Dynamic Gaussian Splatting from Defocused and Motion-blurred Monocular Videos

Xuankai Zhang[1], Junjin Xiao[1], Qing Zhang[1]*
[1]Sun Yat-sen University
zhangxk53@mail2.sysu.edu.cn,xiaojj37@mail2.sysu.edu.cn,
zhangqing.whu.cs@gmail.com
https://dydeblur.github.io/

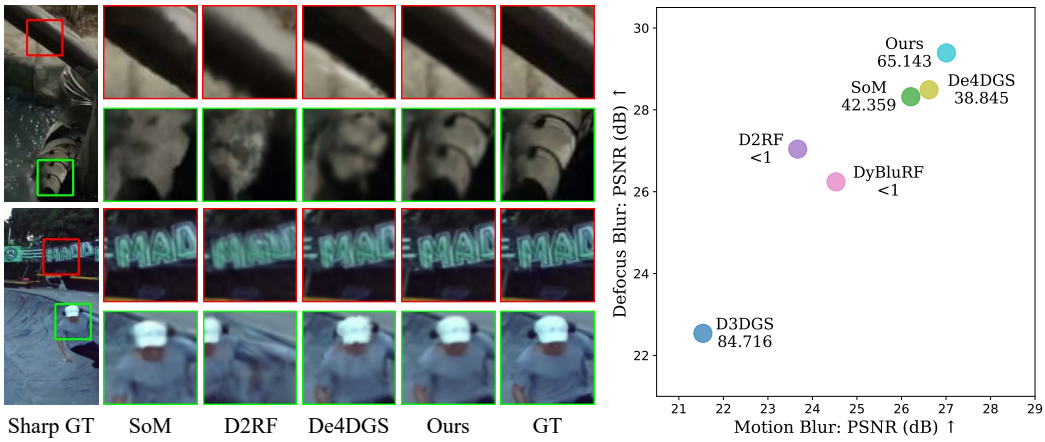

Figure 1: **Performance of our method.** Our method allows to synthesize high-quality sharp novel views for videos with defocus blur (top) and motion blur (bottom). As shown on the right, our method not only obtains significantly better results than existing methods, e.g., D3DGS [15], SoM [52], D2RF [32], DyBluRF [47], and De4DGS [58], but also achieves a performance of 65.143 FPS at a resolution of 512×288 on an NVIDIA RTX 3090 GPU.

Abstract

This paper presents a unified framework that allows high-quality dynamic Gaussian Splatting from both defocused and motion-blurred monocular videos. Due to the significant difference between the formation processes of defocus blur and motion blur, existing methods are tailored for either one of them, lacking the ability to simultaneously deal with both of them. Although the two can be jointly modeled as blur kernel-based convolution, the inherent difficulty in estimating accurate blur kernels greatly limits the progress in this direction. In this work, we go a step further towards this direction. Particularly, we propose to estimate per-pixel reliable blur kernels using a blur prediction network that exploits blur-related scene and camera information and is subject to a blur-aware sparsity constraint. Besides, we introduce a dynamic Gaussian densification strategy to mitigate the lack of Gaussians for incomplete regions, and boost the performance of novel view synthesis by incorporating unseen view information to constrain scene optimization. Extensive experiments show that our method outperforms the state-of-the-art methods in generating photorealistic novel view synthesis from defocused and motion-blurred monocular videos. Our code is available at https://github.com/hhhddddddd/dydeblur.

*Corresponding author.

1 Introduction

Novel view synthesis of dynamic scenes from monocular videos is a very important problem, with applications in various scenarios such as augmented reality, virtual reality, and 3D content creation. Recent progress in this field mostly aims to learn renderable 3D Gaussian representations from monocular videos. Although some of them have demonstrated impressive results of novel view synthesis [24, 45, 52, 63, 53], their performance typically deteriorates significantly on blurry videos, making methods applicable to blurry monocular videos particularly necessary.

