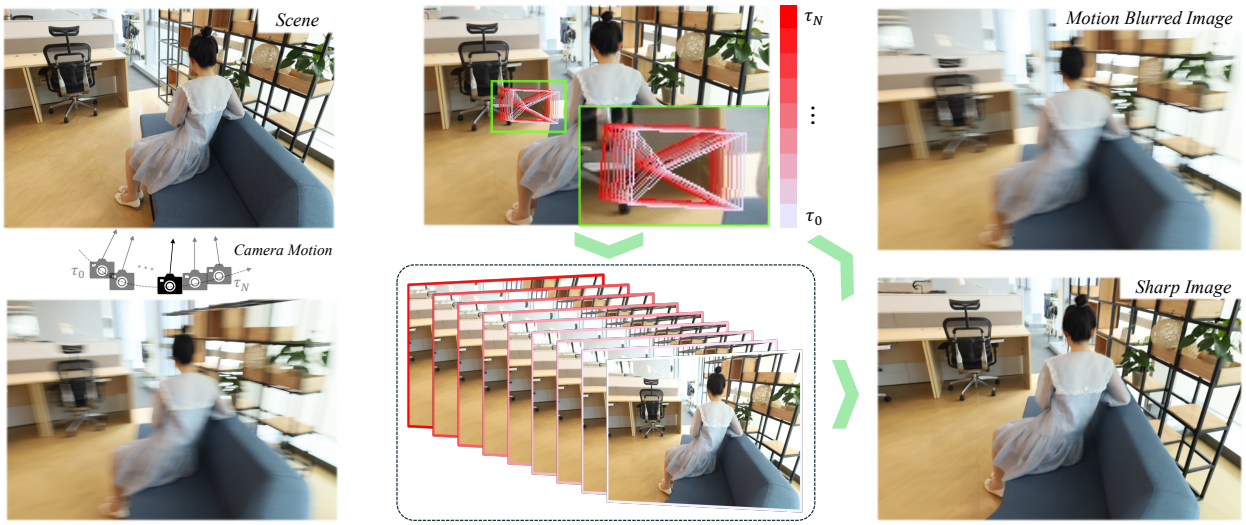
This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

CoMoGaussian: Continuous Motion-Aware Gaussian Splatting from Motion-Blurred Images

Jungho Lee[1] Donghyeong Kim[1] Dogyoon Lee[1] Suhwan Cho[1] Minhyeok Lee[1]
Wonjoon Lee[1] Taeoh Kim[2] Dongyoon Wee[2] Sangyoun Lee[1]
[1]Yonsei University [2]NAVER Cloud



(a) Motion-Blurred Image Acquisition **(b)** Estimated Camera Motion Trajectory **(c)** Rendered Images

Figure 1. We propose CoMoGaussian, a novel framework for reconstructing 3D scenes from camera motion-blurred images. **(a)** Camera motion-blurred images are generated by the continuous movement of the camera during the exposure time. **(b)** The top image represents the camera poses obtained through CoMoGaussian, showing a continuous trajectory indicated by gradual change of colors. The bottom images are the rendering outputs from each camera pose. **(c)** By aggregating the rendered images obtained in step (b), we get the motion-blurred output image shown at the top, while the sharp output image is rendered from the given calibrated camera pose.

Abstract

3D Gaussian Splatting (3DGS) has gained significant attention due to its high-quality novel view rendering, motivating research to address real-world challenges. A critical issue is the camera motion blur caused by movement during exposure, which hinders accurate 3D scene reconstruction. In this study, we propose CoMoGaussian, a ***Co****ntinuous* ***Mo****tion-Aware* ***Gaussian*** *Splatting that reconstructs precise 3D scenes from motion-blurred images while maintaining real-time rendering speed. Considering the complex motion patterns inherent in real-world camera movements, we predict continuous camera trajectories using neural ordinary differential equations (ODEs). To ensure accurate modeling, we employ rigid body transformations, preserving the shape and size of the object but rely on the discrete integration of sampled frames. To better approximate the continuous nature of motion blur, we introduce a continuous motion refinement (CMR) transformation that refines rigid transformations by incorporating additional learnable parameters. By revisiting fundamental camera theory and leveraging advanced neural ODE techniques, we achieve precise modeling of continuous camera trajectories, leading to improved reconstruction accuracy. Extensive experiments demonstrate state-of-the-art performance both quantitatively and qualitatively on benchmark datasets, which include a wide range of motion blur scenarios, from moderate to extreme blur. Project page is available at https://Jho-Yonsei.github.io/CoMoGaussian.*

1. Introduction

Novel view synthesis has recently garnered significant attention, with neural radiance fields (NeRF) [22] making considerable advancements in photo-realistic neural render-

ing. NeRF takes sharp 2D images from multiple views as input to reconstruct precise 3D scenes, which are crucial for applications such as augmented reality (AR) and virtual reality (VR). However, inefficient memory usage of NeRF's volume rendering process poses challenges for real-time applications. To address this, 3D Gaussian Splatting (3DGS) [12] has recently emerged, offering an alternative by explicitly representing 3D scenes and enabling real-time rendering through a differentiable splatting method.

However, for precise 3D scene representation in real-world scenarios, it is essential to address various forms of image quality degradation, such as camera motion blur and defocus blur. Current methods like NeRF and 3DGS rely on sharp images as input, which assumes highly ideal conditions. Obtaining sharp images requires a large depth of field (DoF), necessitating a very small aperture setting [10]. However, a smaller aperture limits light intake, leading to longer exposure times. With longer exposure, even slight camera movement introduces complex motion blur in the images. Therefore, developing techniques that can handle motion blur in input images is crucial for accurate 3D scene reconstruction. Our research tackles this challenge by focusing on reconstructing precise 3D scenes from motion-blurred images, thus broadening the applicability of neural rendering in more realistic conditions.

Recently, several methods have been proposed to achieve sharp novel view rendering from camera motion-blurred images. Inspired by traditional blind image deblurring techniques, Deblur-NeRF [20] firstly introduces a method for deblurring 3D scenes. This approach designs a learnable kernel that intentionally blurs images during training, and during rendering, excludes the learned blurring kernel to render sharp novel view images. Following this approach, various methods targeting camera motion blur [14, 15, 27, 28, 34, 42] have emerged, aiming to improve blurring kernel estimation accuracy. However, all of these methods predict the camera trajectory without enforcing continuity of camera motion throughout the exposure time (*e.g.*, simple spline functions [34, 42]). This oversight is critical because motion blur arises from the continuous integration of scene radiance over the camera's actual movement during the exposure interval. When continuity is not properly accounted for, the predicted motion can exhibit abrupt or piecewise transitions that deviate from physically plausible camera trajectories, resulting in inaccurate blur approximation and suboptimal deblurring performance. While these methods can model simpler motion blur scenarios, they fail to capture complex, smoothly varying camera motion due to their lack of a temporally coherent representation.

