CF-NeRF: Camera Parameter Free Neural Radiance Fields with Incremental Learning

Qingsong Yan\textsuperscript{1}, Qiang Wang\textsuperscript{2,\ast}, Kaiyong Zhao\textsuperscript{3}, Jie Chen\textsuperscript{4}, Bo Li\textsuperscript{5}, Xiaowen Chu\textsuperscript{6,5,\ast}, Fei Deng\textsuperscript{1,7}
\textsuperscript{1}Wuhan University, Wuhan, China, \textsuperscript{2}Harbin Institute of Technology (Shenzhen), Shenzhen, China
\textsuperscript{3}XGRIDS, Shenzhen, China, \textsuperscript{4}Hong Kong Baptist University, Hong Kong SAR, China
\textsuperscript{5}The Hong Kong University of Science and Technology, Hong Kong SAR, China
\textsuperscript{6}The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China
\textsuperscript{7}Hubei Luojia Laboratory, Wuhan, China
yanqs_whu@whu.edu.cn, qiang.wang@hit.edu.cn, kyzhao@xgrids.com, chenjie@comp.hkbu.edu.hk
bli@cse.ust.hk, xwchu@ust.hk, fdeng@sgg.whu.edu.cn

Abstract
Neural Radiance Fields have demonstrated impressive performance in novel view synthesis. However, NeRF and most of its variants still rely on traditional complex pipelines to provide extrinsic and intrinsic camera parameters, such as COLMAP. Recent works, like NeRFmm, BARF, and L2G-NeRF, directly treat camera parameters as learnable and estimate them through differential volume rendering. However, these methods work for forward-looking scenes with slight motions and fail to tackle the rotation scenario in practice. To overcome this limitation, we propose a novel camera parameter free neural radiance field (CF-NeRF), which incrementally reconstructs 3D representations and recovers the camera parameters inspired by incremental structure from motion. Given a sequence of images, CF-NeRF estimates camera parameters of images one by one and reconstructs the scene through initialization, implicit localization, and implicit optimization. To evaluate our method, we use a challenging real-world dataset, NeRFBuster, which provides 12 scenes under complex trajectories. Results demonstrate that CF-NeRF is robust to rotation and achieves state-of-the-art results without providing prior information and constraints.

Introduction
3D reconstruction is a hot topic in computer vision that aims to recover 3D geometry from RGB images. However, traditional methods contain lots of complex procedures, such as feature extraction and matching (Lowe 2004; Yi et al. 2016), sparse reconstruction (Agarwal et al. 2011; Wu 2013; Schonberger and Frahm 2016; Moulon et al. 2016), and dense reconstruction (Yao et al. 2018; Mi, Di, and Xu 2022; Yan et al. 2023). Consequently, traditional methods are not a differential end-to-end reconstruction pipeline and require high-quality results from each sub-module to achieve accurate results. When the quality of results is poor, it is challenging to identify which module is causing the problem.

Recently, Neural Radiance Fields (NeRF) (Mildenhall et al. 2020; Yu et al. 2021a; Müller et al. 2022) have demonstrated a novel way to render highly realistic novel views with impressive quality. Without recovering 3D geometry, NeRF relies on multi-layer perception (MLP) to predict color and sigma for each point in the scene and samples several points along a ray to render a pixel through differential volume rendering. Unlike traditional 3D reconstruction, NeRF simplifies the reconstruction into one step and implicitly represents the 3D scene. Benefiting from the excellent ability of NeRF, it has been further extended to dynamic scenes (Pumarola et al. 2021), large-scale (Turki, Ramanan, and Satyanarayanan 2022), and even surface (Wang et al. 2021a) and material reconstruction (Boss et al. 2021a).
Related Work

In this section, we introduce the development of NeRF-related methods with known camera parameters and several camera parameter estimation methods using SfM&SLAM (simultaneous localization and mapping) and the NeRF.

