NeRFtrinsic Four: An End-To-End Trainable NeRF Jointly Optimizing Diverse Intrinsic and Extrinsic Camera Parameters

Hannah Schieber Fabian Deuser Bernhard Egger Norbert Oswald Daniel Roth

Human-Centered Computing and Extended Reality
Friedrich-Alexander Universität Erlangen-Nürnberg
Erlangen, Germany
hannah.schieber@fau.de,
d.roth@fau.de

Lehrstuhl für Graphische Datenverarbeitung (LGDV)
Friedrich-Alexander Universität Erlangen-Nürnberg
Erlangen, Germany
bernhard.egger@fau.de

Institute for Distributed Intelligent Systems
University of the Bundeswehr Munich
Munich, Germany
fabian.deuser@unibw.de,
norbert.oswald@unibw.de

Abstract

Novel view synthesis using neural radiance fields (NeRF) is the state-of-the-art technique for generating high-quality images from novel viewpoints. Existing methods require a priori knowledge about extrinsic and intrinsic camera parameters. This limits their applicability to synthetic scenes, or real-world scenarios with the necessity of a preprocessing step. Current research on the joint optimization of camera parameters and NeRF focuses on refining noisy extrinsic camera parameters and often relies on the preprocessing of intrinsic camera parameters. Further approaches are limited to cover only one single camera intrinsic. To address these limitations, we propose a novel end-to-end trainable approach called NeRFtrinsic Four. We utilize Gaussian Fourier features to estimate extrinsic camera parameters and dynamically predict varying intrinsic camera parameters through the supervision of the projection error. Our approach outperforms existing joint optimization methods on LLFF and BLEFF. In addition to these existing datasets, we introduce a new dataset called iFF with varying intrinsic camera parameters. NeRFtrinsic Four is a step forward in joint optimization NeRF-based view synthesis and enables more realistic and flexible rendering in real-world scenarios with varying camera parameters.

1. Introduction

Generating novel views and producing rich, photo-realistic images requires multiple camera angles from different viewpoints to generate a detailed 3D scene representation. For the generation of novel views the knowledge of the intrinsic and extrinsic camera parameters of the training

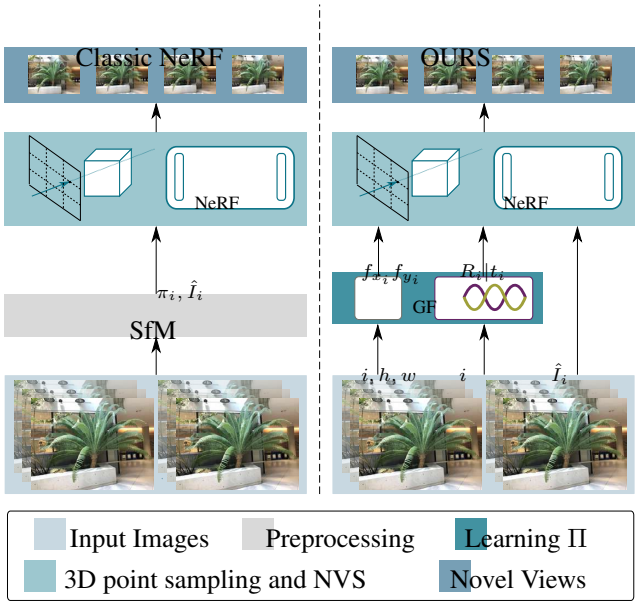

Figure 1: **The classic NeRF framework compared to our NeRFtrinsic Four.** Training a NeRF is usually limited to one type of camera and requires known camera parameters. We present NeRFtrinsic Four which jointly optimizes the camera parameters (Π) of multiple diverse cameras without the necessity of a preprocessing step. Our approach utilizes Gaussian Fourier features (GF) to learn the extrinsic camera parameters. Furthermore, we individually optimize the intrinsic camera parameters per given camera.

images is crucial [1, 2, 3, 4]. Intrinsic camera parameters that are dependent on camera properties like focal length or pixel dimensions impact the image that represents a captured part of a scene. In turn, individual camera parameters influence the projection of pixels into the 3D coordinate space. This projection into the 3D space is required by neural radiance fields (NeRF) [3, 5, 6] which use the intrinsic parameters for the ray projection. Besides intrinsic camera parameters, the camera angle and position, denoted as extrinsic, are decisive.

Traditional approaches use rich structure from motion (SfM) algorithms like COLMAP [7] to determine camera parameters in a preprocessing step and later utilize these estimations to train the NeRF [8, 3]. Although SfM enables training, the preprocessing step is always necessary for new data. This prevents the NeRF from being end-to-end capable. Moreover, standard SfM algorithms highly depend on texture to estimate accurate camera parameters.

One use case is the representation of a 3D scene based on images from varying cameras. To easily generate a 3D representation of such a scenario, the network should be capable of processing the images directly without requiring any preprocessing. To avoid the preprocessing step, existing approaches investigate the joint optimization of camera parameters and the NeRF [9, 5, 10, 6, 11]. Concurrent joint optimization approaches either require given intrinsic camera parameters [9, 5], assume a pretrained NeRF as given [10] or are restricted to one single camera [6, 11].

We propose NeRFtrinsic Four, a novel approach that optimizes diverse intrinsic and extrinsic camera parameters along with novel view synthesis (NVS). Unlike existing approaches, our method is not constrained to one single camera and does not require a preprocessing step to estimate the camera parameters. It also does not rely on a pretrained NeRF. We demonstrate the effectiveness of our approach on three benchmarks, namely LLFF [12], BLEFF [6] and our own iFF. Our evaluation shows that NeRFtrinsic Four outperforms state-of-the-art joint optimization methods in terms of image quality and camera parameter estimation on LLFF, BLEFF and iFF. Overall, our approach provides a more versatile solution for handling real-world scenes with varying cameras.

In summary, our approach contributes:

- A dynamic joint end-to-end trainable optimization framework, capable of handling diverse cameras.
- A pose-multilayer perceptron (MLP), using Gaussian Fourier features for the handling of challenging poses.
- Our novel iFF dataset focusing on the challenge of diverse cameras, on which we demonstrate the advantages of NeRFtrinsic Four[1].

[1] GitHub-NeRFtrinsic Four

2. Related Work

Our approach addresses the joint optimization of NeRF as well as intrinsic and extrinsic camera parameters. We first review approaches, that predict camera parameters, followed by the combination of NeRF and camera pose refinement. Most importantly, we consider approaches estimating camera parameters and optimizing NeRF without prior initialization.

2.1. Camera Parameter Estimation

Traditional approaches which estimate the camera parameters often use SfM. SfM algorithms extract features and match them to find a 2D-3D correspondence. They estimate candidate poses and apply classical or optimized RANSAC to find the best matching poses [13, 14].

In addition to SfM-based approaches, simultaneous localization and mapping (SLAM) algorithms often jointly optimize the camera parameters together with the 3D reconstruction. MonoSLAM [15] and ORB-SLAM [16] estimate the extrinsic camera parameters using feature correspondences. Others apply the photometric loss to optimize the camera pose [17, 18, 19]. COLMAP [7] is typically used to preprocess the camera parameters for NeRF. It first creates a sparse reconstruction using SfM, and afterwards applies dense modelling using multi-view stereo.

Besides SfM or SLAM, deep learning based approaches are applied. Lee et al. [20] estimate the extrinsic camera parameters, assuming the intrinsic parameters to be known. The pose is estimated via the prediction of keypoints by a deep neural network using purely synthetic data. Elmoogy et al. [21] use a convolutional neural network (CNN) to extract features from each image and feed a Graph Neural Network (GNN) to find the extrinsic camera parameters. In addition to only predicting the extrinsic camera parameters, Butt et al. [22] simultaneously estimate intrinsic camera parameters using a CNN. They apply Inception-v3 and a camera projection loss to estimate all camera parameters.

2.2. NeRF and Camera Pose Refinement

Mildenhall et al. [3] introduced NeRF, which encodes a scene representation in a MLP. Subsequent approaches optimize towards a higher image quality, handle fewer input views (e.g. [23, 24, 8]) or use search engine results as input and handle the camera parameters via COLMAP [8]. Others investigate the impact of jointly optimizing extrinsic camera parameters and NeRF to improve NVS quality [25, 9, 5, 26].

