Deblur-GS: 3D Gaussian Splatting from Camera Motion Blurred Images

WENBO CHEN, University of Science and Technology of China, China
LIGANG LIU, University of Science and Technology of China, China

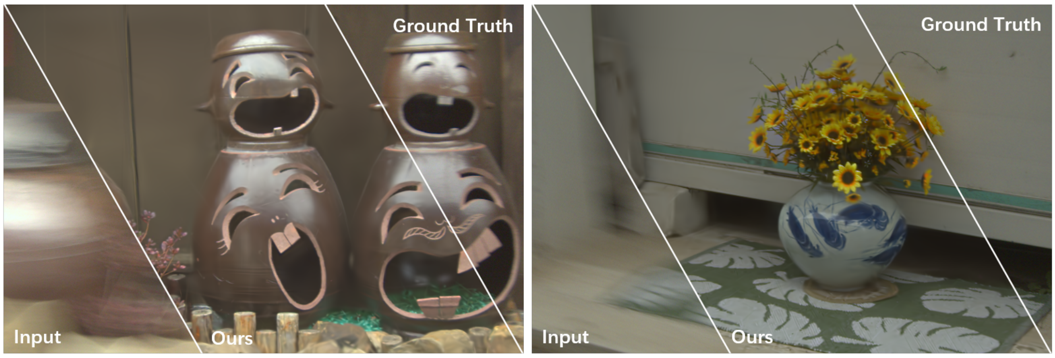

Fig. 1. **Overview of Deblur-GS.** Given a set of multi-view blurry images, our method can reconstruct a sharp 3D Gaussian-based radiance field and render clearly deblurred novel view results in real-time.

Novel view synthesis has undergone a revolution thanks to the radiance field method. The introduction of 3D Gaussian splatting (3DGS) has successfully addressed the issues of prolonged training times and slow rendering speeds associated with the Neural Radiance Field (NeRF), all while preserving the quality of reconstructions. However, 3DGS remains heavily reliant on the quality of input images and their initial camera pose initialization. In cases where input images are blurred, the reconstruction results suffer from blurriness and artifacts. In this paper, we propose the Deblur-GS method for reconstructing 3D Gaussian points to create a sharp radiance field from a camera motion blurred image set. We model the problem of motion blur as a joint optimization challenge involving camera trajectory estimation and time sampling. We cohesively optimize the parameters of the Gaussian points and the camera trajectory during the shutter time. Deblur-GS consistently achieves superior performance and rendering quality when compared to previous methods, as demonstrated in evaluations conducted on both synthetic and real datasets.

CCS Concepts: • **Computing methodologies** → **Computer graphics**.

Additional Key Words and Phrases: Graphics, Rendering, 3D Gaussian Splatting, Motion Blur, Novel View Synthesis

ACM Reference Format:
Wenbo Chen and Ligang Liu. 2024. Deblur-GS: 3D Gaussian Splatting from Camera Motion Blurred Images. *Proc. ACM Comput. Graph. Interact. Tech.* 7, 1, Article 1 (May 2024), 15 pages. https://doi.org/10.1145/3651301

Authors' addresses: Wenbo Chen, University of Science and Technology of China, Hefei Anhui, China, chaf@mail.ustc.edu.cn; Ligang Liu, University of Science and Technology of China, Hefei Anhui, China, lgliu@ustc.edu.cn.

1 INTRODUCTION

Recovering 3D scene information from 2D images has long been a persistent challenge in the fields of computer vision and computer graphics. The emergence of Neural Radiance Fields (NeRF) [Mildenhall et al. 2021] has completely revolutionized the photo-realistic novel view synthesis. To achieve a high-quality NeRF, a crucial prerequisite is to obtain accurate camera poses and a clear input image set. However, in real-world data acquisition, factors like camera jitter and shutter time often result in motion blurring within the dataset. Additionally, commonly used structure-from-motion (SFM) methods, such as colmap [Schönberger and Frahm 2016], frequently introduce errors or even fail to compute camera poses when dealing with blurred images, consequently leading to low-quality reconstructions.

In recent years, several approaches have tackled the problem of NeRF dealing with blurred image sets. Deblur-NeRF[Ma et al. 2022] employs trainable Deformable Sparse Kernels (DSK) to model spatially-varying blur kernels, achieving deblurring through the effect of deconvolution. BAD-NeRF and ExBluRF, on the other hand, achieve the removal of camera motion blur by jointly optimizing camera poses and radiance field parameters. However, NeRF-based methods typically require extensive training time and are not suitable for real-time rendering.

The advent of 3D Gaussian splatting extends the volumetric rendering of NeRF to point cloud rendering, offering a significant boost in rendering speed and reduced training time without sacrificing rendering quality, much like NeRF. However, similar to NeRF, 3DGS critically relies on dataset quality and camera pose accuracy, making it incapable of reconstructing high-quality radiance field results from datasets suffer from motion blur.

In this paper, we propose a motion blur removal and reconstruction method based on 3D Gaussian splatting, aiming to restore clear radiance fields from blurred datasets. When compared to prior methods, our approach demonstrates advantages in reconstruction quality and performance, both on real and synthetic datasets. Our main contributions include:

- We formulate the problem of motion blur removal by modeling it as an estimation of camera motion trajectories within the shutter time, jointly optimizing 3D Gaussian point parameters and camera pose parameters.
- We derived the camera pose derivative for 3D Gaussian splatting and proposed a simplified camera pose estimation method based on its explicit representation.
- We verify through experiments that our method can deblur motion blurry image sets and synthesize sharp results in novel views.

2 BACKGROUND AND RELATED WORK

In this section, we review three main areas of related works: novel view synthesis, image deblurring, and radiance field deblurring.

2.1 Novel View Synthesis

The emergence of Neural Radiance Fields (NeRFs) has completely revolutionized novel view synthesis of scenes from multi-view photos. NeRF learns an implicit neural scene representation that utilizes an MLP to map 3D coordinates (x, y, z) and view dependency (θ, ϕ) to color and density through differentiable volume rendering. Several works are proposed to improve its efficiency [Chen et al. 2022b; Kerbl et al. 2023; Müller et al. 2022; Sun et al. 2022a] and quality [Barron et al. 2023; Suhail et al. 2022; Verbin et al. 2022; Wang et al. 2023a]. The NeRF-based approaches are also applied to numerous 3D vision and graphics application, such as human body [Liu et al. 2021; Peng et al. 2021a,b; Weng et al. 2022], face [Gafni et al. 2021; Sun et al. 2022b; Zhuang et al. 2021], hair [Rosu et al. 2022], large scene reconstruction [Mi and Xu 2023; Tancik et al. 2022], and simultaneous localization and mapping (SLAM) [Li et al. 2022; Rosinol et al. 2023; Zhu et al. 2022]. The neural implicit representation is also suitable for 3D content generation task [Cao et al. 2023; Chan et al. 2021; Gu et al. 2021; Höllein et al. 2023; Lin et al. 2023]. Moreover, some works attempt to reconstruct sharp NeRF from burry input images [Lee et al. 2023b; Ma et al. 2022; Wang et al. 2023b], which regularly occur during image acquisition in real-world scenarios.