In this paper, we propose a **Co**ntinuous **Mo**tion-Aware **Gaussian** Splatting, namly CoMoGaussian, a novel approach with three key contributions. First, we apply neural ordinary differential equations (ODEs) [5] to model continuous camera movement during exposure time, as illustrated in Fig. 1. By continuously modeling the camera trajectory in 3D space, we introduce a model that is fundamentally different from existing methods [15, 20, 27, 34]. Second, we continuously model rigid body transformation over time to accurately capture the shape and size of the static subject throughout the camera movement. By leveraging a continuous representation of rigid motion, our approach better accounts for subtle variations in trajectory, leading to more precise reconstruction of static structures. Third, we introduce a continuous motion refinement (CMR) transformation, which enhances rigid motion modeling by incorporating learnable transformations, enabling more accurate approximation of motion blur trajectories. To integrate these components into a high-quality real-time rendering framework, we adopt Mip-Splatting [41], a differentiable rasterization-based approach built on 3DGS [12]. We evaluate and compare our approach on Deblur-NeRF [20] and ExbluRF [16] datasets, achieving state-of-the-art results across the benchmarks. To demonstrate the effectiveness of CoMoGaussian, we conduct various ablative experiments on the proposed contributions.

2. Related Work

2.1. Neural Rendering

Neural rendering has seen explode in the fields of computer graphics and vision area thanks to the emergence of ray tracing-based neural radiance fields (NeRF) [22], which provide realistic rendering quality from 3D scenes. NeRF has led to a wide range of studies, including TensoRF [4], Plenoxel [9], and Plenoctree [40], which have aimed at overcoming slow rendering speed. Among these advancements, the recent emergence of 3D Gaussian splatting (3DGS) [12], which offers remarkable performance along with fast training and rendering speed, has further accelerated research in the neural rendering. Additionally, there has been active research focusing on the non-ideal conditions of given images, such as sparse-view images [24, 33, 38], and the absence of camera parameters [2, 35]. Moreover, there has been significant research attention on non-ideal conditions inherent to the images themselves, such as low-light [23, 26], blur [15, 17, 20, 27, 28, 34, 42]. Recently, neural rendering from blurry images has attracted attention due to its practical applicability.

2.2. Neural Rendering from Blurry Images

Deblur-NeRF [20] firstly introduced the deblurred neural radiance fields by importing the blind-deblurring mechanism into the NeRF framework. They introduce specific blur kernel in front of the NeRF framework imitating the blind deblurring in 2D image deblurring area. After the emergence of [20], several attempts have been proposed to

model the precise blur kernel with various types of neural rendering baseline, such as TensoRF [4], and 3DGS [12]. DP-NeRF [15] proposes rigid blur kernel that predict the camera motion during image acquisition process as 3D rigid body motion to preserve the geometric consistency across the scene. BAD-NeRF [34] and BAD-Gaussians [42] similarly predict blur kernel as camera motions based on NeRF [22] and 3DGS [12], which assume the simple camera motion and interpolate them between predicted initial and final poses and design simple spline-based methods. Deblurring 3DGS [14] adjusts Gaussian parameters like rotation and scaling to generate blurry images during training, and BAGS [28] proposes CNN-based multi-scale blur-agonostic degradation kernel with blur masking that indicates the blurred areas. In this paper, we implement the continuous motion trajectory that forms camera motion blur in 3D using a neural ODE, which fundamentally differentiates our approach from existing methods.

3. Preliminary

3.1. 3D Scene Blind Deblurring

For conventional image blind deblurring [3, 32, 36], the blurring kernel is estimated without any supervision. The process to acquire blurry images is achieved by convolving the kernel with sharp images, which takes a fixed grid of size around the pixel location p. Deblur-NeRF [20] applies this algorithm to NeRF [22], modeling an adaptive sparse kernel for 3D scene representation from blurry images. Deblur-NeRF acquires the blurry pixel color $\mathbf{c}_{blur}$ by warping the original input ray into multiple rays that constitute the blur, then determining the pixel color from the colors obtained from these rays:

$$\mathbf{c}_{blur} = \sum_{i=0}^{N} w_p^i \mathbf{c}_p^i, \; w.r.t. \; \sum_{i=0}^{N} w_p^i = 1, \tag{1}$$

where N and i respectively denote the number of warped rays for convolution and the correspoding index; w_p is the corresponding weight at each ray's location, and $\mathbf{c}_p$ represents the pixel color of sharp image. In this paper, instead of warping rays, we propose a method to obtain blurry images by applying Eq. (1) to rendered images on estimated camera trajectory.

3.2. 3D Gaussian Splatting

Unlike ray tracing-based methods [1, 4, 22], 3DGS [12] is built on a rasterization-based approach with differentiable 3D Gaussians. These 3D Gaussians are initialized from a sparse point cloud obtained via a Structure-from-Motion (SfM) [30, 31] algorithm and are defined as follows:

$$\mathbf{G}(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\mu)^\top \mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)}, \tag{2}$$

where $\mathbf{x} \in \mathbb{R}^3$ is a point on the Gaussian $\mathbf{G}$ centered at the mean vector $\mu \in \mathbb{R}^3$ with an covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{3\times3}$. The 3D covariance matrix $\mathbf{\Sigma}$ is derived from a learnable scaling vector $\mathbf{s} \in \mathbb{R}^3$ and rotation quaternion $\mathbf{q} \in \mathbb{R}^4$, from which the scaling matrix $\mathbf{S} \in \mathbb{R}^{3\times3}$ and rotation matrix $\mathbf{R} \in \mathbb{R}^{3\times3}$ are obtained and represented as follows: $\mathbf{\Sigma} = \mathbf{RSS}^\top\mathbf{R}^\top$.

For differentiable splatting [39], the Gaussians in the 3D world coordinate system are projected into the 2D camera coordinate system. This projection uses the viewing transformation $\mathbf{W} \in \mathbb{R}^{3\times3}$ and the Jacobian $\mathbf{J} \in \mathbb{R}^{2\times3}$ of the affined approximation of the projective transformation to derive the 2D covariance: $\mathbf{\Sigma}^{2D} = \mathbf{JW\Sigma W}^\top\mathbf{J}^\top$

Each Gaussian includes a set of spherical harmonics (SH) coefficients and an opacity value α to represent view-dependent color $\mathbf{c}$. The pixel color $\mathbf{c}_p$ is then obtained by applying alpha blending to $\mathcal{N}$ ordered Gaussians:

$$\mathbf{c}_p = \sum_{i\in\mathcal{N}} \mathbf{c}_i \alpha_i \prod_{j=1}^{i-1} (1-\alpha_j). \tag{3}$$

In this paper, we continuously model camera poses to project Gaussians $\mathbf{G}$ onto the 2D camera coordinate system over the exposure time and obtain the final blurry image through a rasterization process with the obtained poses.

