Omni-nerf: neural radiance field from 360° image captures

Kai Gu, Thomas Maugey, Sebastian Knorr, Christine Guillemot

OMNI-NERF: NEURAL RADIANCE FIELD FROM 360° IMAGE CAPTURES

Kai Gu, Thomas Maugey*, Sebastian Knorr†,‡ and Christine Guillemot**

*INRIA Rennes - Bretagne Atlantique, France;
†Ernst-Abbe University of Applied Sciences Jena, Germany; ‡Technical University of Berlin, Germany.
*{firstname.lastname}@inria.fr; †{firstname.lastname}@eah-jena.de

ABSTRACT

This paper tackles the problem of novel view synthesis (NVS) from 360° images with imperfect camera poses or intrinsic parameters. We propose a novel end-to-end framework for training Neural Radiance Field (NeRF) models given only 360° RGB images and their rough poses, which we refer to as Omni-NeRF. We extend the pinhole camera model of NeRF to a more general camera model that better fits omni-directional fish-eye lenses. The approach jointly learns the scene geometry and optimizes the camera parameters without knowing the fisheye projection.

***Index Terms*—** Light field, omni-directional imaging, rendering, view synthesis, deep learning

1. INTRODUCTION

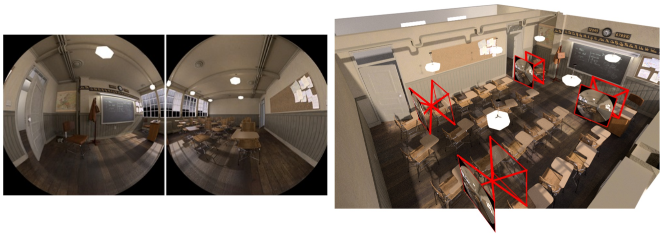
The concept of 6 degrees of freedom (6DOF) video content has recently emerged with the goal of enabling immersive experience in terms of free roaming, i.e. allowing viewing the scene from any viewpoint and direction in space [1]. However, no such real-life full 6DOF light field capturing solution exists so far. Light field cameras have been designed to record orientations of light rays, hence to sample the plenoptic function in all directions, thus enabling view synthesis for perspective shift and scene navigation [14]. Several camera designs have been proposed for capturing light fields, going from uniform arrays of pinholes placed in front of the sensor [4] to arrays of micro-lenses placed between the main lens and the sensor [11], arrays of cameras [17], and coded attenuation masks [9]. However, these light field cameras have a limited field of view. On the other hand, omni-directional cameras allow capturing a panoramic scene with a 360° field of view but do not record information on the orientation of light rays emitted by the scene.

Neural Radiance Fields (NeRF) have been introduced in [10] as an implicit scene representation that allows rendering all light field views with high quality. NeRF models the scene as a continuous function, and is parameterized as a multi-layer perceptron (MLP). The function represents the mapping between the 5D spatial and angular coordinates of light rays emitted by the scene into its three RGB color components and a volume density measure. NeRF is capable of modeling complex large-scale, and even unbounded, scenes. With a proper parameterization of the coordinates and a well-designed foreground-background architecture, NeRF++ [18] is capable of modeling scenes having a large depth, with satisfying resolution in both the near and far fields.

First NeRF-based models need the camera pose parameters to map the pixel coordinates to the ray directions, and use COLMAP [13, 12] for the camera parameter estimation. An end-to-end framework, called NeRF–, is proposed in [16] for training NeRF models without pre-computed camera parameters. The intrinsic and extrinsic camera parameters are automatically discovered via joint optimisation during training of the NeRF model. The authors in [5] consider more complex non-linear camera models and propose a new geometric loss function to jointly learn the geometry of the scene and the camera parameters. Solutions are also proposed in [2] and [3] to enable faster inference as well as using a sparser set of input views, and to enable generalization to new scenes.

Our motivation here is to be able to capture or reconstruct light fields with a very large field of view, in particular 360°. We focus on the question: how do we extract omni-directional information and potentially benefit from it when reconstructing a spherical light field of a large-scale scene with a non-converged camera setup?