More recently, point-based representation [Kopanas et al. 2022; Xu et al. 2022; Yifan et al. 2019; Zhang et al. 2022] has been proposed and widely used for its efficiency in rendering. Zhang et al. [2022] propose a framework that begins with a uniformly-sampled random point cloud and learns per-point position and view-dependent appearance, using a differentiable point rasterizer. Additionally, 3D Gaussian splatting [Kerbl et al. 2023] enables real-time rendering of novel views by its pure explicit representation and the novel differentiable point-based splatting rasterizer. However, most of these approaches still depend on accurately pre-computed camera parameters and are unable to handle blurry input images. Our method utilizes explicit point representation and brings camera motion blur deblurring into the 3D Gaussian splatting framework.

2.2 Image Deblurring

Deblurring is a long-standing problem in the image restoration area due to its ill-posed nature. Existing methods can be generally classified into two main categories. One formulates the deblurring problem as an optimization problem, where the latent sharp images and the blur kernel are jointly optimized using gradient descent [Cho and Lee 2009; Fergus et al. 2006; Krishnan and Fergus 2009; Lee et al. 2018; Levin et al. 2009; Park and Mu Lee 2017; Shan et al. 2008; Xu and Jia 2010]. Another phase of the task is an end-to-end learning problem using deep convolution neural network techniques [Kupyn et al. 2019; Nah et al. 2017; Tao et al. 2018]. Some of the methods can also handle video deblurring [Su et al. 2017]. Our proposed method follows works [Lee et al. 2018; Park and Mu Lee 2017] that jointly estimate multi-view sharp images and the blur kernel formulated by a camera motion trajectory and depth map. They formulate the problem under the classic optimization framework to maximize both the self-view photo-consistency and cross-view photo-consistencies. On the other hand, we use 3D Gaussian points representation and differentiable point splatting rasterizer, which can better preserve the multi-view consistency and provide high-quality novel view synthesis.

2.3 Radiance Field Deblurring

Reconstructing accurate radiance fields using blurry input images has been actively researched by the NeRF community. Deblur-NeRF [Ma et al. 2022] introduces an end-to-end framework that jointly estimates the pixel-wise spatial varying blur kernel and the latent sharp radiance field. However, the blur kernel relies on the training of the deep neural network without geometric and appearance consistency in 3D scene representation. DP-NeRF [Lee et al. 2023a] proposes a novel blurring kernel with two physical priors derived from the physical process of blur acquisition and ray casting to address the issue from Deblur-NeRF. Meanwhile, BAD-NeRF [Wang et al. 2023b] models camera motion blur as camera motion trajectory estimation, requiring a camera pose estimation method [Bian et al. 2023; Chen et al. 2023; Lin et al. 2021; Park et al. 2023; Wang et al. 2021]. ExBluRF [Lee et al. 2023b] extends the BAD-NeRF to more complex camera motion trajectory estimation and voxel-based NeRF, achieving efficiency and better results in extremely blurry images.

However, deblurring remains an open challenge in the context of 3D Gaussian-based radiance field representation. Lee et al. [2024] were the first to address the issue of blurry 3D Gaussian scene reconstruction. They posit that the blurriness of the Gaussian scene is attributed to the covariances of the Gaussian points. To tackle this, they employed an MLP that takes the position, rotation, scale, and viewing direction of 3D Gaussians as inputs and outputs offsets for rotation and scale.

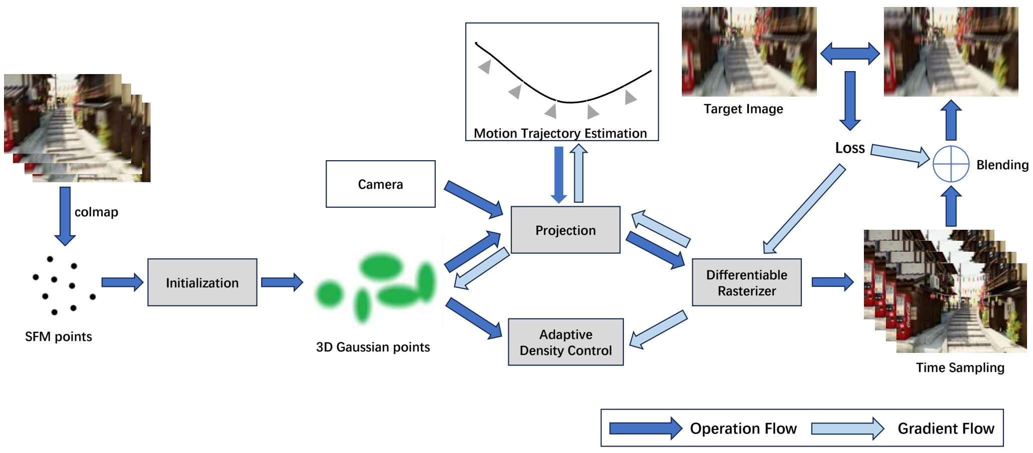

Fig. 2. **The pipeline of Deblur-GS**. We introduce a physics model that generates camera motion blurred images for 3D Gaussian-based radiance field. Through the interpolation of camera poses over the shutter time, we generate multiple images along the camera motion trajectory and blend them to simulate the temporal exposure process. Minimizing the photometric loss between the blending results and the blurred input images allows us to derive the camera motion trajectory along with the representation of the sharp radiance field in 3D Gaussian points.

With predicted offsets for each Gaussian, they selectively enlarged the covariances of the Gaussian points corresponding to the blurred parts in the training images, achieving a deblurring result. It's important to note that enlarging the covariances of the Gaussian points is analogous to applying a Gaussian blur kernel to the corresponding pixel. Therefore, Lee et al. (2024) can effectively address pixel-wise blur effects, such as defocus blur, but they lack the capability to model motion-related blur. In contrast, our method is specifically designed to address camera motion blur effects in 3D Gaussian scenes, addressing the challenge from a physics-based perspective.

3 OVERVIEW

The advent of 3D Gaussian splatting has ushered in high-quality real-time rendering for neural radiation fields. To address challenges arising from inaccurate pose estimation and camera motion-blurred image inputs in 3D Gaussian scene reconstruction, we introduce Deblur-GS——a novel view synthesis method using 3D Gaussian splatting from camera motion-blurred images. The pipeline of our method is shown in Fig.2. Following the framework of 3D Gaussian splatting, we sample the keyframe pose from camera motion trajectory, which can also represent a global rigid transform of the Gaussian points (discussed in Sec.6). By summing up the rendering images along the camera motion trajectory with appropriate weighting, we can simulate the camera motion blur effect. Using a differentiable splatting rasterizer, we can back-propagate the gradient from photometric loss and jointly optimize the Gaussian scene parameters and the motion trajectory representation. We will first review the basic idea of 3D Gaussian scene representation in Sec.4. Then, we will discuss the modeling of the camera motion blur in Sec.5, including the physics-based camera motion blur formulation and camera motion trajectory representation. In Sec.6, we will discuss the camera pose estimation in 3D Gaussian scene representation, which enables us to optimize the camera motion trajectory. We will describe our experiment in detail in Sec.7 and show some evaluation and comparison results in Sec.8.

