This paper tackles the problem of novel view synthesis (NVS) from 2D images without known camera poses or intrinsics. Among various NVS techniques, Neural Radiance Field (NeRF) has recently gained popularity due to its remarkable synthesis quality. Existing NeRF-based approaches assume that the camera parameters associated with each input image are either directly accessible at training, or can be accurately estimated with conventional techniques based on correspondences such as Structure-from-Motion. In this work, we propose an end-to-end framework, termed NeRF−−, for training NeRF models given only RGB images, without pre-computed camera parameters. Specifically, we show that the camera parameters, including both intrinsics and extrinsics, can be automatically discovered via joint optimisation during the training of the NeRF model. On the standard LLFF benchmark, our model achieves novel view synthesis results on par with the baseline trained with COLMAP pre-computed camera parameters. We also conduct extensive analyses to understand the model behaviour under different camera trajectories, and show that in scenarios where COLMAP fails, our model still produces robust results.

1 INTRODUCTION

The ability to fly through our three-dimensional world has been the dream of human beings for thousands of years — from the 3500-year-old story of Daedalus and Icarus in ancient Greek mythology, to the earliest scientific attempts of Leonardo da Vinci to build flying machines in the late 1400s [Niccoli 2006]. Thanks to the recent advances in virtual reality (VR) technology, it is now possible to capture a digital version of our world and generate arbitrary views, allowing us to traverse the world through a virtual lens.

To generate photo-realistic views of a real-world scene from any viewpoint, it not only requires to understand the 3D scene geometry, but also to model complex viewpoint-dependent appearance resulting from sophisticated light transport phenomena. One way to achieve this is by constructing a so-called 5D plenoptic function that directly models the light passing through each point in space [Adelson and Bergen 1991] (or a 4D light field [Gortler et al. 1996; Levoy and Hanrahan 1996] if we restrict ourselves outside the convex hull of the objects of interest). Unfortunately, it is not feasible in practice to physically measure a densely sampled plenoptic function. As an alternative, Novel View Synthesis (NVS) aims to approximate such a dense light field from only sparse observations, such as a small set of images captured from diverse viewpoints.

In literature, a large amount of research effort has been devoted to developing methods for novel view synthesis. One group aims to...
explicitly reconstruct the surface geometry and the appearance on
the surface from the observed sparse views [Chaurasia et al. 2013;
Debevec et al. 1996; Hedman et al. 2017; Waechter et al. 2014; Wiles
et al. 2020; Zitnick et al. 2004]. For the purpose of reconstructing 3D
group of approaches aim to explicitly reconstruct the surface
geometry and model its appearance for novel view rendering. To
reconstruct the 3D geometry from 2D images, traditional techniques,
such as Sim [Faugeras and Luong 2001; Hartley and Zisserman 2003]
and Simultaneous Localisation and Mapping (SLAM) jointly solve
for the 3D geometry and the associated camera parameters, by es-

tablishing feature correspondences (e.g. MonoSLAM [Davison et
al. 2007], ORB-SLAM [Mur-Artal et al. 2015], Bundler [Snapev et
al. 2006], COLMAP [Schonberger and Frahm 2016]), or photometric
errors, e.g. DTAM [Newcombe et al. 2011] and LSD-SLAM [Engel
et al. 2014]. However, many of these methods assume diffuse surface


ture, and do not recover view-dependent appearance, hence re-
sulting in unrealistic novel view rendering. Multi-view photometric
stereo methods [Zhou et al. 2013], on the other hand, aim to explain
view-dependent appearance with sophisticated hand-crafted ma-
terial BRDF models, but suffer from the trade-off between quality
and complexity. Recent works such as [Riegler and Koltun 2020a,b]
integrates meshes and features from images to handle such view-
dependent appearance synthesis. Ultimately, even though explicit
geometry reconstruction facilitates camera parameter estimation,
modelling photo-realistic appearance for novel views is still a chal-


ting task.

As an alternative, volume-based representations have been pro-
posed to directly model the appearance of the 3D space [Flynn
et al. 2016; Mildenhall et al. 2019, 2020; Penner and Zhang 2017;
Seitz and Dyer 1999; Sitzmann et al. 2019; Zhou et al. 2018]. In
recent years, researchers have proposed various volume-based rep-
resentations of this kind, such as Soft3D [Penner and Zhang 2017],
Multi-Plane Images (MPI) [Choi et al. 2019; Flynn et al. 2019; Milden-
hall et al. 2019; Tucker and Snavely 2020; Zhou et al. 2018], Scene


2 RELATED WORK

There is vast literature on novel views synthesis. It can be roughly
divided into two categories, one with explicit surface modelling,
and the other with dense volume-based representations.