NeRF

NeRF (Mildenhall et al. 2020) uses the MLP to represent the 3D scene implicitly and can be trained through differential volume rendering from a set of images with known camera parameters. However, NeRF suffers from efficiency and needs around 1-2 days to train a scene and several minutes to render a novel view at the testing. Instant-NGP (Müller et al. 2022) builds a multi-resolution hash table to store space-aware feature vectors and reduces the complexity of the MLP network. Meanwhile, (Sun, Sun, and Chen 2022; Fridovich-Keil et al. 2022) try to use the coarse-to-fine strategy and (Yu et al. 2021a; Chen et al. 2022; Garbin et al. 2021) update the network structure to speed up training or testing. Besides, NeRF faces another problem that it cannot work for large-scale, unbounded 3D scenes. NeRF++ (Zhang et al. 2020) and MipNeRF360 (Barron et al. 2021, 2022) utilize different sampling strategies for foreground and background to model unbounded 3D scenes by a finite volume. MegaNeRF (Turki, Ramanan, and Satyanarayanan 2022) and BlockNeRF (Tancik et al. 2022) split a large scene into multiple small regions and assign a network for each part. Moreover, (Martin-Brualla et al. 2021; Pumarola et al. 2021; Attal et al. 2021) extend NeRF to dynamic scenes and (Jain, Tancik, and Abbeel 2021; Yu et al. 2021b; Niemeyer et al. 2022; Kim, Seo, and Han 2022) introduce context or geometry information into NeRF to suit scenes with sparse views. In addition to the advances in novel view synthesis, NeRF has made significant progress in geometric reconstruction (Yariv et al. 2021; Wang, Skorokhodov, and Wonka 2022; Darmon et al. 2022; Long et al. 2023; Fu et al. 2022). UniSURF (Oechsle, Peng, and Geiger 2021) and NeUS (Wang et al. 2021a) estimate the zero-level set of an implicit signed distance function instead of the space density. Furthermore, some work (Zhang et al. 2021; Verbin et al. 2022; Boss et al. 2021a; Kuang et al. 2022; Boss et al. 2021b, 2022) even combines BRDF and NeRF to decompose a scene into shape, reflectance, and illumination. However, all of these methods split the reconstruction into two steps and require traditional methods to provide camera parameters, which significantly limits the application of NeRF.

Camera Parameter Estimation

Traditional SfM (Wu 2013; Moulon, Monasse, and Marlet 2013; Schönberger and Frahm 2016; Moulon et al. 2016) and SLAM (Mur-Artal, Montiel, and Tardos 2015; Engel, Koltun, and Cremers 2017) can estimate camera parameters for given images. However, these methods divide the reconstruction pipeline into several non-differentiable modules that need hand-crafted features (Lowe 2004) or learning-based methods (Yi et al. 2016; Teed and Deng 2020) to establish image correspondences, and then reconstruct a sparse scene and camera parameters through multi-view geometry.
Figure 2: The pipeline of CF-NeRF. CF-NeRF can estimate the weight $\theta$ of NeRF $F$ and the camera parameter $\delta$. After initializing through a few selected images, CF-NeRF recovers $\delta$ of the image one by one through implicit localization that only optimizes the newly added image and implicit optimization that refines $\theta$ and $\delta$. Implicit optimization can be divided into partial and global optimization depending on the number of images used. We visualize $\delta$ reconstructed by CF-NeRF and sparse points from COLMAP (Schonberger and Frahm 2016) to show that CF-NeRF can reconstruct rotation in image sequences.