Some recent variants [58, 47, 30, 4, 32] of 3D Gaussian Splatting (3DGS) [15] and Neural Radiance Field (NeRF) [35] have attempted to reconstruct dynamic scenes from blurry monocular videos, where [58, 47, 30, 4] tackles motioned-blurred videos by computing a weighted sum of multiple rendered images within the exposure period, while [32] focuses on dealing with defocus blur using layered Depth-of-Field (DoF) volume rendering. Although these methods demonstrate promising results, due to the significant difference between the formation process of motion blur and defocus blur, their effectiveness is limited to either motion-blurred or defocused videos. Currently, there does not exist a method that can effectively handle both types of blurry videos while enabling high-quality novel view synthesis.

In this work, we present a framework that allows high-quality dynamic Gaussian Splatting from both defocused and motion-blurred monocular videos. To this end, we employ blur kernel based convolution to jointly model the two blur types. To obtain reliable blur kernels from dynamic scenes, we develop a blur prediction network (BP-Net) that is subject to a blur-aware sparsity constraint to simultaneously predict the blur kernel and pixel-level intensity. Moreover, we introduce a dynamic Gaussian densification strategy to mitigate the lack of Gaussians for incomplete regions, and propose to boost the performance of novel view synthesis by incorporating unseen view information to constrain scene optimization. In summary, our main contributions are as follows:

- We introduce a unified framework for dynamic Gaussian Splatting from both defocused and motion-blurred monocular videos, which to our knowledge, makes the first attempt in this field.
- We develop a blur prediction network equipped with a blur-aware sparsity constraint, and introduce a dynamic Gaussian densification strategy as well as a unseen view combined scene optimization scheme.
- We show that our method outperforms previous methods on both defocused and motion-blurred monocular videos.

2 Related Works

4D Reconstruction. Recent works have extended 3DGS to 4D domain to model dynamic scenes [11, 64, 8, 29, 63, 26]. One line of works define motion as a time-conditioned deformation network that warps Gaussians from canonical space to observation space [57, 63, 7, 12, 16, 13, 27]. Among these, DeformableGS [63] employs MLPs to predict Gaussian deformations, while 4DGaussians [57] replaces MLPs with a multi-resolution HexPlane [5]. Another line of works model motion as trajectories of 3D Gaussians [31, 26, 40, 14, 2, 52, 48, 28]. E-D3DGS [2] introduces per-Gaussian and temporal embeddings to encode time-aware information, whereas Shape-of-Motion [52] models motion as a linear combination of $\mathbb{SE}(3)$ motion bases. However, existing dynamic 3DGS methods rely on sharp input images and often struggle to synthesize photorealistic novel views when motion or defocus blur is present in the inputs.

Image deblurring. Early works on image and video deblurring [19, 20, 49, 36, 65] primarily rely on CNNs [46] and RNNs [69]. Later approaches enhance the deblurring process by incorporating additional cues such as depth maps [39, 25, 50, 1], light fields [43, 44], and 3D geometry [37, 38, 55, 59, 61]. Leveraging multi-view and 3D information, NeRF- and 3DGS-based deblurring methods can generate sharp images with improved scene consistency [41, 21, 34, 22, 23, 9]. Among these, some approaches [34, 22, 33, 3, 6] model blur by predicting deformable sparse kernels with per-pixel bases. Others simulate the physical formation of blur by integrating multiple sharp images over the exposure time [68, 51, 38] or use a camera model that learns aperture and focal length parameters [55, 59]. Additionally, some works [47, 32, 58, 30, 4] incorporate 3D geometry into monocular video deblurring. However, existing methods are typically designed for either defocus blur or motion blur.