In this paper, we ask the question: do we really need to pre-
compute camera parameters when training a view synthesis model
such as a NeRF? We show that the answer is no. The NeRF model is
in fact able to automatically discover the camera parameters by itself
during training. Specifically, we propose NeRF++, which jointly opti-
mises the 3D scene representation and the camera parameters (both
extrinsics and intrinsics). On the standard LLFF benchmark, we
demonstrate comparable novel view synthesis results to the base-
line NeRF trained with COLMAP pre-computed camera parameters.
Additionally, we also analyse the model behaviour under different
camera trajectories, showing that in scenarios where COLMAP fails,
our model still produces robust results, which suggests that the
joint optimisation can lead to more robust reconstruction, echoing
the Bundle Adjustment (BA) in classical SIM pipelines [Triggs et al.
2000].
they assume a well-trained NeRF model to begin with, whereas our method is able to automatically discover the camera parameters from only RGB images in a fully unsupervised fashion.

Apart from novel views synthesis using multiple images, there are also learning-based approaches [Niklaus et al. 2019; Shih et al. 2020; Tucker and Snavely 2020; Wiles et al. 2020; Wu et al. 2020; Zhou et al. 2016], which allow for single-image novel view synthesis at inference time by learning a prior over a collection of training data. These methods, however, are either restricted to small camera motions, or produce low quality images, due to the limited information in a single input image.

3 PRELIMINARY

Given a set of images \( I = \{I_1, I_2, \ldots, I_N\} \) captured from \( N \) sparse viewpoints of a scene, with their associated camera parameters \( \Pi = \{\pi_1, \pi_2, \ldots, \pi_N\} \), including both intrinsics and extrinsics, the goal of novel view synthesis is to come up with a scene representation that enables the generation of realistic images from novel, unseen viewpoints. In this paper, we follow the approach proposed in Neural Radiance Fields (NeRF) [Mildenhall et al. 2020].

In NeRF, the authors adopt a continuous function for constructing a volumetric representation of the scene from sparse input views. In essence, it models the view-dependent appearance of the 3D space using a continuous function \( F_\Theta : (x, d) \rightarrow (c, \sigma) \), parameterised by a multi-layer perceptron (MLP). The function maps a location \( x = (x, y, z) \) in 3D space together with a viewing direction \( d = (\theta, \phi) \) to a radiance colour \( c = (r, g, b) \) and a density value \( \sigma \).

To render an image from a NeRF model, the colour at each pixel \( p = (u, v) \) on the image plane \((\hat{I}_i)\) is obtained by a rendering function \( R \), aggregating the radiance along a ray shooting from the camera position \( o_i \), passing through the pixel \( p \) into the volume [Gortler et al. 1996; Max 1995]:

\[
\hat{I}_i(p) = R(p, \pi_i | \Theta) = \int_{h_o}^{h_f} T(h)\sigma(r(h)) c(r(h), d) \, dh, \tag{1}
\]

where

\[
T(h) = \exp \left( -\int_{h_o}^{h_f} \sigma(r(s)) \, ds \right) \tag{2}
\]
denotes the accumulated transmittance along the ray, i.e., the probability of the ray travelling from \( h_o \) to \( h_f \) without hitting any other particle, and \( r(h) = o + h(d) \) denotes the camera ray that starts from camera origin \( o \) and passes through \( p \), controlled by the camera parameter \( \pi_i \), with near and far bounds \( h_o \) and \( h_f \). In practice, the integral in Eq. (1) is approximated by accumulating radiance and densities of a set of sampled points along a ray.

With this implicit scene representation \( F_\Theta(x, d) \) and a differentiable renderer \( R \), NeRF can be trained by minimising the photometric error between the observed views and synthesised ones under known camera parameters:

\[
\mathcal{L} = \sum_i \| I_i - \hat{I}_i \|^2 \tag{3}
\]

\[
\Theta^* = \arg\min_{\Theta} \mathcal{L}(\hat{I} | I, \Pi), \tag{4}
\]

where \( \hat{I} \) denotes the set of synthesised images \( \{\hat{I}_1, \ldots, \hat{I}_N\} \).