In light of these limitations, it is worth exploring to estimate camera parameters during the training process of NeRF. The most direct attempt to utilize NeRF is the visual localization, where iNeRF (Yen-Chen et al. 2021), NeDDF (Ueda et al. 2022), and PNeRFP (Lin et al. 2023) try to estimate the extrinsic camera parameter of a new image by a pre-trained NeRF model. Then, NeRFmm (Wang et al. 2021b) and SiRENmm (Guo and Sherwood 2021) take the NeRF and camera parameters as learnable and prove that it is possible to train the NeRF model from scratch without camera parameters, but they only work for forward-looking scenes. To further enhance accuracy in forward-looking or rotation scenes with initial camera parameters, BARF (Lin et al. 2021) dynamically adjusts the weight of the positional encoding, GARF (Chng et al. 2022) replaces the ReLU activate function with the Gaussian activation function, and L2G-NeRF (Chen et al. 2023) introduces a local-to-global registration. Interestingly, GNeRF (Meng et al. 2021) and VMRF (Zhang et al. 2022) assume there is a prior known distribution of camera parameters to decrease the freedom of camera parameters during training the NeRF model. Meanwhile, other researchers try to add different external restrictions to guide the camera parameter estimation. SCNeRF (Jeong et al. 2021) and Level-$S^2$FM (Xiao et al. 2023) rely on feature matches to guide camera parameters estimation. NoPe-NeRF (Bian et al. 2023), iMap (Sucar et al. 2021), NeRF-SLAM (Rosinol, Leonard, and Carlone 2022), Nice-SLAM (Zhu et al. 2022), and Nicer-SLAM (Zhu et al. 2023) integrate depth maps from active sensors or CNN networks to tune the NeRF. Additionally, LocalLR (Meuleman et al. 2023) combines depth maps and optical flow to train NeRF.

Regrettably, images acquired from real-world scenarios often exhibit a multitude of challenges. These challenges include rotations and the absence of prior information of camera parameters. Furthermore, the introduction of external constraints can augment the intricacy and unpredictability of the reconstruction process. To solve these problems, we propose CF-NeRF inspired by the traditional incremental SfM, which does not require any prior information or external constraints while reconstructing the 3D scene and camera parameters end-to-end from image sequences, demonstrating the powerful reconstruction capability of the NeRF after using a specific training strategy.

Method

In this section, we provide an overview of the proposed method. Firstly, we introduce the preliminary background of the NeRF and the traditional incremental SfM. Then, we explain the details of CF-NeRF that can recover camera parameters from image sequences.

Preliminary Background

NeRF NeRF can generate realistic images from a set of images $I = (I_1, I_2, ..., I_N)$ from $N$ different places without explicitly reconstructing. However, NeRF needs associated camera parameters $\delta$, including camera rotation $\delta_R = (\delta_{R_1}, \delta_{R_2}, ..., \delta_{R_N})$, camera translation $\delta_T = (\delta_{T_1}, \delta_{T_2}, ..., \delta_{T_N})$, and intrinsic camera parameter $\delta_K$. Given a NeRF model $F$ and corresponding weight $\theta$, it can estimate color $c$ and density $\sigma$ through a implicit function $c(x, \bar{d}), \sigma(x) = F_\theta(x, \bar{d})$ with a point $x$ and a view direction.
Figure 3: Estimated Parameters. CF-NeRF estimates the weight $\theta$ of NeRF model and the camera parameter $\delta$, which include the camera rotation $\delta_R$, the camera translation $\delta_T$, and camera intrinsic parameter $\delta_K$. 

\[
\delta_{R_i} + \delta_{R_i}^{-1} \delta_{K_i} \hat{p} + \delta_{T_i} + \delta_{T_i}^{-1} \tilde{p} t \n
\]

To render a pixel $p$, NeRF needs to sample several points $x_p(t) = o + \tilde{d} t$ along a ray shooting from the view position $o$ and generate the color $c$ by the volume rendering function $\mathcal{R}$ as Eq. 1 shows, where $\mathcal{T}(t) = \exp(- \int_{t_n}^{t_f} \sigma(x_p(s)) ds)$ indicates the accumulated transmittance along the ray. $t_n$ and $t_f$ are the near and far bounds of the ray.

\[
c_p = \mathcal{R}(p|\theta) = \int_{t_n}^{t_f} \mathcal{T}(t) \sigma(x_p(t)) c(x_p(t), \tilde{d}) dt \tag{1}
\]