4 3D GAUSSIAN SCENE REPRESENTATION

We present the 3D scene as a set of 3D Gaussian points coupled with opacity and spherical harmonics:

$$\boldsymbol{G} = \{G_i : (\boldsymbol{m}_i, \boldsymbol{\Sigma}_i, \Lambda_i, \boldsymbol{Y}_i) | i = 1, \cdots, N\} \tag{1}$$

Each 3D Gaussian point G_i is defined by position $\boldsymbol{m}_i \in \mathbb{R}^3$, 3D covariance matrix $\boldsymbol{\Sigma}_i \in \mathbb{R}^{3\times3}$, opacity $\Lambda_i \in \mathbb{R}$ and Spherical Harmonic coefficients $\boldsymbol{Y}_i \in \mathbb{R}^k$ (k represents the degrees of freedom). We can also parameterize the 3D covariance matrix as a scaling matrix $\boldsymbol{S}$ and a rotation matrix $\boldsymbol{R}$, as the covariance matrix $\boldsymbol{\Sigma}$ of a 3D Gaussian is analogous to describing the configuration of an ellipsoid:

$$\boldsymbol{\Sigma} = \boldsymbol{R}\boldsymbol{S}\boldsymbol{S}^T\boldsymbol{R}^T \tag{2}$$

We store the scaling and rotation matrix separately: a 3D vector $\boldsymbol{s} \in \mathbb{R}^3$ for scaling and a 4D quaternion $\boldsymbol{q} \in \mathbb{R}^4$ for rotation. Therefore, we can optimize both factors independently, simplifying the learning process of the 3D Gaussian.

Given camera pose $\boldsymbol{T} = \{\boldsymbol{R}_t, \boldsymbol{t}_t\}$ and camera intrinsics matrix $\boldsymbol{K}$, the covariance and mean of the 3D Gaussian points G are projected on the 2D image plane for rendering with:

$$\begin{aligned} \boldsymbol{\Sigma}' &= \boldsymbol{J}\boldsymbol{T}\boldsymbol{\Sigma}\boldsymbol{T}^T\boldsymbol{J}^T \\ \boldsymbol{m}' &= \frac{\boldsymbol{K}\boldsymbol{T}\boldsymbol{m}}{D} \end{aligned} \tag{3}$$

where $\boldsymbol{J}$ is the Jacobian of the affine approximation of the projective transformation and D denotes the z-axis coordinate of projection $\boldsymbol{m}$. After projection, we can calculate the blend weight by multiplying the Gaussian and the opacity of the point:

$$\alpha = \Lambda e^{-\frac{1}{2}\boldsymbol{x}^T\Sigma'^{-1}\boldsymbol{x}} \tag{4}$$

The color of a pixel can be rendered by performing front-to-back α-blending in-depth order:

$$\hat{\boldsymbol{C}} = \sum_{i\in\mathcal{N}} \boldsymbol{c}_i\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j) \tag{5}$$

where c_i is the Gaussian point color obtained by learnable Spherical Harmonic coefficient $\boldsymbol{Y}$. Similarly, the depth of the Gaussian scene can be rendered by:

$$\hat{\boldsymbol{D}} = \sum_{i\in\mathcal{N}} \boldsymbol{d}_i\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j) \tag{6}$$

where d_i is the depth of the center of the Gaussian point, which is obtained by projecting the Z-value of the position in the view space.

To perform scene reconstruction, given the ground truth camera poses that determine the 3D Gaussian projection, we can fit a set of initial Gaussian points to the target scene by optimizing their parameters, i.e., m and Σ. With a differentiable rasterizer, we can optimize all parameters through a photometric loss $\mathcal{L}$. However, with imperfect camera poses and blurry input images, the above process makes it difficult to obtain sharp and consistent novel view synthesis results.

5 CAMERA MOTION BLURRING

Motion blur is a common manifestation of camera shake that occurs when the camera's shutter speed is too slow to maintain a stable camera pose. The mathematical modeling of this process is integrating sharp image results $\boldsymbol{I}_t$ during shutter time:

$$\boldsymbol{B} = \int_0^{\tau} w_t \boldsymbol{I}_t \mathrm{d}t \tag{7}$$

where τ is the shutter time. w_t is the sampling weight, which represents the receiving efficiency of the camera's photosensitive element at timestamp t. In practice, we can uniformly sample N timestamp within the shutter time as keyframe, the i-th keyframe will be at timestamp $t = \frac{i}{N-1}\tau$. The accumulation of the rendering sharp results will become the blurred result as below:

$$\hat{\boldsymbol{B}} = \sum_{i=0}^{N-1} w_i \boldsymbol{I}_i \tag{8}$$

The sampling weight should satisfy $\sum_{i=0}^{N-1} w_i = 1$. For uniform sampling, w_i can be simply set as $1/N$. In our method, $\boldsymbol{w} = (w_0, w_1, \cdots, w_{N-1})$ is a learnable parameter that is jointly optimized with other parameters.

5.1 Camera Motion Trajectory Approximation

Since the camera can move in an arbitrary but smooth trajectory during shutter time, we parameterize the camera pose in $\boldsymbol{T} \in SE(3)$ and approximate the camera trajectory with linear, cubic spline and Bézier curve interpolation in the Lie algebra of $SE(3)$.

Given the typically short shutter time and the relatively small amplitude of camera motion, linear interpolation is generally adequate for most cases. Given start point $\boldsymbol{T}_0$ and endpoint $\boldsymbol{T}_1$, the camera pose at timestamp $t \in [0, \tau]$ can be written as:

$$\boldsymbol{T}_t = \boldsymbol{T}_0 \cdot \exp\left(\frac{t}{\tau} \log\left(\boldsymbol{T}_0^{-1} \cdot \boldsymbol{T}_1\right)\right) \tag{9}$$

where τ is the camera shutter time.