To summarise, NeRF represents a 3D scene as a radiance field parameterised by an MLP, which is trained with a set of sparsely observed images via a photometric reconstruction loss. Note that, the camera parameters \( \pi_i \) for these images are required for training, which are usually estimated by SfM packages, such as COLMAP [Schonberger and Frahm 2016]. For more details of NeRF, we refer the readers to [Mildenhall et al. 2020].

4 METHOD

In this paper, we show that the pre-processing step on estimating camera parameters \( \pi_i \) of the input images is in fact unnecessary. Unlike the training setup of the original NeRF, here, we only assume a set of RGB images \( I \) as inputs, without known camera parameters. We seek to jointly optimise the camera parameters and scene representation during the training. Mathematically, this can be written as:

\[
\Theta^*, \Pi^* = \arg\min_{\Theta, \Pi} \mathcal{L}(\hat{I}, \Pi | I), \tag{5}
\]

where the camera parameters \( \Pi \) include both the camera intrinsics and the camera extrinsics. Apart from simplifying the original two-stage approach, another motivation for such a joint optimisation approach comes from bundle adjustment in classical SfM.
pipelines [Triggs et al. 2000] and SLAM systems [Davison et al. 2007; Engel et al. 2014; Newcombe et al. 2011], which is key step to obtain globally consistent reconstruction results.

In the following sections, we first introduce the representations for the camera parameters and then describe the process of the joint optimisation.

4.1 Camera Parameters

Camera Intrinsics. Assuming a pinhole camera model, the camera intrinsic parameters can be expressed by a matrix:

\[ K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}, \]

where \( f_x \) and \( f_y \) denote camera focal lengths along the width and the height of the sensor respectively, and \( c_x \) and \( c_y \) denote principle points in the image plane.

We assume that the camera principle points are located at sensor centre, i.e. \( c_x \approx W/2 \) and \( c_y \approx H/2 \), where \( H \) and \( W \) denote the height and the width of the image, and all input images are taken by the same camera. As a result, camera intrinsics estimation reduces to estimating two values, the focal lengths \( f_x \) and \( f_y \), which can be directly optimised as trainable parameters during training.

Camera Extrinsic. The camera extrinsic parameters determine the position and orientation of the camera, expressed as a transformation matrix \( T_{wc} = [R|t] \) in SE(3), where \( R \in SO(3) \) denotes the camera rotation and \( t \in \mathbb{R}^3 \) denotes the translation. Since translation vector \( t \) is defined in Euclidean space, it can be directly optimised as trainable parameters during training.

As for the camera rotation, which is defined on SO(3), we adopt the axis-angle representation: \( \phi := a \omega, \phi \in \mathbb{R}^3 \), where a rotation is represented by a normalised rotation axis \( \omega \) and a rotation angle \( a \). This can be converted to a rotation matrix \( R \) using the Rodrigues’ formula:

\[ R = I + \sin(\alpha) \frac{\mathbf{\phi}}{\alpha} + \frac{1 - \cos(\alpha)}{\alpha^2} (\mathbf{\phi} \cdot \mathbf{\phi}), \]

where the skew operator \( \mathbf{\phi} \) converts a vector \( \mathbf{\phi} \) to a skew matrix:

\[ \mathbf{\phi} = \begin{pmatrix} \phi_0 \\ \phi_1 \\ \phi_2 \end{pmatrix} = \begin{pmatrix} 0 & -\phi_2 & \phi_1 \\ \phi_2 & 0 & -\phi_0 \\ -\phi_1 & \phi_0 & 0 \end{pmatrix}. \]

With this parameterisation, we can optimise the camera extrinsics for each input image \( I_i \) with the trainable parameters \( \phi_i \) and \( t_i \) during training.

To summarise, the set of camera parameters that we directly optimise in our model are the camera intrinsics \( f_x \) and \( f_y \) shared by all input images, and the camera extrinsics parameterised by \( \phi_i \) and \( t_i \) specific to each image \( I_i \).

4.2 Joint Optimisation of NeRF and Camera Parameters

Our goal is to train a NeRF model given only RGB images as input, without known camera parameters. In other words, we need to find out the camera parameters associated with each input image while training the NeRF model.