To address this question, we propose a method for learning a 360° neural radiance field from omni-directional images captured with fisheye cameras. We adapt the pinhole camera model of NeRF to a general camera model that fits omni-directional fisheye lenses. Finally, we extend our approach to allow accurate camera parameter estimation (intrinsic and extrinsic) for omni-directional fisheye lenses. To evaluate our approach, we render photo-realistic panoramic fisheye views from 2 Blender scenes: an indoor ”Classroom” scene and a larger scale scene ”Lone Monk” of an atrium surrounded by buildings. With these datasets, we first evaluate how spherical sampling improves the performance of view synthesis compared to planar sampling. We prove that our model can learn the fisheye distortion from scratch with ground truth camera pose. We also assess the camera extrinsic parameter estimation, with a noisy initialization which simulates the case



(a) 360° fisheye images (b) Spatial camera grid

Fig. 1. Space sampling with 360° cameras (fisheye camera pairs) in a spatial grid.

where the camera parameters are imperfect or measured with error. Finally, we train and evaluate our model on two real scenes captured by fisheye cameras.

2. NERF: BACKGROUND

In this section we briefly introduce the method of synthesizing novel views of complex static scenes by representing the scenes as Neural Radiance Fields (NeRF) [10].

In NeRF, a scene is represented with a Multi-Layer Perceptron (MLP). Given the 3D spatial coordinate location $\mathbf{X} = (x, y, z)$ of a voxel and the viewing direction $\mathbf{d} = (\theta, \phi)$ in a scene, the MLP predicts the volume density σ and the view-dependent emitted radiance or color $\mathbf{c} = [R, G, B]$ at that spatial location for volumetric rendering. In the concrete implementation, $\mathbf{X}$ is first fed into the MLP, and outputs σ and intermediate features, which are then fed into an additional fully connected layer to predict the color $\mathbf{c}$. The volume density is hence decided by the spatial location $\mathbf{X}$, while the color $\mathbf{c}$ is decided by both $\mathbf{X}$ and viewing direction $\mathbf{d}$. The network can thus yield different illumination effects for different viewing directions. This procedure can be formulated as

$$[R, G, B, \sigma] = F_\Theta(x, y, z, \theta, \phi), \tag{1}$$

with $\Theta = \{W_i, b_i\}$ being weights and biases of the MLP. The network is trained by fitting the rendered (synthesized) views with the reference (ground-truth) views via the minimization of the total squared error between the rendered and true pixel colors in the different views. NeRF uses a rendering function based on classical volume rendering [6]. Rendering a view from NeRF requires calculating the expected color $C(\mathbf{r})$ by integrating the accumulated transmittance T along the camera ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, for $\mathbf{o}$ being the origin of cast ray from the desired virtual camera, with near and far bounds t_n and t_f. This integral can be expressed as

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t)\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt, \tag{2}$$

where $T(t) = exp(-\int_{t_n}^{t} \sigma(\mathbf{r}(s)ds)$. In practice, the integral is numerically approximated by sampling discretely along the ray.

While the original NeRF in [10] requires the knowledge of the camera pose parameters, a pose-free solution has been introduced in [16] which estimates intrinsic and extrinsic camera parameters while training the NeRF model.

Assuming a pinhole camera model, the camera parameters can be expressed with the camera projection matrix

$$\mathbf{P} = \mathbf{KR}[\mathbf{I}| - \mathbf{t}]. \tag{3}$$

$\mathbf{I}$ is a 3 identity matrix and $\mathbf{K}$ is the camera calibration matrix

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}, \tag{4}$$

with the intrinsic camera parameters f_x and f_y as the focal length, which are identical for square pixels, and c_x and c_y as the offsets of the principal point from the top-left corner of the image. The extrinsic camera parameters in Eq. (3) contain the rotation and translation of the camera with respect to a 3-D world coordinate system and are expressed by the 3×3 rotation matrix $\mathbf{R}$ and the 3×1 translation vector $\mathbf{t}$.

When these camera parameters are unknown, they need to be estimated. For camera rotation, the axis-angle representation is adopted in NeRF–:

$$\boldsymbol{\Phi} := \alpha\boldsymbol{\omega}, \boldsymbol{\Phi} \in \mathbb{R}^3 \tag{5}$$

where α is the rotation angle and $\boldsymbol{\omega}$ is the unit vector represent the rotation axis. $\boldsymbol{\Phi}_i$ can be converted to the rotation matrix $\mathbf{R}$ using the Rodrigues' formula [16]. With such parametrization, the i-th camera extrinsics can be optimized by searching for the parameters of $\boldsymbol{\Phi}_i$ and $\mathbf{t}_i$. To render the color of the m-th pixel $\mathbf{p}_{i,m} = (u, v)$ from the i-th camera, we cast the ray $\mathbf{r}_{i,m}$ from the image plane as