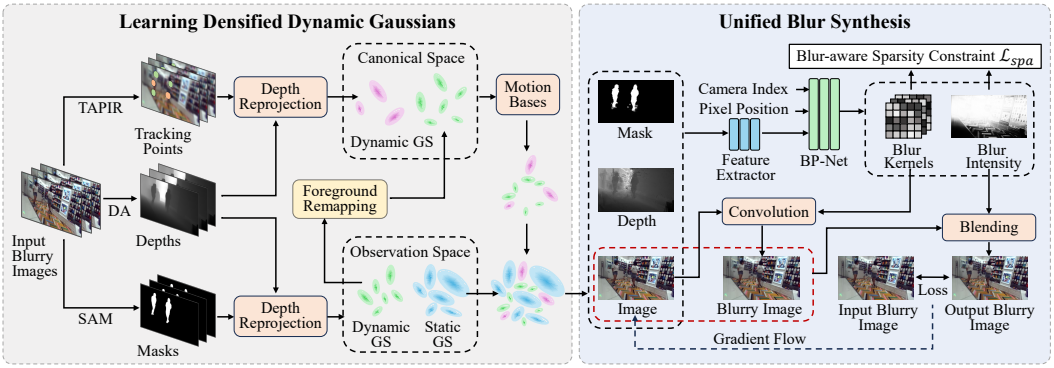

Figure 2: **Overview of our method.** We initialize static Gaussians via depth reprojection and dynamic Gaussians from tracking points, modeling their transformations with learnable motion bases. After stable training, we densify dynamic Gaussians using foreground remapping. For blur modeling, a network predicts per-pixel blur kernels and intensity, enabling blur synthesis through convolution and blending operation. The reconstruction loss between synthesized and input blurry images optimizes the Gaussians for sharper results.

In contrast, our approach effectively removes both motion blur and defocus blur from monocular videos, producing high-quality novel-view images of dynamic scenes.

3 Method

Our method aims to achieve dynamic Gaussian Splatting that enables high-fidelity yet sharp novel view synthesis from any given monocular video with defocus blur or motion blur. Figure 2 presents the overview of our method.

3.1 Learning Densified Dynamic Gaussians

Representing scene as dynamic 3D Gaussians. We adopt Shape-of-Motion [52] to initialize 3D Gaussians and model scene motion. It separately models dynamic and static Gaussians, representing motion with a compact set of $\mathbb{SE}(3)$ motion bases. Specifically, dynamic Gaussians are initialized by reprojecting the depth of 2D tracking points in the canonical frame t_0, obtained via Depth-Anything [62] and TAPIR [10]. Static Gaussians are initialized by reprojecting background depth across frames and share the same parameter composition as the original 3D Gaussians. Dynamic Gaussians include an additional motion coefficient $\mathbf{w} \in \mathbb{R}^{N_b}$, which combines $\mathbb{SE}(3)$ motion bases to model their motion. The affine transformation from the canonical frame t_0 to the observation frame t is computed as follows:

$$\mathrm{T}_{t_0 \to t} = \sum_{b=0}^{N_b} \mathbf{w}^{(b)} \mathrm{T}^{(b)}_{t_0 \to t}, \tag{1}$$

Where N_b denotes the number of $\mathbb{SE}(3)$ motion bases, and $\mathrm{T}^{(b)}_{t_0 \to t}$ represents the affine transformation of motion base b. The affine transformation $\mathrm{T}_{t_0 \to t}$ consists of two components: rotation matrix $\mathrm{R}_{t_0 \to t}$ and translation vector $\mathrm{t}_{t_0 \to t}$. The transformation of the pose parameters (μ, R) for the dynamic Gaussian from the canonical frame t_0 to the observation frame t is defined as follows:

$$\mu_t = \mathrm{R}_{t_0 \to t} \mu_{t_0} + \mathrm{t}_{t_0 \to t}, \quad \mathrm{R}_t = \mathrm{R}_{t_0 \to t} \mathrm{R}_{t_0}. \tag{2}$$

The transformed dynamic Gaussians in observation space, along with the static Gaussians, are then processed through a differentiable rasterization pipeline to generate the final rendered image $\tilde{I}$, depth map $\tilde{D}$, and mask $\tilde{M}$.

Dynamic Gaussian densification.