For cubic spline interpolation, however, four control points $\boldsymbol{T}_0, \boldsymbol{T}_1, \boldsymbol{T}_2$, and $\boldsymbol{T}_3$ are required to represent the camera trajectory. We introduce the parameterization $u = t/\tau \in [0, 1]$. Based on the De Boor-Cox formula, the matrix representation of cumulative basis functions $\tilde{\boldsymbol{B}}(u)$ for the splines can be written as follows [Mueggler et al. 2018]:

$$\tilde{\boldsymbol{B}}(u) = \boldsymbol{C} \begin{bmatrix} 1 \\ u \\ u^2 \\ u^3 \end{bmatrix} \quad \boldsymbol{C} = \frac{1}{6} \begin{bmatrix} 6 & 0 & 0 & 0 \\ 5 & 3 & -3 & 1 \\ 1 & 3 & 3 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

the interpolated camera pose at timestamp t can be represented as:

$$\boldsymbol{T}(u(t)) = \boldsymbol{T}_0 \prod_{j=0}^{2} \exp\left(\tilde{\boldsymbol{B}}_{j+1}(u(t)) \cdot \boldsymbol{\Omega}_j\right) \tag{11}$$

where $\tilde{\boldsymbol{B}}_j$ is the j-th entry (0-based) of vector $\tilde{\boldsymbol{B}}$. The incremental pose from T_{i-1} to T_i is encoded by the twist

$$\boldsymbol{\Omega}_i = \log\left(\boldsymbol{T}_{i-1}^{-1} \cdot \boldsymbol{T}_i\right) \tag{12}$$

For some extremely blurry input images, linear and cubic spline interpolation may not be enough. Therefore, we also implement K order Bézier curve interpolation, which requires $K + 1$ control points $T_i(i = 0, 1, \cdots, K)$. Applying De Casteljau's algorithm in Lie algebra representation, the interpolated camera pose is derived as follows:

$$\boldsymbol{T}_t = \prod_{j=0}^{K} \exp\left(\binom{K}{j}(1-u)^{K-j}u^j \cdot \log(\boldsymbol{T}_i)\right) \tag{13}$$

where $u = t/\tau \in [0, 1]$. It can be derived that $\boldsymbol{T}_t$ is differentiable with respect to $\boldsymbol{T}_i(i = 0, 1, 2, \cdots, K)$, which allows us to optimize the motion blurring process by optimizing $\boldsymbol{T}_i(i = 0, 1, 2, \cdots, K)$.

6 DIFFERENTIABLE POSE ESTIMATION

The photometric loss $\mathcal{L}$ is L_1 loss combined with a D-SSIM loss:

$$\mathcal{L} = \lambda\mathcal{L}_1 + (1-\lambda)\mathcal{L}_{D-SSIM} \tag{14}$$

We use $\lambda = 0.2$ for all experiments.

According to eq.5 and eq.3, we can derive the derivative of $\mathcal{L}$ with respect to the camera poses.

$$\begin{aligned}
\frac{\partial\mathcal{L}}{\partial\boldsymbol{T}} &= \frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{C}}}\left(\sum_{i\in\mathcal{N}}\frac{\partial\hat{\boldsymbol{C}}}{\partial\boldsymbol{c}_i}\frac{\partial\boldsymbol{c}_i}{\partial\boldsymbol{T}} + \sum_{i\in\mathcal{N}}\frac{\partial\hat{\boldsymbol{C}}}{\partial\alpha_i}\frac{\partial\alpha_i}{\partial\boldsymbol{T}}\right) \\
&= \sum_{i\in\mathcal{N}}\frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{C}}}\left(\frac{\partial\hat{\boldsymbol{C}}}{\partial\boldsymbol{c}_i}\frac{\partial\boldsymbol{c}_i}{\partial\boldsymbol{T}} + \frac{\partial\hat{\boldsymbol{C}}}{\partial\alpha_i}\left(\frac{\partial\alpha_i}{\partial\boldsymbol{\Sigma}'_i}\frac{\partial\boldsymbol{\Sigma}'_i}{\partial\boldsymbol{T}} + \frac{\partial\alpha_i}{\partial\boldsymbol{m}'_i}\frac{\partial\boldsymbol{m}'_i}{\partial\boldsymbol{T}}\right)\right) \\
&= \sum_{i\in\mathcal{N}}\frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{C}}}\left(\frac{\partial\hat{\boldsymbol{C}}}{\partial\boldsymbol{c}_i}\frac{\partial\boldsymbol{c}_i}{\partial\boldsymbol{T}} + \frac{\partial\hat{\boldsymbol{C}}}{\partial\alpha_i}\left(\frac{\partial\alpha_i}{\partial\boldsymbol{\Sigma}'_i}\frac{\partial(\boldsymbol{J}\boldsymbol{T}\boldsymbol{\Sigma}_i\boldsymbol{T}^T\boldsymbol{J}^T)}{\partial\boldsymbol{T}} + \frac{\partial\alpha_i}{\partial\boldsymbol{m}'_i}\frac{\partial(\boldsymbol{K}\boldsymbol{T}\boldsymbol{m}_i)}{\partial\boldsymbol{T}D_i}\right)\right)
\end{aligned} \tag{15}$$

We split three components from Eq.15 into color term $\frac{\partial\boldsymbol{c}_i}{\partial T}$, covariance term $\frac{\partial(\boldsymbol{J}\boldsymbol{T}\boldsymbol{\Sigma}_i\boldsymbol{T}^T\boldsymbol{J}^T)}{\partial\boldsymbol{T}}$ and position term $\frac{\partial(\boldsymbol{K}\boldsymbol{T}\boldsymbol{m}_i)}{\partial\boldsymbol{T}D_i}$.

The color term is associated with the shading process using spherical harmonics, which encode low-frequency radiance information into a spherical basis and compute the shading result with the viewpoint direction. Given that we optimize the camera pose with a short step size, we can assume that the color of the Gaussian points remains unchanged throughout the process. Therefore, we can just eliminate the color term to simplify our calculation.

The covariance term is related to the projection of the 3D covariance of the Gaussian points. When the distribution of the Gaussian points approaches a sphere, the covariance term becomes small. However, if the distribution of the Gaussian points forms an elongated ellipsoid, the covariance term could become significantly large, causing instability. In practice, we observed that ignoring the covariance term can enhance the robustness and stability of the optimization process.

Finally, the only term we should take care of for the gradient back-propagation is the position term. The gradient with respect to camera pose then can be written as:

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{T}} = \sum_{i\in\mathcal{N}}\frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{C}}}\frac{\partial\hat{\boldsymbol{C}}}{\partial\alpha_i}\frac{\partial\alpha_i}{\partial\boldsymbol{m}'_i}\frac{\partial(\boldsymbol{K}\boldsymbol{T}\boldsymbol{m}_i)}{\partial\boldsymbol{T}D_i} \tag{16}$$

Due to the explicit representation of 3D Gaussian, we can derive that transforming the camera pose is equivalent to transforming the position of the Gaussian points.

$$\boldsymbol{K}(\boldsymbol{T}\hat{\boldsymbol{T}})\boldsymbol{m}_i = \boldsymbol{K}\boldsymbol{T}(\hat{\boldsymbol{T}}\boldsymbol{m}_i) \tag{17}$$

where $\hat{\boldsymbol{T}}$ is an external transformation applied to the camera pose $\boldsymbol{T}$, equivalent to directly applying it to the position of the Gaussian points. Therefore, instead of optimizing the camera pose directly, we optimize a global transformation for all Gaussian points per camera. This approach allows us to pass the pre-transformed Gaussian points position $\hat{\boldsymbol{m}}_i = \hat{\boldsymbol{T}}\boldsymbol{m}_i$ to the Gaussian differentiable renderer without any modification. The differentiable renderer will manage the gradient of $\mathcal{L}$ with respect to $\hat{\boldsymbol{m}}$, while PyTorch automatic differentiation will handle the gradient of $\hat{\boldsymbol{m}}$ with respect to $\hat{\boldsymbol{T}}$. Combining these, we obtain the desired gradient.