3.3. Neural Ordinary Differential Equations

Neural ODEs [5] are first proposed as an approach that interprets neural networks as the derivatives of a ODE systems, where ODEs represent the dynamics inherent to the hidden states. Specifically, neural ODEs are utilized to represent parameterized, time-continuous dynamics in the latent space, providing a unique solution given an initial value and numerical differential equation solvers [18].

Neural ODEs model a continuous and differentiable latent state $\mathbf{z}(\tau)$. Within an infinitesimally small step limit ϵ in the latent space, the local continuous dynamics are modeled as $\mathbf{z}(\tau+\epsilon) = \mathbf{z}(\tau) + \epsilon \cdot \frac{d\mathbf{z}(\tau)}{d\tau}$. The derivative of the latent state, $\frac{d\mathbf{z}(\tau)}{d\tau}$, is represented by a neural network $f(\mathbf{z}(\tau), \tau; \phi)$ parameterized by learnable parameters ϕ. The latent state at any arbitrary time τ_s is obtained by solving the ODE from the initial time τ_0:

$$\mathbf{z}(\tau_s) = \mathbf{z}(\tau_0) + \int_{\tau_0}^{\tau_s} f(\mathbf{z}(\tau), \tau; \phi) d\tau. \tag{4}$$

The derivative f modeled by the neural network is expressed as a uniformly Lipschitz continuous non-linear function in $\mathbf{z}$ and τ [11]. Therefore, the solution obtained through the solver for any given integration interval (τ_i, τ_j) is always unique in the integration of the continuous dynamics. The simplest method to solve ODE is the Euler method [8], which is a fixed-step-size first-order solver. Additionally, the Runge-Kutta [13] methods are preferred as a higher-order solvers as they offer enhanced stability.

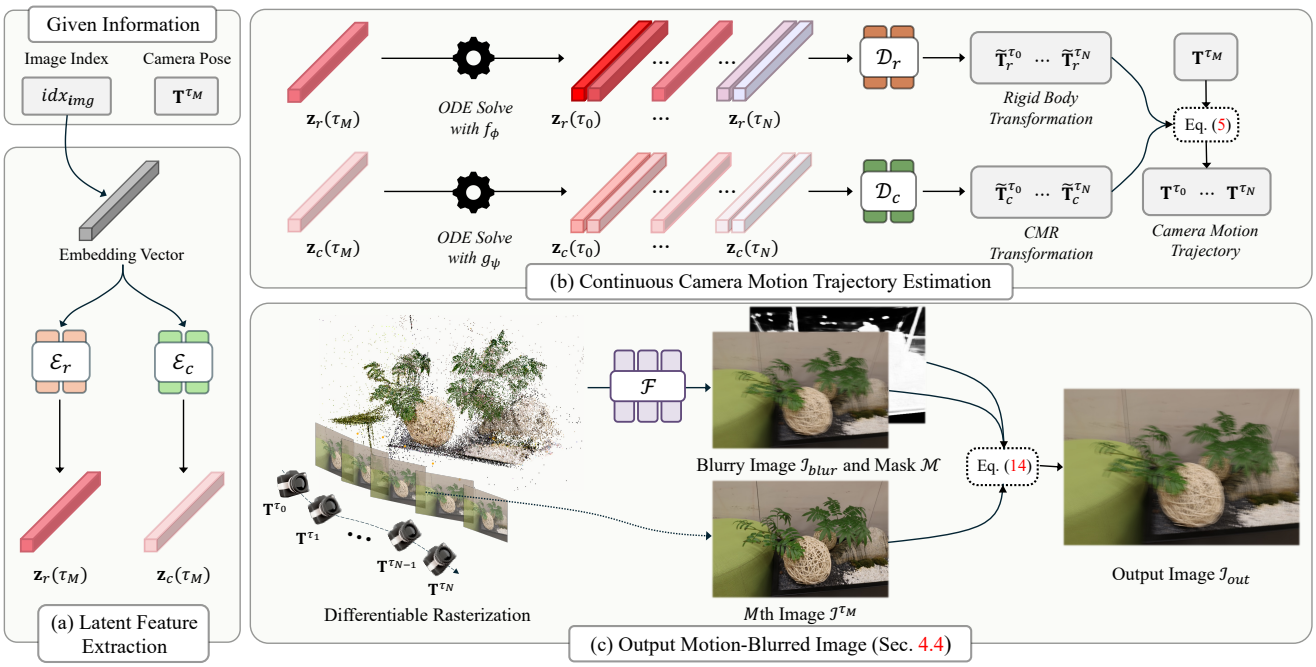


Figure 2. Pipeline of CoMoGaussian. (a) The given image index is embedded and passed through the encoders $\mathcal{E}_r$ and $\mathcal{E}_c$ to obtain latent features for the rigid body transformation and the CMR transformation. (b) CoMoGaussian solves the ODEs using the latent features and the neural derivatives f_ϕ and g_ψ, and obtains N transformed camera poses via Eq. (5). (c) Then, N images are rendered through Mip-Splatting [41], and the output motion-blurred image is obtained through the pixel-wise weighted-sum and scalar pixel mask (Sec. 4.4)

We model the 3D camera trajectory as time-continuous using neural ODEs in the latent space for a continuous representation. In our experiments, we adopt the fourth-order Runge-Kutta solver for all experiments, following [5].