Jeong et al. [25] tackle the joint extrinsic camera parameter and NVS optimization with the geometric loss. This loss jointly optimizes the NeRF and extrinsic camera parameters focusing on forward-facing scenes. Besides optimizing initialized camera parameters, their approach can estimate unknown parameters by finding correspondences.

Lin et al. [9] enhance this idea by using a coarse-to-fine annealing schedule in BARF. BARF jointly optimizes the 3D scene representation and registers the camera poses, initialized by noisy camera poses. Chen et al. [27] improve BARF using Gaussian activation functions. While Truong et al. [26] also optimize on noisy poses, they additionally consider only sparse input views.

Apart from BARF-like approaches, Meng et al. [5] combine NeRF and a GAN in GNeRF. GNeRF first randomly samples poses from a predefined pose sampling space and then optimizes the NeRF. The discriminator differentiates between real and fake images. An inversion network then learns the camera pose. Finally, the pose embedding and NeRF are optimized using the photometric loss.

2.3. NeRF Without Known Camera Parameters

While previous methods initialize their pose regression from noisy poses [9, 26, 27] or sample from a predefined space [5], iNerf [10] optimizes the camera pose by inverting a pretrained NeRF. It starts from an initial pose and applies gradient descent to minimize the residual between the pixels in the rendered image and the observed image. Wang et al. [6] introduce NeRF-- which optimizes not only the extrinsic but also the intrinsic camera parameters. The camera parameters are jointly optimized with the NeRF using the photometric loss. However, the intrinsic camera parameter estimation is limited to only one camera. They also limit their NeRF to focus on forward-facing scenes with a perturbation $\leq 20°$ and do not provide specific ray sampling for $360°$ scenes. SiNeRF [11] extends this approach by utilizing sinusoidal activation functions for radiance mapping and a novel Mixed Region Sampling (MRS). The MRS ensures efficient training and prevents poor supervision from the lack of ray diversity. Their approach also focuses on one single camera intrinsic. Although they use MRS the applicability is limited to forward-facing scenes.

The most relevant work to ours is NeRF-- [6] and SiNeRF [11] as they jointly optimize the NeRF, intrinsic camera parameters and extrinsic camera parameters, without any priors. However, the intrinsic camera parameters are restricted to one type of camera. Thus, we further optimize the camera intrinsic estimation to generalize on differing cameras at the same time. Moreover, we not only learn the six extrinsic parameters but show that a MLP using Gaussian Fourier features can minimize the translation and rotation error and improves NVS quality.

3. Method

In this work, we investigate the end-to-end trainable joint optimization of varying intrinsic and extrinsic camera parameters as well as NVS. The extrinsic camera parameters are learned by employing Gaussian Fourier feature mapping. For the intrinsic camera parameters, we apply dynamic parameter learning for each given camera. This allows NeRFtrinsic Four to learn independent intrinsic camera parameters for each camera. As depicted in Fig. 2, the output of extrinsic and intrinsic camera parameter estimation is then used for the NeRF training and jointly optimized with the NeRF.

3.1. Camera Parameters

Following the definition of the pinhole camera model [28], $\Pi = K\,[R|t]$, the extrinsic camera parameters $[R|t]$ are used to convert from the world coordinate system to the camera coordinate system. The intrinsic camera parameters K convert the points from the camera coordinate system into the image coordinate system. These parameters are the field of view, the focal length f_x, f_y, the principal point, skew and the geometric distortion. In summary, the camera parameters Π influence which part of the scene around the camera is later part of the captured image. In turn, for NeRF, this defines the projection of rays and is essential for training NeRF.

3.1.1 Intrinsic Camera Parameters

The number of possible images that are used for NeRF is known a priori. For this, we construct a lightweight parameter array. The length of the array equals the number of varying cameras to optimize each image differentiated by their given camera towards their focal length. Thus, the images with varying height and width are independent from each other and independent $f = (f_x, f_y)$ values can be learned. Following the assumption of Wang et al. [6], we consider the camera's principle points $c_x \approx W/2$ and $c_y \approx H/2$, where H, W denote the image height and width respectively. Thus, initialization is done individually for each intrinsic. The focal length parametrization, see equation 1, is a learned scaling factor s_i, which is initialized with 1.0 for each camera i. Optimizing the square root of s, leads to better results in NeRF--, which is why we adopted this 2nd-order trick.

$$f_{x_i} = s_i^2 W,\; f_{y_i} = s_i^2 H \tag{1}$$

3.1.2 Gaussian Fourier Feature Mapping for Extrinsic Camera Parameters

The extrinsic camera parameters, rotation $R \in SO(3)$ and translation $t \in \mathbb{R}^3$, are represented by a camera-to-world transformation matrix $[R|t]$ in $SE(3)$. We use Gaussian Fourier feature mapping in a MLP to learn these parameters. By utilizing different Fourier feature mappings, MLPs can learn high-frequency features from a low-dimensional input $\mathbf{v}$. Tancik et al. [29] compared positional encoding given in equation 2 and Gaussian Fourier feature mapping given in equation 3.

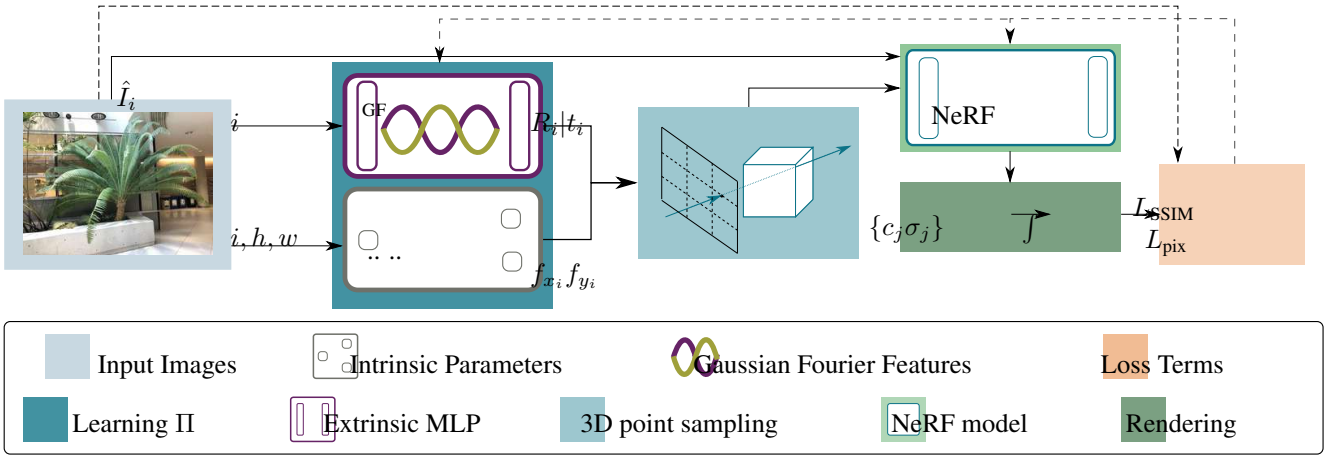

Figure 2: **Architecture of NeRFtrinsic Four, our end-to-end trainable approach.** We utilize Gaussian Fourier features (GF) in our pose MLP to learn the extrinsic camera parameters and learn individual intrinsic camera parameters per given camera. We further stabilize the convergence of the pose MLP with a SSIM loss function. Our framework is jointly optimized using the SSIM loss (L_{SSIM}) and photometric loss (L_{pix}).

$$\gamma(\mathbf{v}) = \left[\cos(2\pi\sigma^{j/m}\mathbf{v}), \sin(2\pi\sigma^{j/m}\mathbf{v})\right]^T, j = 0, ..., m-1 \tag{2}$$

For both mappings, the scale σ is determined through a hyperparameter search. While the positional encoding is deterministic, every value in matrix $\mathbf{B}$ for the Gaussian mapping is sampled from $N(0, \sigma^2)$. Due to this sampling of random features, the bias towards axis aligned data, as in positional encoding, is avoided. Tancik et al. applied this only the image input to boost the performance of NeRF.

$$\gamma(\mathbf{v}) = [\cos(2\pi\mathbf{B}\mathbf{v}), \sin(2\pi\mathbf{B}\mathbf{v})]^T, \mathbf{B} \in \mathbb{R}^{m\times d} \tag{3}$$

We instead apply this for our extrinsic camera parameter estimation. The Gaussian Fourier feature mapping is used to map the index of each camera to a higher-dimensional space. As frequency parameter m for the matrix B, we use 128 which results in an embedding size of 256. Our lightweight MLP consists of three layers with a hidden size of 64 and GELU activation functions. The embedding scale σ is set to 10.