$$\mathbf{r}_{i,m}(t) = \mathbf{o} + t\mathbf{d}, \tag{6}$$

where

$$\mathbf{d} = \mathbf{R}_i^{-1} \begin{pmatrix} (u - c_x)/f_x \\ -(v - c_y)/f_y \\ -1 \end{pmatrix} \tag{7}$$

and $\mathbf{o} = \mathbf{t}_i$ using the current estimation of camera parameters

$$\boldsymbol{\pi}_i = (f_x, f_y, c_x, c_y, \boldsymbol{\Phi}_i, \mathbf{t}_i). \tag{8}$$

Then, the calculated coordinate and direction of the sampled ray are fed into the NeRF model. The current estimated color value of the pixel is rendered by Eq. (2) and can be compared with the ground-truth color value by estimating the squared error. Finally, the parameters Θ of the NeRF model and the camera parameters $\boldsymbol{\pi}_i$ are optimized jointly by minimising the photometric loss as described in [16] and [7].

3. OMNI-NERF: OMNI-DIRECTIONAL NERF

3.1. Fisheye projection

Here, we focus on panoramic fisheye lenses which usually have a field of view (FOV) greater than 180°. To adapt NeRF

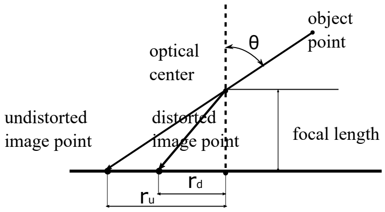


Fig. 2. Distorted image projection.

to omni-directional images, the key is to recover the true ray direction of pixel coordinates on the image plane. In order to do so, we need to model the projection of the fisheye lens onto the image plane. We can model these projections as relations between r_d, the radial distance from the image center to the distorted image point, and θ, the incoming angle measured from the lens axis as depicted in Fig. 2. For the specific ”equisolid” projection, the radial distance can be calculated as

$$r_d = 2f \cdot \sin\left(\frac{\theta}{2}\right). \tag{9}$$

Different projections exist for panoramic fisheye lenses [15]. In order to unify the description, we consider the sum of the 4 first terms of the infinite series as an odd-order polynomial representation of the incoming ray direction θ defined as

$$\theta = \theta_d + k_1\theta_d^3 + k_2\theta_d^5 + k_3\theta_d^7, \tag{10}$$

where

$$\theta_d = \arctan\left(\frac{r_d}{f}\right), \tag{11}$$

f is the focal length and k_1, k_2 and k_3 are coefficients to fit the different fisheye projections. Given a pixel $\mathbf{p}$ with its pixel coordinates (u, v) on the image plane, the radial distance r_d can then be expressed as

$$r_d = \sqrt{(u - c_x)^2 + (v - c_y)^2}, \tag{12}$$

where (c_x, c_y) are the coordinates of the principle point. The actual direction of the ray of pixel $\mathbf{p}$ in the camera coordinate system can then be expressed as a vector $\mathbf{d}_c = (x, y, z)^T$ with

$$x = \sin(\theta) \cdot (u - c_x) \tag{13}$$

$$y = \sin(\theta) \cdot (v - c_y) \tag{14}$$

$$z = \cos(\theta). \tag{15}$$

By applying a rotation and translation using the extrinsic parameters, the ray direction in world coordinates is then defined as

$$\mathbf{d} = [\mathbf{R}^{-1}|\mathbf{t}]\mathbf{d}_c. \tag{16}$$

The rotation matrix $\mathbf{R}$ can be parameterized by the axis-angle representation $\mathbf{\Phi}$ as described in Section 2.

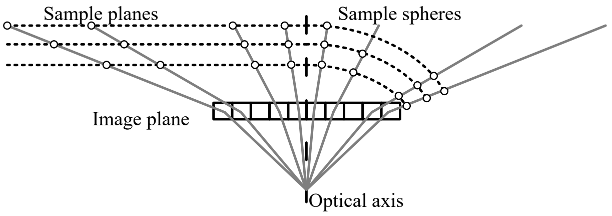


Fig. 3. Planar sampling vs. spherical sampling.