$$\frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{T}}} = \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{m}}} \frac{\partial \hat{\boldsymbol{m}}}{\partial \hat{\boldsymbol{T}}} \tag{18}$$

7 IMPLEMENTATION

7.1 Dataset

To evaluate the performance of our method, we utilized both synthetic and real datasets from ExBluRF [Lee et al. 2023b]. The ExBluRF synthetic dataset, derived from DeblurNeRF [Ma et al. 2022], was synthesized using Blender [Community 2018]. Each scene consists of 29 blurry training images and 5 sharp test images. The original Deblur-NeRF datasets assume linear camera motion in constant velocity within the shutter time, generating blurry images by averaging rendering results obtained through linearly interpolating poses between the original and randomly perturbed poses for each view. Blender exported ground truth camera poses during rendering. ExBluRF introduces more challenging motion-blurred images with random 6-DOF camera motion trajectories. The real datasets from ExBluRF were captured by a dual-camera system, where one camera captured a single blurry image with a long exposure time, and the other captured a sequence of sharp images during the exposure time of the blurry image. Eight scenes were captured, each consisting of 20 to 40 multi-view blurry images and corresponding sequences of sharp images. Camera poses and sparse point clouds for the real datasets were estimated using structure-from-motion (i.e., COLMAP [Schönberger and Frahm 2016]) applied to the corresponding sharp images.

7.2 Implementation Detail

Deblur-GS is implemented in Python with the PyTorch framework. We extend the differentiable Gaussian splatting rasterizer to support depth, pose, and cumulative opacity for both forward and backward propagation. Both the Gaussian scene parameters and camera pose parameters are optimized with the two separate Adam optimizer [Kingma and Ba 2014]. The learning rate of the pose optimizer exponentially decays from 1×10^{-3} to 1×10^{-5}. We choose 7 order Bézier interpolation and sample 31 points on the camera motion trajectory for comparison. We train our model for 90K iterations on a single NVIDIA RTX 3090 GPU. The control points of the camera trajectory overlap during initialization and they will be optimized to different positions during training. We use COLMAP [Schönberger and Frahm 2016] to initialize the Gaussian points and control points.

8 RESULTS

We compare the deblurring and novel view synthesis performance with Deblur-NeRF [Ma et al. 2022], BAD-NeRF [Wang et al. 2023b], ExBluRF [Lee et al. 2023b] and 2D image deblurring methods [Chen et al. 2022a; Zamir et al. 2022] with 3D Gaussian splatting. Deblur-NeRF jointly optimizes the neural radiance field and 2D pixel-wise blur kernel estimation. BAD-NeRF models the camera motion using the 6-DOF linear trajectory. We choose two state-of-the-art 2D image deblurring methods for comparison, which are Restormer [Zamir et al. 2022] and NAFNet [Chen et al. 2022a]. We employ the image deblurring method to deblur the training images independently, and subsequently, the deblurred images are utilized for Gaussian scene reconstruction. We follow the default configuration of the official implementation of Deblur-NeRF [Ma et al. 2022], BAD-NeRF [Wang et al. 2023b], and ExBluRF [Lee et al. 2023b], training them for 90K iterations instead of 200K in the original papers.

Table 1. Quantitative comparison of image deblurring on the real dataset of ExBluRF.

	camellia			jars			jars2			stone_lantern			sunflowers		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
3DGS + Restormer	24.50	0.65	0.50	25.71	0.71	0.46	25.12	0.74	0.41	25.53	0.78	0.40	24.55	0.76	0.47
3DGS + NAFNet	24.07	0.61	0.56	25.38	0.69	0.54	23.70	0.718	0.51	23.11	0.75	0.50	26.93	0.77	0.48
Deblur-NeRF	28.15	0.70	0.41	27.85	0.73	0.44	26.92	0.76	0.44	26.58	0.78	0.42	31.34	0.84	0.36
BAD-NeRF	14.93	0.42	0.66	13.19	0.44	0.65	10.99	0.41	0.65	9.56	0.21	0.68	17.69	0.60	0.60
ExbluRF	25.41	0.63	0.35	25.65	0.67	0.37	26.94	0.78	0.34	26.62	0.78	0.37	28.66	0.80	0.32
Deblur-GS (**Ours**)	**29.16**	**0.79**	**0.28**	**31.35**	**0.84**	**0.27**	**31.27**	**0.85**	**0.28**	**30.61**	**0.86**	**0.29**	**33.22**	**0.89**	**0.27**

Table 2. Quantitative comparison of novel view synthesis on the real dataset of ExBluRF.

	camellia			jars			jars2			stone_lantern			sunflowers		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
3DGS + Restormer	24.93	0.65	0.48	26.96	0.75	0.41	24.96	0.75	0.41	24.55	0.77	0.41	24.51	0.77	0.47
3DGS + NAFNet	24.07	0.61	0.56	25.32	0.68	0.52	25.13	0.76	0.43	23.19	0.77	0.45	25.80	0.73	0.50
Deblur-NeRF	27.33	0.67	0.44	27.12	0.70	0.47	26.64	0.74	0.46	26.67	0.76	0.45	30.10	0.81	0.39
BAD-NeRF	15.12	0.41	0.66	13.29	0.43	0.66	10.82	0.40	0.65	9.50	0.20	0.67	17.78	0.58	0.61
ExbluRF	24.79	0.61	0.38	25.16	0.64	0.41	26.63	0.76	0.37	25.78	0.76	0.41	27.62	0.77	0.35
Deblur-GS (**Ours**)	**28.78**	**0.78**	**0.29**	**31.59**	**0.84**	**0.27**	**30.56**	**0.84**	**0.29**	**30.98**	**0.87**	**0.28**	**33.05**	**0.89**	**0.28**

Table 3. Quantitative comparison of novel view synthesis on the synthesis dataset of ExBluRF.

	cozyroom			factory			pool			tanabata		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
3DGS + Restormer	21.25	0.73	0.25	18.88	0.51	0.47	26.62	0.74	0.33	20.25	0.63	0.36
3DGS + NAFNet	20.82	0.70	0.36	18.68	0.49	0.54	24.81	0.67	0.49	19.34	0.58	0.45
Deblur-NeRF	25.94	0.83	0.22	19.42	0.51	0.52	28.56	0.79	0.30	23.10	0.73	0.34
BAD-NeRF	11.28	0.18	0.66	10.14	0.20	0.68	12.69	0.39	0.73	9.42	0.10	0.70
ExbluRF	28.56	0.89	0.14	26.92	0.81	0.29	27.30	0.76	0.35	26.81	0.85	0.22
Deblur-GS (**Ours**)	**29.97**	**0.89**	**0.086**	**27.11**	**0.83**	**0.27**	**29.45**	**0.82**	**0.24**	**28.48**	**0.90**	**0.16**

Table 4. Standard deviation comparison of Deblur-GS on the real dataset of ExBluRF.

	camellia			jars			jars2			stone_lantern			sunflowers		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIP	PSNR	SSIM	LPIPS
Image Deblurring	1.073	0.0170	0.018	1.000	0.014	0.016	0.970	0.008	0.014	1.356	0.027	0.028	0.703	0.013	0.015
Novel View Synthesis	1.252	0.030	0.038	1.055	0.015	0.021	1.338	0.016	0.017	1.394	0.023	0.017	1.010	0.012	0.018

It is based on the observation that our method achieves full convergence by the 90K iteration mark, thereby allowing for a more efficient alignment with our experiment. Note that we use cubic spline interpolation for BAD-NeRF in our experiment. We evaluate the deblurring and novel view image synthesis performance with the dataset from ExBluRF [Lee et al. 2023b].

