SMURF: Continuous Dynamics for Motion-Deblurring Radiance Fields

Jungho Lee Dogyoon Lee Minhyeok Lee Donghyeong Kim Sangyoun Lee

School of Electrical and Electronic Engineering, Yonsei University

https://Jho-Yonsei.github.io/SMURF/

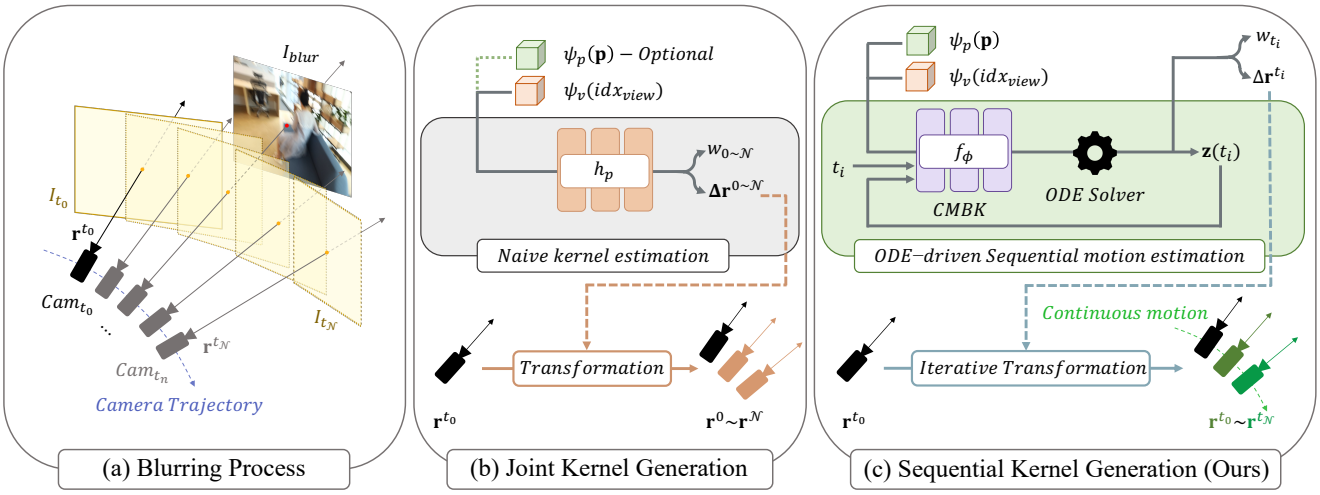

Figure 1. **Comparison of joint kernel generation and our sequential kernel generation.** A blurry image I_{blur} is acquired as the camera moves over the exposure time ($t_0 \sim t_{\mathcal{N}}$), with images I_t captured at each camera pose being composited together. Previous methods generate warped rays of the blurring kernel in a single step by a parameterized network h_p without considering the temporal sequence of camera motion. However, our approach iteratively estimates warped rays along the sequential camera motion trajectory.

Abstract

Neural radiance fields (NeRF) has attracted considerable attention for their exceptional ability in synthesizing novel views with high fidelity. However, the presence of motion blur, resulting from slight camera movements during extended shutter exposures, poses a significant challenge, potentially compromising the quality of the reconstructed 3D scenes. While recent studies have addressed this issue, they do not consider the continuous dynamics of camera movements during image acquisition, leading to inaccurate scene reconstruction. To effectively handle this issue, we propose sequential motion understanding radiance fields (SMURF), a novel approach that employs neural ordinary differential equations (Neural-ODEs) to model continuous camera motion and leverages the explicit volumetric representation method for robustness to motion-blurred input images. The core idea of the SMURF is continuous motion blurring kernel (CMBK), a unique module designed to model a continuous camera movements for processing blurry inputs. Our model, rigorously evaluated against benchmark datasets, demonstrates state-of-the-art performance both quantitatively and qualitatively.

1. Introduction

Reconstructing complex 3D scenes from 2D images of different views and re-rendering novel view images represent a core problem in computer vision and graphics, with significant applications in augmented reality (AR) and virtual reality (VR). Over the past few years, photo-realistic novel view synthesis has greatly advanced with the emergence of Neural Radiance Fields (NeRF) [33]. It takes 3D spatial coordinates and 2D viewing directions (*i.e.* 5D vector) as inputs for mapping to the radiance and volume density, where the process is parameterized through an implicit multi-layer perceptron (MLP) based model. Exploiting classical volume rendering techniques [18], NeRF integrates the output radiance and volume density along the emitted rays, making the volume rendering process fully differentiable.

Generally, NeRF variants have reconstructed 3D scenes using well-captured, noise-free images as inputs along with calibrated camera parameters. The training of complex geometry in 3D scenes necessitates sharp images; however, in real-world scenarios, obtaining such images may not always be feasible due to various factors. For instance, to capture a sharp image, it is essential to set the camera's aperture to a small size, thereby increasing the depth of field. However, a smaller aperture demands a significant amount of light, which consequently leads to longer exposure times. During these extended exposure, any movement of the hand-held camera results in undesirable camera motion-blurred images. Recently, many works [25, 26, 31, 40, 57] have been conducted on NeRF that takes camera motion-blurred images as input. Deblur-NeRF [31] first proposes a method that models blurring kernel by imitating the deconvolution method for blind image deblurring, to reconstruct 3D scenes from motion-blurred images and render sharp novel view images. Since the inception of Deblur-NeRF, various methods [25, 26, 40, 57] for estimating the blurring kernel have been proposed. However, their blurring kernel does not continuously model the camera movement during exposure time. Since actual camera movement is normally continuous, it can be represented as a continuous function over time, and such continuous modeling allows for a more precise tracking of the camera movement path, even when the movement is complex or irregular. Previous works model the blurring kernel without considering the sequential camera movement, making the kernel for images with complex camera motion imprecise.

In this paper, we propose a sequential motion understanding radiance fields (SMURF), which incorporates a novel continuous motion blur kernel (CMBK) for modeling precise camera movements from blurry images. The CMBK estimates the small change in pose regarding continuous camera motion. The output values of the CMBK are not computed jointly as in previous studies but are designed to be computed sequentially according to camera motion, as shown in Fig. 1. Particularly, recognizing that the sequential camera movements share a single continuous function, we employ a neural ordinary differential equations (Neural-ODEs) [7] to ensure that the sequentially computed camera movements exhibits continuity. Specifically, to reflect the physics inherent in camera motion, we apply continuous dynamics in the latent space to CMBK, and transform the latent feature into changes in ray within the physical space. Additionally, we introduce two regularization strategies to prevent the divergence of camera motion generated through CMBK. The first is residual momentum, which ensures that the warped ray does not deviate significantly from the previous ray, thereby preventing overfitting. The second is the output suppression loss, which also serves as a regularization method to ensure that the warped ray does not diverge significantly from the initial ray.

Our SMURF leverages explicit volumetric representation methods [6, 17, 34, 51] as its backbone. Specifically, the tensor factorization-based approach, Tensorial Radiance Fields (TensoRF) [6], facilitates compact and efficient 3D scene reconstruction. Considering the fact that some parts may exhibit significant blur while others are almost free of blur in a single image, incomplete blurred information and complete sharp information merge into a few adjacent voxels via blurring kernel and ensure a coherent 3D scene. Therefore, we adopt the TensoRF as our backbone as it effectively mitigates the uncertainty of information caused by motion blur and enables faster training and high quality rendering compared to previous approaches. While there exists a deblurring approach [26] that uses voxel-based radiance fields as a backbone and shares the common goal of fast training with ours, a key difference lies in their use of Plenoxels [60] as the backbone. Furthermore, while they adopt the different explicit representation method from ours with another goal of efficient memory consumption, we diverge in our objectives, as we aim to leverage the advantages of a 3D voxel-based method when utilizing blurry images as input.

To demonstrate the effectiveness of the proposed SMURF, we conduct extensive experiments on synthetic and real-world scenes. Our experimental results elucidate the advantages of the CMBK in comparison to previously presented blurring kernels.

Our main contributions are summarized as follows:

- We propose a continuous motion blur kernel (CMBK) to sequentially estimate continuous camera motion from blurry images mimicking the real-world motion blur.
- We propose two regularization strategies that guide the warped rays to prevent divergence: residual momentum, and output suppression loss.
- Our sequential motion understanding radiance fields (SMURF) exhibit higher visual quality and quantitative performance compared to existing approaches.

2. Related Work

2.1. Neural Radiance Fields (NeRF)

The synthesis of photo-realistic novel views from images of different viewpoints has attracted considerable attention with the advancement of NeRF [33]. In particular, NeRF uses 3D spatial coordinates and 2D viewing directions to map radiance and volume density, a process facilitated by implicit neural representation (INR). Following the success of NeRF, various sub-domains considering real-world scenarios have been explored. Many recent studies have extensively applied NeRF across various fields, including dynamic 3D scene modeling [3, 27, 28, 30, 37, 38, 44, 55, 63], scene relighting [2, 32, 42, 50, 54], and 3D reconstruc-

tion [53, 56, 58]. Furthermore, recent studies have applied NeRF for deblurring 3D scenes [25, 31, 40, 57], assuming blurry input images from real-world conditions and aiming to produce clean images. Additionally, due to the slow rendering caused by the costly MLP layers required to evaluate each pixel's density and color, some studies [6, 22, 34, 36, 43, 51, 60] design networks for faster training and rendering. To address these challenges, various voxel-based explicit rendering methods [6, 34, 51, 60] have been proposed, offering competitive performance to NeRF while significantly reducing training and rendering times. We adopt an explicit representation method, TensoRF [6] as the backbone, which facilitates fast training and achieves accurate and efficient novel-view rendering.

2.2. NeRFs from Blurry Images

Recently, several studies have been conducted to deblur 3D scenes and synthesize clean novel view images using blurry input images from different views. Deblur-NeRF [31] proposes, for the first time, an implicit blurring kernel in 3D space inspired by blind deblurring methods for 2D vision [16, 21, 52]. This blurring kernel is intentionally trained close to the pixels of the blurry image, and during rendering, a clean image is obtained without the trained kernel. DP-NeRF [25] models a rigid kernel that assumes physical scene priors when blurry images are captured. BAD-NeRF [57], assuming a very short camera exposure time, designs a kernel that linearly interpolates the camera motion trajectory in using simple spline-based method. Recently, methods utilizing 3DGS have been proposed to enable faster rendering. For instance, Deblurring 3DGS [24] adjusts Gaussian parameters like rotation and scaling to generate blurry images during training, and BAGS [41] proposes a blur-agnostic kernel and a blur mask using convolutional neural networks (CNNs). However, these approaches estimate the change in camera pose jointly, not modeling the fact that light traces left by camera motion during exposure time appear continuously. Therefore, we aim to design a kernel where the elements of the blurring kernel, namely the small changes in camera pose, are estimated sequentially over time.