Shape-of-Motion [52] initializes dynamic Gaussians using point clouds by reprojecting 2D tracking points from the canonical frame into 3D space. Although the canonical frame is selected as the one containing the most visible 2D tracking points across all frames, tracking points that are invisible in

the canonical frame can still lead to partial dynamic Gaussians with inaccurate depth. This occurs because the depth values at the corresponding locations of invisible 2D tracking points do not reflect the true depth after reprojection. To address this, we initialize dynamic Gaussians using only visible 2D tracking points in the canonical frame. While this improves initialization accuracy, it may lead to incomplete dynamic regions. To compensate, we supplement dynamic Gaussians by reprojecting dynamic regions with depth maps from all observation frames. Then, we transform the dynamic Gaussians in the observation frames to the canonical frame using foreground remapping. Since this operation requires relatively stable and accurate motion bases, we perform dynamic Gaussian densification only once after the first N_d training iterations.

To this end, we first identify dynamic pixels in the training images using motion masks M, obtained via the off-the-shelf method SAM [18]. A random subset of pixels is selected in each observation frame. For a dynamic pixel g in frame t, the mean μ_t^g of the corresponding Gaussian G in observation space is defined as:

$$\mu_t^g = P_t^{-1}(\pi_t^{-1}(g, D(g))), \tag{3}$$

where $D(g)$ is the depth of pixel g, π_t is the projection function for frame t, and P_t represents the camera extrinsics for frame t. We use foreground remapping to compute the corresponding position $\mu_{t_0}^g$ of Gaussian G in the canonical frame. Specifically, we first determine all affine transformations for all existing dynamic Gaussians from t_0 to t. We then select the affine transformation $\mathrm{T}_{t_0\to t}^{G'} = [\mathrm{R}_{t_0\to t}^{G'}, \mathrm{t}_{t_0\to t}^{G'}]$ corresponding to the dynamic Gaussian G' that is closest to μ_t^g after transformation, which gives:

$$\mu_{t_0}^g = (\mathrm{R}_{t_0\to t}^{G'})^{-1}(\mu_t^g - \mathrm{t}_{t_0\to t}^{G'}). \tag{4}$$

3.2 Unified Blur Synthesis

During training, we explicitly model the blurring process and jointly optimize a sharp 3DGS representation along with the blur parameters, ensuring that the synthesized blurry images match the input blur image. Although the formation processes of defocus blur and motion blur are fundamentally different, both types of blur can be approximated as a weighted combination of a pixel and its neighboring pixels. Thus, the generation processes of motion blur and defocus blur can be unified under a simple yet powerful mathematical model:

$$\tilde{B}(x) = \sum_{x_i \in \mathcal{N}(x)} \tilde{I}(x_i) k_x(x_i) \text{ s.t } \sum_{x_i \in \mathcal{N}(x)} k_x(x_i) = 1, \tag{5}$$

where $\tilde{I}(x_i)$ is the clear pixel value at pixel coordinates x_i around the neighborhood of x, and k_x denotes the blur kernel for pixel x.

Using the per-pixel blur kernel k_x from Eq. (5), we can generate a blurry image $\tilde{B}$ from the sharp image $\tilde{I}$. The main challenge is estimating a reasonable blur kernel k_x for each pixel x. An intuitive solution would be to use a CNN to estimate k_x for each pixel in the sharp image $\tilde{I}$ obtained from rasterization. However, jointly optimizing the 3DGS and the CNN blur kernel prediction network under the constraint of blurry images B can lead to non-rigid distortions in the reconstructed 3DGS. This is in line with expectations because it is possible that the reconstructed 3DGS and the CNN blur kernel prediction network deform together without affecting the reconstructed blurry result. To address this, we design the Blur Prediction Network (BP-Net) F_Θ, which simultaneously predicts the per-pixel blur kernel k_x and the corresponding blur intensity m_x. Using the blur intensity m_x, we compute the output blurry image pixel value $\hat{B}(x)$ by blending the sharp image pixel value $\tilde{I}(x)$ and the blurry image pixel value $\tilde{B}(x)$:

$$\hat{B}(x) = (1 - m_x) \cdot \tilde{I}(x) + m_x \cdot \tilde{B}(x). \tag{6}$$

Computing the reconstruction loss between the output blurry image $\hat{B}$ and the input blurry image B directly constrains the rendered image $\tilde{I}$ using the sharp regions in B, progressively guiding it toward a sharper 3DGS-based scene representation, while ensuring that most of the blur is explicitly modeled by the CNN blur kernel prediction network.