The rendered images are evaluated by PSNR, SSIM [Hore and Ziou 2010] and LPIPS [Zhang et al. 2018] as metrics. The three metrics are vulnerable to misalignment. Note that the reconstructed sharp image can be any rendering result along the estimated camera trajectory. To align with the ground truth image, we keep the Gaussian scene parameters fixed and optimize the global transform to determine the suitable camera pose for evaluation. This process is very efficient, typically requiring only 10-15 seconds to converge.

8.1 Quantities Evaluation Results

Table 1 and Table 2 present the quantities evaluation results for image deblurring and novel view synthesis respectively, using the real dataset from ExBluRF. Additionally, Table 3 showcases the

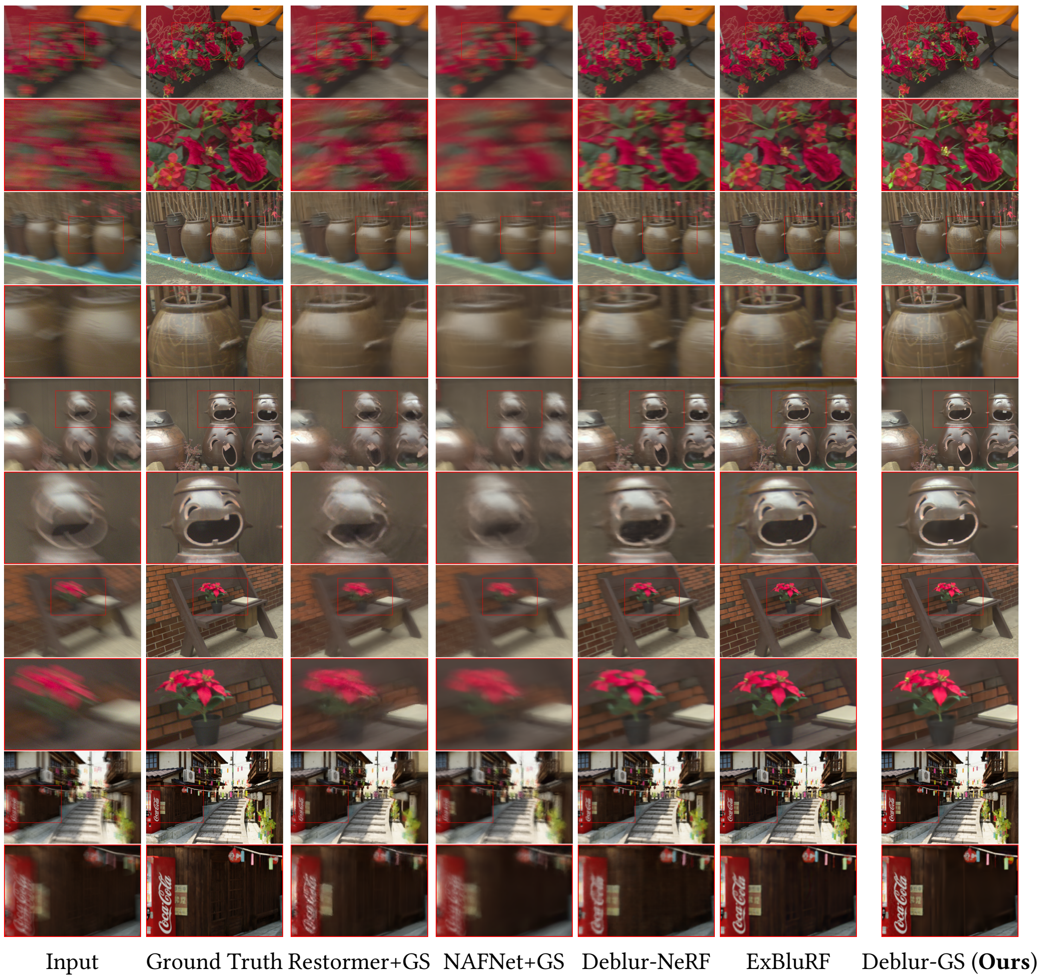

Fig. 3. **Qualitative results of different methods** The experimental results demonstrate that our method achieves superior performance over prior methods in the dataset as well.

results obtained with the synthesis dataset from ExBluRF. They show that our method consistently outperforms previous approaches on image deblurring and novel view synthesis. Table 4 presents the standard deviation observed across various views within the real dataset from ExBluRF, highlighting the method's consistent performance. For the 2D image deblurring preprocess method using Restormer and NAFNet, they struggle to generate deblurred results due to the absence of 3D scene information during the deblurring process. Like our method, BAD-NeRF [Wang et al. 2023b] jointly optimizes the camera motion trajectory with the BARF [Lin et al. 2021] based radiance field, however, it failed to reconstruct the scene due to instability and insufficient training iterations. On the other hand, our method, ExBluRF [Lee et al. 2023b] and Deblur-NeRF [Ma et al. 2022] can consistently reconstruct the 3D scene, while our method produces sharper results than the other two methods in the same training iteration.

8.2 Qualitative Evaluation Results

We also evaluate the qualitative performance of our method against the other methods. Fig.3 demonstrates that our method also outperforms other methods as in the quantitative evaluation

Table 5. **Comparison of memory cost of our method and baselines.**

	camellia	jars	jars2	stone_lantern
Restormer+GS	0.86GB	0.96GB	1.12GB	1.4GB
NAFNet+GS	0.77GB	0.74GB	1.11GB	1.46GB
Deblur-NeRF	13.53GB	13.19GB	13.23GB	13.5GB
BAD-NeRF	12.18GB	11.55GB	11.42GB	12.82GB
ExBluRF	10.04GB	9.76GB	9.77GB	10.1GB
Deblur-GS (**Ours**)	1.54GB	1.39GB	1.35GB	1.49GB

section. For the single image deblurring methods(i.e. Restormer [Zamir et al. 2022], NAFNet [Chen et al. 2022a]) indeed can achieve impressive results on some datasets. However, they failed to restore the sharp images and bring artifacts for the blurry images in Gaussian scene reconstruction. Compared to the single image deblurring methods, Deblur-NeRF [Ma et al. 2022] can produce better results. However, Deblur-NeRF [Ma et al. 2022] models motion blur by image convolution, using MLP to estimate the point spread function, which is lack of camera motion and scene occlusion information. Moreover, in the case of extremely blurry images, Deblur-NeRF [Ma et al. 2022] requires an extended training duration with a larger blur kernel size. Otherwise, the latent 3D scene tends to exhibit a slight blurriness, potentially leading to a lack of continuity in the reconstruction of blurred views during training. Even though ExbluRF [Lee et al. 2023b] can already achieve great results, with the help of 3D Gaussian scene representation, our method can converge faster and produce sharper image results.