2.3. Neural Ordinary Differential Equations

Neural-ODEs [7] are proposed to interpret neural networks within the framework of ordinary differential equations, representing the underlying dynamics inherent in hidden parameters. Neural-ODEs model a parameterized time-continuous latent state and output a unique solution of the integral of continuous dynamics, utilizing given initial values and various numerical differential equation solvers (*e.g.* Euler's method [11], Runge-Kutta method [23], Dormand-Prince-Shampine method [10]). They are extensively used to continuously connect the latent space embedded in videos [12, 15, 19, 39] or model the continuous dynamics of irregularly sampled time-series data [45]. Inspired by the fact that camera motion is continuous, we exploit Neural-ODEs to create a precise blurring kernel that models the hidden continuous dynamics inherent in camera motion.

3. Preliminary

Blind Deblurring for 3D Scene. For image blind deblurring [5, 49, 59], the blurring kernel h is estimated in a fully unsupervised manner, and this process is achieved by convolving h with sharp images. Deblur-NeRF [31] apply this algorithm to NeRF, modeling an adaptive kernel h_p for each ray,

$$\mathbf{c}_{blur}(\mathbf{r}) = \mathbf{c}_p(\mathbf{r}) * h_p(\mathbf{r}), \tag{1}$$

where $\mathbf{c}_{blur}$ and $\mathbf{c}_p$ respectively denote the color of the blurry pixel and the sharp pixel; $*$ indicates the convolution operator.

The conventional image deblurring kernel takes a fixed grid of size $K \times K$ centered around the location p. However, for 3D scene deblurring, applying such a kernel based on NeRF requires computing $\mathbf{c}_p$ for K^2 rays, which is inefficient in terms of memory and training time. To optimize a kernel for the 3D scene, it is necessary to produce a sparse kernel with fewer rays. Deblur-NeRF designs the sprase kernel to acquire blurry color, and for the temporally continuous camera motion, we extend the process as follows:

$$\mathbf{c}_{blur}(\mathbf{r}) = \sum_{i=0}^{\mathcal{N}} w_p^{t_i} \mathbf{c}_p(\mathbf{r}^{t_i}),\ w.r.t.\ \sum_{i=0}^{\mathcal{N}} w_p^{t_i} = 1, \tag{2}$$

where $\mathcal{N}$ and t_i respectively denote the number of warped rays in camera motion and the instantaneous time during exposure; w_p is the corresponding weight at each ray's location, and $\mathbf{r}^{t_i}$ represents the warped ray due to the camera movement. As the warped rays are determined by parameterized learning, the blurring kernel is deformable, not in a fixed grid manner.

Tensorial Radiance Fields (TensoRF). We follow the architecture of TensoRF [6], an explicit voxel-based volumetric representation utilizing CANDECOMP/PARAFAC (CP) [4, 13] and block term decomposition [9], which models an efficient view-dependent sparse voxel grid. TensoRF optimizes two 3D grid tensors, $\mathcal{G}_\sigma, \mathcal{G}_\mathbf{c} \in \mathbb{R}^{FXYZ}$, for estimating volume density and view-dependent appearance feature, employing the vector-matrix (VM) decomposition that effectively extends CP decomposition. Exploiting this method, the grid (3rd-order) tensor is decomposed into low-rank 1D vectors and 2D matrices across three modes, reducing the space complexity from $\mathcal{O}(n^3)$ to $\mathcal{O}(n^2)$ compared

to conventional explicit representations [60]. This grid is represented as follows:

$$\mathcal{G} = \sum_{r=1}^{R_1} \mathbf{v}_r^X \circ \mathbf{M}_r^{YZ} + \sum_{r=1}^{R_2} \mathbf{v}_r^Y \circ \mathbf{M}_r^{XZ} + \sum_{r=1}^{R_3} \mathbf{v}_r^Z \circ \mathbf{M}_r^{XY}, \quad (3)$$

where $\circ$ denotes the outer product; R_1, R_2, and R_3 indicates the number of low-rank components. $\mathbf{v}^X \in \mathbb{R}^X$ is the vector along the X axes in each mode, and $\mathbf{M}^{YZ} \in \mathbb{R}^{YZ}$ is the matrix in the YZ plane for each mode. Once the grids are defined, the radiance field at a 3D point $\mathbf{x}$ for computing volume density σ and color $\mathbf{c}$ is defined as follows:

$$\sigma(\mathbf{x}), \mathbf{c}(\mathbf{x}) = \mathcal{G}_\sigma(\mathbf{x}), \mathcal{S}(\mathcal{G}_\mathbf{c}(\mathbf{x}), d), \quad (4)$$

where $\mathcal{S}$ denotes a parameterized shallow MLP that converts viewing direction d and appearance feature $\mathcal{G}_\mathbf{c}(\mathbf{x})$ into color. The features obtained from the grid, $\mathcal{G}_\sigma(\mathbf{x})$ and $\mathcal{G}_\mathbf{c}(\mathbf{x})$, are trilinearly interpolated from adjacent voxels.

To render an image for a given view, TensoRF uses a differentiable classic volume rendering technique [18] with σ and $\mathbf{c}$. The color $\mathbf{c}_p$ for each pixel reached by ray $\mathbf{r}$ is computed as follows:

$$\mathbf{c}_p(\mathbf{r}) = \sum_{i=1}^{N} T_i \left(1 - \exp\left(-\sigma_i \delta_i\right)\right) \mathbf{c}_i, \quad (5)$$

where $T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right)$ is the transmittance; N and δ denote the number of sampled points and the step size between adjacent samples on ray $\mathbf{r}$, respectively.

In this paper, we employ TensoRF as our backbone, as it effectively reduces the uncertainty in information caused by motion blur, enabling faster training and higher-quality rendering compared to previous implicit neural representation.

4. Method

4.1. Continuous Motion Blur Kernel

Continuous Latent Space Modeling for Camera Motion. Camera motion blur in images arises from camera movement during exposure time. Such movement, due to the motion of the hand, allows for the representation of camera pose changes as a temporally continuous function, as shown in Fig. 1. Therefore, our goal is to model the continuous movement of the camera within the exposure time ($t_0 \leq t < t_\mathcal{N}$). To reflect the physics inherent in camera motion, we apply continuous dynamics to the latent space of camera motion, and transform the latent feature into changes in ray within the physical space. For this process, we transform the given information of the rays into latent features and apply them to a Neural-ODEs [7] to design a continuous latent space.

As shown in Fig. 4, we embed information about the initial ray into the latent space using a parameterized encoder $\mathcal{E}_\theta$, utilizing the scene's view index idx_{view} and 2D pixel location $\mathbf{p}$:

$$\mathbf{z}(t_0) = \mathcal{E}_\theta\left(\psi_v(idx_{view}),\ \psi_p(\mathbf{p})\right), \quad (6)$$

where $\mathbf{z}(t_0) \in \mathbb{R}^d$ is the latent feature of initial ray with hidden dimension d, and ψ_v and ψ_p denote embedding functions for view and pixel information, respectively. Within a small step limit of the latent feature $\mathbf{z}(t)$, the local continuous dynamic is expressed as $\mathbf{z}(t+\epsilon) = \mathbf{z}(t) + \epsilon \cdot d\mathbf{z}/dt$. To apply continuous dynamics to $\mathbf{z}(t)$, we model a parameterized neural derivative function f in the latent space. The derivative of the continuous function is defined as follows:

$$\frac{d\mathbf{z}(t)}{dt} = f(\mathbf{z}(t), t; \phi), \quad (7)$$

where ϕ denotes the learnable parameters of f. With $\mathbf{z}(t_0)$ and the derivative function f, we define an initial value problem (IVP), and the features of subsequential rays in the latent space are obtained by the integral of f from t_0 to the desired time. This dynamic leads to the format of ODEs, and the process of obtaining latent features of the camera motion trajectory using various ODE solvers [10, 11, 23] is expressed as follows:

$$\mathbf{z}\left(t_{n+1}\right) = \mathbf{z}\left(t_n\right) + \int_{t_n}^{t_{n+1}} f(\mathbf{z}(t), t; \phi) dt, \quad (8)$$

where $0 \leq n < \mathcal{N}$, and $\mathcal{N}$ is the number of rays in the camera motion. As a result of this approach, we obtain the latent features $\mathbf{Z} \in \mathbb{R}^{\mathcal{N} \times d}$ for $\mathcal{N}$ rays.

Motion-blurred images all have different exposure times if camera setting is not fixed. Therefore, assuming a common exposure time for all images when obtaining latent features through the ODE solver may lead to suboptimal local minima as all the images exhibit varying degrees of blur due to different exposure times. Inspired by the difference in exposure time for images of a single scene, we define a chrono-view embedding function $\mathbf{\Psi}$ with a single-layer MLP that simultaneously embeds given time t and view index idx_{view}. Then, f is expressed as follows:

$$f(\mathbf{z}(t), t; \phi) \rightarrow f(\mathbf{z}(t), t, \mathbf{\Psi}(t; idx_{view}); \phi). \quad (9)$$

Exploiting $\mathbf{\Psi}$, when the conditions of t_n and idx_{view} are given to the ODE solver for the IVP, the subsequential latent feature is defined as an unique solution by Picard's existence theorem [29].