Recall that NeRF is trained by minimising the photometric reconstruction error on the input views. Specifically, for each training image \( I_i \), we randomly select \( M \) pixel locations \( \{ p_{i,m} \}_{m=1}^M \) which we would like to reconstruct from the NeRF model \( F_\theta \). To render the colour of each pixel \( p_{i,m} = (u, v) \), we shoot a ray \( \hat{r}_{i,m}(h) \) from the camera position through the pixel into the radiance field, with the current estimates of the camera parameters \( \hat{\pi}_i = (\hat{f}_x, \hat{f}_y, \hat{\phi}_i, \hat{t}_i) \):

\[ \hat{r}_{i,m}(h) = \hat{o}_i + h\hat{d}_{i,m}, \]

where

\[ \hat{o}_i = t_i \text{ and } \hat{R}_i \text{ is computed from } \hat{\phi}_i \text{ using Eq. (7)}. \]

We then sample a number of 3D points \( \{ s_j \} \) along the ray and evaluate the radiance colours \( \{ c_j \} \) and the density values \( \{ \sigma_j \} \) at these locations via the NeRF network \( F_\theta \). The rendering function Eq. (1) is then applied to obtain the colour of that pixel \( \hat{I}_{i,m} \) by aggregating the predicted radiance and densities along the ray.

For each reconstructed pixel, we compute photometric loss using Eq. (3) by comparing its predicted colour \( \hat{I}_{i,m} \) against the ground-truth colour \( I_{i,m} \) sampled from the input image. Since the entire pipeline is fully differentiable, we can jointly optimise both the parameters of the NeRF model \( \Theta \) and the camera parameters \( \{ \pi_i \} \) by minimising the reconstruction loss. The pipeline is summarised in Algorithm 1.

For initialisation, the cameras for all input images are located at origin looking towards \(-z\)-axis, i.e. all \( R_i \) are initialised with identity matrices and all \( t_i \) with zero vectors, and the focal lengths \( f_x \) and \( f_y \) are initialised to be the width \( W \) and the height \( H \) respectively, i.e. FOV = 53°.

Refinement. Although the above joint optimisation of both the camera parameters and the NeRF model from scratch produces reasonable results, the model could fall into local minima where the optimised camera parameters are sub-optimal, resulting in slightly blurry synthesised images. Thus, we introduce an optional refinement step to further improve the quality of the synthesised images. Specifically, after the first training process is completed, we drop the trained NeRF model and re-initialise it with random parameters while keeping the pre-trained camera parameters. We then repeat the joint optimisation using the pre-trained camera parameters as initialisation. We find this additional refinement step generally leads to sharper images and improves the synthesis results, as evidenced by the comparison in Table 1.

Additionally, the camera parameters can also be initialised with estimated values from external toolboxes, where they are available, and jointly refined during the training of the NeRF model. We conduct experiments to refine the camera parameters estimated using COLMAP during NeRF training, and find the novel view results slightly improved through the joint refinement, as shown in Table 1.

5 EXPERIMENTS

We conduct experiments on diverse scenes and compare with the original baseline NeRF, where camera parameters of input images are estimated with COLMAP. In the following sections, we describe the experiment setup, followed by various results and analyses. We
Figure 3. Qualitative comparison between our NeRF−− model with unknown cameras and the baseline NeRF on LLFF-NeRF dataset. For each example, on the left, we show the synthesised novel views from the baseline NeRF with COLMAP pre-computed camera parameters and from our model with jointly optimised camera parameters; on the right, we compare our optimised camera trajectories with the ones estimated from COLMAP, aligned using ATE. Our proposed NeRF−− model recovers accurate camera poses and produces high quality novel views comparable to the baseline NeRF.