3.2. Neural Radiance Fields

Using the camera parameters Π, NeRF enable the generation of novel views. A scene in NeRF [3] is represented as 5D vector function f consisting of the 3D location $\mathbf{x} = (x, y, z)$ and 2D viewing direction $\mathbf{d} = (\theta, \phi)$ as input. A NeRF maps the 3D location $\mathbf{x}$ and viewing direction $\mathbf{d}$ to a radiance color $\mathbf{c} = (r, g, b)$ and volume density σ, namely $f : \mathbb{R}^5 \rightarrow \mathbb{R}^4$. The 5D coordinates are sampled along camera rays and the output of the MLP is used in classical volumetric rendering techniques.

This differential volumetric rendering allows a fully optimizable pipeline for obtaining the pixel values based on the input coordinates. To render images from NeRF, the color from each pixel $\mathbf{p} = (u, v)$ on the image plane $\hat{I}_i$ is obtained by rendering the function $\mathcal{R}$, considering known camera parameters Π [3, 6], see equation 4.

$$\hat{I}_i(p) = \mathcal{R}(\mathbf{p}, \pi_i|\Theta)) = \int_{h_n}^{h_f} T(h)\sigma(\mathbf{r}(h))\mathbf{c}(r(h), \mathbf{d})\, \mathrm{d}h, \tag{4}$$

The near and far bounds are denoted as h_n (near) and h_f (far) [6]. π_i denotes the camera parameters and $T(h) = \exp(-\int_{h_n}^{h_f})\sigma(\mathbf{r}(s)ds)$ describes the accumulated transmission factor along the ray. To optimize the radiance field, NeRF minimize the mean squared error between rendered color and ground truth color also called photometric loss. In summary, the general NeRF framework can be formulated as $\Theta^* = \arg\min \mathcal{L}(\hat{I}|I, \Pi)$.

Wang et al. [6] adapt this framework to jointly optimize the NeRF as well as the intrinsic and extrinsic camera parameters, with the focus on forward-facing scenes. To achieve the joint optimization, Wang et al. reformulate the NeRF framework as denoted in equation 5. However, this framework is restricted to one single camera.

$$\Theta^*, \Pi^* = \arg\min \mathcal{L}(I, PI|I) \tag{5}$$

Our approach investigates the end-to-end trainable joint optimization of differing intrinsic and extrinsic camera parameters and NeRF. NeRFtrinsic Four learns varying intrinsic and extrinsic camera parameters along with the scene representation, see Fig. 2. As input, a set of RGB images, potentially captured by varying cameras are used. We espe-

cially investigate intrinsic and extrinsic parameter optimization. Therefore, we adapt the NeRF framework, see equation 6. Our framework considers differing intrinsic camera parameters K^*_{cam} and utilizes Gaussian Fourier features to predict the pose $[R|t]^*$ of each camera.

$$\Theta^*, K^*_{cam}, [R|t]^* = \arg\min \mathcal{L}(\hat{I}\hat{K}_{cam}, [\hat{R}|\hat{t}]|I) \quad (6)$$

Consequently, we jointly optimize the intrinsic K^*_{cam} and extrinsic $[R|t]^*$ camera parameters along with the NeRF, while also allowing that images can be taken by different cameras (*cam*). Our approach is depicted in Fig. 2, which shows the individual steps.

4. Evaluation

We evaluate the camera parameter estimation and NVS quality of NeRFtrinsic Four on two real-world datasets, namely Local Light Field Fusion (LLFF) and our intrinsic forward-facing (iFF) dataset. On LLFF, we compare our approach in detail with the results from NeRF-- [6] and SiNeRF [11]. To evaluate the performance of joint optimization approaches when diverse intrinsic camera parameters are present, we compare NeRF-- and our approach on iFF. Additionally, we compare our results with NeRF-- [6] on the synthetic Blender Forward Facing (BLEFF) dataset.

4.1. Datasets

LLFF: Eight forward-facing scenes are included in LLFF [12]. These scenes have a varying number of images ranging from 20 up to 62. The pseudo ground truth of LLFF derives from COLMAP.

BLEFF: Wang et al. [6] presented BLEFF to evaluate camera parameter estimation accuracy and NVS rendering quality. BLEFF contains 14 scenes with 31 images each with a resolution of 1040×1560. For comparability, with NeRF-- we followed the downscaling to a resolution of 520×780 and the $t010r010$ BLEFF setup.

iFF: We captured five real-world scenes namely T1, brick house, bike, fireplug and stormtrooper with 31 images each. The scenes were captured with an OAK-D Lite, an iPhone mini 13, and an iPad Air 2 with varying resolutions. Additionally, we added resizing, to show the influence of various image sizes. We received the pseudo ground truth by applying COLMAP. Details about the focal length and example images are included in the supplementary material.

4.2. Evaluation Metrics

To evaluate our approach, we consider two aspects. First, the rendering quality of the novel views. Here, we report peak signal-to-noise ratio (PSNR), similarity index measure (SSIM) [30] and learned perceptual image patch similarity (LPIPS) [31]. The second aspect is the camera parameter estimation. To report on the intrinsic camera parameter quality, we measure the focal length error in pixels. For the extrinsic camera parameters, we use the absolute trajectory error (ATE) [32, 33] and similarity transformation $Sim(3)$ to align the ground truth and predicted poses. As metric, we report the rotation and translation error.

Method	PSNR↑	Focal Err.	Rot. Err.	Trans. Err
LLFF (Real-World)				
COLMAP [6]	23.52	-	-	-
OURS^{COLMAP}	**23.70**	**10.4**	**0.87**	**0.002**
NeRF-- [6]	22.48	143.3	3.75	0.031
SiNeRF [11]	21.49	155.2	14.81	0.033
OURS	**23.14**	**103.7**	**2.10**	**0.008**
BLEFF (Synthetic)				
COLMAP [6]	**33.92**	14.89	13.65	**0.012**
NeRF-- [6]	33.24	20.55	4.45	0.065
OURS	33.57	**8.57**	**2.51**	0.018
iFF (Real-World)				
NeRF--	23.78	198.94	8.71	**0.159**
OURS	**27.15**	**75.14**	**5.46**	0.191

Table 1: **Overview of the mean PSNR values and the camera parameter estimation errors on LLFF, BLEFF and iFF**. Our approach outperforms the existing joint optimization approaches on LLFF, BLEFF and iFF. When using COLMAP initialization and applying our method to fine-tune the camera parameters, we also outperform COLMAP-based NeRF on LLFF.

4.3. Loss Function

The commonly used loss $L_{pix} = \sum_{r \in R_i} ||I(r) - \hat{I}(r)||^2$, also called photometric loss, is applied to train the joined optimization. Moreover, we found that the pose prediction stability can be strengthened by applying the SSIM loss $L_{\text{SSIM}}(P) = \frac{1}{N} \sum_{p \in P} 1 - \text{SSIM}(p)$ during the starting phase of the training.

4.4. Implementation Details

NeRFtrinsic Four is implemented in PyTorch. It (a) excludes the hierarchical sampling strategy; (b) has a layer dimension of 128 instead of 256; (c) samples 1024 pixels from each input image and 128 points along each ray [6]. To initialize our NeRF, we use Kaiming initialisation [34]. The focal length is initialized by the individual camera height and width considering the given resize factor of the individual dataset and optimized during training. The camera poses are initialized in $-z$ direction.

We use Adam optimizer for the camera parameters and NeRF. The initial learning rate is set to 10^{-3} for all models. We decay the NeRF learning rate all 10 epochs by multiplying with 0.9954. The focal and pose learning rate are decayed every 100 epochs by multiplying 0.9.