Benefiting from the differentiability of the volume rendering, NeRF can be trained end to end by minimizing the difference between $c_p$ and observed color $I(p)$ as Eq. 2 shows, where $\mathcal{L}$ is the loss function. To be noted, NeRF only estimates $\theta$ and borrows $\delta$ from traditional SFM methods. However, NeRFmm (Wang et al. 2021b) prove that it is possible to estimate $\theta$ and $\delta$ simultaneously under the forward-looking situation.

\[
\arg \min_\theta \left\{ \sum_{I_i \in I} \sum_{p \in I_i} \mathcal{L}(\mathcal{R}(p|\theta), I_i(p)) \right\} \tag{2}
\]

Incremental SFM Given a set of images, the incremental SFM can recover $\delta$ one by one in a linear time (Wu 2013) and contains four steps (Schonberger and Frahm 2016):

**Initialization** The selection of an initial two-view is essential because a suitable initial two-view improves the robustness and quality of the reconstruction. With a given two-view and its matched features, incremental SFM computes the relative pose by multi-view geometry (MVG) and triangulates 3D points to initial the scene.

**Image Registration** After initialization, incremental SFM adds images to the scene in order. Given a new image, incremental SFM builds the 2D-3D relationship by matching its features with images in the scene and recovers the camera parameter by Perspective-n-Point (PnP).

**Triangulation** As a newly added image observes additional information that can extend the scale of the scene, incremental SFM triangulates more 3D points based on the new image and matched features.

**Bundle Adjustment** Adding new images and 3D points without refinement leads to drift. Therefore, it is essential to apply bundle adjustment (BA) by minimizing the re-projection error. In terms of efficiency, incremental SFM proposes partial BA that refines only a subset of images, and global BA that optimizes all images.

**CF-NeRF** Fusing the differentiability of NeRF and the reconstruction strategy of SFM, we propose CF-NeRF, which is capable of estimating the camera parameter under complex movement from sequential images. CF-NeRF consists of three modules: initialization, implicit localization, and implicit optimization, as Figure 2 shows. To convenient later introduction, we define the set of images we have completed estimating the camera parameter as $E$, which starts from $\emptyset$.

**Parameter** CF-NeRF estimates camera parameter $\delta$, which includes $\delta_R$, $\delta_T$, and $\delta_K$, and the weight $\theta$ of NeRF, as Figure 3 shows. During the differential volume rendering, we calculate the ray $\tilde{r}_p(t) = \delta_{R_i} + \delta_{R_i}^{-1} \tilde{p} t$ of pixel $p$ in image $I_i \in I$, where $\tilde{p}$ is the homogeneous expression of $p$. Following NeRFmm (Wang et al. 2021b), we use the axis-angle to represent $\delta_R$ and assume all images have the same camera intrinsic parameter without distortion so that $\delta_K$ only contains the focal length. We initialize $\delta_R$ and $\delta_T$ to zero, and set $\delta_K$ to $53^\circ$ by a common field of view. The activation function determines how to initialize $\theta$. NeRF using ReLU are initialized according to NeRF (Mildenhall et al. 2020), while NeRF using sine are initialized according to SIREN (Sitzmann et al. 2020).

**Initialization** Similar to incremental SFM, CF-NeRF requires initialize $\theta$, $\delta_R$, $\delta_T$, and $\delta_K$ before adding images to $E$. We select the first $N_{init}$ images $I_{init}$ from $I$ to optimise these parameters by Eq. 3 with $\xi_{init}$ iterations. Since the rotation between adjacent images is not large and NeRF is hard to estimate rotation (Lin et al. 2021), we do not estimate the rotation in the initialization to reduce the freedom. After initialization, we add $I_1$ to $E$ and keep $\theta$, $\delta_R$, $\delta_T$, and $\delta_K$ but discard other camera parameters. Note that, unlike the initialization in the previous section, the initialization here is data-specific, similar to the warm-up procedure.