Blur prediction network. The Blur Prediction Network (BP-Net) is a four-layer CNN model that takes camera and scene information as inputs, both of which significantly contribute to blur

Figure 3: **Visual comparison of novel view synthesis on the D2RF defocus blur dataset [32].**

in monocular videos. Camera motion and suboptimal camera settings can introduce blur, while dynamic objects and scene depth strongly correlate with blur magnitude. The camera information is represented by a learnable embedding vector $e(i)$ obtained from the camera view i. The scene information is encoded into f_{scene} through a three-layer CNN Scene Feature Extractor, which uses the rendered image $\tilde{I}$, depth $\tilde{D}$, and motion mask $\tilde{M}$ as inputs. Given the variation in blur across different pixels in real blurry images, we also include the pixel coordinate positional encoding $p(x)$ as input to the BP-Net:

$$k_x, m_x = F_\Theta(e(i), f_{scene}(x), p(x)), \tag{7}$$

where k_x and m_x denote the blur kernel and blur intensity for pixel x, respectively. To improve the scene information representation, we add skip connections from f_{scene} after the first two layers of the BP-Net.

Blur-aware sparsity constraint. The blur kernel weights for mildly blurred pixels should be more concentrated around the center, while those for severely blurred pixels should be more uniformly distributed. To prevent unrealistic blur kernels from excessively blurring mildly blurred regions, we use the blur intensity m_x to constrain the weight distribution of the corresponding blur kernel k_x. Specifically, we use the blur kernel center weight $k_x(c)$ to quantify the sparsity of the blur kernel k_x. A smaller $k_x(c)$ suggests a more dispersed kernel, implying that the corresponding pixel is more severely blurred. Based on this, we design a blur-aware center weight c_x for the blur kernel k_x as follows:

$$c_x = \text{sigmoid}(scale \cdot (1 - \texttt{sg}(m_x))), \tag{8}$$

where $scale$ is a scale factor (set to 5), and $\texttt{sg}$ denotes the stop-gradient operation. Clearly, the pixel-wise blur intensity m_x is negatively correlated with the corresponding blur-aware center weight c_x; as m_x increases, c_x decreases. We then compute the $\mathcal{L}_1$ loss between the blur-aware center weight c_x and the blur kernel center weight $k_x(c)$:

$$\mathcal{L}_{spa} = \mathcal{L}_1(c_x, k_x(c)). \tag{9}$$

During training, we adopt the blur-aware sparsity constraint $\mathcal{L}_{spa}$ for training only when the blur intensity m_x has been trained for a number of N_{spa} iterations to be stable.

3.3 Training Strategy

Monocular reconstruction of complex dynamic scenes is ill-posed and prone to local minima due to the limited information provided by monocular videos, where each timestamp captures only a single training view. This lack of sufficient views makes it difficult to accurately learn the 3D structure of dynamic scenes, often leading to overfitting to the training views. To mitigate overfitting and extract more cues from monocular videos, we obtain potential appearance information for unseen views surrounding the training views. During training, we use this appearance information to supervise the optimization process every N_u iterations. Since the appearance information from unseen views is