To ensure comparability with NeRFtrinsic Four, we retrain SiNeRF with a layer dimension of 128 as the original

Scene	PSNR↑			SSIM↑			LPIPS↓			Rot. Err.			Trans. Err.		
	NeRF--	SiNeRF	OURS	NeRF--	SiNeRF	OURS	NeRF--	SiNeRF	OURS	NeRF−−	SiNeRF	OURS	NeRF−−	SiNeRF	OURS
Fern	21.67	20.99	**21.82**	0.61	0.59	**0.62**	0.50	0.53	**0.49**	1.78	**1.17**	1.34	0.029	**0.006**	**0.006**
Flower	25.34	**25.66**	25.39	0.71	**0.73**	0.72	**0.37**	**0.37**	**0.37**	4.84	1.38	**0.89**	0.016	**0.007**	**0.007**
Fortress	26.20	26.74	**27.28**	0.63	0.67	**0.70**	0.49	0.45	**0.41**	1.36	2.02	**0.91**	0.025	0.048	**0.006**
Horns	22.53	17.29	**24.00**	0.61	0.45	**0.67**	0.50	0.66	**0.45**	5.55	83.34	**1.89**	0.044	0.133	**0.014**
Leaves	18.88	17.38	**18.97**	**0.53**	0.43	0.50	**0.47**	0.52	0.49	3.90	14.46	**2.65**	0.016	0.100	**0.005**
Orchids	16.73	16.77	**17.41**	0.39	0.40	**0.43**	0.55	0.53	**0.52**	4.96	3.97	**2.75**	0.051	0.014	**0.009**
Room	25.84	24.84	**27.12**	**0.84**	0.80	0.82	0.44	0.51	**0.43**	2.77	4.92	**1.33**	0.030	0.022	**0.006**
T-Rex	22.67	22.14	**22.92**	0.72	0.68	**0.74**	0.44	0.49	**0.43**	4.67	7.19	**3.90**	0.036	0.027	**0.008**
Mean	22.48	21.49	**23.14**	0.63	0.60	**0.66**	0.47	0.51	**0.44**	3.73	14.81	**2.10**	0.031	0.033	**0.008**

Table 2: **Quantitative comparison between NeRFtrinsic Four (OURS), NeRF-- [6], and SiNeRF [11] on LLFF**. For SiNeRF we retrained the approach with a layer dimension of 128, to ensure comparability with our approach and NeRF--. We report PSNR, SSIM and LPIPS to show the results on NVS. For the extrinsic camera parameters we report the translation error and rotation error.

Figure 3: **Visual results on LLFF.** As shown in the scene room, our approach (right) gets finer details compared to NeRF-- (left). This can be seen, for example, when looking at the chair in the front, the cables on the table or the box in the back on right side of the room.

SiNeRF has a layer dimension of 256. For SiNeRF we tried 10 random seeds per scene and report the best results on LLFF. In our supplementary material we show the comparison of NeRFtrinsic Four and SiNeRF with a layer dimension of 256.

4.5. Novel View Synthesis Quality

We compare our approach with COLMAP-based NeRF, on LLFF and BLEFF. The COLMAP-based NeRF follows the same 128-layer dimension as NeRF--. This results in lower NVS quality compared to the vanilla NeRF [3].

As shown in Table 1, for LLFF and BLEFF we achieve competitive NVS compared to COLMAP without any need for preprocessing and outperform existing joint optimization approaches. On LLFF, we outperform these approaches in PSNR, SSIM and LPIPS. When using COLMAP initialization for the joint optimization we also outperform COLMAP-based NeRF. Detailed results for the COLMAP initialization can be found in the supplementary material. The detailed results on LLFF for NVS quality can be found in Table 2. We outperform NeRF-- on all scenes on LLFF and SiNeRF on seven out of eight scenes in PSNR. Overall we achieve better results for mean PSNR, mean SSIM and mean LPIPS compared to all other joint optimization methods. We also report visual results in Fig. 3, comparing our result on NVS to NeRF--. As shown in the room scene of LLFF, we better reconstruct fine-grained details in the focus of the image and in the corners.

Moreover, we outperform NeRF-- on BLEFF in PSNR, and perform equally in SSIM. The results for each scene are reported in Table 4.

On iFF we compared NeRF-- and NeRFtrinsic Four (OURS), as shown in Table 5. Here, we first tested only our intrinsic camera prediction module, as iFF contains diverse focal lengths. Using our intrinsic camera module alone, we achieved an improved NVS quality. Thereby, we already outperformed NeRF-- in PSNR and SSIM. Combined with our Gaussian Fourier feature mapping for the extrinsic camera prediction we achieved an even better NVS quality. As shown in Table 5, we outperform NeRF-- on all scenes in PSNR and SSIM.

4.6. Camera Parameter Estimation

4.6.1 Extrinsic Camera Parameters

For each scene in LLFF, we report the rotation and translation error of NeRF--, SiNeRF128 and our approach in Table 2. We outperform both approaches with NeRFtrinsic Four. Furthermore, we show the advantage of the SSIM loss, exemplarily for the scenes in LLFF in our supplementary material. For BLEFF and iFF, we report the detailed results of NeRF-- and NeRFtrinsic Four in Table 4 and in Table 5. Our approach shows an improved extrinsic camera parameter estimation on all datasets.

Positional Encoding Approaches like BARF [9] recognize the benefits of Fourier features for noisy pose estimation. However, they do not use Gaussian Fourier features

Skip/Scene	Fern		Flower		Fortress		Horns		Leaves		Orchids		Room		T-Rex	
	NeRF--	OURS	NeRF--	OURS	NeRF--	OURS	NeRF--	OURS	NeRF--	OURS	NeRF--	OURS	NeRF--	OURS	NeRF--	OURS
2																
3																
4																
5																
6																

Table 3: **Breaking point analysis of our Gaussian Fourier feature-based pose MLP vs. NeRF-- on LLFF.** We used every second, third, fourth, fifth or sixth image during training. A rotation error below 20° is considered as success (green). We outperform NeRF-- as we succeed on 31 scenes and NeRF-- only on 24.

Scene	PSNR		SSIM		Focal. Err.		Rot. Err.		Trans. Err.	
	NeRF--	OURS	NeRF--	OURS	NeRF--	OURS	NeRF--	OURS	NeRF--	OURS
Airplane	**30.57**	29.99	0.83	**0.86**	0.87	1.77	**0.61**	1.04	**0.003**	0.015
Balls	32.12	**34.39**	0.81	0.64	15.44	**14.53**	13.43	**0.81**	0.285	0.004
Bathroom	**31.58**	31.28	0.94	0.92	0.39	**0.19**	1.50	1.49	0.004	0.001
Bed	32.41	**33.82**	0.94	**0.95**	0.39	**0.02**	**2.21**	**2.21**	0.004	**0.001**
Castle	**32.74**	31.86	**0.89**	0.87	**3.23**	16.06	**3.17**	4.41	**0.020**	0.053
Chair	**32.24**	31.54	**0.81**	**0.81**	6.12	**5.74**	**3.52**	5.44	0.078	**0.006**
Classroom	**25.14**	24.90	0.86	**0.88**	2.29	**0.12**	8.14	**5.38**	0.032	**0.004**
Deer	**42.01**	41.63	**0.99**	**0.99**	**6.29**	31.46	6.17	**5.51**	0.166	**0.001**
Halloween	29.30	**32.11**	0.91	**0.94**	9.77	**6.53**	6.74	**0.97**	**0.022**	0.030
Jugs	**42.5**	42.09	**0.99**	**0.99**	**0.16**	0.99	2.30	**0.72**	0.065	**0.033**
Root	35.45	**37.00**	0.97	**0.98**	36.42	**19.54**	4.51	**0.45**	0.100	**0.016**
Roundtable	39.88	**39.91**	**0.99**	0.98	206.10	**17.92**	9.68	**2.72**	0.139	**0.020**
Stone	**31.74**	31.36	**0.86**	0.85	**0.11**	3.12	**0.12**	0.14	**0.001**	**0.001**
Valley	27.66	**27.92**	**0.75**	0.74	**0.05**	1.82	0.25	**0.15**	**0.001**	**0.001**
Mean	33.24	**33.57**	**0.90**	**0.90**	20.55	**8.57**	4.45	**2.15**	0.064	**0.018**

Table 4: **Quantitative comparison of NeRF-- [6] and NeRFtrinsic Four (OURS) on BLEFF.** We report PSNR, SSIM, translation error and rotation error. We outperform NeRF-- in PSNR, focal length error, rotation error and translation error.

but instead the so-called positional encoding. Our experiments show that the use of Gaussian Fourier features in comparison leads to a better convergence. For this purpose, we train our network with Gaussian Fourier features and positional encoding with ten different random seeds on each scene. The Gaussian Fourier features lead to an average rotation error of *26.91* compared to the positional encoding with an average rotation error of *45.95*. While the gap between the translation error remains small, the Gaussian Fourier features still outperform the positional encoding with an error of 0.039 compared to 0.042. The high error is caused by mirror poses in the ten runs. The runs are included in the score calculation to show that mirror poses are less likely with Gaussian Fourier features. A detailed overview of the error in each scene can be found in the supplementary material.