8.3 Performance Evaluation Results

We examine the efficiency of our method in terms of memory cost as shown in Tab.5. We observe that our method consumes significantly less memory compared to other methods. Due to the potent scene representation capability of 3D Gaussian, we can efficiently represent our scene using an acceptable number of Gaussian points and minimal parameters, as opposed to employing a neural network with a large number of parameters. Note that the 3D Gaussian splatting method can achieve high framerate real-time rendering during inference, which it's a challenge for neural network-based radiance field methods.

9 ABLATION STUDY

9.1 Number of Key Frames

We choose *tanabata* scene from ExBluRF [Lee et al. 2023b] to evaluate the effect of the number of interpolation sampling keyframes within the shutter time. As depicted in Fig. 4, the error bar shows the image deblurring quantitative of the scene and standard deviation across different input images. The reconstruction quality shows a gradual improvement, accompanied by a consistent decline in the standard deviation as the number of keyframes increases, until reaching a point of stabilization. However, the inclusion of more keyframes requires rendering additional intermediate results during the training process, exerting a significant influence on training performance. To enhance training performance without compromising reconstruction quality, we set the default number of keyframes to 31 for our experiment.

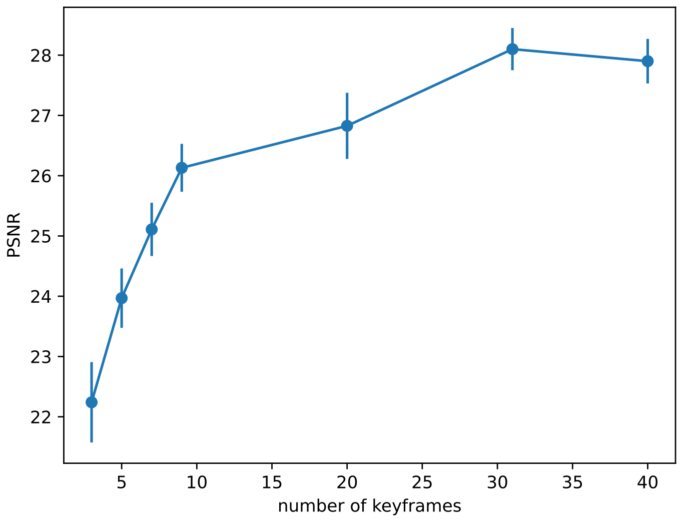

Fig. 4. **The effect of the number of keyframes.** The result demonstrates that the quality saturates as the number of keyframes increases.

Table 6. **Quantitative comparison of image deblurring of different camera motion trajectory representations.**

	pool			wine		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Linear	29.992	0.845	0.163	16.81	0.459	0.426
Cubic Spline	29.993	**0.846**	**0.159**	15.793	0.377	0.389
Bézier	**30.043**	0.837	0.234	**27.628**	**0.874**	**0.183**

9.2 Representation of Camera Motion Trajectory

We evaluate the effect of the representation of camera motion trajectory within the shutter time. We choose *pool* and *wine* scenes from ExBluRF [Lee et al. 2023b] which represent low-level and high-level motion blur to evaluate image deblurring quantitative respectively shown in Tab.6. For the low-level motion blur scene (i.e. *pool*), the camera only has small movements within camera exposure. Therefore, linear interpolation is enough and the quality is close to the higher-order curve. However, for the high-level motion blur scene (i.e. *wine*), the camera may move fast within camera exposure, and the motion trajectory could be complex. At this time, simple curve interpolation might not be enough, while high-order Bézier curves work well. Note that higher higher-order curve requires more control points, which means more keyframe samples and learnable parameters are needed, leading to longer training time. Therefore, we should choose different trajectory modeling strategies based on the blurriness level of the input images to achieve a balance between performance and quality.

10 CONCLUSION

In this paper, we proposed Deblur-GS, a camera motion deblurring method for 3D Gaussian scene representation. Our approach models camera motion blur through a 6-DOF camera motion trajectory, and we validate that the proposed model yields a sharp 3D radiance field when converged to the input images. Using 3D Gaussian representation, our method outperforms existing deblurring approaches both on synthetic and real datasets, demonstrating superior performance with lower memory cost, reduced training iterations, and faster inference speed.

ACKNOWLEDGMENTS

This work is supported by the National Key R&D Program of China (2022YFB3303400) and the National Natural Science Foundation of China (62025207).

REFERENCES

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. 2023. Zip-NeRF: Anti-aliased grid-based neural radiance fields. *arXiv preprint arXiv:2304.06706* (2023).

Wenjing Bian, Zirui Wang, Kejie Li, Jiawang Bian, and Victor Adrian Prisacariu. 2023. NoPe-NeRF: Optimising Neural Radiance Field with No Pose Prior. *CVPR*.

Yukang Cao, Yan-Pei Cao, Kai Han, Ying Shan, and Kwan-Yee K Wong. 2023. Dreamavatar: Text-and-shape guided 3d human avatar generation via diffusion models. *arXiv preprint arXiv:2304.00916* (2023).

Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. 2021. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 5799–5809.

Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. 2022b. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*. Springer, 333–350.

Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. 2022a. Simple baselines for image restoration. In *European Conference on Computer Vision*. Springer, 17–33.

Yue Chen, Xingyu Chen, Xuan Wang, Qi Zhang, Yu Guo, Ying Shan, and Fei Wang. 2023. Local-to-global registration for bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8264–8273.

Sunghyun Cho and Seungyong Lee. 2009. Fast motion deblurring. In *ACM SIGGRAPH Asia 2009 papers*. 1–8.

Rob Fergus, Barun Singh, Aaron Hertzmann, Sam T Roweis, and William T Freeman. 2006. Removing camera shake from a single photograph. In *Acm Siggraph 2006 Papers*. 787–794.

Guy Gafni, Justus Thies, Michael Zollhofer, and Matthias Nießner. 2021. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8649–8658.

Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. 2021. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv preprint arXiv:2110.08985* (2021).

Lukas Höllein, Ang Cao, Andrew Owens, Justin Johnson, and Matthias Nießner. 2023. Text2room: Extracting textured 3d meshes from 2d text-to-image models. *arXiv preprint arXiv:2303.11989* (2023).

Alain Hore and Djemel Ziou. 2010. Image quality metrics: PSNR vs. SSIM. In *2010 20th international conference on pattern recognition*. IEEE, 2366–2369.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics* 42, 4 (2023).