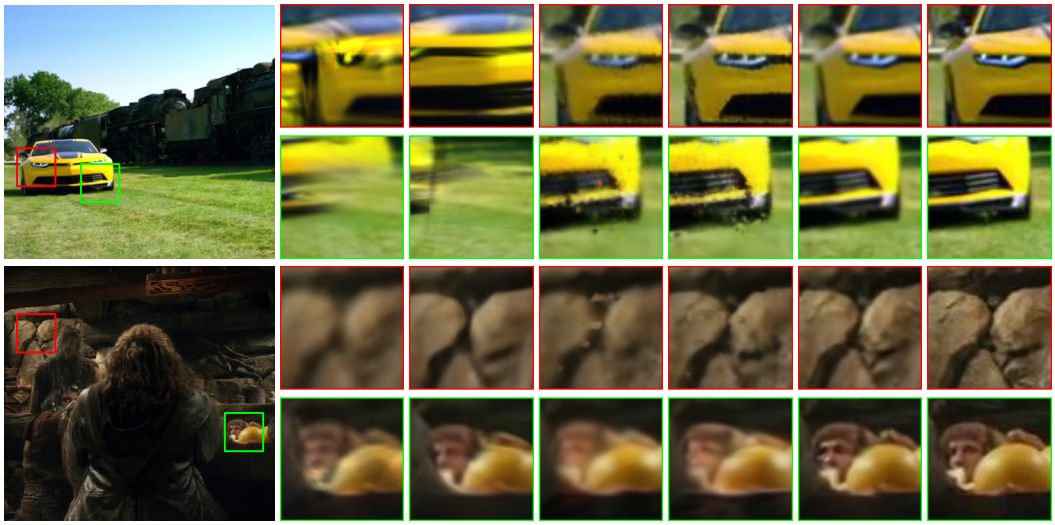

Figure 4: **Visual comparison of novel view synthesis on the D2RF defocus blur dataset [32].** Here, we also compare with methods fed with deblurred images produced by a state-of-the-art video deblurring method [66] to manifest the effectiveness of our method.

less accurate than that from training views, we apply slightly different loss functions for them, as detailed in Section 3.4.

We leverage the geometry and appearance information from the training views to obtain the appearance of unseen views. For instance, consider a pixel p_s in the training view V_s. The corresponding pixel p_t in the unseen view V_t can be expressed as:

$$p_t = K P_t^{-1} P_s D_s(p_s) K^{-1} p_s, \tag{10}$$

Where K is the camera intrinsic matrix, D_s is the depth map for the training view V_s, P_s and P_t are the camera extrinsics for the training and unseen views, respectively. We then use the color $B_s(p_s)$ of pixel p_s to derive the color $B_t(p_t)$ of the corresponding pixel p_t in the unseen view via reversed bilinear sampling [60, 54]. Similarly, the motion mask $M_t(p_t)$ in the unseen view can be derived from $M_s(p_s)$.

To ensure the accuracy of appearance information in unseen views, we restrict the selection to unseen views close to the training views. In most monocular videos, the camera poses of training views typically form a sequence of consecutive poses directed towards the scene. Therefore, we select unseen views on both sides of the training view sequence and insert unseen views between adjacent training views. We implement this by generating two types of unseen views: (i) parallel-unseen views: generated by interpolating between adjacent training views along the camera trajectory, (ii) perpendicular-unseen views: generated by first computing a local perpendicular direction to the camera trajectory and then perturbing the training view's camera center along this perpendicular direction by a distance of [0.5, 1] (normalized units). Importantly, the timestamp of an unseen view matches the timestamp of the training view from which it is generated.

3.4 Loss Function

Reconstruction loss. We supervise the training process with a reconstruction loss to align per-frame pixel-wise color input $\mathbf{B}$. We compute the blurry image $\hat{\mathbf{B}}$ according to Eq. (5) and Eq. (6). The blurry image $\hat{\mathbf{B}}$ is supervised by the following reconstruction loss:

$$\mathcal{L}_{rec} = (1-\beta)\mathcal{L}_1(\hat{\mathbf{B}}, \mathbf{B}) + \beta \mathcal{L}_{ssim}(\hat{\mathbf{B}}, \mathbf{B}), \tag{11}$$

where $\mathcal{L}_1$ and $\mathcal{L}_{ssim}$ are the $_1$ loss and SSIM [56] loss, respectively, and β is set to 0.2. We also constrain the scene geometry using the depth $\tilde{\mathbf{D}}$ and mask $\tilde{\mathbf{M}}$:

$$\mathcal{L}_{geo} = \lambda_{depth}\mathcal{L}_1(\tilde{\mathbf{D}}, \mathbf{D}) + \lambda_{mask}\mathcal{L}_1(\tilde{\mathbf{M}}, \mathbf{M}), \tag{12}$$

Figure 5: **Visual comparison of novel view synthesis on the DyBluRF motion blur dataset [47].**

where λ_{depth} and λ_{mask} are set to 0.075. During iterations with unseen views, we only use the mask, excluding depth, to prevent the inaccurate geometry of unseen views from distorting the scene geometry.