\[
\arg \min_{\theta, \delta_R, \delta_T, \delta_K} \left\{ \sum_{I_i \in I_{init}} \sum_{p \in I_i} \mathcal{L}(\mathcal{R}(p|\theta, \delta_T, \delta_K), I_i(p)) \right\} \tag{3}
\]

**Implicit Localization** After initialization, CF-NeRF estimates the camera parameter of the remaining images one by one and determines $\delta_R$, $\delta_T$, and $\delta_K$ for each new image $I_n$ by localization. Specifically, we first initialize $\delta_R$ and $\delta_T$ by $\delta_{R_{n-1}}$ and $\delta_{T_{n-1}}$, and then optimize them by minimizing Eq. 4 with fixed $\theta$ through $\xi_{loc}$ iterations. The localization is similar to iNeRF (Yen-Chen et al. 2021), but CF-NeRF does not have a pre-trained $F$.

\[
\arg \min_{\delta_R, \delta_T} \left\{ \sum_{p \in I_n} \mathcal{L}(\mathcal{R}(p|\delta_R, \delta_T), I_n(p)) \right\} \tag{4}
\]
Implicit Optimization Although implicit localization can roughly determine $\delta_{R_n}$ and $\delta_{T_n}$, it faces two problems: the observation from $I_n$ is not added to NeRF, and the localization does not take the multi-view consistency into account to reduce drift. Incremental SfM solves these problems using two separate steps: triangulation and BA, while CF-NeRF benefits from the volume rendering and deals with these problems together. However, it is time-consuming to optimize all images in $E$ every time a new image is added. Therefore, CF-NeRF splits optimization into implicit partial optimization and implicit global optimization.

Each time localizing a new image $I_n$, CF-NeRF performs implicit partial optimization. We select $I_n$ and previous $N_{part} - 1$ images to construct the partial image set $I_{part}$, then optimizes them with $\xi_{part}$ iterations, as Eq.5 shows.

$$\arg \min_{\theta, \delta_R, \delta_T} \left\{ \sum_{I_i \in I_{part}} \sum_{p \in I_i} \mathcal{L}(R(p|\theta, \delta_R, \delta_T), I_i(p)) \right\}$$ (5)

When the number of images in $E$ can be evenly divided by $N_{glob}$, CF-NeRF employs implicit global optimization for $\theta$ and all images in $E$ to enhance the overall accuracy and reduce drifts with $\xi_{glob}$ iterations, as Eq.6 shows.

$$\arg \min_{\theta, \delta_R, \delta_T, \delta_K} \left\{ \sum_{I_i \in E} \sum_{p \in I_i} \mathcal{L}(R(p|\theta, \delta_R, \delta_T, \delta_K), I_i(p)) \right\}$$ (6)

Coarse-to-Fine CF-NeRF uses a coarse-to-fine strategy to improve robustness. CF-NeRF first constructs a Gaussian pyramid with depth $d_G$, then recovers all parameters at a low-resolution image through the incremental pipeline. Finally, CF-NeRF directly performs implicit global optimization with a higher resolution in each scale of the Gaussian pyramid with $\xi_{glob}$ iterations.

Loss Function To improve robustness, we employ the Smooth-L1 loss function, as Eq. 7 shows, where $gt$ represents the ground truth, $pr$ is the estimated value, and $\beta$ is the set to 1.0 by default.

$$\mathcal{L}(pr, gt) = \begin{cases} 0.5 \ast \frac{(gt - pr)^2}{\beta} & \text{if} |gt - pr| < \beta \\ |gt - pr| - 0.5 \ast \beta & \text{otherwise} \end{cases}$$ (7)

Experiments

Dataset

We evaluate our method using a real-world dataset NeRF-Buster (Warburg et al. 2023), mainly rotating around an object. We sample around 50 frames for each scene and resize all images to $480 \times 270$ with ground truth (GT) camera parameters from COLMAP (Schonberger and Frahm 2016).