Motion-Blurring Kernel Generation. The latent features $\mathbf{Z}$ of all rays on the camera motion trajectory, obtained through continuous dynamics modeling, must be decoded

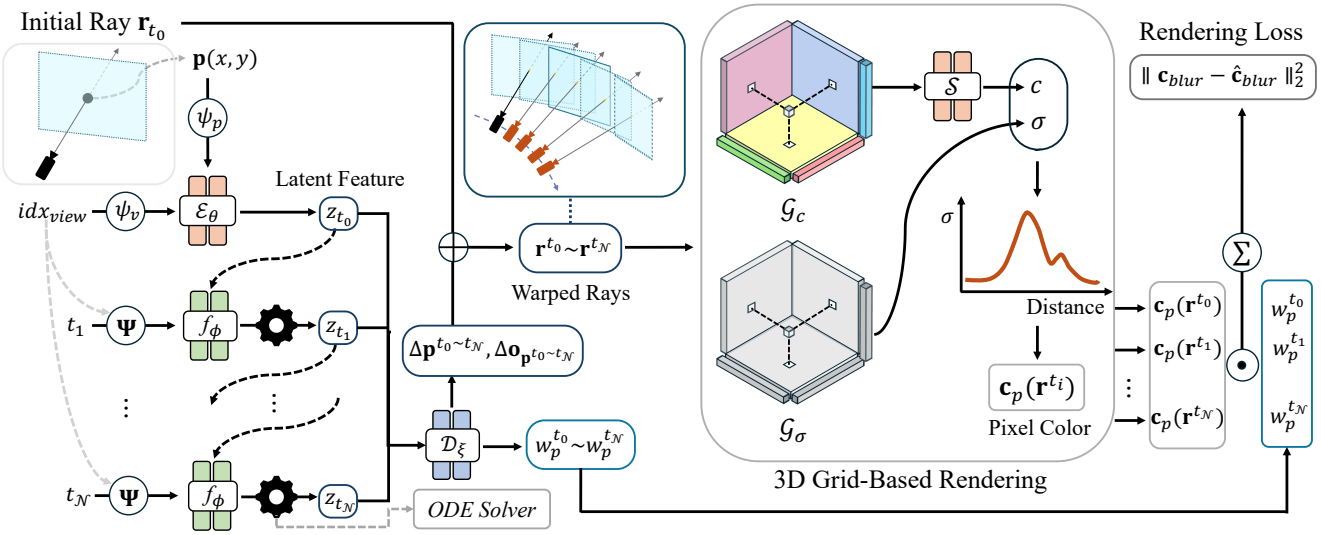

Figure 2. **Method overview of the SMURF**. The CMBK encoder $\mathcal{E}_\theta$ transforms the embedded 2D pixel location $\mathbf{p}$ of the initial ray and view index into a latent feature. This feature is extended into an IVP with a parameterized derivative function f_ϕ in the latent space. Then, it is solved by Neural-ODEs along with given time t and a chrono-view embedding function $\mathbf{\Psi}$, obtaining latent features for all warped rays. These features are transformed into changes of the ray (*i.e.*, $\Delta\mathbf{p}$ and $\Delta\mathbf{o}$) through a decoder $\mathcal{D}_\xi$, and we get the warped rays by applying Eq. (11) to the initial ray. These rays are rendered into 2D pixel colors through a 3D grid-based method, and a blurry color is acquired by summing up the colors with weights from the decoder.

into changes in camera pose. We define a decoder $\mathcal{D}_\xi$, represented by a shallow MLP parameterized by ξ, which outputs three components: the change in 2D kernel location $\Delta\mathbf{p}$, the change in ray origin $\Delta\mathbf{o}_\mathbf{p}$, and the corresponding weight w_p as defined in Eq. (2):

$$(\Delta\mathbf{p}^t, \Delta\mathbf{o}_{\mathbf{p}^t}, w_p^t) = \mathcal{D}_\xi(\mathbf{z}(t)). \tag{10}$$

Then, t-th warped ray $\mathbf{r}^t$ is generated from the initial ray $\mathbf{r} = \mathbf{o} + \tau\mathbf{d}$ by following equation:

$$\mathbf{r}^t = (\mathbf{o} + \Delta\mathbf{o}_{\mathbf{p}^t}) + \tau\mathbf{d}_{\mathbf{p}^t},\ \mathbf{p}^t = \mathbf{p} + \Delta\mathbf{p}^t, \tag{11}$$

where $\mathbf{o}$ and $\mathbf{p}$ denote the ray origin and the 2D pixel location of initial ray, respectively; $\mathbf{d}_{\mathbf{p}^t}$ stands for the warped direction by $\mathbf{p}^t$. After acquiring sharp pixel colors $\mathbf{c}_p(\mathbf{r}^t)$ for all $\mathcal{N}$ warped rays with inherent continuous camera movement, the blurry pixel color is computed using Eq. (2).

Residual Momentum. Latent features of the camera motion trajectory are predicted by CMBK and are expected to be optimized continuously. In this process, predicted origin and direction of the ray $\mathbf{r}^t$ may diverge from the initial ray $\mathbf{r}$, potentially leading to a suboptimal kernel. Deblur-NeRF prevents ray divergence by applying a hyperbolic tangent function to $\Delta\mathbf{p}^t$ for regularization. However, such regularization is inefficient due to the varying intensity of motion blur across all the images. To address this issue, we apply a residual term to the latent derivative function f, implementing regularization that ensures the predicted ray $\mathbf{r}^{t_i}$ does not significantly deviate from the previous ray $\mathbf{r}^{t_{i-1}}$:

$$f_\phi(\mathbf{z}(t_{i-1})) \rightarrow \Lambda(f_\phi(\mathbf{z}(t_{i-1})) + \mathbf{z}(t_{i-1})), \tag{12}$$

where the Λ is a shallow MLP for regularization and $f_\phi(\mathbf{z})$ is the simplified version of $f(\mathbf{z}, t, \mathbf{\Psi}(t; idx_{view}); \phi)$. This approach prevents the divergence of the camera trajectory, allowing rays to be warped regardless of the motion blur intensity of the image. Note that while the proposed residual momentum is implemented similarly to the methodology of residual connections in ResNet [14], the underlying concepts of the two approaches are distinct.

Output Suppression Loss. We encode the ray and the view information into the latent space using $\mathcal{E}_\theta$, and decode it into changes of the ray in camera motion trajectory using $\mathcal{D}_\xi$. In this process, since latent feature the initial ray $\mathbf{z}(t_0)$ serves as the initial value for the ODE, there should be no change in the initial ray. Therefore, we apply an output suppression loss as follows:

$$\mathcal{L}_{supp} = \lambda_{supp} \left\| \mathcal{D}_\xi(\mathbf{z}(t_0)) \right\|_2, \tag{13}$$

where λ is the weight for the loss $\mathcal{L}_{supp}$. In this process, the color weight $w_p^{t_0}$ is not included for the loss. This concept is similar to the cycle consistency in CycleGAN [64]. Minimizing the changes of initial ray to zero value, we ensure that not just the initial ray but also the warped rays do not diverge, providing a regularization effect.

Table 1. **Quantitative comparisons on synthetic and real-world scene dataset.** We evaluate the quantitative performance using three metrics, demonstrating that our proposed model significantly ourperforms existing ones even with faster training. The orange and yellow cells respectively indicate the highest and second-highest value.

Methods	Synthetic Scene Dataset			Real-World Scene Dataset		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Naive NeRF [33]	23.78	0.6807	0.3362	22.69	0.6347	0.3687
NeRF+MPR [61]	25.11	0.7476	0.2148	23.38	0.6655	0.3140
NeRF+PVD [48]	24.58	0.7190	0.2475	23.10	0.6389	0.3425
Deblur-NeRF [31]	28.77	0.8593	0.1400	25.63	0.7645	0.1820
PDRF-10* [40]	28.86	0.8795	0.1139	25.90	0.7734	0.1825
BAD-NeRF* [57]	27.32	0.8178	0.1127	22.82	0.6315	0.2887
DP-NeRF [25]	29.23	0.8674	0.1184	25.91	0.7751	0.1602
SMURF	30.98	0.9147	0.0609	26.52	0.7986	0.1013

4.2. Optimization

Following Sec. 3, our goal is to minimize the difference between the blurry pixel color $\hat{\mathbf{c}}_{blur}(\mathbf{r})$ obtained by the blurring kernel and the ground truth pixel color of the motion-blurred image $\mathbf{c}_{blur}(\mathbf{r})$. Hence, the photometric loss $\mathcal{L}_{photo}$ is represented by $\mathcal{L}_{photo} = \|\mathbf{c}_{blur}(\mathbf{r}) - \hat{\mathbf{c}}_{blur}(\mathbf{r})\|_2^2$. Furthermore, the parameterized 3D voxel grids $\mathcal{G}_\sigma$ and $\mathcal{G}_c$ are regularized through the total variation [1] losses $\mathcal{L}_{TV}^\sigma$ and $\mathcal{L}_{TV}^c$, respectively. Additionally, to prevent divergence of the warped rays and ensure initial ray consistency, Eq. (13) is applied, resulting in our combined objective as follows:

$$\mathcal{L} = \mathcal{L}_{photo} + \lambda_{TV}^\sigma \mathcal{L}_{TV}^\sigma + \lambda_{TV}^c \mathcal{L}_{TV}^c + \lambda_{supp} \mathcal{L}_{supp} \quad (14)$$

where the λ_{TV}^σ and the λ_{TV}^c are the weights for $\mathcal{L}_{TV}^\sigma$ and $\mathcal{L}_{TV}^c$, respectively.

5. Experiments

5.1. Implementation Details

Datasets. In this experiment, we utilize the camera motion blur dataset published by [31], which is divided into synthetic and real-world scenes. The synthetic scene dataset comprises five scenes synthesized using Blender [8], with multi-view cameras set up to render images with applied camera motion. The camera motion trajectory is formed by linearly interpolating between the poses of the first and last cameras, and the rendered multi-view images are combined in the linear RGB space to obtain the final blurry images. The real-world scene dataset consists of ten scenes captured with an actual camera. The blurry images are obtained by physically shaking the camera during the exposure time, and the camera poses are calculated using the images obtained with COLMAP [46, 47].

Training Details. We implement our model on TensoRF [6], an explicit grid-based method, as our backbone renderer. We upsample the voxel counts of grids $\mathcal{G}_\sigma$ and $\mathcal{G}_c$ from 128^3 to 480^3, and set the feature component F of the grids to 36 and 96, respectively. To implement the CMBK, we set the number of warped rays $\mathcal{N}$ on the camera motion trajectory to 8 and adopt the Euler method [11] as the solver for the Neural-ODEs. Single scene training is conducted for 40k iterations, with the learning rates for the grids decaying from 0.02 to 0.002, and the learning rate for CMBK decaying from 0.001 to 0.0001. The weights for the losses λ_{TV}^σ, λ_{TV}^c, and λ_{supp} are all set to 0.1. All our experiments are conducted on a single NVIDIA RTX 3090.