Algorithm 1: NeRF−− Pipeline

Input: $N$ images $I = \{I_i\}_{i=1}^N$
Output: NeRF model $F_\Theta$, camera parameters $[\phi_i], [t_i], f_x$ and $f_y$

// Initialisation
$\{\phi_i\} = \text{nn.Parameter(shape=(N, 3), require_grad=True)}$
$\{t_i\} = \text{nn.Parameter(shape=(N, 3), require_grad=True)}$
$f_x, f_y = \text{nn.Parameter(shape=(2), require_grad=True)}$

// NeRF structure see our supp.
$F_\Theta = \text{NeRF\_Module(require_grad=True)}$

// Training
for $i$ in range ($N$) do
    for $m$ in range ($M$) do
        $\hat{d}_{i,m} = \text{construct\_ray}(\phi_i, \hat{t}_i, f_x, f_y, p_{i,m})$ // Eq. 10
        for $h$ from $h_a$ to $h_f$ do
            $x_j = \text{sample\_point}(\hat{d}_{i,m}, h)$ // Eq. 5
            $c_h, \sigma_j = F_\Theta(x_h, \hat{d}_{i,m})$ // forward NeRF
        end
        $\hat{I}_{i,m} = \text{render\_ray}(\{c_h\}, \{\sigma_h\})$
    end
    $L = \text{loss}(\hat{I}_{i}, I_i)$ // Eq. 5
end
L.backward()
optimiser.update($[\phi_i], [t_i], f_x, f_y, \Theta$)

also include a discussion on the limitations of the current method at the end of the section.

5.1 Setup

5.1.1 Dataset. We first conduct experiments on the same forward-facing dataset as that in NeRF, namely, LLFF-NeRF [Mildenhall et al. 2019], which has 8 forward-facing scenes captured by mobile phones or consumer cameras, each containing 20-62 images. In all experiments, we follow the official pre-processing procedures and train/test splits, i.e. the resolution of the training images is $756 \times 1008$, and every 8-th image is used as the test image.

To understand the behaviour of NVS under different camera motion scenarios, such as rotation, traversal (horizontal motion) and zoom-in, we additionally collected a number of diverse scenes extracted from the short video segments in RealEstate10K [Zhou et al. 2018] and Tanks&Temples [Knaptisch et al. 2017] dataset, as well as a few more clips captured by ourselves. In particular, we only select the video sequences with the desired motion type from corresponding datasets. The image resolution in these sequences varies between $480 \times 640$ and $1080 \times 1920$ and the frame rate ranges from 24 fps to 60 fps. We sub-sample the frames and reduce the frame rates to 3-6 fps, and each sequence contains 7-40 images.

5.1.2 Metrics. We evaluate the proposed framework from two aspects: First, to measure the quality of novel view rendering, we use the common metrics: Peak Signal-to-Noise Ratio (PSNR), Structural
We will release the code. We use three separate Adam optimisers for NeRF, camera poses and we demonstrate the results for novel view synthesis in terms of 1024 we randomly sample every 10\(^9\) pixels from every input image and 128 points along each ray. We use Kaiming initialisation [He et al. 2015] for the NeRF model, and do not except that, for computation efficiency, we: (a) do not use the hierarchical sampling strategy; (b) reduce the hidden layer dimension from 256 to 128; and (c) sample only 128 points along each ray. We use Kaiming initialisation [He et al. 2015] for the NeRF model, and initialise all cameras to be at origin looking at \(z\) direction, with focal lengths \(f_x\) and \(f_y\) to be the width and the height of the image. We use three separate Adam optimisers for NeRF, camera poses and focal lengths respectively, all with an initial learning rate of 0.001, except that we lower the initial NeRF learning rate to 0.0005 for Fortress scene. The learning rate of the NeRF model is decayed every 10 epochs by multiplying with 0.9954 (exponential decay), and learning rates of the pose and focal length parameters are decayed every 100 epochs with a multiplier of 0.9. For each training epoch, we randomly sample 1024 pixels from every input image and 128 points in NeRF along each ray to synthesise the colour of the pixels. All models are trained for 10000 epochs unless otherwise specified. More technical details are included in the supplementary material. We will release the code.

5.2 Results

In this section, we present the experimental results and in-depth analyses on the proposed framework, i.e. \textit{NeRF}—. In Section 5.2.1, we demonstrate the results for novel view synthesis in terms of perceptual qualities. In Section 5.2.2, we show the evaluation of the optimised camera parameters. Lastly, in Section 5.2.3, to understand the model behaviour under different camera motion patterns, we demonstrate some qualitative results and discussion for sequences under controlled camera motions, e.g. rotational, traversal, and zoom-in. More results and visualisations are provided in the supplementary material.