Implementation

CF-NeRF is implemented using PyTorch. Similar to NeRFmm (Wang et al. 2021b), CF-NeRF does not have hierarchical sampling and uses the coarse network, which has eight layers and the dimension of the hidden layers is set to 128. Moreover, we use the sine activation function instead of the ReLU, as SiRENmm (Guo and Sherwood 2021) is more robust than NeRFmm. We utilize the Adam optimizer to optimize all learnable parameters. Specifically, we set the learning rate of $\theta$ to 0.001, which undergoes a decay
of 0.9954 every 200 epochs. Similarly, the learning rate of $\delta$ is set to 0.001 and undergoes a decay of 0.9000 every 200 epochs. Here, we describe how to set the hyper-parameters in CF-NeRF. We set $N_{\text{init}}$ and $N_{\text{part}}$ to 3 to meet the minimum requirements that can filter outliers based on MVG. To balance drift and efficiency, we set $N_{\text{glob}}$ to 5. Considering the input image resolution, we set $d_{GC}$ to 3 to reconstruct all parameters by coarse-to-fine strategy. The most important parameter in CF-NeRF is iteration, which is the epoch number for each image. During initialization, we set $\xi_{\text{init}}$ to 3000 to guarantee that $\theta$ and $\delta$ can be correctly initialized with fewer images. Subsequently, during the incremental training, we maintain a consistent value of $\xi$, setting $\xi = \xi_{\text{init}} = \xi_{\text{part}} = \xi_{\text{glob}} = \xi_{\text{global}}$ to 900, thus reconstructing the scene from images one by one. Throughout all our experiments, we use the NVIDIA RTX3090.

### Evaluation

To demonstrate the performance of the proposed method, we conduct a comprehensive comparison between CF-NeRF and several state-of-the-art models, including NeRFmm (Wang et al. 2021b), SiRENmm (Guo and Sherwood 2021), BARF (Lin et al. 2021), GARF (Chng et al. 2022), and L2G-NeRF (Chen et al. 2023). We use all images for camera parameter estimation without employing a train/test split. To evaluate the quality of the camera parameters, we calculate the average translation error $\Delta T$ and the average rotation error $\Delta R$ by aligning the estimated camera parameters $\delta_{R}$ and $\delta_{T}$ with COLMAP using a similarity transformation Sim(3) (Lin et al. 2021). It is worth noting that $\delta_{T}$ represents a relative translation error rather than an absolute measurement, as COLMAP can not reconstruct an absolute scale of the scene. We further evaluate the estimated camera parameters through a novel view synthesis by PSNR and LPIPS. To ensure a fair comparison and avoid the influence of varying network backbones across different methods, we uniformly use the NerfAcc (Li, Tancik, and Kanazawa 2022) as the network backbone across different methods, we uniformly use the NerfAcc (Li, Tancik, and Kanazawa 2022), where we select one image for testing in every eight images and the remaining is for training.

### Results

We performed qualitative and quantitative evaluations of these methods on 12 scenes of the NeRFBuster (Warburg et al. 2023) dataset. Notably, BARF (Lin et al. 2021), GARF (Chng et al. 2022), and L2G-NeRF (Chen et al. 2023) require manual setting the focal length. In contrast, NeRFmm (Wang et al. 2021b), SiRENmm (Guo and Sherwood 2021), and CF-NeRF have the ability to estimate the focal length.

Table 1 shows the results of qualitative experiments. Our method obtains the highest accuracy camera parameters, while all other methods fail outright. It is important to understand that $\Delta R$ and $\Delta T$ are calculated by aligning the camera positions with Sim(3) and that a slight difference in camera position can lead to huge errors. The rotation error $\Delta R$ of our method CF-NeRF is roughly around $10^\circ$, while the other methods are around $10^\circ$. Moreover, the translation error $\Delta T$...
The Thirty-Eighth AAAI Conference on Artificial Intelligence (AAAI-24)

Table 2: Ablation experiments. We compare the accuracy of camera parameters of CF-NeRF under different hyper-parameter settings, including the iteration \( \xi \), the global optimization frequency \( N_{glob} \), and the coarse-to-fine strategy, where \( C \) means the coarse stage and \( F \) means the fine stage.