5.2. Novel View Synthesis Results

Quantitative Results. We show the quantitative evaluation of our network, SMURF, in comparison to various baselines on the two benchmark datasets proposed by Deblur-NeRF [31], as shown in Tab. 1. NeRF+MPR and NeRF+PVD are trained with a naive NeRF, where the single-image deblurring methods MPR [62] and PVD [48] to the input data. Additionally, since PDRF-10 [40] does not specify LPIPS in their experiment, we obtain performance of three evaluation metrics using their released code. Furthermore, as BAD-NeRF [57] specifies results from experiments on a modified benchmark dataset, for a fair evaluation, we apply the released code of BAD-NeRF to the benchmark dataset. According to Tab. 1, despite significantly reduced training times compared to previous methods, we demonstrate superior quantitative performance across all the metrics. Especially, LPIPS of SMURF demonstrates SMURF's exceptional perceptual quality across all datasets. Quantitative results of indiividual scenes are in the **appendix**.

Qualitative Results. We validate the effectiveness of SMURF's quantitatively high performance through qualitative evaluation via novel view rendering. According to Fig. 3, we compare our results with previous 3D scene deblurring methods for five scenes ("FACTORY", "TANABATA, GIRL", "BUICK", and "POOL"). For the "FACTORY" scene, when comparing the perceptual quality of the reconstructed lowest part of the stairs, it is evident that BAD-NeRF and SMURF best estimate the 3D geometry. However, BAD-NeRF shows an inability to capture the overall color as accurately as other methods. In the rendering results for the "TANABATA" scene, while other methods fail to accurately estimate the position of the power lines, SMURF restores them most similarly to the reference image. In the "GIRL" and "BUICK" scenes, our method notably restores the visual quality of the shelf's black support rods and the car logo, respectively, most closely to the reference images. For the "BASKET" scene, the holes of the

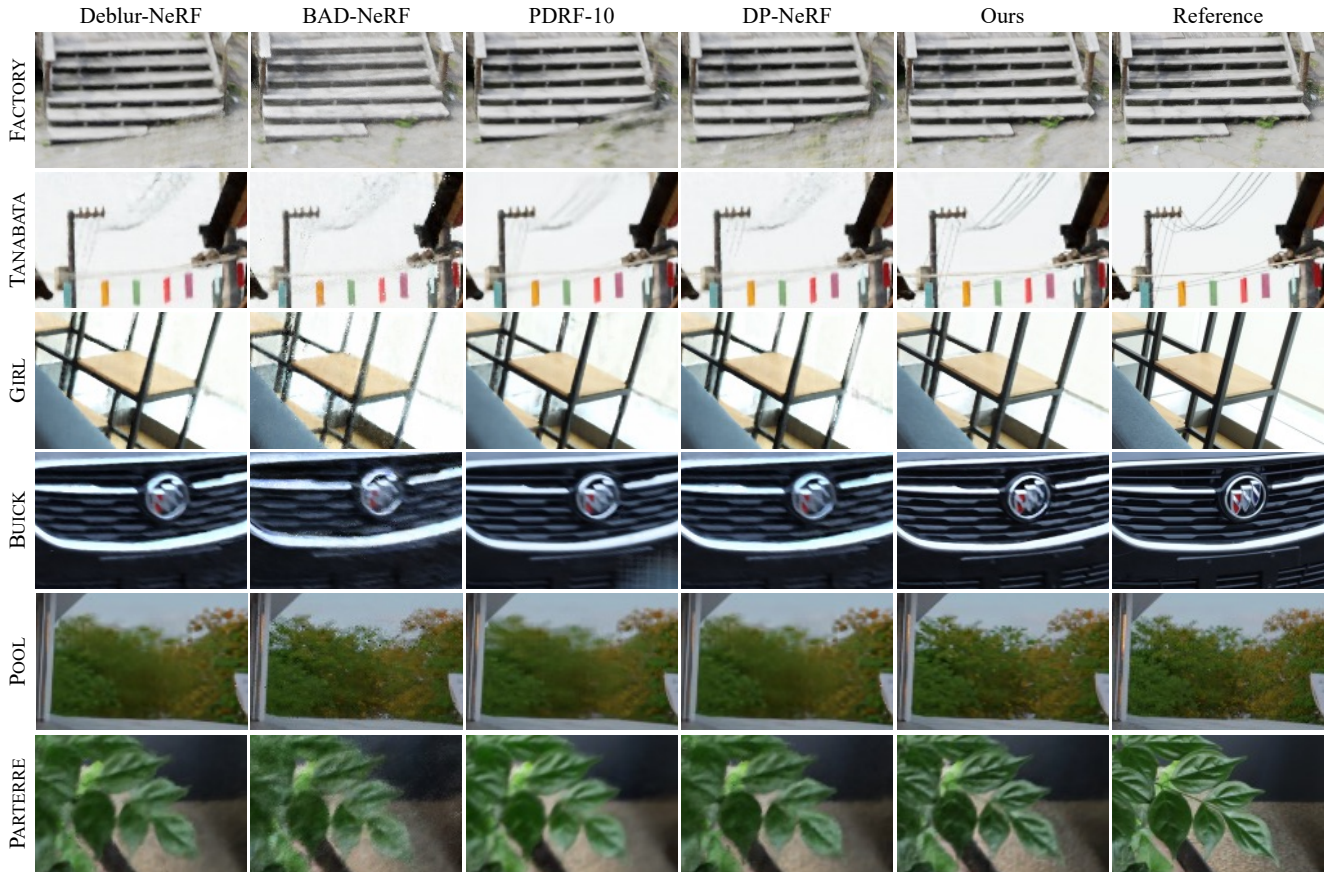

Figure 3. **Qualitative comparisons on synthetic scenes and real-world scenes.** SMURF produces results most similar to the reference images and models the detailed aspects that are not captured by previous methods.

Table 2. **Comparison of performance for individual scenes on the synthetic dataset.** SMURF exhibits higher performance across all scenes, with the exception of "COZYROOM," where it shows slightly lower performance relative to others.

Synthetic Scene	FACTORY			COZYROOM			POOL			TANABATA			TROLLEY		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Naive NeRF	19.32	0.4563	0.5304	25.66	0.7941	0.2288	30.45	0.8354	0.1932	22.22	0.6807	0.3653	21.25	0.6370	0.3633
MPR+NeRF	21.70	0.6153	0.3094	27.88	0.8502	0.1153	30.64	0.8385	0.1641	22.71	0.7199	0.2509	22.64	0.7141	0.2344
PVD+NeRF	20.33	0.5386	0.3667	27.74	0.8296	0.1451	27.56	0.7626	0.2148	23.44	0.7293	0.2542	23.81	0.7351	0.2567
Deblur-NeRF	25.60	0.7750	0.2687	32.08	0.9261	0.0477	31.61	0.8682	0.1246	27.11	0.8640	0.1228	27.45	0.8632	0.1363
PDRF-10*	25.87	0.8316	0.1915	31.13	0.9225	0.0439	31.00	0.8583	0.1408	28.01	0.8931	0.1004	28.29	0.8921	0.0931
BAD-NeRF*	24.43	0.7274	0.2134	29.77	0.8864	0.0616	31.51	0.8620	0.0802	25.32	0.8081	0.1077	25.58	0.8049	0.1008
DP-NeRF	25.91	0.7787	0.2494	32.65	0.9317	0.0355	31.96	0.8768	0.0908	27.61	0.8748	0.1033	28.03	0.8752	0.1129
SMURF	29.87	0.8958	0.1057	32.48	0.9285	0.0379	32.34	0.8884	0.0779	29.91	0.9300	0.0436	30.30	0.9307	0.0397

basket further demonstrate the superior qualitative outcome of our results. Additional novel view rendering results, are in the **appendix**.

5.3. Ablation Study

We conduct ablative experiments on all scenes in the given datasets, with the results presented in Tab. 3. Additionally, we perform experiments on the number of warping rays, $\mathcal{N}$, which are provided in the **appendix**.

Chrono-View Embedding Function. To demonstrate the effectiveness of the chrono-view embedding function $\mathbf{\Psi}$ in making continuous latent space modeling robust to exposure time, as discussed in Sec. 4, we conduct experiments

Table 3. **Ablations on embedding type and regularization strategies.** "Emb.", "O.S. Loss", and "R.M." refer to embedding type, the output suppression loss, and the residual momentum, respectively. "C.V" and "T" denote chrono-view and time embeddings.

Emb.	O.S.	R.M.	Synthetic Dataset			Real-World Dataset		
			PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
C.V			26.17	0.8186	0.0759	25.27	0.7576	0.1121
C.V		✓	28.49	0.8727	0.0675	25.60	0.7748	0.1057
C.V	✓		30.72	0.9052	0.0684	25.99	0.7846	0.1181
–	✓	✓	30.54	0.8995	0.0709	25.74	0.7755	0.1194
T.	✓	✓	30.94	**0.9158**	0.0610	26.34	0.7881	0.1102
C.V	✓	✓	**30.98**	0.9147	**0.0609**	**26.52**	**0.7986**	**0.1013**

by dividing the embedding types into three categories. The time embedding assumes that all images have the same exposure time without incorporating view information, while the chrono-time embedding applies view information, ensuring that all images have distinct exposure time information. As shown in Tab. 3, the application of time embedding yields higher performance in general, but chrono-time embedding shows higher performance only in real-world scenes, while its effect in synthetic scenes is minimal. This is attributed to the characteristics of the synthetic dataset created by Deblur-NeRF using Blender, where motion blur images are acquired by linearly interpolating between camera poses. In other words, the synthetic scenes set a constant speed for camera motion throughout the exposure time for all images, implying that all input images share the same exposure time. Our chrono-view embedding function assumes that all images have varying exposure times, leading to its minimal effect on the synthetic dataset. However, the real-world dataset consists of images captured with an actual camera, where the speed of camera motion is not constant during the exposure time. Therefore, applying this function to the real-world scenes yield improved performance due to these variations.

Regularzation Strategies. To validate the effectiveness of the regularization strategies presented in Sec. 4, namely output suppression loss and residual momentum, we conduct various experiments as shown in Tab. 3. The output suppression loss is designed to suppress changes in the origin and direction of the initial ray on the camera motion trajectory to zero, yielding higher performance when applied. This is because the changes in rays are prevented from diverging since the estimated poses also share the same derivative function f, similar to the effect of the alignment loss proposed by Deblur-NeRF [31]. This similarity suggests that aligning ray origins is robust to inaccuracies in camera pose, which can be ambiguous due to motion blur. Consequently, Tab. 3 demonstrates that the performance of PSNR and SSIM in synthetic and real-world scenes improves more significantly.

Residual momentum offers a similar effect to output suppression loss. While the change in camera poses is estimated through a parameterized differential equation, the shape of the derivative obtained in an unsupervised manner could diverge. Therefore, we use residual momentum to ensure the next pose does not deviate significantly from the previous one. Tab. 3 shows that applying another regularization strategy, residual momentum, shows better performance in terms of LPIPS than output suppression loss. Combining two strategies, the SMURF results in best performance across all metrics.