When these camera parameters are not known, e.g. when using real world captures, they need to be estimated. For this, we follow the approach of NeRF– and SCNeRF[5] where the estimated parameters are:

$$\pi_i = (f_x, f_y, c_x, c_y, \mathbf{\Phi}_i, \mathbf{t}_i, k_1, k_2, k_3), \tag{17}$$

i.e., extended by the parameters k_1, k_2 and k_3 which approximate the projection as in Eq. (10).

Assuming that the specific fisheye projection model is unknown, we initialize (k_1, k_2, k_3) to be $(0, 0, 0)$, i.e., the pinhole perspective camera model. We then optimize the coefficients of the polynomial to fit the specific fisheye projection by minimizing the photometric loss as mentioned in Section. 2. With different combinations of (k_1, k_2, k_3) the polynomial distortion model can approximate fisheye lenses or mirrors with FOVs of up to 360°.

3.2. Spherical sampling

In the original NeRF model, the pixels are rendered by sampling 5D coordinates (location and viewing direction) along the camera rays. For a camera ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ with near and far bounds t_n and t_f, where $\mathbf{o}$ is the origin of ray and $\mathbf{d}$ is a vector giving the ray direction. The interval $[t_n, t_f]$ is originally partitioned into N evenly-spaced bins, and one sample is then uniformly drawn at random from each bin. Such a sampling pattern is equivalent to placing sampling planes parallel to the image plane as depicted in Fig. 3.

For fisheye lenses, however, the rays on the border are more sparsely sampled than the rays close to the optical axis, and the spacing tends to infinity when the angle of the incidence ray approaches 90° (see Fig. 3). As the sampling is critical in the neural representation of the radiance field, such large bins have a high chance to skip thin objects in the scene and cause artifacts, i.e., resulting in degraded image quality.

Hence, we use a sampling on a sphere instead of the sampling on planes to resolve the above issue. For a ray direction $\mathbf{d} = (x, -y, -1)^T$, we define the normalized direction as

$$\mathbf{d_n} = \left(\frac{x}{\|\mathbf{d}\|}, -\frac{y}{\|\mathbf{d}\|}, -\frac{1}{\|\mathbf{d}\|}\right)^T \tag{18}$$

With this spherical sampling scheme, the rays at the border of the scene projected on an image have the same importance as the rays in the center. Thus, the spherical sampling offers a more uniform sampling of the whole scene. We therefore define the near and far bounds to be concentric spheres with radius t_{near} and t_{far} centered at the projection center.

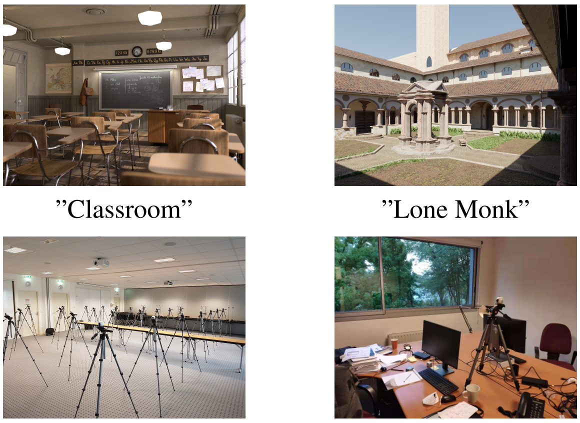


Fig. 4. Overview of all the scenes

Similarly, we partition $[t_{near}, t_{far}]$ into N evenly-spaced bins. Hence, the bins for rays in all the directions are equal as shown in Fig. 3, and a ray is now expressed with

$$\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}_n. \tag{19}$$

Please note that the spherical sampling is not only preferred for fisheye lenses, but also improves the image quality for any wide-angle lenses.

4. EXPERIMENTAL RESULTS

4.1. Dataset

The dataset consists of four scenes, two synthetic scenes from the Blender demo ”Classroom” (by Christophe Seux.) and ”Lone Monk” (by Carlo Bergonzini) and two real scenes, the FTV-indoor-sit dataset captured in a big meeting room [8] and a densely sampled scene in an office room, both captured with Samsung Gear360 cameras. Fig. 4 shows screenshots of the test scenes.