5.2.1 On Novel View Synthesis Quality. In this section, we compare the perceptual qualities from the novel views rendered by \textit{baseline NeRF} (where camera parameters are estimated from COLMAP), and our proposed model \textit{NeRF}—, which jointly optimises the camera parameters and the 3D scene representation from only RGB images.

| Scene  | colmap  | ours  | colmap+r | ours+r |
|--------|---------|-------|----------|--------|
| Fern   | 22.24   | 21.83 | 23.31    | 22.14  |
| Flower | 25.25   | 25.34 | 25.67    | 25.51  |
| Fortress | 27.68  | 26.55 | 27.53    | 27.44  |
| Horns  | 24.35   | 23.13 | 24.51    | 23.65  |
| Leaves | 18.82   | 18.73 | 19.00    | 18.88  |
| Orchids | 18.92  | 16.50 | 19.23    | 16.86  |
| Room   | 27.83   | 25.73 | 27.89    | 26.21  |
| Trex   | 23.21   | 22.49 | 23.36    | 22.97  |
| Mean   | 23.54   | 22.54 | 23.69    | 22.96  |

Table 1. Quantitative comparison between our model and the \textit{baseline NeRF} on LLFF-NeRF dataset, with optional refinement (+r). The results show that: (1) The NVS quality of our method with unknown cameras is comparable to the \textit{baseline NeRF} (colmap vs. ours: ΔPSNR = 1.0, ΔSSIM and ΔLPIPS < 0.05), (2) Additional joint refinement of the camera parameters with a re-initialised NeRF model leads to slightly more optimal results on both our model and the \textit{baseline NeRF} (colmap+r/ours+r vs. colmap/ours).

![Figure 4](image-url)
Since our optimised camera parameters might lie in different spaces from the ones estimated using COLMAP, for evaluation, we first align the two trajectories globally with a Sim(3) transformation using an ATE toolbox [Zhang and Scaramuzza 2018], followed by a more fine-grained gradient-driven camera pose alignment by minimising the photometric error on the synthesised image, while keeping the NeRF model fixed. Finally, we compute the metrics between the test image and our synthesised image rendered from the best possible viewpoint. Simply put, all the above mentioned processing aims to eliminate the effect from camera mis-alignment and make a fair comparison on quality of the 3D scene representation.

We report the quantitative evaluations in Table 1 and visual results in Figure 3. Overall, our joint optimisation model, which does not require camera parameters as inputs, achieves similar NVS quality compared to the baseline NeRF model. This confirms that jointly optimising the camera parameters and 3D scene representation is indeed possible. Nevertheless, we observe that for both the Orchids and the Room, our NeRF+ model produces slightly worse results compared to the baseline NeRF. We also notice from Table 2 that the difference between optimised camera focal lengths and COLMAP estimation are most noticable for these two scenes (196.50 and 343.6). This suggests that the optimisation might have fallen into local minima with sub-optimal intrinsics. More discussion can be found in Section 5.3.

In Table 1, we also show the results with additional refinement step, which has shown to improve the NVS quality for both the baseline NeRF and our proposed NeRF+ model slightly.

5.2.2 On Camera Parameter Estimation. We evaluate the accuracy of the camera parameter estimation on the LLFF-NeRF dataset. As explained in Section 5.1.2, the ground-truth camera parameters for these sequences are not available, we therefore treat the COLMAP estimation as references, and report the difference between our optimised camera parameters and theirs on the training images.

In Table 2, we show the L1 difference on the estimated focal lengths, and metrics on camera rotation and translation computed with the ATE toolbox [Zhang and Scaramuzza 2018], which accounts for global scale ambiguity. In the first set of columns (Focal + Pose + NeRF), the camera poses obtained from our model are close to those estimated from COLMAP, confirming the effectiveness of the joint optimisation pipeline. This can also be visualised by the aligned camera trajectories on xy-plane in Figure 3. The error on camera intrinsics is however much larger. This is due to the notorious ambiguity between camera intrinsics and the scale of the camera translation [Pollefeys and Van Gool 1997], especially for these forward facing scenes.

We conduct another two sets of experiments: 1) We fix the camera poses to be same as from COLMAP, and only optimise the camera focal lengths jointly with the NeRF model. We then measure the difference between the optimised focal lengths and the COLMAP estimated ones. As indicated by the second set of columns (Focal + NeRF) in Table 2, by fixing the camera extrinsics, the focal length has been recovered. 2) We fix the focal lengths to be same as from COLMAP estimation, and only the camera extrinsics are jointly optimised with the NeRF model. The results are reported in the same table Pose+NeRF, showing similar performance as the joint optimisation.