Breaking Point Analysis A breaking point analysis for our pose MLP shows an improved stability on the pose predictions by using fewer images. We compare our approach with NeRF-- on LLFF. A rotation error greater than 20° is considered as failed. For the training, we used every second, third, fourth, fifth and sixth image. As shown in Table 3, we succeed in 31 scenes, while NeRF-- only succeeds in 24.

4.6.2 Intrinsic Camera Parameters

In the focal length estimation, we outperform existing approaches on LLFF and BLEFF, see Table 1. This indicates, that the joint optimization and an improved extrinsic camera parameter estimation also supports the intrinsic camera parameter estimation. As described in detail in the supplementary material, we outperform NeRF-- on five out of eight scenes on LLFF in the focal length estimation. On BLEFF we achieve a better focal length estimation on eight out of 14 scenes, see Table 4. The scenes in both datasets are captured by a single camera. For this reason, we introduce iFF. The focal lengths in the scenes vary as we used different cameras and rescaling factors. While NeRF-- averages the scaling factor over all images, we learn the individual intrinsic camera parameter per given camera. To show the influence of varying camera parameters, we trained NeRF--, NeRF-- combined with our intrinsic camera parameter method and our NeRFtrinsic Four on iFF. As presented in Table 5, the network better estimates the focal lengths. Using only our intrinsic module we already show an improved focal length estimation. NeRFtrinsic Four pre-

Scene	PSNR			SSIM			Focal Err.			Rot Err.			Trans Err.		
	NeRF--	I	I+GF	NeRF--	I	I+GF	NeRF--	I	I+GF	NeRF--	I	I+GF	NeRF--	I	I+GF
T1	24.36	26.22	**26.77**	0.82	0.84	**0.85**	214.36	**55.89**	70.24	5.97	9.45	2.30	0.120	**0.018**	0.042
Brick House	21.09	23.98	**24.91**	0.66	**0.71**	**0.71**	254.73	**67.80**	89.99	**4.53**	4.49	5.44	0.075	0.090	**0.052**
Fireplug	22.52	22.95	**25.15**	0.62	0.63	**0.71**	267.89	123.24	**92.14**	**5.45**	5.61	5.51	0.006	**0.001**	0.007
Bike	15.38	17.72	**22.12**	0.32	0.55	**0.70**	173.21	132.38	**68.93**	20.77	19.99	**6.58**	0.042	**0.035**	0.060
Stormtropper	36.23	36.61	**36.67**	**0.97**	**0.97**	0.99	121.18	103.93	**51.51**	7.83	8.97	**7.47**	0.550	**0.540**	0.796
Mean	23.92	25.50	**27.12**	0.68	0.74	**0.80**	206.27	96.65	**74.56**	8.91	9.70	**5.46**	0.159	**0.136**	0.191

Table 5: **Quantitative comparison between NeRF-- [6], NeRF-- and our improved intrinsic estimation (I) and NeRFtrinsic Four (I+GF) on iFF.** We report PSNR, SSIM, rotation error, translation error and focal length error.

Figure 4: **Visual results on iFF.** As shown in scene T1 (left), our approach (right) provides a more accurate image compared to NeRF-- (center). For example, the rear part of the car and the plant show a more accurate result with NeRFtrinsic Four. This demonstrates the influence of a correct focal length prediction when comparing both approaches.

dicts the focal length even better, possibly due to the improved prediction of the camera pose. A more accurate focal length has an impact on the results, as demonstrated in Fig. 4. Unlike the image produced by NeRF--, the image produced by NeRFtrinsic Four shows no distortion and is more accurate compared to the ground truth.

5. Discussion

Our breaking point analysis shows that we outperform NeRF-- on LLFF and iFF. However, on some scenes our method needs a sufficient number of images for successful joint optimization, as indicated in Table 3 and the supplementary material. When dealing with multiple intrinsic camera parameters on iFF, the same conclusion emerges. Nevertheless, our approach can handle diverse intrinsic camera parameters while NeRF-- learns only an average over the input. Fig. 4 shows that accurately estimated intrinsic camera parameters are crucial for a valid image representation. While an average over the images can lead to high distortions, our precise estimation of differing intrinsic camera parameters supports accurate image synthesis.

We also observe that a rotation error of 180° leads to a mirror pose. This was already reported by Wang et al. [6]. We provide an example image in the supplementary material. While mirror poses still lead to a high PSNR value, the position of objects in the scene is shifted. To further stabilize the pose MLP, we added the SSIM loss. Our supplementary material shows that our approach with the SSIM loss achieves better convergence for a fixed random seed. However, using the loss over the whole training process leads to a degradation in performance for the PSNR value.

Our experiments highlight the importance of accurate intrinsic and extrinsic camera parameter estimations for an end-to-end trainable NeRF. Incorrect extrinsic camera parameters can shift objects in the scene, while incorrect intrinsic camera parameters can distort the resulting image.

6. Limitations

A limitation of our work, inherited from previous research [6, 11], is the focus on forward-facing scenes. This means that our approach is not suitable for 360° scenes. While current approaches that can handle non-forward-facing scenes exist [9, 26, 10], they typically rely on noisy poses or a pretrained NeRF for initialization, which are not transferable to real-world scenarios. Additionally, we found that predicting extrinsic camera parameters becomes more challenging in our joint optimization approach when dealing with varying camera intrinsic parameters.

Moreover, the initialization of the pose MLP is challenging. Therefore, we see potential in a regularization method for coordinate based MLPs [35].

7. Conclusion

We introduce NeRFtrinsic Four, an end-to-end trainable NeRF that jointly optimizes the extrinsic camera parameters, the intrinsic camera parameters, and the scene repre-

sentation. Unlike other joint optimization frameworks, it is not limited to one type of camera. To validate our approach, we present our real-world iFF dataset, which demonstrates that NeRFtrinsic Four achieves better results compared to NeRF-- when a diverse set of cameras is used. Additionally, our work outperforms the joint optimization frameworks NeRF-- and SinNeRF on existing datasets. By using our Gaussian Fourier feature-based pose MLP, we achieve a better camera pose estimation, which consequently enables an improved estimation of the intrinsic camera parameters and higher-quality NVS results. In summary, NeRFtrinsic Four allows the estimation of both diverse intrinsic and extrinsic camera parameters in joint optimization with NeRF.

References

[1] B. Heigl, R. Koch, M. Pollefeys, J. Denzler, and L. Van Gool, "Plenoptic modeling and rendering from image sequences taken by a hand-held camera," in *Mustererkennung 1999: 21. DAGM-Symposium Bonn, 15.–17. September 1999*. Springer, 1999, pp. 94–101. 2

[2] M. Levoy and P. Hanrahan, "Light field rendering," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 1996, pp. 31–42. 2

[3] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing Scenes as Neural Radiance Fields for View Synthesis," in *European Conference on Computer Vision*. Springer, 2020, pp. 405–421. 2, 4, 6

[4] R. Szeliski and P. Golland, "Stereo matching with transparency and matting," in *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*. IEEE, 1998, pp. 517–524. 2

[5] Q. Meng, A. Chen, H. Luo, M. Wu, H. Su, L. Xu, X. He, and J. Yu, "GNeRF: GAN-Based Neural Radiance Field Without Posed Camera," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2021, pp. 6351–6361. 2, 3

[6] 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. 2, 3, 4, 5, 6, 7, 8, 14, 15, 16

[7] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4104–4113. 2

[8] R. Martin-Brualla, N. Radwan, M. S. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. 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*, 2021, pp. 7210–7219. 2

[9] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey, "BARF: Bundle-Adjusting Neural Radiance Fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2021, pp. 5741–5751. 2, 3, 6, 8, 13, 14, 16

[10] L. Yen-Chen, P. Florence, J. T. Barron, A. Rodriguez, P. Isola, and T.-Y. Lin, "iNeRF: Inverting Neural Radiance Fields for Pose Estimation," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 1323–1330. 2, 3, 8

[11] Y. Xia, H. Tang, R. Timofte, and L. V. Gool, "SiNeRF: Sinusoidal Neural Radiance Fields for Joint Pose Estimation and Scene Reconstruction," in *33rd British Machine Vision Conference 2022, BMVC 2022, London, UK, November 21-24, 2022*. BMVA Press, 2022. [Online]. Available: https://bmvc2022.mpi-inf.mpg.de/0131.pdf 2, 3, 5, 6, 8, 15, 16