4. Method

4.1. CoMoGaussian Framework

Our goal is to reconstruct a sharp 3D scene using only camera motion-blurred images as input, thereby obtaining deblurred novel view images. Inspired by image blind deblurring methods, we follow the approach of Deblur-NeRF [20], learning a kernel that intentionally blurs images and excluding this kernel during rendering to produce sharp images. Since sharp image supervision is not available during training, we adhere to the fundamental principle of accurately simulating motion-blurred images. In this framework, learning to reconstruct blurry images inherently drives the model to capture the underlying sharp image representation. As illustrated in Fig. 1, our blurring process consists of camera poses along the camera motion trajectory, generated continuously in time order through neural ODEs [5]. Each pose is composed of a rigid body transformation $\tilde{\mathbf{T}}_r = [\tilde{\mathbf{R}}_r|\tilde{\mathbf{t}}_r]$ (Sec. 4.2), which maintains the shape and size of the subject, and a continuous motion refinement (CMR) transformation $\tilde{\mathbf{T}}_c = [\tilde{\mathbf{R}}_c|\tilde{\mathbf{t}}_c]$ (Sec. 4.3), which compensates for the limitations of approximating continuous motion using a discrete sum of rigid transformations by introducing learnable adjustments. Note that $\mathbf{R} \in \mathbb{R}^{3\times3}$ and $\mathbf{t} \in \mathbb{R}^3$ represent rotation matrix and translation vector, and $\tilde{\mathbf{R}} \in \mathbb{R}^{3\times3}$ and $\tilde{\mathbf{t}} \in \mathbb{R}^3$ stand for multiplicative offsets of change of the rotation matrix and translation vector, respectively. Given the two transformation matrices $\tilde{\mathbf{T}}_r = [\tilde{\mathbf{R}}_r|\tilde{\mathbf{t}}_r]$ and $\tilde{\mathbf{T}}_c = [\tilde{\mathbf{R}}_c|\tilde{\mathbf{t}}_c]$, the subsequent pose $\mathbf{T}^{\tau_s} (0 \leq s < N)$ in the camera motion is derived as follows:

$$\mathbf{T}^{\tau_s} = \mathbf{T}^{\tau_M}\tilde{\mathbf{T}}_r^{\tau_s}\tilde{\mathbf{T}}_c^{\tau_s},\ where\ \mathbf{T} = [\mathbf{R}|\mathbf{t}] = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}, \tag{5}$$

where $\mathbf{T}^{\tau_M} (M = N/2)$ denotes given calibrated camera pose. We assume that calibrated camera pose is located at the midpoint of the camera motion trajectory, as it represents an averaged position influenced by the integration of motion blur over the exposure time. By rendering N images from the obtained N camera poses and computing their pixel-wise weighted sum (Sec. 4.4), we obtain the final blurred image. Our framework is shown in Fig. 2.

4.2. Continuous Rigid Body Motion

To design the camera motion trajectory within the exposure time, we apply rigid body transformation in that the shape and size of the object remain unchanged. Rigid body transformation requires three components for the unit screw axis $\mathcal{S} = (\hat{\omega}, v)$: the unit rotation axis $\hat{\omega} \in \mathbb{R}^3$, the rotation angle θ about the axis, and the translation component $v \in \mathbb{R}^3$

for translation. Considering that the direction and extent of blur vary for each image in a single scene, we embed the image index of the scene to obtain different embedded features for each image as shown in Fig. 2. These features are then passed through an encoder $\mathcal{E}_r$ with a single-layer MLP, transforming them into the latent features $\mathbf{z}_r(\tau_0)$, which potentially represent for $\hat{\omega}$, θ, and v of $\mathcal{S}$. Then, we model the continuous latent space for the screw axis using a neural ODEs [5] to assign the latent continuity to the screw axis in camera motion trajectory. The neural derivative f of the latent features for the screw axis is expressed as $\frac{d\mathbf{z}_r(\tau)}{d\tau} = f(\mathbf{z}_r(\tau), \tau; \phi)$, and the latent features at an arbitrary time τ_s can be obtained by ODE solver, numerically integrating f from τ_M to τ_s:

$$\mathbf{z}_r(\tau_s) = \mathbf{z}_r(\tau_M) + \int_{\tau_M}^{\tau_s} f(\mathbf{z}_r(\tau), \tau; \phi) d\tau. \tag{6}$$

To obtain N poses along the camera trajectory, we uniformly sample N time points within the fixed exposure time and get N latent features by applying the Eq. (6) to them. We then transform the latent features obtained by the ODE solver into the unit screw axis $\mathcal{S}$ using a single-layer MLP decoder $\mathcal{D}_r$. In Nerfies [25] and DP-NeRF [15], the angular velocity $\omega = \hat{\omega}\theta$ is modeled and then decomposed into the unit rotation axis $\hat{\omega}$ and the rotation angle θ, making the two elements dependent on each other. However, since θ represents the rotation amount about the axis, these two elements should be independent. Therefore, we model $\hat{\omega}$ with normalization and θ independently through the decoder $\mathcal{D}_r$:

$$\mathcal{D}_r(\mathbf{z}_r(\tau)) = (\hat{\omega}^\tau, \theta^\tau, v^\tau), \ where \ ||\hat{\omega}^\tau|| = 1. \tag{7}$$

According to [19], the screw axis $\mathcal{S}^\tau = (\hat{\omega}^\tau, v^\tau)$ is a normalized twist, so the infinitesimal transformation matrix $[\mathcal{S}^\tau] \in \mathbb{R}^{4\times4}$ is represented as follows:

$$[\mathcal{S}^\tau] = \begin{bmatrix} [\hat{\omega}^\tau] & v^\tau \\ 0 & 0 \end{bmatrix} \in \mathfrak{se}(3), \tag{8}$$

where $[\hat{\omega}^\tau] \in \mathfrak{so}(3)$ is a 3×3 skew-symmetric matrix of vector $\hat{\omega}^\tau$. To derive the infinitesimal transformation matrix $[\mathcal{S}^\tau]\theta^\tau \in \mathfrak{se}(3)$ on the Lie Algebra to the transformation matrix $\mathbf{T}_r^\tau \in$ *SE(3)* in the Lie Group, we use the matrix exponential $e^{[\mathcal{S}^\tau]\theta^\tau}$, whose rotation matrix and translation vector are $\mathbf{R}_r^\tau = e^{[\hat{\omega}^\tau]\theta^\tau}$ and $\mathbf{t}_r^\tau = G(\theta^\tau)v^\tau$, respectively. These matrices are expressed as follows using Rodrigues' formula [29] through Taylor expansion:

$$e^{[\hat{\omega}^\tau]\theta^\tau} = I + \sin\theta^\tau[\hat{\omega}^\tau] + (1-\cos\theta^\tau)[\hat{\omega}^\tau]^2 \in \textit{SO(3)}, \tag{9}$$

$$G(\theta^\tau) = I\theta^\tau + (1-\cos\theta^\tau)[\hat{\omega}^\tau] + (\theta^\tau - \sin\theta^\tau)[\hat{\omega}^\tau]^2. \tag{10}$$

Through the whole process, we obtain N continuous rigid body transformation matrices $\mathbf{T}_r = e^{[\mathcal{S}]\theta}$, and by multiplying these with the input pose as in Eq. (5), we obtain the transformed poses. We show whole derivation process of Eqs. (9) and (10) in the **appendix**, referring [19].