Visualisation of the optimisation process. For a better understanding of the optimisation process, we provide a visualisation of the camera poses at various training epochs for the scene Flower from the LLFF-NeRF dataset (Figure 5). The pose estimations are initialised to be identity matrices at the beginning, and converged after about 1000 epochs, subject to a similarity transformation between optimised camera parameters and those estimated from COLMAP.

5.2.3 On Different Camera Motion Patterns. To inspect how our system performs under different camera motions, we pick a number of sequences from the additional datasets (RealEstate10K, Tanks & Temples), with the camera motions following the desired patterns, such as rotation, traversal (horizontal motion) and zoom-in. To give an overview of the experimental results, both the baseline NeRF and our joint training approach work well for zoom-in camera motions, whereas in rotational and traversal movements, we find that the COLMAP sometimes produces incorrect camera poses or simply fails to converge. We provide more discussions for each motion pattern in the following sections.

Rotational Motion. Despite being one common camera motion in hand-held video capturings, rotational motion is notoriously difficult to model in SM or SLAM systems, as no 3D points can be triangulated under such a motion [Pirchheim et al. 2013; Svoboda et al. 1998; Szeliski and Shum 1997]. In the literature, numerous approaches have been proposed to deal with this problem, for example, through rotation averaging [Bustos et al. 2019; Hartley et al. 2013].

In Figure 6, we show the NVS results of a sequence from the RealEstate10K dataset, where the camera motion is dominated by a rotation. In this case, COLMAP produces incorrect camera poses with extreme outliers, leading to a failure for training baseline NeRF. After manually correcting two extreme outlier poses by assigning to their closest ones, we then re-train the NeRF model, shown as the last row in Figure 6. Even with such manually corrected poses, the baseline NeRF still produces blurry synthesis results and fails to model the geometry correctly. In comparison, our joint optimisation model recovers much more accurate geometry and consequently higher quality view synthesis results.

Apart from picking video sequence from the public datasets, we also show results on the sequence recorded by ourselves (Figure 7).
Table 2. Quantitative evaluation of our optimised focal lengths and camera poses on LLFF-NeRF dataset. We report the difference between our optimised camera parameters and COLMAP computed ones for the lack of ground-truth on real scenes. The results show that: (1) our optimised camera poses are very close to COLMAP estimations (E1 - Rot. & Trans.); (2) our model converges to a different solution for camera intrinsics as it is highly ambiguous (E1 - Foc.); (3) with the same camera poses, our model is able to recover similar focal lengths (E2 - Foc.); (4) similarly, with the same focal lengths, our model still recovers similar camera poses (E3 - Rot. & Trans.). See more discussions in Section 5.2.2.

Figure 6. Camera motion analysis - rotation-dominant. We compare our results with baseline NeRF on a rotation-dominant sequence. Row (a) shows thumbnails of our training images. Row (b) shows novel view and depth renderings of our method. Row (c) illustrates that the baseline NeRF fails on this scene due to incorrect camera pose estimations from COLMAP. Row (d) shows the results of the baseline NeRF with COLMAP poses where outliers are manually corrected. On the left, we show a zoomed-in region of both (a) and (d). Our method recovers accurate geometry and produces high quality novel views, whereas the baseline NeRF fails with COLMAP poses and still produces poor results even with manually corrected poses.

Figure 7. Camera motion analysis - pure rotation. Top: input images. Bottom: optimised camera poses, rendered novel view and rendered depth map. Our model is able to render high-quality novel views, even though the depth map is incorrect, which is expected since no geometry information is captured in a rotation sequence.

Figure 8. Camera motion analysis - traversal pattern. Top: input images. Bottom: optimised camera poses, rendered novel view and rendered depth map. Our model is able to render high-quality novel views, even though the depth map is incorrect, which is expected since no geometry information is captured in a traversal motion.

Figure 9. Zoom-in Motion. In Figure 10, we show an example captured with a zooming-in camera, both COLMAP and our system recover reasonable camera trajectories and view synthesis results.

5.3 Limitations and Future Work

Although the proposed framework for jointly optimising camera parameters and 3D scene representation demonstrates promising results, we still observe a few limitations.