[12] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar, "Local light field fusion: Practical view synthesis with prescriptive sampling guidelines," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–14, 2019, publisher: ACM New York, NY, USA. 2, 5, 13

[13] E. Brachmann and C. Rother, "Visual camera relocalization from RGB and RGB-D images using DSAC," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 9, pp. 5847–5865, 2021, publisher: IEEE. 2

[14] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981, publisher: ACM New York, NY, USA. 2

[15] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007, publisher: IEEE. 2

[16] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap

slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021, publisher: IEEE. 2

[17] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II 13*. Springer, 2014, pp. 834–849. 2

[18] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 15–22. 2

[19] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2011, pp. 2320–2327. 2

[20] T. E. Lee, J. Tremblay, T. To, J. Cheng, T. Mosier, O. Kroemer, D. Fox, and S. Birchfield, "Camera-to-robot pose estimation from a single image," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9426–9432. 2

[21] A. Elmoogy, X. Dong, T. Lu, R. Westendorp, and K. Reddy, "Pose-GNN: Camera pose estimation system using graph neural networks," *arXiv preprint arXiv:2103.09435*, 2021. 2

[22] T. H. Butt and M. Taj, "Camera Calibration Through Camera Projection Loss," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 2649–2653. 2

[23] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, "Mip-nerf: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5855–5864. 2

[24] S. Kulkarni, P. Yin, and S. Scherer, "360FusionNeRF: Panoramic Neural Radiance Fields with Joint Guidance," *arXiv preprint arXiv:2209.14265*, 2022. 2

[25] Y. Jeong, S. Ahn, C. Choy, A. Anandkumar, M. Cho, and J. Park, "Self-calibrating Neural Radiance Fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5846–5854. 2

[26] P. Truong, M.-J. Rakotosaona, F. Manhardt, and F. Tombari, "SPARF: Neural Radiance Fields from Sparse and Noisy Poses," *arXiv preprint arXiv:2211.11738*, 2022. 2, 3, 8, 13, 16

[27] S.-F. Chng, S. Ramasinghe, J. Sherrah, and S. Lucey, "Gaussian Activated Neural Radiance Fields for High Fidelity Reconstruction and Pose Estimation," in *European Conference on Computer Vision*. Springer, 2022, pp. 264–280. 3

[28] R. Hartley and A. Zisserman, "Multiple View Geometry In Computer Vision." Cambridge University Press, 2004, p. 153 ff, section: 6. 3

[29] M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng, "Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 7537–7547. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/55053683268957697aa39fba6f231c68-Paper.pdf 3

[30] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004, publisher: IEEE. 5

[31] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595. 5

[32] Z. Zhang and D. Scaramuzza, "A Tutorial on Quantitative Trajectory Evaluation for Visual(-Inertial) Odometry," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2018. 5

[33] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 573–580. 5

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2015, pp. 1026–1034. 5

[35] S. Ramasinghe, L. MacDonald, and S. Lucey, "On Regularizing Coordinate-MLPs," Feb. 2022, arXiv:2202.00790 [cs]. [Online]. Available: http://arxiv.org/abs/2202.00790 8

[36] G. Schweighofer and A. Pinz, "Robust Pose Estimation from a Planar Target," *IEEE Transactions on*

Pattern Analysis and Machine Intelligence, vol. 28, no. 12, pp. 2024–2030, Dec. 2006, conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence. 14

[37] D. Oberkampf, D. DeMenthon, and L. Davis, "Iterative Pose Estimation Using Coplanar Feature Points," *Computer Vision and Image Understanding*, vol. 63, pp. 495–511, May 1996. 14

Supplementary Material

Scene	Pos. Enc.		Gaussian F.F.	
	Rot. Err.	Trans. Err.	Rot. Err.	Trans. Err.
Fern	36.18	0.035	**23.28**	**0.026**
Flower	39.64	**0.025**	**5.36**	0.027
Fortress	38.05	**0.033**	**27.47**	0.047
Horns	46.51	**0.051**	**4.90**	**0.051**
Leaves	42.55	**0.020**	**4.03**	0.022
Orchids	**22.48**	**0.025**	23.73	**0.025**
Room	64.49	0.086	**41.03**	**0.069**
T-Rex	**77.77**	0.065	85.50	**0.052**
Mean	45.95	0.042	**26.91**	**0.039**

Table S1: **Quantitative comparison between positional encoding and Gaussian Fourier features on LLFF.** The results are computed after 500 epochs. The high error values are the result of ten random seeds for each scene.

Scene	PSNR		Rot. Err.		Trans. Err.	
	64	256	64	256	64	256
Fern	21.63	22.75	1.59	0.94	0.009	0.004
Flower	24.66	25.95	3.98	1.59	0.023	0.017
Fortress	27.24	28.39	1.30	1.32	0.014	0.015
Horns	22.46	25.20	1.05	1.45	0.014	0.005
Leaves	17.24	19.00	4.26	4.03	0.012	0.016
Orchids	17.39	16.87	2.86	4.12	0.005	0.024
Room	27.04	27.54	0.97	1.15	0.009	0.036
T-Rex	23.04	24.25	3.36	3.85	0.012	0.009
Mean	22.59	**23.74**	2.42	**2.31**	**0.012**	0.016

Table S2: **Quantitative comparison between two different embedding sizes for the pose MLP, i.e. 64 and 256 on LLFF.** The embedding size of 256 leads to an higher PSNR and lower rotation error.

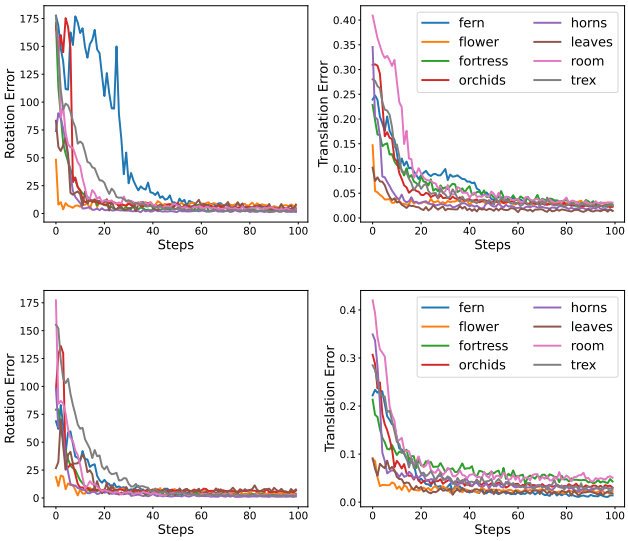

Figure S1: **Rotation error and translation error for the first 100 epochs between two different embedding sizes i.e. 64 (top) and 256 (bottom).** The embedding size 256 shows less outliers compared to an embedding size of 64.

1. Camera Parameter Experiments

NeRFtrinsic Four aims to improve the extrinsic camera parameter estimation using Gaussian Fourier features and to predict differing intrinsic camera parameters if they are given in a dataset. We first show a deeper analysis of extrinsic camera parameter estimation. Secondly, we show additional results for the camera intrinsic parameter estimation. We further analyze novel view synthesis. Lastly, we provide details about the focal length and example images of our iFF dataset.

1.1. Extrinsic Camera Parameters

For the extrinsic camera parameters we conducted additional experiments. In these we compare positional encoding and Gaussian Fourier features as well as different embedding sizes for our pose MLP. Furthermore, we analyze mirror poses, investigate the COLMAP initialization, the influence of the SSIM loss and conduct another breaking point analysis.

1.1.1 Positional Encoding

Our pose MLP uses Gaussian Fourier features while another approach is the use of positional encoding [9, 26]. Therefore, we test positional encoding in our pose MLP. We train our pose MLP on each scene of the LLFF dataset [12]. For training we use ten random seeds. In Table S1 we compare the results of Gaussian Fourier features and positional encoding after 500 epochs. Gaussian Fourier features averagely outperform positional encoding in mean rotation and translation error over the ten runs. The high mean errors occur due to mirror poses in the ten runs. We also include the experiments with a mirror pose in our calculation to show that mirror poses are less likely to occur when using Gaussian Fourier features.