First, we deploy virtual fisheye and perspective cameras in the synthetic 3D scenes to create a dataset that samples the 3D scenes from different viewpoints. A pair of virtual cameras facing forward and backward are placed in a bounded cuboid at each vertex of the spatial grid to get full 360° FOV. By subdividing the edge of the cuboid, we use different sampling of the space and therefore with a different number of cameras. In particular, we investigated two different sampling grids, namely 6x6x3 (108 view points) and 9x9x3 (243 view points) in each scene with fisheye and perspective camera pairs, which are our training datasets. Furthermore, we render a smooth path of 400 intermediate views in the sampled 3D space by varying rotation and translation for both camera types, which are our test datasets. Both training and test images have a resolution of 600x600 pixels. The fisheye views and perspective views are rendered with equisolid projection of 180° FOV and perspective projection of 119° FOV, respectively. We demonstrate the performance of Omni-NeRF compared to SCNeRF using the synthetic datasets. Finally, we show that our model is capable of reconstructing real scenes. For configuration details of the dataset and camera setup, please refer to the provided supplementary material.

sample method	Classroom		Lone Monk	
(samples)	FE	WA	FE	WA
planar(128)	22.46	24.77	15.13	24.68
planar(256)	24.78	**25.19**	25.38	**25.04**
spherical(128)	**28.69**	25.18	**28.45**	25.03

Table 1. PSNR values obtained with different sampling methods and synthetic datasets. ”FE” and ”WA” denote fisheye and wide-angle perspective rendering, respectively.

4.2. Fisheye and Spherical Sampling

Our first experiment aims to evaluate how spherical sampling improves the performance of view synthesis compared with planar sampling. For this, we train Omni-NeRF with synthetic data using ground truth camera parameters and different sampling schemes.

We use the hierarchical sampling strategy of NeRF and consider training and rendering with planar sampling of 128 and 256 coarse samples along the rays. For spherical sampling we use 128 coarse samples. Then, we allocate 128 fine samples biased towards the relevant parts of the volume for all the cases. The reported PSNR in Table 1 are averaged values over 400 test views of fisheye and perspective cameras. The table shows that spherical sampling outperforms the planar sampling for fisheye rendering with much fewer samples. For wide-angle perspective rendering, the spherical sampling achieves similar performance with only half the number of samples.

Fig. 5 shows small patches of rendered views with synthetic data. The rendered views with planar sampling show typical artifacts close to the border while the spherical sampling with same or a lower number of samples resolves the issue.

4.3. Fisheye distortion estimation

In our second experiment, we show that our model can learn the fisheye distortion from scratch using ground truth camera pose. And we also show that our model can optimize the camera parameters using noisy camera pose as initialization. We train our model and also SCNeRF on ”Classroom” and ”Lone Monk” datasets. We first investigate the estimation of radial distortion with different models, so we fix the extrinsics and intrinsics to their ground truth values and we initialize both models to correspond to pinhole projections. Fig. 6 shows how our model fit to fisheye projection.

We evaluate our model with the average PSNR of the test

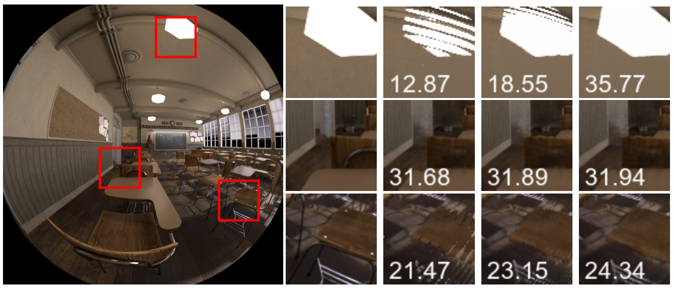


Fig. 5. Rendered views with planar (P) and spherical (S) sampling with different numbers of inputs, with the ”Classroom” scene. (GT denotes ground truth, PSNR is shown on each sub-figure).

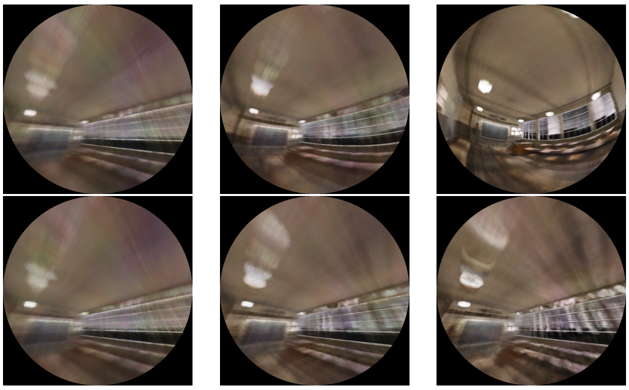


Fig. 6. We initialize the projection as pinhole camera and start to optimize the projection model. The figures show how our model (first row) and SCNeRF (second row) fit the fisheye projection during training.