4.3. Continuous Motion Refinement

We predict the continuous camera trajectory using only rigid body transformations; however, according to Eq. (1), the computation of blurry pixels through numerical integration is inherently discrete. In other words, while a blurry pixel should ideally be formed by continuous integration over time, we approximate it using only N discrete samples. Consequently, rigid body transformations alone are insufficient to accurately represent the continuous motion of the camera under discretized numerical integration. To address this limitation, we introduce the CMR Transformation. This transformation has higher degrees of freedom from a learning perspective and is simple to implement. The CMR transformation optimizes the transformation matrix close to *SE(3)* without explicitly enforcing its constraints. Unlike traditional rigid body transformations that strictly adhere to *SE(3)* by parameterizing motion through a screw axis representation $(\hat{\omega}, v)$, the CMR transformation learns a transformation matrix $\tilde{\mathbf{R}}_c$ and a translation vector $\tilde{\mathbf{t}}_c$ in an unconstrained space. This process begins by encoding the image index into the latent state $\mathbf{z}_c(\tau_M)$ of $\tilde{\mathbf{R}}_c$ and $\tilde{\mathbf{t}}_c$ using the encoder $\mathcal{E}_c$ with a single-layer MLP, similar to Sec. 4.2. The latent state $\mathbf{z}_c(\tau_s)$ at any arbitrary time τ_s is obtained using the neural derivative g parameterized by ψ and a solver:

$$\mathbf{z}_c(\tau_s) = \mathbf{z}_c(\tau_M) + \int_{\tau_M}^{\tau_s} g(\mathbf{z}_c(\tau), \tau; \psi)\, d\tau. \tag{11}$$

The latent features are then decoded into rotation matrix component $\mathbf{A}_c^\tau$ and translation vector $\tilde{\mathbf{t}}_c^\tau$ through a single-layer MLP decoder $\mathcal{D}_c$:

$$\mathcal{D}_c(\mathbf{z}_c(\tau)) = (\mathbf{A}_c^\tau, \tilde{\mathbf{t}}_c^\tau) \rightarrow \tilde{\mathbf{T}}_c^\tau = \begin{bmatrix} \tilde{\mathbf{R}}_c^\tau & \tilde{\mathbf{t}}_c^\tau \\ 0 & 1 \end{bmatrix}, \tag{12}$$

where $\tilde{\mathbf{R}}_c^\tau = \mathbf{A}_c^\tau + \mathbf{I}$ and $\mathbf{I}$ is the identity matrix to initialize $\tilde{\mathbf{R}}_c^\tau$ to identity transformation. Since the CMR transformation is intended to refine the rigid body transformation, it should not significantly affect the rigid body transformation during the initial stages of training. Therefore, we initialize the weights of the decoder to approximate $\mathbf{A}_c^\tau$ to zero matrix using a uniform distribution $\mathcal{U}(-10^{-5}, 10^{-5})$, which means that $\tilde{\mathbf{R}}_c^\tau$ is initialized close to identity matrix. To ensure that $\tilde{\mathbf{R}}_c$ remains close to a vaild rotation matrix, we apply explicit regularization, allowing for greater flexibility while maintaining consistency with rigid motion. The proposed loss is $\mathcal{L}_o$, which ensures the orthogonality condition of rotation matrix: $\mathcal{L}_o = \|\tilde{\mathbf{R}}_c^\top \tilde{\mathbf{R}}_c - \mathbf{I}\|_2$. Finally, we apply Eq. (1) using $\tilde{\mathbf{T}}_c^{\tau_s}$ and $\tilde{\mathbf{T}}_r^{\tau_s}$ obtained from Sec. 4.2 to get the refined transformed camera pose $\mathbf{T}^{\tau_s}$.

4.4. Optimization

Pixel-wise Weight and Mask. Once the N images along the camera motion trajectory are rendered from the N camera poses, we apply a pixel-wise weighted sum to create the blurry image $\mathcal{I}_{blur}$, following previous research [15, 17, 20, 27]. To satisfy Eq. (1), we use a shallow CNN $\mathcal{F}$ and a softmax function to compute the pixel-wise weights $\mathcal{P} \in \mathbb{R}^{N\times H\times W\times 3}$ for the resulting images, as follows:

$$\mathcal{I}_{blur} = \sum_{i=1}^{N} \mathcal{I}^{\tau_i} \cdot \mathcal{P}^{\tau_i}, \; where \; \mathcal{P} = \text{softmax}(\mathcal{F}(\mathcal{I})), \quad (13)$$

where $\mathcal{I}^{\tau_i}$ is the i-th image along the camera motion, and $\mathcal{P}^{\tau_i}$ is the pixel-wise weight of i-th image. Additionally, we adopt per-pixel scalar mask [28] $\mathcal{M} \in \mathbb{R}^{H\times W\times 3}$ to generate final output blurry image $\mathcal{I}_{out}$ by blending the sharp image $\mathcal{I}^{\tau_M}$ at camera pose $\mathbf{T}^{\tau_M}$ and the blurry image $\mathcal{I}_{blur}$ acquired by pixel-wise weight:

$$\mathcal{I}_{out} = (\mathbf{1} - \mathcal{M}) \cdot \mathcal{I}^{\tau_M} + \mathcal{M} \cdot \mathcal{I}_{blur}. \quad (14)$$

The scalar mask aims to decide whether each pixel is blurry or not, and a small sparsity constraint is applied to the mask to assign a larger weight on $\mathcal{I}^{\tau_M}$, namely mask sparsity loss $\mathcal{L}_{\mathcal{M}}$, which is a mean value of the scalar mask. This method allows us to obtain the final precise blurry image, which is then optimized against the ground truth blurry image.

Objective. We optimize the learning process using the $\mathcal{L}_1$ loss and D-SSIM between the generated output blurry image $\mathcal{I}_{out}$ and the ground truth blurry image, similar to 3D-GS [12]. The $\mathcal{L}_1$ loss ensures pixel-wise accuracy, while D-SSIM captures perceptual differences. Additionally, we apply the regularization loss $\mathcal{L}_o$ to regularize the CMR transformation and mask sparsity loss $\mathcal{L}_{\mathcal{M}}$. The final objective $\mathcal{L}$ is defined as follows:

$$\mathcal{L} = (1 - \lambda_c)\mathcal{L}_1 + \lambda_c \mathcal{L}_{\text{D-SSIM}} + \lambda_o \mathcal{L}_o + \lambda_{\mathcal{M}} \mathcal{L}_{\mathcal{M}}, \quad (15)$$

where λ_c is a factor for balancing $\mathcal{L}_1$ and $\mathcal{L}_{\text{D-SSIM}}$, and λ_o and $\lambda_{\mathcal{M}}$ are factors for $\mathcal{L}_o$ and $\mathcal{L}_{\mathcal{M}}$, respectively.