1.1.2 Embedding Size

In addition to testing positional encoding, we investigate the influence of different embedding sizes for the pose MLP, see Table S2. We apply Gaussian Fourier feature mapping and use the embedding sizes 64 and 256 respectively. As visualized in Fig. S1 the embedding size of 64 shows more outliers for the rotation convergence, for example in the fern and orchids scene, and for the translation convergence in the

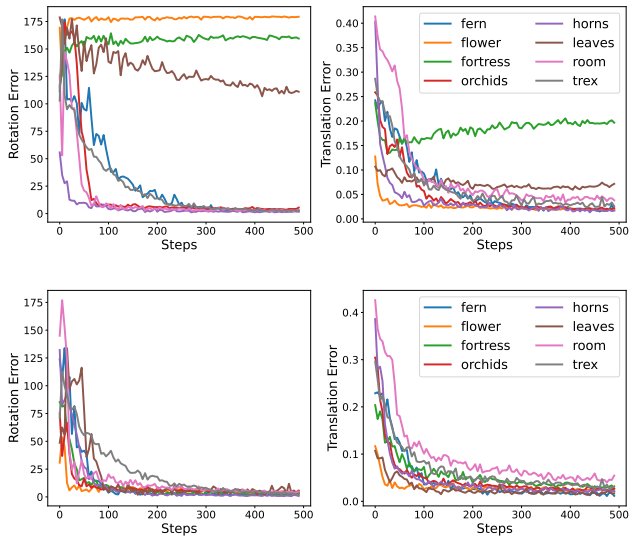

Figure S1: **Comparison of ATE statistics on all LLFF scenes.** We show the comparison without SSIM loss (top) and with SSIM loss (bottom) for 500 epochs.

room scene. Based on the results showing more stability of 256, we select the embedding size of 256 for our pose MLP.

1.1.3 Mirror Poses

Wang et al. [6] mentioned the occurrence of mirror poses when using COLMAP. In their experiments on the BLEFF dataset, COLMAP shows an error close to $180°$ for the roundtable scene. For the parameter-based or MLP-based pose estimation similar errors can appear. This can occur due to an ambiguous pose estimation caused by several local minima in the loss function [36, 37].

Nevertheless, the NeRF does not fail to learn and generate accurate novel views. Despite high image quality, a clear shift of the objects in the scene is visible. Especially for the roundtable scene, low rotation errors lead to a better result in the rotation comparison. Still, the visual novel view quality is poor despite high SSIM and PSNR values. An example of this can be seen in Fig. S1. The picture in the center of Fig. S1 shows a mirror pose that occurred during training with a train PSNR of 47.13 and an average rotation error above $150°$. However, the generated image is of high quality compared to the image on the right. There, the scene has a comparatively low average rotation error of below $30°$, but only a train PSNR of 39.69. Although the rotation error in the right image is $< 30°$, the PSNR value for the middle figure is much higher. Nevertheless, the outline of the ground truth scene shows a highly incorrect camera pose in the center image. Thus, both metric types, NVS quality metrics and ATE, are decisive for joint optimization approaches.

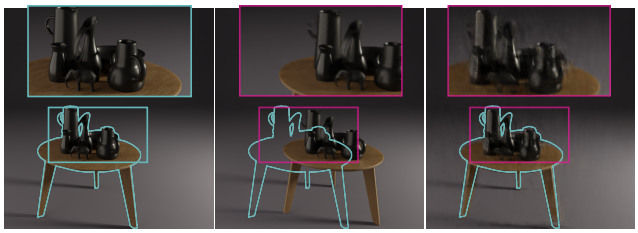

Figure S1: **Comparison of the same scene of BLEFF and same frame at different rotation error levels.** We highlight the original shift from the ground truth scene (left) with a blue outline around the table. In the pink rectangles we show the same cutout from the same coordinates in the image.

1.1.4 SSIM Loss

To test whether the additional SSIM loss supports the pose estimation, we train on each scene of LLFF for 500 epochs with fixed seeds. As can be observed in Fig. S1 the altered training objective leads to a convergence in scenes where a mirror pose would otherwise occur. The convergence speed for the rotation and translation error also increases. Since the PSNR is lower when SSIM loss is used in the whole training process, we apply it only in the first 500 epochs.

1.1.5 Breaking Point Analysis on iFF

Our main paper includes a breaking point analysis on LLFF, where we outperform NeRF--. We also run this comparison on iFF. A rotation error lower than $20°$ is considered as success. For the training, we used every second, third, fourth and fifth image. As shown in Table S3, we succeed in one more scene than NeRF--. This allows the conclusion, that differing intrinsic camera parameters make the estimation of camera extrinsic parameters more challenging.

1.1.6 COLMAP Initialization for Camera Parameter Estimation

NeRFtrinsic Four per se learns the camera parameters from scratch. Besides learning the camera parameters without initializing the pose MLP, COLMAP preprocessed poses can be used for its initialization. Research on $360°$ pose estimation and joint NVS optimization initializes the pose network either directly with COLMAP or refines from noisy poses also calculated from COLMAP poses [9]. We initialize the pose MLP with COLMAP poses and then train the joint optimization. The results are denoted in Table S4. Again, the results show that our improved pose prediction reduces the focal length error and improves NVS.

Skip/Scene	Fireplug		Stormtrooper		T1		Bike		Brick House	
	NeRF--	OURS	NeRF--	OURS	NeRF--	OURS	NeRF--	OURS	NeRF--	OURS
2										
3										
4										
5										

Table S3: **Breaking point analysis of our Gaussian Fourier feature-based Pose MLP vs. NeRF-- on iFF.** We used every second, third, fourth or fifth image during training. A rotation error below 20° is considered as success (green). We slightly outperform NeRF-- as we succeed on 15 scenes while NeRF-- only succeeds on 14 scenes.

Scene	Rot. Err.				Trans. Err.				Focal. Err.			
	NeRF--	SiNeRF	OURS	OURS$^{(COLMAP)}$	NeRF--	SiNeRF	OURS	OURS$^{(COLMAP)}$	NeRF--	SiNeRF	OURS	OURS$^{(COLMAP)}$
Fern	1.78	1.17	1.34	**0.41**	0.029	0.006	0.006	**0.001**	153.5	112.7	161.6	**1.76**
Flower	4.84	1.38	0.89	**0.39**	0.016	0.007	0.007	**0.002**	**13.2**	80.4	53.6	**18.63**
Fortress	1.36	2.02	0.91	**0.15**	0.025	0.048	0.006	**0.002**	144.1	59.7	129.3	**1.05**
Horns	5.55	83.34	1.89	**0.41**	0.044	0.133	0.014	**0.002**	156.2	282.5	92.5	**25.8**
Leaves	3.90	14.46	**2.65**	5.04	0.016	0.10	0.005	**0.004**	59.0	18.8	24.4	**10.3**
Orchids	4.96	3.97	2.75	**0.31**	0.051	0.014	0.009	**0.004**	199.3	155.7	165.9	**11.96**
Room	2.77	4.92	1.33	**0.14**	0.030	0.022	0.006	**0.001**	331.8	313.6	**102.7**	**6.06**
T-Rex	4.67	7.19	**3.90**	**0.14**	0.036	0.027	0.008	**0.002**	89.3	38.3	100.4	**7.54**
Mean	3.73	14.81	**2.10**	**0.83**	0.031	0.033	0.008	**0.002**	143.3	155.2	103.7	**10.4**

Table S4: **Quantitative comparison between NeRFtrinsic Four (OURS), NeRFtrinsic Four initialized with COLMAP (OURS^{COLMAP}), NeRF-- [6] and SiNeRF [11] on LLFF.** For SiNeRF we retrained the approach with a layer dimension of 128 to ensure comparability with our approach and NeRF--. We report the camera extrinsic parameter errors, in detail translation and rotation error. We tested ten random seeds for SiNeRF; for NeRF-- we report the values from Wang et. al [6].

Scene	Focal. Err.		
	NeRF--	SiNeRF	OURS
Fern	153.5	**112.7**	161.6
Flower	**13.2**	80.4	53.6
Fortress	144.1	**59.7**	129.3
Horns	156.2	282.5	**92.5**
Leaves	59.0	198.8	**24.4**
Orchids	199.3	**155.7**	165.9
Room	331.8	313.6	**102.7**
T-Rex	89.3	**38.3**	100.4
Mean	143.3	155.2	**103.7**

Table S5: **Quantitative comparison between NeRFtrinsic Four (OURS), NeRF-- [6] and SiNeRF [11] with a NeRF layer dimension of 128 on LLFF.** We report the focal pixel error. Our approach outperforms NeRF-- [6] and SiNeRF [11].

