[2] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 425–432, 2001. 2

[3] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics (TOG)*, 32(3):1–12, 2013. 2

[4] Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1511–1520, 2017. 5

[5] Nathaniel Chodosh, Chaoyang Wang, and Simon Lucey. Deep convolutional compressed sensing for lidar depth completion. In *Asian Conference on Computer Vision*, pages 499–513. Springer, 2018. 3

[6] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7781–7790, 2019. 3, 6, 7

[7] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. 5

[8] Abe Davis, Marc Levoy, and Fredo Durand. Unstructured light fields. In *Computer Graphics Forum*, volume 31, pages 305–314. Wiley Online Library, 2012. 2

[9] Martin Eisemann, Bert De Decker, Marcus Magnor, Philippe Bekaert, Edilson De Aguiar, Naveed Ahmed, Christian Theobalt, and Anita Sellent. Floating textures. In *Computer graphics forum*, volume 27, pages 409–418. Wiley Online Library, 2008. 2

[10] Abdelrahman Eldesokey, Michael Felsberg, and Fahad Shahbaz Khan. Propagating confidences through cnns for sparse data regression. *arXiv preprint arXiv:1805.11913*, 2018. 3

[11] Abdelrahman Eldesokey, Michael Felsberg, and Fahad Shahbaz Khan. Confidence propagation through cnns for guided sparse depth regression. *IEEE transactions on pattern analysis and machine intelligence*, 42(10):2423–2436, 2019. 3

[12] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2367–2376, 2019. 2

[13] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5515–5524, 2016. 2

[14] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 43–54, 1996. 2

[15] Peter Hedman, Tobias Ritschel, George Drettakis, and Gabriel Brostow. Scalable inside-out image-based rendering. *ACM Trans. Graph.*, 35(6), Nov. 2016. 1, 2

[16] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2821–2830, 2018. 1, 3

[17] Petr Kellnhofer, Lars C Jebe, Andrew Jones, Ryan Spicer, Kari Pulli, and Gordon Wetzstein. Neural lumigraph rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4287–4297, 2021. 2

[18] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017. 5

[19] Johannes Kopf, Michael F Cohen, and Richard Szeliski. First-person hyper-lapse videos. *ACM Transactions on Graphics (TOG)*, 33(4):1–10, 2014. 2

[20] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42, 1996. 2

[21] Ruihui Li, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-gan: a point cloud upsampling adversarial network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7203–7212, 2019. 1

[22] Christian Lipski, Christian Linz, Kai Berger, Anita Sellent, and Marcus Magnor. Virtual video camera: Image-based viewpoint navigation through space and time. In *Computer Graphics Forum*, volume 29, pages 2555–2568. Wiley Online Library, 2010. 2

[23] Lee-Kang Liu, Stanley H Chan, and Truong Q Nguyen. Depth reconstruction from sparse samples: Representation, algorithm, and sampling. *IEEE Transactions on Image Processing*, 24(6):1983–1996, 2015. 3

[24] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Bundled camera paths for video stabilization. *ACM Transactions on Graphics (TOG)*, 32(4):1–10, 2013. 1

[25] Fangchang Ma and Sertac Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 4796–4803. IEEE, 2018. 1, 3

[26] Dhruv Mahajan, Fu-Chung Huang, Wojciech Matusik, Ravi Ramamoorthi, and Peter Belhumeur. Moving gradients: a path-based method for plausible image interpolation. *ACM Transactions on Graphics (TOG)*, 28(3):1–11, 2009. 2

[27] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021. 2

[28] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 2

[29] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer, 2020. 2

[30] Eunbyung Park, Jimei Yang, Ersin Yumer, Duygu Ceylan, and Alexander C Berg. Transformation-grounded image generation network for novel 3d view synthesis. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 3500–3509, 2017. 2

[31] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *European Conference on Computer Vision*, pages 623–640. Springer, 2020. 1, 2, 3, 5, 6, 7

[32] Gernot Riegler and Vladlen Koltun. Stable view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12216–12225, 2021. 1, 2, 3, 6, 7

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 4

[34] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 1, 3

[35] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997. 5

[36] Yujiao Shi, Hongdong Li, and Xin Yu. Self-supervised visibility learning for novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9675–9684, 2021. 3, 6, 7

[37] Yujiao Shi, Liu Liu, Xin Yu, and Hongdong Li. Spatial-aware feature aggregation for image based cross-view geo-localization. *Advances in Neural Information Processing Systems*, 32:10090–10100, 2019. 1

[38] Yujiao Shi, Xin Yu, Liu Liu, Tong Zhang, and Hongdong Li. Optimal feature transport for cross-view image geo-localization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11990–11997, 2020. 1

[39] Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. 3d photography using context-aware layered depth inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8028–8038, 2020. 2

[40] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[41] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 175–184, 2019. 2

[42] Timo Stich, Christian Linz, Christian Wallraven, Douglas Cunningham, and Marcus Magnor. Perception-motivated interpolation of image sequences. *ACM Transactions on Applied Perception (TAP)*, 8(2):1–25, 2011. 2

[43] Shao-Hua Sun, Minyoung Huh, Yuan-Hong Liao, Ning Zhang, and Joseph J Lim. Multi-view to novel view: Synthesizing novel views with self-learned confidence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 155–171, 2018. 2

[44] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d representation and rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15182–15192, 2021. 3

[45] Shubham Tulsiani, Richard Tucker, and Noah Snavely. Layer-structured 3d scene inference via view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 302–317, 2018. 2

[46] Wouter Van Gansbeke, Davy Neven, Bert De Brabandere, and Luc Van Gool. Sparse and noisy lidar completion with rgb guidance and uncertainty. In *2019 16th international conference on machine vision applications (MVA)*, pages 1–6. IEEE, 2019. 1, 3, 4

[47] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2021. 3

[48] Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 5

[49] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7467–7477, 2020. 2

[50] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015. 5

[51] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 767–783, 2018. 1, 3

[52] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1983–1992, 2018. 5

[53] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021. 3

[54] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2018. 1

[55] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 2

[56] Yinda Zhang and Thomas Funkhouser. Deep depth completion of a single rgb-d image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 175–185, 2018. 1, 3, 4

[57] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *SIGGRAPH*, 2018. 2

[58] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *European conference on computer vision*, pages 286–301. Springer, 2016. 2

[59] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9308–9316, 2019. 5