<table>
<thead>
<tr>
<th>( \Delta R )</th>
<th>( \Delta T )</th>
<th>( G, \xi, N_{glob} )</th>
<th>aloe</th>
<th>art</th>
<th>car</th>
<th>century</th>
<th>flowers</th>
<th>garbage</th>
<th>picnic</th>
<th>pikachu</th>
<th>pipe</th>
<th>plant</th>
<th>roses</th>
<th>table</th>
</tr>
</thead>
</table>

Ablation Experiments

We conduct several ablation experiments on the iteration \( \xi \), the global optimization frequency \( N_{glob} \), and the coarse-to-fine strategy to validate the influence of hyper-parameters in CF-NeRF, and results are presented in Table 2.

The iteration \( \xi \) The iteration \( \xi \) is the most important hyper-parameter in our method, determining how many times to optimize the camera parameters for each image. We compare two configurations: \( F, \xi = 600, N_{glob} = 10 \) and \( F, \xi = 900, N_{glob} = 10 \). Table 2 reveals that increasing \( \xi \) improves the final results for almost all scenes. This observation aligns with NeRF (Mildenhall et al. 2020) and iNeRF (Yen-Chen et al. 2021), where NeRF requires a large number of iterations to converge, and iNeRF enhances the quality of camera parameters through more iterations.

The global optimization frequency \( N_{glob} \) To mitigate drift while maintaining efficiency, CF-NeRF employs the implicit global optimization when every \( N_{glob} \) image is added \( E \). We conduct two experiments \( F, \xi = 900, N_{glob} = 10 \) and \( F, \xi = 900, N_{glob} = 5 \) to find out the influence of \( N_{glob} \). As highlighted in Table 2, reducing \( N_{glob} \) yields improved final results, which can be attributed to the fact that global optimization ensures global consistency to avoid NeRF trap into a local minimum.

The coarse-to-fine strategy CF-NeRF adopts a coarse-to-fine strategy to avoid directly estimating camera parameters on high-resolution images, where the fine stage refines initial results from the coarse stage. We conduct two experiments \( C, \xi = 900, N_{glob} = 5 \) and \( F, \xi = 900, N_{glob} = 5 \). Results in Table 2 demonstrate that the fine stage outperforms the coarse stage across almost all scenes. The coarse-to-fine strategy facilitates the training process of CF-NeRF, as the pixel gradient is smoother at the coarse stage and has less RGB information to learn.

Limitation

Although CF-NeRF achieves state-of-the-art results in camera parameter estimation, surpassing other NeRF-based methods, there are still some gaps between CF-NeRF and COLMAP (Schonberger and Frahm 2016), and the accuracy can be further improved through the adjustment of the sample space (Wang et al. 2023) or the utilization of a more robust function (Sabour et al. 2023).

Conclusion

This paper presents CF-NeRF, a novel end-to-end method that does not require prior camera parameters to deal with image sequences with complex trajectories. Following the pipeline of incremental SFM, CF-NeRF contains three major sub-modules: initialization, implicit localization, and implicit optimization. Experiments on the NeRFBuster dataset demonstrate that CF-NeRF achieves state-of-the-art results, while NeRFmm, SiRENmm, BARF, GARF, and L2G-NeRF only work for forward-looking scenes and get trapped in the local minimum on the NeRFBuster dataset. More importantly, CF-NeRF highlights the unlimited potential of NeRF and differential volume rendering, showing that NeRF has impressive reconstruction capabilities and can also be used to estimate camera parameters in complex trajectories.
Acknowledgments

The research was supported in part by a RGC RIF grant under the contract R6021-20, RGC CRF grants under the contracts C7004-22G and C1029-22G, and RGC GRF grants under the contracts 16209120, 16200221, and 16207922. This research was also supported by the National Natural Science Foundation of China (No. 62302126), and the Shenzhen Science and Technology Program (No. RCBS20221008093125065, No. JCYJ20220818102414030).

References