1.2. Intrinsic Camera Parameters

In addition to the extrinsic camera parameter estimation, we compare our intrinsic camera parameter estimation against NeRF-- and SiNeRF on LLFF. As denoted in Table S5, we outperform both approaches in the mean focal length prediction. While we clearly outperform NeRF-- in the individual values, SiNeRF shows comparable results. However, SiNeRF produces a significant error in the room scene. Thus, their mean focal length error suffers. When excluding the room scene from the mean value calculation, we still outperform SiNeRF, with a focal pixel error of 103.9 compared to 132.6.

Moreover, the COLMAP initialization tested on LLFF shows also a lower focal length error. As denoted in Table S4, a reduced pose error leads to a lower focal length error.

1.3. A Higher NeRF Layer Dimension and its Influence on Camera Parameters

NeRFtrinsic Four is off-the-shelf comparable with NeRF--, as we use a layer dimension of 128. SiNeRF was originally trained with a layer dimension of 256. When applying this layer dimension to our NeRF, the results for the intrinsic and extrinsic camera parameters change due to the joint optimization. For the focal length we report a mean error of 72.12. With a layer dimension of 128 we reported a mean focal length error of 103.7, see Table S5. The extrinsic camera parameter results are denoted in Table S7. NeRFtrinsic Four outperforms the other joint optimization approaches in rotation error and translation error when using a layer dimension of 256.

Scene	PSNR↑				SSIM↑				LPIPS↓			
	NeRF	NeRF--	OURS	OURS$^{(COLMAP)}$	NeRF	NeRF--	OURS	OURS$^{(COLMAP)}$	NeRF	NeRF--	OURS	OURS$^{(COLMAP)}$
Fern	**22.22**	21.67	21.82	22.19	**0.64**	0.61	0.62	**0.64**	0.47	0.50	0.49	0.41
Flower	25.25	25.34	25.39	**25.78**	0.71	0.71	0.72	**0.73**	0.36	0.37	0.37	**0.35**
Fortress	27.60	26.20	27.28	**28.13**	0.73	0.63	0.70	**0.74**	0.38	0.49	0.41	**0.37**
Horns	24.25	22.53	24.00	**24.45**	0.68	0.61	0.67	**0.68**	**0.44**	0.50	0.45	**0.44**
Leaves	18.81	18.88	**18.97**	16.65	0.52	**0.53**	0.50	0.43	**0.47**	**0.47**	0.49	0.53
Orchids	19.09	16.73	17.41	**19.30**	0.51	0.39	0.43	**0.52**	**0.46**	0.55	0.52	**0.46**
Room	**27.77**	25.84	27.12	27.72	**0.87**	0.84	0.82	**0.87**	0.40	0.44	0.43	**0.41**
T-Rex	23.19	22.67	22.92	**23.20**	**0.74**	0.72	**0.74**	**0.74**	**0.41**	0.44	0.43	**0.41**
Mean	**23.52**	22.48	23.14	23.20	0.68	0.63	0.66	**0.74**	0.42	0.47	0.44	**0.41**

Table S6: **Quantitative comparison between NeRFtrinsic Four (OURS), NeRFtrinsic Four with COLMAP initialization (OURS^{COLMAP}), NeRF-- [6] and COLMAP-based NeRF [6] on LLFF**. We follow the results of Wang et. al [6] for COLMAP-based NeRF. Thus, the reported NVS quality is overall lower compared to vanilla NeRF as the layer size is 128 instead of 256. We report PSNR, SSIM and LPIPS. When using COLMAP initialization we outperform COLMAP-based NeRF in NVS quality.

Scene	PSNR↑			SSIM↑			LPIPS↓			Rot. Err.			Trans. Err.		
	NeRF--	SiNeRF	OURS	NeRF--	SiNeRF	OURS	NeRF--	SiNeRF	OURS	NeRF--	SiNeRF	OURS	NeRF--	SiNeRF	OURS
Fern	22.15	22.48	**22.75**	0.65	0.67	**0.69**	0.46	0.44	**0.41**	1.57	0.74	0.94	0.008	0.004	0.004
Flower	26.61	**27.23**	25.95	0.77	**0.80**	0.74	0.30	0.30	0.33	3.21	0.51	2.3	0.012	0.008	0.022
Fortress	25.60	27.47	**28.39**	0.60	0.72	**0.77**	0.54	0.39	**0.31**	2.41	1.77	1.32	0.060	0.041	0.015
Horns	23.17	24.14	**25.20**	0.64	0.68	**0.73**	0.51	0.43	**0.37**	3.04	2.66	1.45	0.015	0.022	0.005
Leaves	**19.74**	19.15	19.00	**0.61**	0.57	0.55	**0.39**	**0.39**	0.43	6.78	8.76	4.03	0.006	0.008	0.016
Orchids	15.86	**16.92**	16.87	0.35	**0.41**	**0.41**	0.55	0.53	**0.50**	5.46	3.24	4.19	0.022	0.013	0.024
Room	25.68	26.10	**27.54**	0.84	0.84	**0.87**	0.41	0.43	**0.37**	3.75	2.08	1.15	0.021	0.021	0.036
T-Rex	23.38	**24.94**	24.25	0.76	**0.82**	0.79	0.39	0.36	0.36	6.34	0.86	3.36	0.015	0.005	0.009
Mean	22.77	23.55	**23.74**	0.65	**0.69**	**0.69**	0.44	0.41	**0.39**	4.07	2.58	**2.32**	0.019	**0.015**	0.016

Table S7: **Quantitative comparison between NeRFtrinsic Four (OURS), NeRF-- [6] and SiNeRF [11] on LLFF, with a layer dimension of 256 for all NeRF frameworks.** We follow Xia et. al [11] for the results of NeRF-- with a layer dimension of 256. We report PSNR, SSIM, LPIPS and the extrinsic camera error, in detail translation and rotation error. NeRFtrinsic Four outperforms NeRF-- [6] and SiNeRF in PSNR and LPIPS.

2. Novel View Synthesis

2.1. COLMAP Initialization for NVS Quality

Initializing the pose MLP with COLMAP shows an improved prediction of the intrinsic and extrinsic camera parameters, see Table S4. Our further experiments demonstrate that a better pose estimation leads to an improved NVS. Table S6 shows the PSNR, SSIM and LPIPS results for all scenes on LLFF. The NVS quality is higher in the joint optimization than when purely training on COLMAP poses. These findings corrospond with other related work [9, 26].

2.2. Higher Layer Dimension for NVS Quality

In our ablation study we compare NeRFtrinsic Four with the original SiNeRF and their adapted NeRF-- with a layer dimension of 256, see Table S7. NeRFtrinsic Four outperforms both other joint optimization approaches. We achieve better results in PSNR and LPIPS. For SSIM we perform equally compared to SiNeRF but outperform NeRF--.

Scene	Focal Lengths
T1	740.90, 1673.73, 2980.71
Fireplug	3531.73, 2663.47, 683.74
Bike	3022.43, 756.16
Stormtrooper	3022.43, 756.16
Brick House	3000.51, 1483.85, 3152.86, 746.25

Table S8: **Focal lengths of the individual scenes from iFF.** The camera parameters were estimated using COLMAP.

3. Dataset

In our iFF dataset we captured five real world scenes. These five scenes are namely T1, brick house, bike, fireplug and stormtrooper. Example images from iFF are depicted in Fig. S2. The scenes are captured with an OAK-D Lite camera with varying resolution, an iPhone mini 13 and an iPad Air 2. We also applied varying resizing factors to later receive more differing intrinsic estimations from COLMAP. The focal lengths of the individual scenes are denoted in

(a) Brick House (b) Stormtrooper (c) Bike (d) Fireplug (e) T1

Figure S2: **Example images from iFF.** We have three indoor and two outdoor scenes. The brick house is captured with an iPad Air 2 and an iPhone 13 mini. The stormtrooper, bike and fireplug are captured with an iPhone 13 mini with different resolutions. The T1 is captured with an iPhone 13 mini and the OAK D-Lite camera.

Table S8. All scenes were captured from a video stream. From this video stream we extract one frame per second. This results in the final images of the dataset. To receive the pseudo ground truth we applied COLMAP.

4. Acknowledgment

The authors gratefully acknowledge the computing time granted by the Institute for Distributed Intelligent Systems and provided on the GPU cluster Monacum One at the University of the Bundeswehr Munich