5. Experiments

Datasets. CoMoGaussian is evaluated using three datasets: the Deblur-NeRF [20] synthetic dataset, the Deblur-NeRF real-world dataset, and the ExbluRF [16] real-world dataset. The Deblur-NeRF synthetic dataset consists of 5 scenes generated using Blender [7], where the images are combined in linear RGB space to create the final blurred images. The Deblur-NeRF real-world dataset includes 10 scenes captured with a CANON EOS RP camera, where the exposure time is manually set to produce blurry images. The ExbluRF real-world dataset consists of 8 scenes with challenging camera motion. We obtain the camera poses for each image and the initial point clouds by applying COLMAP [30, 31].

Table 1. Comparisons on Deblur-NeRF synthetic and real-world scene dataset. "*" denotes the results obtained by reproducing the released code. The orange and yellow cells respectively indicate the highest and second-highest value.

Methods	Synthetic Scene [20]			Real-World Scene [20]		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Naive NeRF [22]	23.78	0.6807	0.3362	22.69	0.6347	0.3687
Mip-Splatting [41]	23.28	0.6765	0.2822	21.87	0.6270	0.3066
Deblur-NeRF [20]	28.77	0.8593	0.1400	25.63	0.7645	0.1820
DP-NeRF [15]	29.23	0.8674	0.1184	25.91	0.7751	0.1602
BAD-Gaussians* [42]	22.01	0.6377	0.1001	21.69	0.6471	0.1262
Deblurring 3DGS [14]	28.24	0.8580	0.1051	26.61	0.8224	0.1096
BAGS [28]	27.34	0.8353	0.1116	26.70	0.8237	0.0956
CoMoGaussian	31.02	0.9167	0.0492	27.85	0.8431	0.0822

Table 2. Comparisons on ExbluRF real-world scene dataset.

Methods	ExbluRF Real-World Scene [37]		
	PSNR↑	SSIM↑	LPIPS↓
Mip-Splatting [41]	24.52	0.5662	0.6003
ExbluRF [16]	23.45	0.5495	0.3805
BAD-Gaussians [42]	26.83	0.6625	0.3221
Deblurring 3DGS [14]	27.36	0.6795	0.3989
BAGS [28]	24.70	0.5843	0.5278
CoMoGaussian	30.15	0.7559	0.3107

5.1. Novel View Synthesis Results

For quantitative results, we evaluate CoMoGaussian using three metrics: peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM), and learned perceptual image patch similarity (LPIPS). We compare our method with both ray-based [15, 20, 22] and rasterization methods [14, 28, 41, 42]. The overall results for all datasets are shown in Tabs. 1 and 2, demonstrating that our method achieves superior performance compared to all other methods. Additionally, benefiting from using Mip-Splatting [41] as our backbone, our proposed method ensures fast training and rendering speeds. Although BAD-Gaussians [42] achieves relatively good LPIPS scores on ExbluRF dataset, their PSNR and SSIM scores are lower, indicating that their camera poses are not properly optimized.

To qualitatively evaluate the results, we visualize rendering results of several scenes and compare to other methods as shown in Fig. 3. Our method shows superior qualitative results compared to the state-of-the-art rasterization-based methods such as BAD-Gaussians [42], BAGS [28], and Deblurring 3DGS [14]. The bottom two rows of Fig. 3 stand for visualizations of ExbluRF [16] STONE LANTERN and SUNFLOWERS scenes, where our CoMoGaussian best pre-