6. Limitations and Future Work

By adopting TensoRF [6], a 3D tensor factorization-based method, as our backbone, we ensure high quantitative performance, superior perceptual quality, and faster training. However, with the advent of 3D Gaussian Splatting [20], which allows for GPU-based rasterization instead of ray tracing-based methods, backbones that facilitate more faster training and rendering become available. Although our backbone shows slower training and rendering speed compared to 3D Gaussian Splatting, applying the main idea of CMBK, *continuous dynamics*, to a rasterization-based method is expected to result in faster training and rendering with superior perceptual quality. Furthermore, by demonstrating the applicability of *continuous dynamics* to the 3D scene deblurring, we anticipate the possibility of designing models that cover not only camera motion blur but also object motion blur, which is caused by the movement of objects within the scene.

7. Conclusion

We have proposed SMURF, a novel approach that sequentially models accurate camera motion for reconstructing sharp 3D scenes from motion-blurred images. Unlike previous approaches that estimate camera motion in a single step, SMURF incorporates, for the first time, a kernel for estimating sequential camera motions, named the CMBK. This camera motion is represented with continuity by solving continuous dynamics in the latent space using Neural-ODEs. To prevent the divergence of rays estimated by CMBK beyond the motion blur range, we apply regularization techniques: residual momentum and output suppression loss. Furthermore, we model the 3D scene using tensor factorization-based representation, which allows for the integration of incomplete blurred information and complete sharp information within adjacent voxels via CMBK, thereby reducing the uncertainty of blurry information. SMURF significantly outperform previous works quantitatively with faster training, and its qualitative evaluation is demonstrated through novel view rendering results.

References

[1] Amir Beck and Marc Teboulle. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. *IEEE transactions on image processing*, 18(11):2419–2434, 2009. 6

[2] Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824*, 2020. 2

[3] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 130–141, 2023. 2

[4] J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319, 1970. 3

[5] 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

[6] 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, 6, 8, 4

[7] 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, 1

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

[9] Lieven De Lathauwer. Decompositions of a higher-order tensor in block terms—part ii: Definitions and uniqueness. *SIAM Journal on Matrix Analysis and Applications*, 30(3): 1033–1066, 2008. 3

[10] John R Dormand and Peter J Prince. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26, 1980. 3, 4

[11] Leonhard Euler. *Institutionum calculi integralis*. impensis Academiae imperialis scientiarum, 1845. 3, 4, 6

[12] Steven Fernandes, Sunny Raj, Eddy Ortiz, Iustina Vintila, Margaret Salter, Gordana Urosevic, and Sumit Jha. Predicting heart rate variations of deepfake videos using neural ode. In *Proceedings of the IEEE/CVF international conference on computer vision workshops*, pages 0–0, 2019. 3

[13] Richard A Harshman et al. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970. 3

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[15] Florian Hofherr, Lukas Koestler, Florian Bernard, and Daniel Cremers. Neural implicit representations for physical parameter inference from a single video. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2093–2103, 2023. 3

[16] Neel Joshi, C Lawrence Zitnick, Richard Szeliski, and David J Kriegman. Image deblurring and denoising using color priors. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1550–1557. IEEE, 2009. 3

[17] Kim Jun-Seong, Kim Yu-Ji, Moon Ye-Bin, and Tae-Hyun Oh. Hdr-plenoxels: Self-calibrating high dynamic range radiance fields. *arXiv preprint arXiv:2208.06787*, 2022. 2

[18] James T Kajiya and Brian P Von Herzen. cing volume densities. *ACM SIGGRAPH computer graphics*, 18(3):165–174, 1984. 1, 4

[19] David Kanaa, Vikram Voleti, Samira Ebrahimi Kahou, and Christopher Pal. Simple video generation using neural odes. *arXiv preprint arXiv:2109.03292*, 2021. 3

[20] 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. 8, 4

[21] Dilip Krishnan, Terence Tay, and Rob Fergus. Blind deconvolution using a normalized sparsity measure. In *CVPR 2011*, pages 233–240. IEEE, 2011. 3

[22] Andreas Kurz, Thomas Neff, Zhaoyang Lv, Michael Zollhöfer, and Markus Steinberger. Adanerf: Adaptive sampling for real-time rendering of neural radiance fields. In *European Conference on Computer Vision*, pages 254–270. Springer, 2022. 3

[23] Wilhelm Kutta. *Beitrag zur näherungsweisen Integration totaler Differentialgleichungen*. Teubner, 1901. 3, 4

[24] Byeonghyeon Lee, Howoong Lee, Xiangyu Sun, Usman Ali, and Eunbyung Park. Deblurring 3d gaussian splatting. *arXiv preprint arXiv:2401.00834*, 2024. 3

[25] 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, 6, 4

[26] 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

[27] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5521–5531, 2022. 2

[28] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. 2

[29] 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. 4, 1

[30] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*, 2019. 2

[31] 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, 6, 8

[32] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021. 2

[33] 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, 6

[34] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 2, 3

[35] Shree K Nayar and Moshe Ben-Ezra. Motion-based motion deblurring. *IEEE transactions on pattern analysis and machine intelligence*, 26(6):689–698, 2004. 2

[36] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Joerg H Mueller, Chakravarty R Alla Chaitanya, Anton Kaplanyan, and Markus Steinberger. Donerf: Towards real-time rendering of compact neural radiance fields using depth oracle networks. In *Computer Graphics Forum*, pages 45–59. Wiley Online Library, 2021. 3

[37] 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. 2

[38] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228*, 2021. 2

[39] Sunghyun Park, Kangyeol Kim, Junsoo Lee, Jaegul Choo, Joonseok Lee, Sookyung Kim, and Edward Choi. Vid-ode: Continuous-time video generation with neural ordinary differential equation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2412–2422, 2021. 3

[40] 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, 3, 6

[41] 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. 3

[42] Julien Philip, Sébastien Morgenthaler, Michaël Gharbi, and George Drettakis. Free-viewpoint indoor neural relighting from multi-view stereo. *ACM Transactions on Graphics (TOG)*, 40(5):1–18, 2021. 2

[43] Martin Piala and Ronald Clark. Terminerf: Ray termination prediction for efficient neural rendering. In *2021 International Conference on 3D Vision (3DV)*, pages 1106–1114. IEEE, 2021. 3

[44] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 2

[45] Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019. 3

[46] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 6

[47] 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. 6

[48] Hyeongseok Son, Junyong Lee, Jonghyeop Lee, Sunghyun Cho, and Seungyong Lee. Recurrent video deblurring with blur-invariant motion estimation and pixel volumes. *ACM Transactions on Graphics (TOG)*, 40(5):1–18, 2021. 6

[49] 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

[50] Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7495–7504, 2021. 2

[51] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 2, 3

[52] Jian Sun, Wenfei Cao, Zongben Xu, and Jean Ponce. Learning a convolutional neural network for non-uniform motion blur removal. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 769–777, 2015. 3

[53] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15598–15607, 2021. 3

[54] Marco Toschi, Riccardo De Matteo, Riccardo Spezialetti, Daniele De Gregorio, Luigi Di Stefano, and Samuele Salti. Relight my nerf: A dataset for novel view synthesis and

relighting of real world objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20762–20772, 2023. 2

[55] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12959–12970, 2021. 2

[56] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 3

[57] 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, 6

[58] Bowen Wen, Jonathan Tremblay, Valts Blukis, Stephen Tyree, Thomas Müller, Alex Evans, Dieter Fox, Jan Kautz, and Stan Birchfield. Bundlesdf: Neural 6-dof tracking and 3d reconstruction of unknown objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 606–617, 2023. 3

[59] 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

[60] Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. *arXiv preprint arXiv:2112.05131*, 2021. 2, 3, 4

[61] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14821–14831, 2021. 6

[62] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14821–14831, 2021. 6

[63] Jiakai Zhang, Xinhang Liu, Xinyi Ye, Fuqiang Zhao, Yanshun Zhang, Minye Wu, Yingliang Zhang, Lan Xu, and Jingyi Yu. Editable free-viewpoint video using a layered neural representation. *ACM Transactions on Graphics (TOG)*, 40(4):1–18, 2021. 2

[64] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 5

SMURF: Continuous Dynamics for Motion-Deblurring Radiance Fields

Supplementary Material

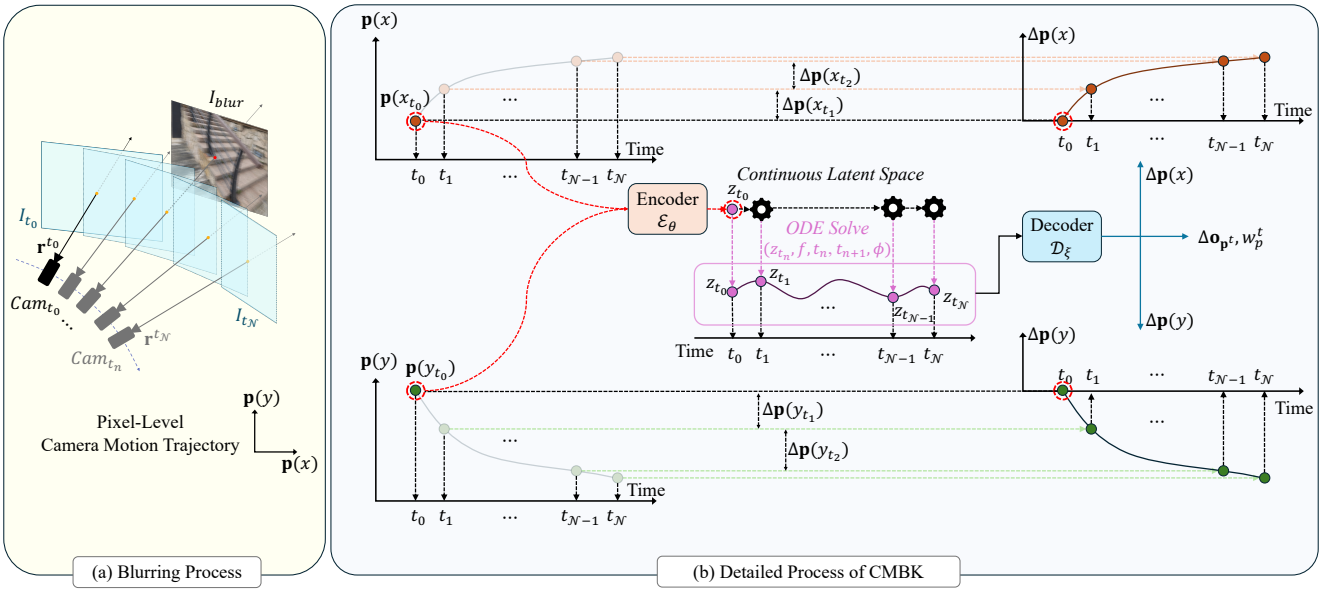

Figure 4. **Detailed process for generating kernel**. To highlight the continuous dynamics in the latent space, all embedding functions are omitted from the figure. $\mathbf{p}(x_{t_0})$ and $\mathbf{p}(y_{t_0})$ correspond to the x and y coordinates of the pixel associated with the initial ray, respectively.