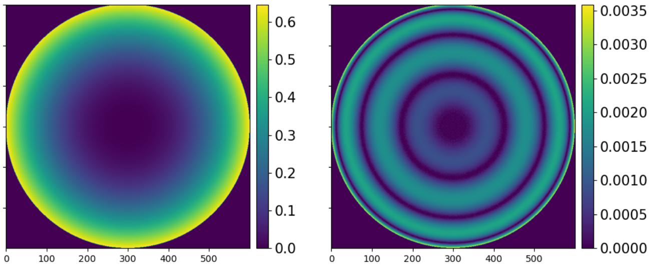
set and the mean absolute error (MAE) of angles e_θ between estimated ray directions $\hat{\mathbf{d}}$ and ground truth ray directions $\mathbf{d}$ of all valid pixels in the fisheye view, given by

$$e_\theta = \arccos\left(\frac{\hat{\mathbf{d}} \cdot \mathbf{d}}{\|\hat{\mathbf{d}}\| \|\mathbf{d}\|}\right). \tag{20}$$

We then add uniformly distributed random noise to the ground truth camera pose as initialization and optimize the noisy pose jointly with the projection model. we add random rotation angle $\Delta_\theta \sim U(-7.5, 7.5)(rad)$ along random axis, and translation $\Delta_t \sim U(0.075, 0.075)(meter)$ along (x,y,z) axes for each training view. We train our model and SCNeRF with the same noisy initialization on the synthetic datasets. We report the result of different scenes using ground truth pose and noisy pose in Table 2 and visualize the projected ray error in Fig. 7. The result shows that our model, with the proposed polynomial representation of ray direction, can estimate the fisheye distortion accurately, with ground truth or noisy camera poses, while SCNeRF fails to learn the correct projection.

	Classroom		Lone Monk	
Method	PSNR	MAE	PSNR	MAE
SCNeRF (gt. pose)	13.04	0.21	12.94	0.28
Ours (gt. pose)	27.12	0.001	26.55	0.001
SCNeRF (noisy pose)	13.74	0.13	13.92	0.21
Ours (noisy pose)	21.65	0.004	24.68	0.003

Table 2. Comparison between SCNeRF and Omni-NeRF in terms of fisheye distortion estimation. We report the average PSNR of the smooth-path test set and the mean absolute error of estimated ray direcitions after a same amount of iterations.



(a) SCNeRF (b) Ours

Fig. 7. Error of estimated ray directions. Please note that the scales of the two figures are different.

We finally train our model on real captured 360° fisheye data. The model has been initialized with imperfect camera pose parameters and projection model. Fig. 8 shows exemplary the reference views and rendered views using our model with 2 different scenes.

4.4. Discussion

We found that NeRF– and other differentiable camera models have a limited search range of camera parameters. For our non-planar and non-converging camera configurations, rough initialization is necessary. Due to the peculiarities of panoramic fisheye lenses, the camera usually captures some of the pixels of the camera mount or the photographer, leading to problems of inconsistency between frames, which have an impact on the scene reconstruction and parameter estimation.

5. CONCLUSION

We proposed Omni-NeRF, a NeRF-based method that reconstructs the scene with 360° information from panoramic fisheye imaging. We optimized the parameterization for the estimation of wide-angle fisheye distortion and we stressed the importance of sampling in this case. We have shown that the spherical sampling improves the performance when training with panoramic fisheye or wide-angle perspective images. Furthermore, We have shown that our parametrization is better suited than the one of SCNerf when using panoramic fisheye lenses. We have also shown that NeRF can be used to reconstruct scenes with 360° panoramic information, using a non-converged camera setup. Finally, we demonstrated