Firstly, as with other photometry-based reconstruction methods, it often struggles to reconstruct scenes with large texture-less regions or in the presence significant photometric inconsistency across frames, such as motion blur, changes in brightness or colour. For example, the joint optimisation struggles to converge on the Fortress scene from the LLFF-NeRF dataset (although it works well with a lower learning rate on the NeRF model). This is likely to
NeRF−−: Neural Radiance Fields Without Known Camera Parameters

Figure 8. Camera motion analysis - roughly traversal. Top: input images. Middle: our rendered novel view and rendered depth. Bottom: visualisation of our optimised camera trajectory and COLMAP estimated trajectory. Both our model and the baseline NeRF model produce high-quality results, although COLMAP seems to produce unlikely camera trajectory illustrated in the zoomed-in segment.

Figure 9. Camera motion analysis - traversal. Top: input images. Middle: our rendered novel view and rendered depth. Bottom: visualisation of our optimised camera trajectory. While our model produces high-quality results with accurate geometry, COLMAP fails to estimate the camera poses from the input images, hence no results from baseline NeRF.

Figure 10. Camera motion analysis - zoom-in. Left: visualisation of our optimised camera trajectory. Right: input images, our rendered novel view and rendered depth. Our approach produces similar pose estimations compared to COLMAP.

be caused by large areas of repeated textures, which could potentially be mitigated by incorporating feature-level losses or explicitly attending to distinctive feature points during training.

Secondly, jointly optimizing camera parameters and scene reconstruction is notoriously challenging and could potentially fall into local minima. For instance, as discussed in Section 5.2.1, our joint optimisation pipeline produces inferior synthesis results on Orchids and Room compared to baseline NeRF shown in Table 1, largely due to sub-optimal optimisation results for the camera intrinsics (as indicated by the significant difference between our optimised focal lengths and the ones from COLMAP reported in Table 2). Incorporating additional components for explicit geometric matching might be useful in guiding the optimisation process.

Lastly, the proposed framework is limited to roughly forward-facing scenes and relatively short camera trajectories, since the NeRF model still struggles to model real scenes in 360° or large camera displacements [Zhang et al. 2020]. As for future work, exploiting the temporal information in sequences can be an effective regularisation for longer trajectories.

6 CONCLUSIONS

In this work, we present an end-to-end NeRF-based pipeline, called NeRF−−, for novel view synthesis from sparse input views, which does not require any information about the camera parameters for training. Specifically, our model jointly optimises the camera parameters for each input image while simultaneously training the NeRF model. This eliminates the need of pre-computing the camera parameters using potentially erroneous SfM methods (e.g. COLMAP) and still achieves comparable view synthesis results as the COLMAP-based NeRF baseline. We present extensive experimental results and demonstrate the effectiveness of this joint optimisation framework under different camera trajectory patterns, even when the baseline COLMAP fails to estimate the camera parameters. Despite its current limitations discussed above, our proposed joint optimisation pipeline has demonstrated promising results on this highly challenging task, which presents a step forward towards novel view synthesis on more general scenes with an end-to-end approach.

ACKNOWLEDGEMENT

Shangzhe Wu is supported by Facebook Research. The authors would like to thank Tim Yuqing Tang for insightful discussions and proofreading.

REFERENCES

Edward H. Adelson and James R. Bergen. 1991. The Plenoptic Function and the Elements of Early Vision. In Computational Models of Visual Processing. Álvaro Parra Bustos, Tat-Jen Chin, Anders Eriksson, and Ian Reid. 2019. Visual slam: Why bundle adjust?. In ICRA.

Gaurav Chaurraya, Sylvain Duchêne, Olga Sorkine-Hornung, and George Drettakis. 2013. Depth Synthesis and Local Warps for Plausible Image-based Navigation. In SIGGRAPH.

Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. 2019. Extreme view synthesis. In CVPR.

Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. 2007. MonoSLAM: Real-time single camera SLAM. TPAMI (2007).

Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. 1996. Modeling and Rendering Architecture from Photographs: A Hybrid Geometry- and Image-Based Approach. In SIGGRAPH.

Jakob Engel, Thomas Schöps, and Daniel Cremers. 2014. LSD-SLAM: Large-scale direct monocular SLAM. In ECCV.