8. Details of CMBK

We elaborate on the continuous dynamics of the proposed CMBK as shown in Fig. 4. We assume that camera motion encompasses inherent dynamics with a unique solution. The assumed solution is represented by the lighter circles in the left plots of Fig. 4 (b). Rather than implementing the inherent dynamics in a simple physical space, we refine them within a latent space with parametric learning. The continuous dynamics of CMBK involve transforming the pixel coordinates corresponding to the ray into latent features via a parameterized encoder $\mathcal{E}_\theta$, and a unique numerical solution [29] is obtained by solving the initial value problem on the latent space through a neural ordinary differential equation [7]. The solution in the latent space is transformed to the physical space by the decoder $\mathcal{D}_\xi$, which represents the change in ray origin, corresponding weight, and the change in pixel of the warped ray relative to the pixel of the initial ray. This change defines the pixel corresponding to the warped ray, and we specify Eq. (11) from the main paper:

$$\begin{aligned} \mathbf{p}^{t_i} &= (\mathbf{p}(x_{t_i}),\ \mathbf{p}(y_{t_i}))\,, \\ \mathbf{p}(x_{t_{i+1}}) &= \mathbf{p}(x_{t_i}) + \Delta\mathbf{p}(x_{t_{i+1}}), \\ \mathbf{p}(y_{t_{i+1}}) &= \mathbf{p}(y_{t_i}) + \Delta\mathbf{p}(y_{t_{i+1}}). \end{aligned} \tag{15}$$

Following our assumption, the change in the initial ray must necessarily be zero, so we proposed the *output suppression loss* to ensure it.

9. Motion Blur in Real-World

In real-world applications in 3D reconstruction, obtaining sharp images necessitates a small aperture size to ensure a substantial depth of field. This small aperture size inherently requires longer exposure times due to the diminished light intake, contradicting the assumption of very short exposure times. Longer exposure times inevitably lead to more intricate blur such as non-uniform blur as shown in Fig. 5, which are not adequately represented by simple modelings such as linear motion assumptions.

Therefore, using neural-ODEs to model camera motion is particularly advantageous as they can accurately capture both linear and non-linear paths. The neural-ODEs are capable of representing a continuum of functions, thus effectively capturing the variations in camera motion regardless of their complexity or the amount of motion. Therefore, our approach is not only theoretically sound but also highly applicable in practical scenarios where the limitations of linear motion assumptions may lead to suboptimals.

Figure 5. Blurry images caused by nonlinear camera path and kernels predicted by BAD-NeRF [57] and our method.

10. Justification for Continuous Camera Motion

The impact of camera movements on image quality is profoundly influenced by the dynamic camera motion, including non-linear trajectories. Therefore, capturing the exact position and orientation of the camera is crucial, which is hard without considering the continuous nature of the motion [35]. While BAD-NeRF [57] uses linear-spline interpolation for camera path estimation, our method surpasses BAD-NeRF, as detailed in the main paper. Spline models are useful for predictable movements, but they lack the flexibility to accurately model the intricate dynamics of camera motion that frequently occurs in real-world. They are constrained to fixed intervals and predefined degrees of freedom, which can oversimplify the motion path. With neural-ODEs, we consider sequences of continuous camera motion that were previously unaccounted for. This approach is advantageous as they can accurately capture both linear and non-linear paths. The neural-ODEs are capable of representing a continuum of functions, thus effectively capturing the variations in camera motion regardless of their motion complexity. As shown in Fig. 5, while the warped rays of kernel of BAD-NeRF are linear, SMURF shows a non-linear camera motion path more close to actual motion blur. Therefore, our approach is not only theoretically sound but also highly applicable in practical scenarios where the limitations of linear motion assumptions may lead to suboptimals.

While neural-ODEs may appear to be mathematically complex, they simply utilize neural networks to solve numerical differential equations, effectively combine traditional calculus with the learning capabilities of modern computing. They not only streamline the modeling of dynamic systems, but it is also straightforward to implement, making it a low-cost solution. Furthermore, it is important to clarify that our primary goal is to model continuous functions. The fact that the deblurring kernel is time-discrete does not contradict this objective. Ours is derived from a model that continuously defines the motion dynamics. We will revise this section to ensure clarity. Moreover, in real-world dataset of Deblur-NeRF [31], there is no ground truth for the actual camera path, and it is also impractical to obtain such ground truth in real scenarios, making it impossible to quantify the approximation error of the camera path.

11. Number of Warped Rays

We conduct extensive experiments to analyze the performance of the proposed CMBK based on the number of warped rays, $\mathcal{N}$. As shown in Sec. 11, larger $\mathcal{N}$ requires the more pixels to be rendered, resulting in an almost linear increase in training time. Moreover, Fig. 6 shows the performance according to the number of warped rays. Across all datasets, an increase in the number of warped rays tends to improve the PSNR and SSIM metrics. LPIPS noticeably decreases with more rays, interpreting that a higher number of warped rays ensures better perceptual quality. The performance for individual scenes in the synthetic dataset is shown in Sec. 11, showing that LPIPS decreases with larger value of $\mathcal{N}$, and overall performance peaks when $\mathcal{N}$ is 8 or 9 except for the "COZYROOM" scene, . Analysis for the "COZYROOM" scene is conducted in Sec. 12. As indicated in Tab. 6, for the real-world dataset, a larger $\mathcal{N}$ generally guarantees higher performance across most scenes. However, there are scenes where performance drops when $\mathcal{N}$ exceeds 9, suggesting that the optimal $\mathcal{N}$ might be smaller than 9. Even with $\mathcal{N}$ set to 5, which is the same condition to DP-NeRF [25] and Deblur-NeRF [31], SMURF outperforms them across all the metrics. Furthermore, SMURF achieves higher performance with fewer warped rays than PDRF-10 [40], which set $\mathcal{N}$ at 10, validating the effectiveness of our proposed ideas.

Table 4. Performance and training time of SMURF according to the number of warped rays. The red, orange, and yellow cells respectively indicate the highest, second-highest, and third-highest value.

Methods	$\mathcal{N}$	Synthetic Scene Dataset			Real-World Scene Dataset			Training Time (h)
		PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	
SMURF	4	29.52	0.8829	0.1014	25.81	0.7710	0.1388	1.16
SMURF	5	30.30	0.9013	0.0926	25.91	0.7811	0.1253	1.27
SMURF	6	29.85	0.8929	0.0851	25.98	0.7822	0.1155	1.43
SMURF	7	30.23	0.8964	0.0751	26.40	0.7944	0.1052	1.56
SMURF	8	30.98	0.9147	0.0609	26.52	0.7986	0.1013	1.72
SMURF	9	30.41	0.9086	0.0575	26.24	0.7922	0.1021	1.88

Table 5. Per-scene quantitative performance of SMURF, according to the number of warped rays.

Synthetic	$\mathcal{N}$	FACTORY			COZYROOM			POOL			TANABATA			TROLLEY		
		PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
SMURF	4	25.13	0.7697	0.2127	32.46	0.9294	0.0407	33.04	0.8993	0.0825	28.38	0.9059	0.0825	28.57	0.9101	0.0890
SMURF	5	27.33	0.8381	0.2075	32.72	0.9315	0.0375	32.47	0.8904	0.0797	29.34	0.9222	0.0631	29.65	0.9241	0.0755
SMURF	6	26.74	0.8159	0.1897	31.74	0.9246	0.0381	32.56	0.8922	0.0778	29.22	0.9179	0.0604	28.99	0.9139	0.0599
SMURF	7	26.15	0.8010	0.1679	32.12	0.9269	0.0378	32.59	0.8926	0.0773	29.69	0.9269	0.0490	30.58	0.9348	0.0438
SMURF	8	29.87	0.8958	0.1057	32.48	0.9285	0.0379	32.34	0.8884	0.0779	29.91	0.9300	0.0436	30.30	0.9307	0.0397
SMURF	9	30.52	0.9065	0.0807	31.43	0.9198	0.0428	32.10	0.8849	0.0771	28.93	0.9184	0.0448	29.05	0.9136	0.0423
Real	$\mathcal{N}$	BALL			BASKET			BUICK			COFFEE			DECORATION		
		PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
SMURF	4	26.47	0.7424	0.1754	28.30	0.8653	0.0847	26.57	0.8270	0.1066	30.66	0.8676	0.1228	24.59	0.7972	0.1471
SMURF	5	26.63	0.7528	0.1549	27.96	0.8648	0.0741	26.03	0.8254	0.0947	30.93	0.8734	0.1172	24.22	0.7799	0.1497
SMURF	6	27.31	0.7678	0.1434	27.19	0.8463	0.0773	26.75	0.8315	0.0905	30.76	0.8731	0.1004	24.16	0.7744	0.1465
SMURF	7	27.68	0.7821	0.1325	29.24	0.8862	0.0617	27.05	0.8395	0.0867	30.67	0.8649	0.0931	24.97	0.8092	0.1220
SMURF	8	27.50	0.7760	0.1298	28.95	0.8842	0.0619	27.10	0.8409	0.0839	31.33	0.8879	0.0874	24.90	0.8114	0.1190
SMURF	9	27.16	0.7698	0.1315	28.52	0.8766	0.0631	26.92	0.8366	0.0813	31.41	0.8802	0.0870	24.12	0.7753	0.1405
Real	$\mathcal{N}$	GIRL			HERON			PARTERRE			PUPPET			STAIR		
		PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
SMURF	4	24.66	0.8410	0.1084	23.36	0.7144	0.1863	25.43	0.7644	0.1611	24.63	0.7496	0.1277	23.38	0.5406	0.1687
SMURF	5	24.73	0.8354	0.1028	23.47	0.7244	0.1717	25.22	0.7640	0.1474	24.69	0.7514	0.1239	25.22	0.6391	0.1171
SMURF	6	25.10	0.8473	0.0940	23.60	0.7386	0.1533	24.99	0.7569	0.1402	24.51	0.7493	0.1145	25.38	0.6371	0.0956
SMURF	7	25.43	0.8570	0.0884	23.66	0.7352	0.1457	25.22	0.7744	0.1302	24.70	0.7587	0.1076	25.42	0.6370	0.0846
SMURF	8	25.66	0.8592	0.0829	23.59	0.7317	0.1381	25.47	0.7825	0.1207	25.19	0.7702	0.1077	25.48	0.6421	0.0822
SMURF	9	25.56	0.8567	0.0822	23.81	0.7828	0.1333	24.91	0.7583	0.1216	24.71	0.7554	0.1035	25.31	0.6302	0.0777