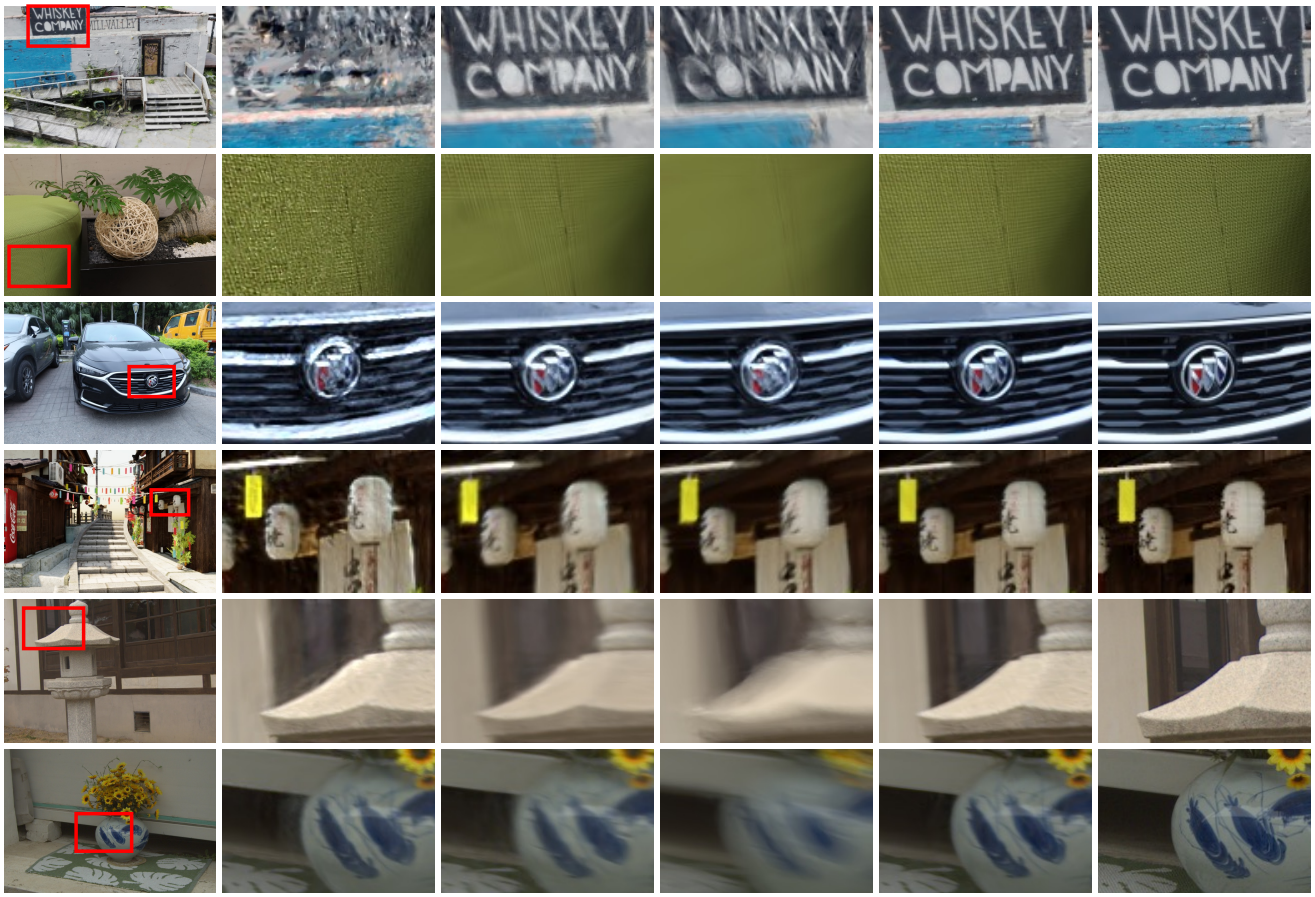


Figure 3. Qualitative comparison on the Deblur-NeRF [20] synthetic and real-world scenes, and ExbluRF [16] real-world scenes.

Table 3. Ablation study on our main components.

Rigid Motion	CMR	Orth. Loss	PSNR↑	SSIM↑	LPIPS↓
			21.87	0.6270	0.3066
✓			27.19	0.8249	0.1005
✓	✓		27.74	0.8402	0.0878
✓	✓	✓	**27.85**	**0.8431**	**0.0822**

serves the edge details of the ground truth, even when given input images with extreme motion blur.

5.2. Ablation Study

To thoroughly demonstrate the effectiveness of rigid body and CMR transformations, we conduct ablation experiments as shown in Tab. 3. Additionally, we perform further ablation studies for our continuous modeling, presented in Tab. 4. All the experiments for Tabs. 3 and 4 are conducted on Deblur-NeRF [20] real-world scenes. Additional ablation experiments are provided in the **appendix**.

Ablation on Transformations. We analyze the impact of the main components of CoMoGaussian: the rigid body transformation and the CMR transformation. The baseline in Tab. 3 is Mip-Splatting [41], and when pixel-wise weights are not applied, the N rendered images are averaged to produce a blurry image. We observe that incorporating the rigid body transformation alone into the baseline results in significantly improved performance across all metrics compared to the baseline. Furthermore, adding the CMR transformation to the rigid body motion, which compensates for the discretized integration, consistently enhances performance across all metrics. Finally, applying the orthogonality loss $\mathcal{L}_o$, a component of the CMR transformation, leads to further improvements in LPIPS.

Ablation on Neural ODE. To validate the effectiveness of implementing rigid body and CMR transformations continuously in latent space, we conduct ablation studies using various approaches, as shown in Tab. 4. For linear interpolation and cubic B-spline, we adopt the implementations from BAD-Gaussians [42]. Since these methods operate in 3D physical space rather than latent space, they exhibit lower performance compared to other latent space-based methods.

For latent space-based methods, we test MLP and GRU [6] as alternatives to neural ODEs. MLP-based

Table 4. Ablation study on the type of camera motion estimator.

Methods	Physical Space	Latent Space	PSNR↑	SSIM↑	LPIPS↓
Linear Interpolation [42]	✓		20.97	0.6085	0.1461
Cubic B-Spline [42]	✓		21.69	0.6471	0.1262
MLP		✓	27.43	0.8295	0.1010
GRU		✓	27.40	0.8302	0.0988
Neural ODE		✓	**27.85**	**0.8435**	**0.0822**

method replaces the neural ODE by extracting N latent features from the latent feature of τ_M using a two-layer MLP. GRU-based method replaces the neural ODE with GRU cells to generate latent features. Our neural ODE-based approach consistently outperforms both methods. Notably, in our neural ODE framework, we extract features for all time points by propagating forward ($\tau_M \rightarrow \tau_N$) and backward ($\tau_M \rightarrow \tau_0$) from the midpoint τ_M of the exposure time. Since all these processes share the same neural derivative, they remain within the same function space. However, in the GRU-based approach, forward and backward time features require separate GRU cells, making it theoretically less suitable than our neural ODE-based method.

Table 5. Experimental results on NeRF-LLFF [21, 22] dataset.

Methods	PSNR↑	SSIM↑	LPIPS↓
Mip-Splatting [41]	27.71	0.8685	0.0494
Deblurring 3DGS [14]	17.42	0.5075	0.3187
BAGS [28]	26.35	0.8490	0.0528
CoMoGaussian	27.56	0.8669	0.0457

Experiments on Sharp Images. To evaluate the generalization capability of CoMoGaussian, we conduct experiments on the NeRF-LLFF [21, 22] dataset, a forward-facing scenes consisting of sharp images, with the results presented in Tab. 5. CoMoGaussian performs comparably to Mip-Splatting [41], demonstrating that our method is not limited to motion-blurred images. Additionally, we evaluate the performance of other methods, Deblurring 3DGS [14] and BAGS [28]. They struggle to optimize certain scenes and fall short of CoMoGaussian in quantitative performance. This highlights the superior generalization ability of our approach compared to existing methods.

5.3. Camera Trajectory Visualization

We visualize the continuous camera trajectories for the HERON (left) and STAIR (right) scenes from the Deblur-NeRF real-world dataset, as shown in Fig. 4. The camera trajectory for a single motion-blurred image is represented by colored cones, with the cone's color gradually transitioning from red to light purple as time progresses from t_0 to t_N. The visualized trajectories confirm that the camera paths generated by CoMoGaussian are smoothly continuous, validating that the continuous transformations described in Secs. 4.2 and 4.3 function as intended. Additionally, the output blurry images in Fig. 4 align with the visualized camera poses, accurately reflecting both the positions and directions of the camera. Specifically, even though the camera path in the HERON scene on the left is nonlinear, CoMoGaussian accurately predicts this path and generates a precise blurry image. Visualizations of camera motion for various types of motion blur, including extreme and moderate motion blur, are provided in the **appendix**.

Figure 4. Camera Trajectory Visualization. Red cones stand for camera poses, and the images below are the output blurry images.

6. Limitation and Future Work

While our CoMoGaussian achieves both high speed and high-quality results, it lacks the ability to distinguish between moderate and severe blur. In other words, moderate blur can be effectively modeled with fewer warped camera poses, whereas severe blur benefits from a greater number of warped poses for more accurate representation. Therefore, adaptively determining the number of warped poses based on the degree of blur could improve training efficiency and mitigate potential overfitting issues.