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

Georgios Kopanas, Thomas Leimkühler, Gilles Rainer, Clément Jambon, and George Drettakis. 2022. Neural point catacaustics for novel-view synthesis of reflections. *ACM Transactions on Graphics (TOG)* 41, 6 (2022), 1–15.

Dilip Krishnan and Rob Fergus. 2009. Fast image deconvolution using hyper-Laplacian priors. *Advances in neural information processing systems* 22 (2009).

Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. 2019. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *Proceedings of the IEEE/CVF international conference on computer vision*. 8878–8887.

Byeonghyeon Lee, Howoong Lee, Xiangyu Sun, Usman Ali, and Eunbyung Park. 2024. Deblurring 3D Gaussian Splatting. arXiv:2401.00834 [cs.CV]

Dogyoon Lee, Minhyeok Lee, Chajin Shin, and Sangyoun Lee. 2023a. DP-NeRF: Deblurred Neural Radiance Field With Physical Scene Priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

12386–12396.

Dongwoo Lee, Jeongtaek Oh, Jaesung Rim, Sunghyun Cho, and Kyoung Mu Lee. 2023b. ExBluRF: Efficient Radiance Fields for Extreme Motion Blurred Images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 17639–17648.

Dongwoo Lee, Haesol Park, In Kyu Park, and Kyoung Mu Lee. 2018. Joint blind motion deblurring and depth estimation of light field. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 288–303.

Anat Levin, Yair Weiss, Fredo Durand, and William T Freeman. 2009. Understanding and evaluating blind deconvolution algorithms. In *2009 IEEE conference on computer vision and pattern recognition*. IEEE, 1964–1971.

Kejie Li, Yansong Tang, Victor Adrian Prisacariu, and Philip HS Torr. 2022. Bnv-fusion: Dense 3d reconstruction using bi-level neural volume fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 6166–6175.

Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. 2023. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 300–309.

Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. 2021. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 5741–5751.

Lingjie Liu, Marc Habermann, Viktor Rudnev, Kripasindhu Sarkar, Jiatao Gu, and Christian Theobalt. 2021. Neural Actor: Neural Free-view Synthesis of Human Actors with Pose Control. *ACM SIGGRAPH Asia* (2021).

Li Ma, Xiaoyu Li, Jing Liao, Qi Zhang, Xuan Wang, Jue Wang, and Pedro V Sander. 2022. Deblur-nerf: Neural radiance fields from blurry images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12861–12870.

Zhenxing Mi and Dan Xu. 2023. Switch-NeRF: Learning Scene Decomposition with Mixture of Experts for Large-scale Neural Radiance Fields. In *International Conference on Learning Representations (ICLR)*. https://openreview.net/forum?id=PQ2zoIZqvm

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

Elias Mueggler, Guillermo Gallego, Henri Rebecq, and Davide Scaramuzza. 2018. Continuous-time visual-inertial odometry for event cameras. *IEEE Transactions on Robotics* 34, 6 (2018), 1425–1440.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)* 41, 4 (2022), 1–15.

Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. 2017. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3883–3891.

Haesol Park and Kyoung Mu Lee. 2017. Joint estimation of camera pose, depth, deblurring, and super-resolution from a blurred image sequence. In *Proceedings of the IEEE International Conference on Computer Vision*. 4613–4621.

Keunhong Park, Ben Henzler, Philipp; Mildenhall, Jonathan T. Barron, and Ricardo Martin-Brualla. 2023. CamP: Camera Preconditioning for Neural Radiance Fields. *ACM Trans. Graph.* (2023).

Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. 2021a. Animatable neural radiance fields for modeling dynamic human bodies. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 14314–14323.

Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. 2021b. Neural Body: Implicit Neural Representations with Structured Latent Codes for Novel View Synthesis of Dynamic Humans. In *CVPR*.

Antoni Rosinol, John J Leonard, and Luca Carlone. 2023. Nerf-slam: Real-time dense monocular slam with neural radiance fields. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 3437–3444.

Radu Alexandru Rosu, Shunsuke Saito, Ziyan Wang, Chenglei Wu, Sven Behnke, and Giljoo Nam. 2022. Neural Strands: Learning Hair Geometry and Appearance from Multi-View Images. *ECCV* (2022).

Johannes Lutz Schönberger and Jan-Michael Frahm. 2016. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Qi Shan, Jiaya Jia, and Aseem Agarwala. 2008. High-quality motion deblurring from a single image. *Acm transactions on graphics (tog)* 27, 3 (2008), 1–10.

Shuochen Su, Mauricio Delbracio, Jue Wang, Guillermo Sapiro, Wolfgang Heidrich, and Oliver Wang. 2017. Deep video deblurring for hand-held cameras. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1279–1288.

Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. 2022. Light field neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8269–8279.

Cheng Sun, Min Sun, and Hwann-Tzong Chen. 2022a. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5459–5469.

Jingxiang Sun, Xuan Wang, Yong Zhang, Xiaoyu Li, Qi Zhang, Yebin Liu, and Jue Wang. 2022b. Fenerf: Face editing in neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7672–7682.

Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul Srinivasan, Jonathan T. Barron, and Henrik Kretzschmar. 2022. Block-NeRF: Scalable Large Scene Neural View Synthesis. *arXiv* (2022).

Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. 2018. Scale-recurrent network for deep image deblurring. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 8174–8182.

Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. 2022. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).* IEEE, 5481–5490.

Peng Wang, Yuan Liu, Zhaoxi Chen, Lingjie Liu, Ziwei Liu, Taku Komura, Christian Theobalt, and Wenping Wang. 2023a. F2-NeRF: Fast Neural Radiance Field Training with Free Camera Trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 4150–4159.

Peng Wang, Lingzhe Zhao, Ruijie Ma, and Peidong Liu. 2023b. BAD-NeRF: Bundle Adjusted Deblur Neural Radiance Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 4170–4179.

Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. 2021. NeRF−: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064* (2021).

Chung-Yi Weng, Brian Curless, Pratul P. Srinivasan, Jonathan T. Barron, and Ira Kemelmacher-Shlizerman. 2022. HumanNeRF: Free-Viewpoint Rendering of Moving People From Monocular Video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).* 16210–16220.

Li Xu and Jiaya Jia. 2010. Two-phase kernel estimation for robust motion deblurring. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part I 11.* Springer, 157–170.

Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. 2022. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 5438–5448.

Wang Yifan, Felice Serena, Shihao Wu, Cengiz Öztireli, and Olga Sorkine-Hornung. 2019. Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–14.

Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. 2022. Restormer: Efficient transformer for high-resolution image restoration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 5728–5739.

Qiang Zhang, Seung-Hwan Baek, Szymon Rusinkiewicz, and Felix Heide. 2022. Differentiable point-based radiance fields for efficient view synthesis. In *SIGGRAPH Asia 2022 Conference Papers.* 1–12.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 586–595.

Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. 2022. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 12786–12796.

Yiyu Zhuang, Hao Zhu, Xusen Sun, and Xun Cao. 2021. MoFaNeRF: Morphable Facial Neural Radiance Field. *arXiv preprint arXiv:2112.02308* (2021).