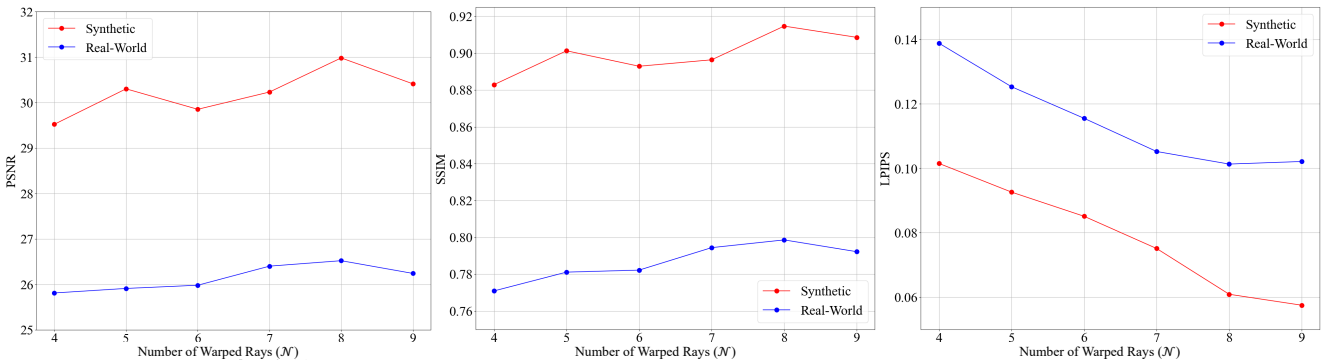

Figure 6. Variation in performance with the number of warped rays.

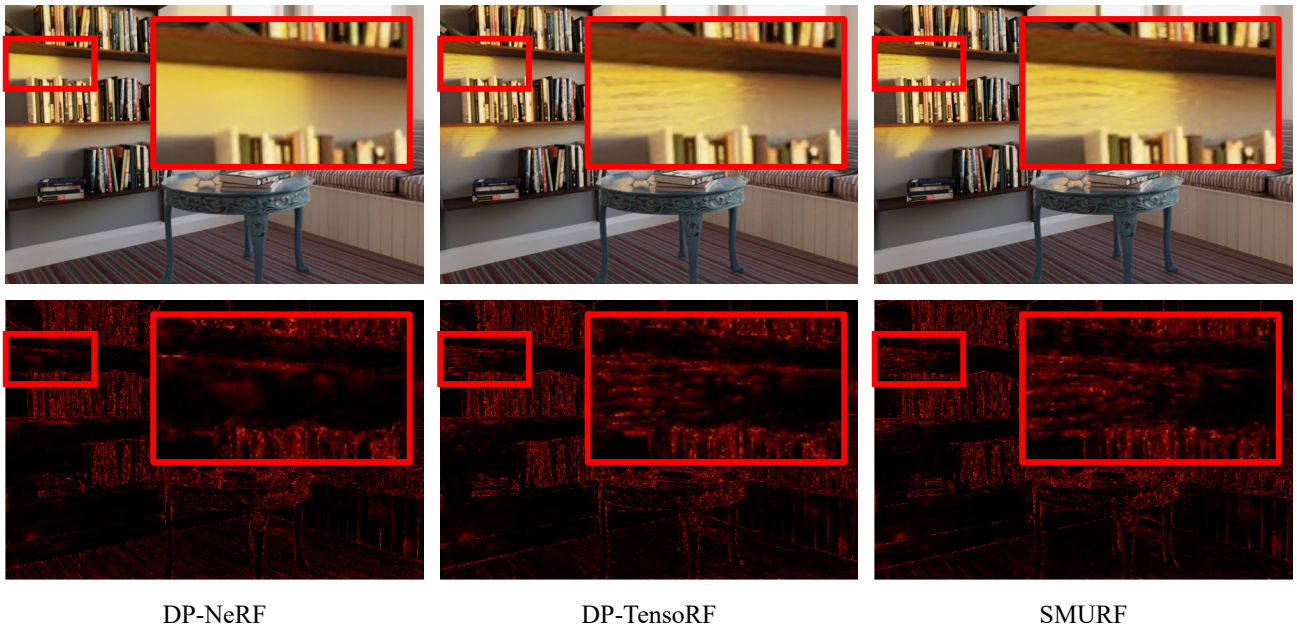

Figure 7. **Qualitative comparison of rendering results and error maps for the Cozyroom scene.** DP-TensoRF is a model that applies the kernel proposed by DP-NeRF [25] to TensoRF [6].

12. Analysis for Low-PSNR Scenes

In this section, we analyze individual scenes that showed slightly lower performance from the main paper. Notably, we visualize the error maps for the "COZYROOM" scene from the synthetic dataset for DP-NeRF [25], DP-TensoRF [6, 25], and SMURF in Fig. 7, where DP-TensoRF is a model that applies the blurring kernel proposed by DP-NeRF to TensoRF. The error map for DP-NeRF shows bright wall rendered cleanly without noise, whereas DP-TensoRF and SMURF exhibit noise on the wall. This indicates that the cause of noise is not the proposed CMBK but the inherent characteristics of the backbone model, TensoRF. As shown in Fig. 8, aside from the noise on the wall, the rendering results of SMURF show slightly sharper image quality in other areas except the wall. Moreover, the "COFFEE", "PARTERRE", and "PUPPET" scenes from real-world dataset, SMURF shows the best LPIPS score but somewhat lower PSNR. However, when comparing the rendering results in Figs. 8 and 9, it is observable that the results of SMURF are most similar to the reference images for these scenes.

13. Limitations and Future Work

By adopting TensoRF [6], a 3D tensor factorization-based method, as our backbone, we ensure high quantitative performance, superior perceptual quality, and faster training. However, with the advent of 3D Gaussian Splatting [20], which allows for GPU-based rasterization instead of optimizing per ray, backbones that facilitate more faster training and rendering become available. Although our backbone may be slower compared to 3D Gaussian Splatting, applying the main idea of CMBK, *continuous dynamics*, to a rasterization-based method is expected to result in faster training and rendering. Furthermore, by demonstrating the applicability of *continuous dynamics* to the 3D scene deblurring, we anticipate the possibility of designing models that cover not only camera motion blur but also object motion blur, which is caused by the movement of objects within the scene.

14. Per-Scene Quantitative Results

To demonstrate the superiority of SMURF, we present the individual performance results for all synthetic and real-world scenes in Tab. 6. For synthetic scenes, except for "COZYROOM," all scenes show quantitatively high performance, with this scene also displaying no significant difference when compared to DP-NeRF. Additionally, for the real-world scenes, despite a few scenes exhibiting somewhat lower PSNR, they demonstrate better perceptual quality through superior LPIPS scores. For a fair comparison, we also include the performance of DP-TensoRF, which applies the state-of-the-art DP-NeRF [25] to the explicit volumetric rendering method using the TensoRF [6] backbone. Although DP-TensoRF benefits from reduced training time due to the use of the TensoRF backbone, it shows negligible

performance differences when compared to DP-NeRF, and our SMURF outperforms both. Notably, DP-TensoRF generally exhibits lower PSNR and SSIM scores than DP-NeRF on real-world scenes. This indicates that the performance of our SMURF is not significantly dependent on the TensoRF backbone.

15. Additional Rendering Results

Additional rendering results are shown in Fig. 8 and Fig. 9, demonstrating that our SMURF offers the best perceptual quality when compared to the reference images. Please refer to the *supplementary videos* for comparisons of rendered videos.

Synthetic	Factory			CozyRoom			Pool			Tanabata			Trolley		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Naive NeRF	19.32	0.4563	0.5304	25.66	0.7941	0.2288	30.45	0.8354	0.1932	22.22	0.6807	0.3653	21.25	0.6370	0.3633
MPR+NeRF	21.70	0.6153	0.3094	27.88	0.8502	0.1153	30.64	0.8385	0.1641	22.71	0.7199	0.2509	22.64	0.7141	0.2344
PVD+NeRF	20.33	0.5386	0.3667	27.74	0.8296	0.1451	27.56	0.7626	0.2148	23.44	0.7293	0.2542	23.81	0.7351	0.2567
Deblur-NeRF	25.60	0.7750	0.2687	32.08	0.9261	0.0477	31.61	0.8682	0.1246	27.11	0.8640	0.1228	27.45	0.8632	0.1363
PDRF-10*	25.87	0.8316	0.1915	31.13	0.9225	0.0439	31.00	0.8583	0.1408	28.01	0.8931	0.1004	28.29	0.8921	0.0931
BAD-NeRF*	24.43	0.7274	0.2134	29.77	0.8864	0.0616	31.51	0.8620	0.0802	25.32	0.8081	0.1077	25.58	0.8049	0.1008
DP-NeRF	25.91	0.7787	0.2494	32.65	0.9317	0.0355	31.96	0.8768	0.0908	27.61	0.8748	0.1033	28.03	0.8752	0.1129
DP-TensoRF	25.54	0.7798	0.2250	32.13	0.9252	0.0397	32.14	0.8826	0.0877	28.22	0.9007	0.0917	28.59	0.9075	0.0963
SMURF	29.87	0.8958	0.1057	32.48	0.9285	0.0379	32.34	0.8884	0.0779	29.91	0.9300	0.0436	30.30	0.9307	0.0397