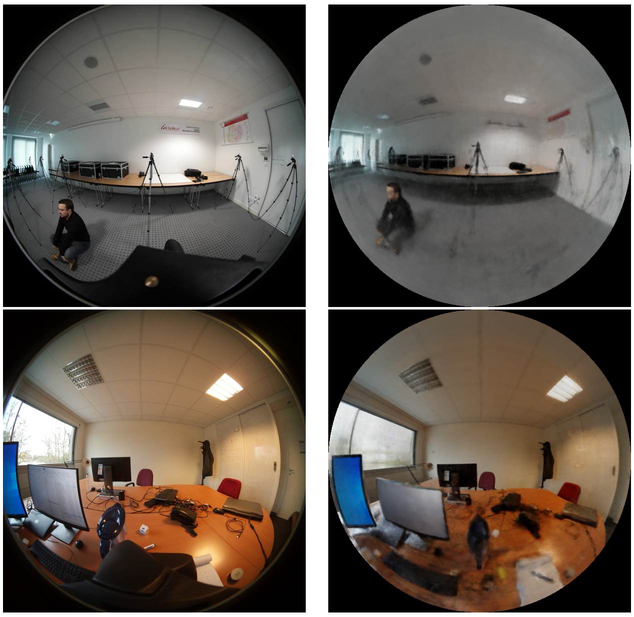


Fig. 8. Rendered view with Omni-NeRF trained for the FTV-indoor-Sit [8] dataset and our office dataset. The model has been initialized with imperfect camera pose parameters and projection model.

that our method is also applicable to data collected with real 360 cameras, and that our model successfully reconstructs the scene. Since we are dealing with the problem of reconstructing large-scale scenes, the question of how to optimally deploy cameras in 3D space is worth further exploration.

6. ACKNOWLEDGEMENT

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 956770.

7. REFERENCES

[1] M. Broxton, J. Flynn, R. Overbeck, D. Erickson, P. Hedman, M. DuVall, J. Dourgarian, J. Busch, M. Whalen, and P. Debevec. Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics*, 39(4):86:1–86:15, 2020.

[2] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su. MVSNeRF: Fast Generalizable Radiance Field Reconstruction from Multi-View Stereo. *arXiv preprint arXiv:2103.15595*, 2021.

[3] J. Chibane, A. Bansal, V. Lazova, and G. Pons-Moll. Stereo Radiance Fields (SRF): Learning View Synthesis for Sparse Views of Novel Scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[4] H. E. Ives. Parallax panoramagrams made with a large diameter lens. *Journal of the Optical Society of America*, 20(6):332–342, 1930.

[5] Y. Jeong, S. Ahn, C. Choy, A. Anandkumar, M. Cho, and J. Park. Self-Calibrating Neural Radiance Fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[6] J. T. Kajiya and B. P. Von Herzen. Ray Tracing Volume Densities. *ACM Computer Graphics*, 18(3):165–174, 1984.

[7] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey. BARF: Bundle-Adjusting Neural Radiance Fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[8] T. Maugey, L. Guillo, and C. L. Cam. Ftv360: a multiview 360° video dataset with calibration parameters. In *ACM Multimedia Systems Conference*, 2019.

[9] E. Miandji, J. Unger, and C. Guillemot. Multi-shot single sensor light field camera using a color coded mask. In *European Signal Processing Conference (EUSIPCO)*, 2018.

[10] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, 2020.

[11] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan. Light Field Photography with a Handheld Plenoptic Camera. Research Report CSTR 2005-02, Stanford University, 2005.

[12] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys. Pixelwise view selection for unstructured multiview stereo. In *European Conference on Computer Vision (ECCV)*, 2016.

[13] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[14] J. Shi, X. Jiang, and C. Guillemot. Learning fused pixel and feature-based view reconstructions for light fields. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[15] M. Thoby. Photographic lenses projections: computational models, correction, conversion... Available: *http://michel.thoby.free.fr/Fisheye_history_short/Projections/Fisheye_projection-models.html.*, 2012.

[16] Z. Wang, S. Wu, W. Xie, M. Chen, and V. A. Prisacariu. NeRF–: Neural Radiance Fields Without Known Camera Parameters. *arXiv preprint arXiv:2102.07064*, 2021.

[17] B. Wilburn, N. Joshi, V. Vaish, E. V. Talvala, E. Antunez, A. Barth, A. Adams, M. Horowitz, and M. Levoy. High performance imaging using large camera arrays. *ACM Transactions on Graphics*, 24(3):765–776, 2005.

[18] K. Zhang, G. Riegler, N. Snavely, and V. Koltun. NeRF++: Analyzing and Improving Neural Radiance Fields. *ArXiv*, abs/2010.07492, 2020.