7. Conclusion

We propose CoMoGaussian, a novel method for reconstructing sharp 3D scenes from blurry images caused by camera motion. We apply a rigid body transformation and further enhance it with CMR transformation to compensate discretized numerical integration. These transformations are continuously modeled using neural ODEs, capturing continuous camera motion trajectories. Drawing inspiration from prior 3D scene deblurring research, we incorporate a pixel-wise weighting strategy with a lightweight CNN. CoMoGaussian surpasses state-of-the-art methods in 3D scene deblurring, with extensive experiments validating the effectiveness of each component.

Acknowledgements. This work was supported by the Yonsei Signature Research Cluster Program of 2024 (2024-22-0161), the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT)(No. RS-2024-00340745), and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT)(No. RS-2024-00423362).

References

[1] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 3

[2] Wenjing Bian, Zirui Wang, Kejie Li, Jia-Wang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4160–4169, 2023. 2

[3] Ayan Chakrabarti. A neural approach to blind motion deblurring. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 221–235. Springer, 2016. 3

[4] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*, pages 333–350. Springer, 2022. 2, 3, 12

[5] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018. 2, 3, 4, 5

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 7

[7] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 6

[8] Leonhard Euler. *Institutionum calculi integralis*. impensis Academiae imperialis scientiarum, 1845. 3

[9] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022. 2

[10] Eugene Hecht. *Optics*. Pearson Education India, 2012. 2

[11] Edward L Ince. *Ordinary differential equations*. Courier Corporation, 1956. 3

[12] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4):1–14, 2023. 2, 3, 6, 13

[13] Wilhelm Kutta. *Beitrag zur näherungsweisen Integration totaler Differentialgleichungen*. Teubner, 1901. 3

[14] Byeonghyeon Lee, Howoong Lee, Xiangyu Sun, Usman Ali, and Eunbyung Park. Deblurring 3d gaussian splatting. *arXiv preprint arXiv:2401.00834*, 2024. 2, 3, 6, 8, 13, 16

[15] Dogyoon Lee, Minhyeok Lee, Chajin Shin, and Sangyoun Lee. Dp-nerf: Deblurred neural radiance field with physical scene priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12386–12396, 2023. 2, 3, 5, 6, 16

[16] Dongwoo Lee, Jeongtaek Oh, Jaesung Rim, Sunghyun Cho, and Kyoung Mu Lee. Exblurf: Efficient radiance fields for extreme motion blurred images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17639–17648, 2023. 2, 6, 7, 16

[17] Jungho Lee, Dogyoon Lee, Minhyeok Lee, Donghyung Kim, and Sangyoun Lee. Smurf: Continuous dynamics for motion-deblurring radiance fields. *arXiv preprint arXiv:2403.07547*, 2024. 2, 6, 12

[18] Ernest Lindelöf. Sur l'application de la méthode des approximations successives aux équations différentielles ordinaires du premier ordre. *Comptes rendus hebdomadaires des séances de l'Académie des sciences*, 116(3):454–457, 1894. 3

[19] Kevin M Lynch and Frank C Park. *Modern robotics*. Cambridge University Press, 2017. 5, 13

[20] Li Ma, Xiaoyu Li, Jing Liao, Qi Zhang, Xuan Wang, Jue Wang, and Pedro V Sander. Deblur-nerf: Neural radiance fields from blurry images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12861–12870, 2022. 2, 3, 4, 6, 7, 16

[21] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (ToG)*, 38(4):1–14, 2019. 8, 12

[22] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I*, pages 405–421, 2020. 1, 2, 3, 6, 8, 12, 16

[23] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P Srinivasan, and Jonathan T Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16190–16199, 2022. 2

[24] Michael Niemeyer, Jonathan T Barron, Ben Mildenhall, Mehdi SM Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5480–5490, 2022. 2

[25] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 5

[26] Naama Pearl, Tali Treibitz, and Simon Korman. Nan: Noise-aware nerfs for burst-denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12672–12681, 2022. 2

[27] Cheng Peng and Rama Chellappa. Pdrf: progressively deblurring radiance field for fast scene reconstruction from blurry images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2029–2037, 2023. 2, 6

[28] Cheng Peng, Yutao Tang, Yifan Zhou, Nengyu Wang, Xijun Liu, Deming Li, and Rama Chellappa. Bags: Blur agnostic gaussian splatting through multi-scale kernel modeling. *arXiv preprint arXiv:2403.04926*, 2024. 2, 3, 6, 8, 13, 16

[29] Olinde Rodrigues. *De l'attraction des sphéroïdes, Correspondence sur l'É-cole Impériale Polytechnique*. PhD thesis, PhD thesis, Thesis for the Faculty of Science of the University of Paris, 1816. 5

[30] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *European conference on computer vision*, pages 501–518. Springer, 2016. 3, 6

[31] Qi Shan, Jiaya Jia, and Aseem Agarwala. High-quality motion deblurring from a single image. *Acm transactions on graphics (tog)*, 27(3):1–10, 2008. 3, 6

[32] Pratul P Srinivasan, Ren Ng, and Ravi Ramamoorthi. Light field blind motion deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3958–3966, 2017. 3

[33] Guangcong Wang, Zhaoxi Chen, Chen Change Loy, and Ziwei Liu. Sparsenerf: Distilling depth ranking for few-shot novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9065–9076, 2023. 2

[34] Peng Wang, Lingzhe Zhao, Ruijie Ma, and Peidong Liu. Bad-nerf: Bundle adjusted deblur neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4170–4179, 2023. 2, 3

[35] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. Nerf–: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021. 2

[36] Oliver Whyte, Josef Sivic, Andrew Zisserman, and Jean Ponce. Non-uniform deblurring for shaken images. *International journal of computer vision*, 98:168–186, 2012. 3

[37] Zijin Wu, Xingyi Li, Juewen Peng, Hao Lu, Zhiguo Cao, and Weicai Zhong. Dof-nerf: Depth-of-field meets neural radiance fields. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 1718–1729, 2022. 6

[38] Jiawei Yang, Marco Pavone, and Yue Wang. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8254–8263, 2023. 2

[39] Wang Yifan, Felice Serena, Shihao Wu, Cengiz Öztireli, and Olga Sorkine-Hornung. Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (TOG)*, 38(6):1–14, 2019. 3

[40] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5752–5761, 2021. 2

[41] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19447–19456, 2024. 2, 4, 6, 7, 8, 11, 12, 13, 16

[42] Lingzhe Zhao, Peng Wang, and Peidong Liu. Bad-gaussians: Bundle adjusted deblur gaussian splatting. *arXiv preprint arXiv:2403.11831*, 2024. 2, 3, 6, 7, 8, 13, 16