Real-World	Ball			Basket			Buick			Coffee			Decoration		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Naive NeRF	24.08	0.6237	0.3992	23.72	0.7086	0.3223	21.59	0.6325	0.3502	26.48	0.8064	0.2896	22.39	0.6609	0.3633
Deblur-NeRF	27.36	0.7656	0.2230	27.67	0.8449	0.1481	24.77	0.7700	0.1752	30.93	0.8981	0.1244	24.19	0.7707	0.1862
PDRF-10*	27.37	0.7642	0.2093	28.36	0.8736	0.1179	25.73	0.7916	0.1582	31.79	0.9002	0.1133	23.55	0.7508	0.2145
BAD-NeRF*	21.33	0.5096	0.4692	26.44	0.8080	0.1325	21.63	0.6429	0.2593	28.98	0.8369	0.1956	22.13	0.6316	0.2894
DP-NeRF	27.20	0.7652	0.2088	27.74	0.8455	0.1294	25.70	0.7922	0.1405	31.19	0.9049	0.1002	24.31	0.7811	0.1639
DP-TensoRF	25.85	0.7164	0.2106	27.04	0.8434	0.1099	25.02	0.7981	0.1603	29.67	0.8424	0.1323	22.78	0.7362	0.1801
SMURF	27.50	0.7760	0.1298	28.95	0.8842	0.0619	27.10	0.8409	0.0839	31.33	0.8879	0.0874	24.90	0.8114	0.1190

Real-World	Girl			Heron			Parterre			Puppet			Stair		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Naive NeRF	20.07	0.7075	0.3196	20.50	0.5217	0.4129	23.14	0.6201	0.4046	22.09	0.6093	0.3389	22.87	0.4561	0.4868
Deblur-NeRF	22.27	0.7976	0.1687	22.63	0.6874	0.2099	25.82	0.7597	0.2161	25.24	0.7510	0.1577	25.39	0.6296	0.2102
PDRF-10*	24.12	0.8328	0.1679	22.53	0.6880	0.2358	25.36	0.7601	0.2263	25.02	0.7496	0.1532	25.20	0.6235	0.2288
BAD-NeRF*	18.10	0.5652	0.3933	22.18	0.6479	0.2226	23.44	0.6243	0.3151	22.48	0.6249	0.2762	21.52	0.4237	0.3341
DP-NeRF	23.33	0.8139	0.1498	22.88	0.6930	0.1914	25.86	0.7665	0.1900	25.25	0.7536	0.1505	25.59	0.6349	0.1772
DP-TensoRF	21.32	0.7775	0.1614	22.62	0.6861	0.2039	25.37	0.7708	0.1761	24.29	0.7376	0.1495	23.52	0.6022	0.1752
SMURF	25.66	0.8592	0.0829	23.59	0.7317	0.1381	25.47	0.7825	0.1207	25.19	0.7702	0.1077	25.48	0.6421	0.0822

Table 6. **Comparison of performance for individual scenes.** SMURF exhibits higher performance across all synthetic scenes, with the exception of "CozyRoom," where it shows slightly lower performance relative to others. For real-world scenes, while SMURF shows slightly lower PSNR and SSIM in some scenes, it shows significantly better LPIPS across all scenes compared to previous methods.

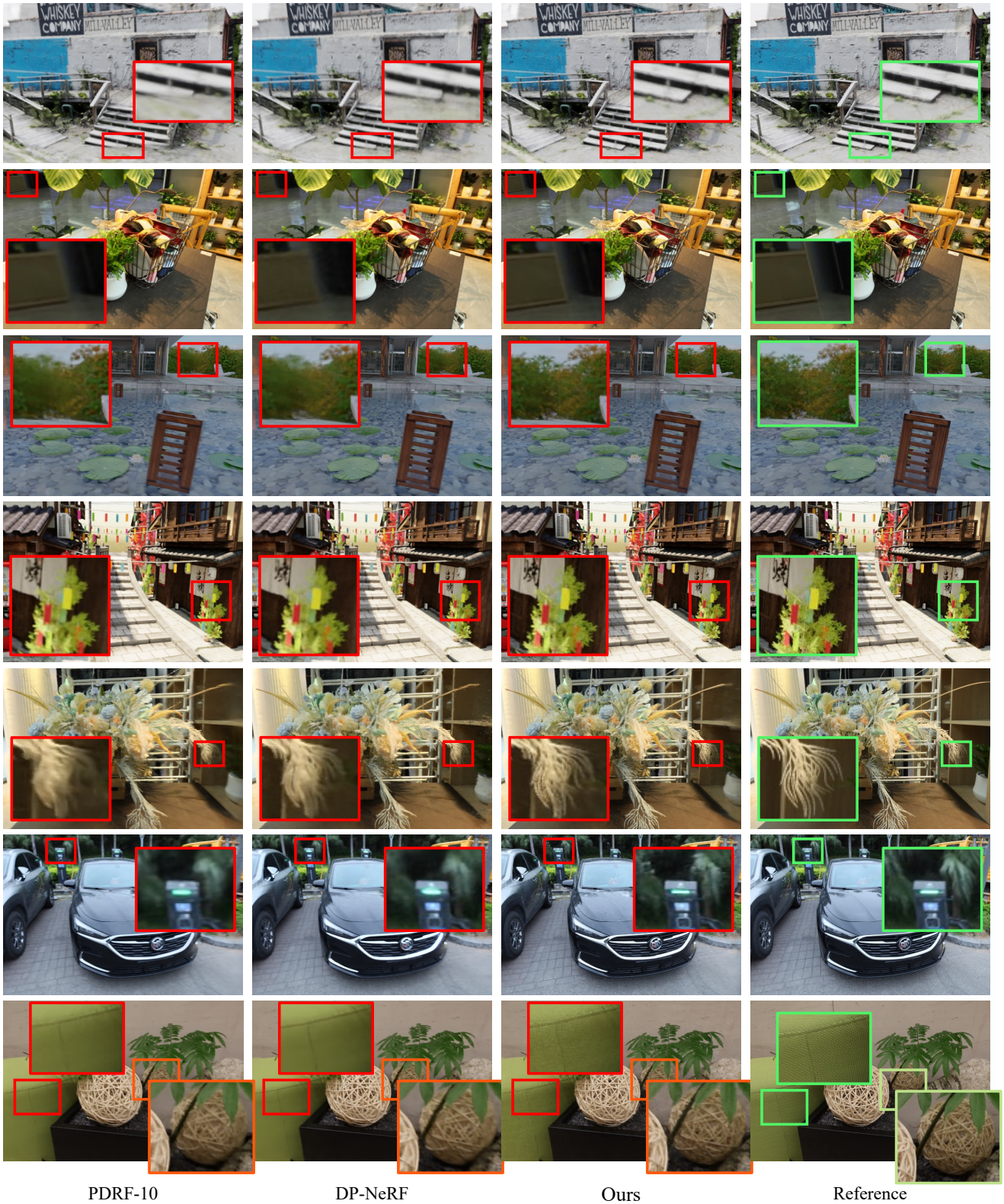

Figure 8. Qualitative comparison for individual scenes.

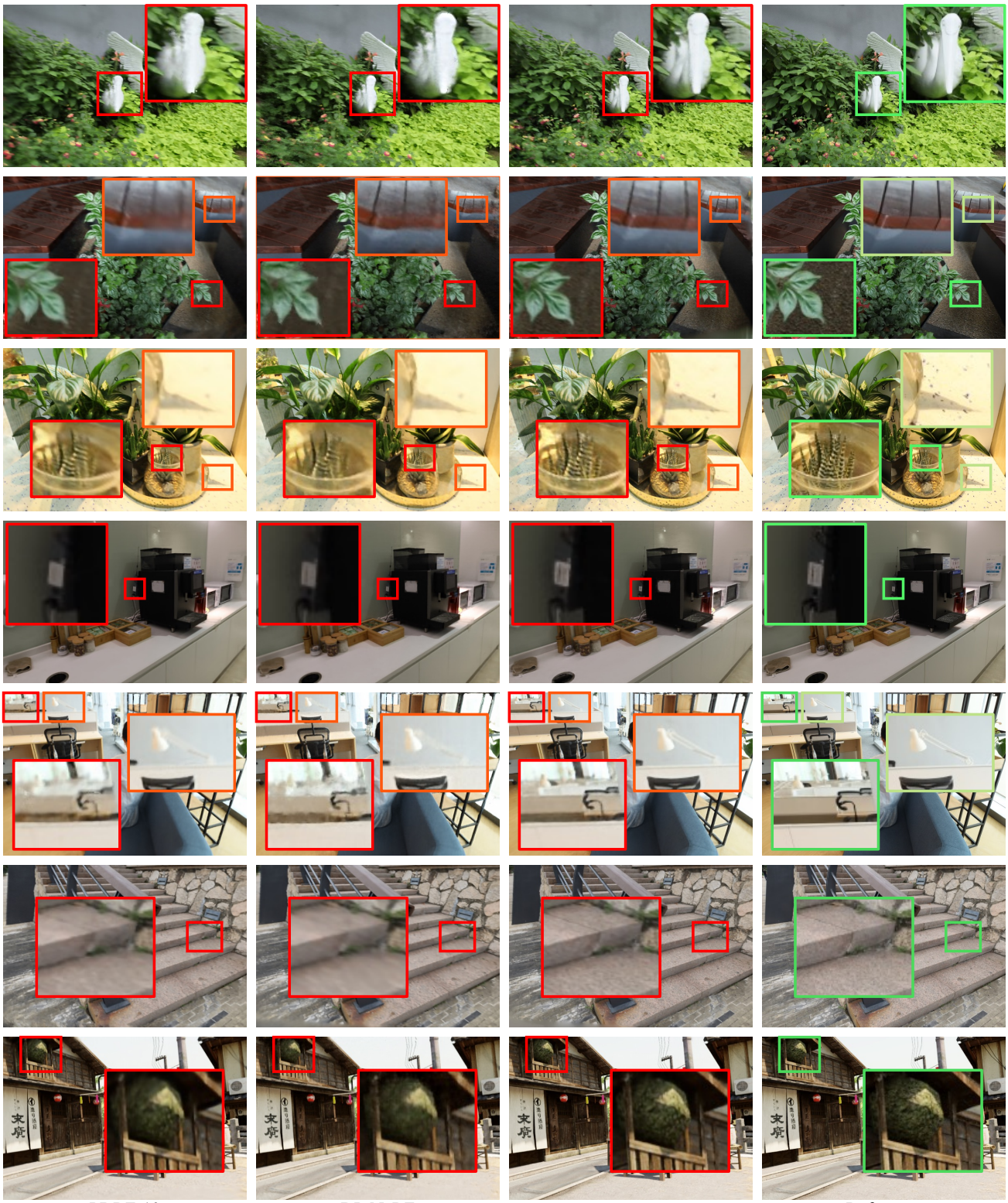

Figure 9. Qualitative comparison for individual scenes.