Table 1: **Quantitative comparison of novel view synthesis on the D2RF defocus blur dataset [32] and the DyBluRF motion blur dataset [47].**

Method	Defocus Blur			Motion Blur			Param.	Training Time
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓		
D3DGS [63]	22.54	0.715	0.215	21.54	0.675	0.287	42.6M	10 mins
SoM [52]	28.32	0.784	0.164	26.21	0.823	0.109	164.2M	10 mins
D2RF [32]	27.04	0.808	0.128	23.67	0.745	0.120	2.7M	48 hrs
DyBluRF [47]	26.24	0.788	0.159	24.53	0.864	0.079	1.3M	48 hrs
De4DGS [58]	28.49	0.791	0.154	26.62	0.871	0.059	754.6M	20 hrs
Ours	**29.39**	**0.859**	**0.078**	**27.01**	**0.876**	**0.056**	192.2M	1 hr

Smoothing loss. To improve scene motion representation, we introduce a smoothing constraint $\mathcal{L}_{smo}$ for the dynamic Gaussians' deformation. Similar to Shape-of-Motion [52], $\mathcal{L}_{smo}$ includes two components: a smoothness constraint on the $\mathbb{SE}(3)$ motion bases across adjacent frames and a smoothness constraint on the mean of dynamic Gaussians.

The overall training objective for the network is:

$$\mathcal{L} = \mathcal{L}_{rec} + \mathcal{L}_{geo} + \mathcal{L}_{smo} + \mathcal{L}_{spa}. \tag{13}$$

4 Experiments

Evaluation datasets. We evaluate our method on two datasets, including the one from D2RF [32] of defocus blur and the other one from DyBluRF [47] of motion blur. The first dataset from D2RF consists of 8 dynamic scenes, where each scene contains sharp stereo image sequences and their corresponding blurry images. The other one from DyBluRF contains 6 motion-blurred scenes composed of blurry stereo images and the corresponding sharp images. We train and evaluate our method on all sequences from the two datasets, using their left-view blurred sequences for training and the corresponding right-view sharp sequences for evaluation. Note that, similar to previous methods, the downsampled images in the two datasets are utilized for training and evaluation.

Metrics. Following previous work [52, 58, 32, 47], we quantitatively evaluate our performance on novel view synthesis using PSNR, SSIM [56], and LPIPS [67].

Implementation Details. We set the number of motion bases N_b to 20, and the blur kernel size K to 9. We employ the Adam optimizer [17] to jointly optimize the Gaussians and $\mathbb{SE}(3)$ motion bases and the BP-Net. The learning rates are set to 1.6×10^{-4} for motion bases, and 5×10^{-4} for

Table 2: **Quantitative comparison of novel view synthesis on the D2RF defocus blur dataset [32] and the DyBluRF motion blur dataset [47].** Note, BSSTNet [66] is a state-of-the-art video deblurring method, and "[66] + D3DGS [63]" means feeding the dynamic scene reconstruction [63] with deblurred images produced by [66].

Method	Defocus Blur			Motion Blur		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D3DGS [63]	22.54	0.715	0.215	21.54	0.675	0.287
[66] + D3DGS [63]	24.42	0.723	0.179	21.72	0.653	0.279
SoM [52]	28.32	0.784	0.164	26.21	0.823	0.109
[66] + SoM [52]	28.56	0.786	0.164	26.33	0.825	0.105
Ours	**29.39**	**0.859**	**0.078**	**27.01**	**0.876**	**0.056**

Table 3: **Quantitative ablation study on the D2RF defocus blur dataset [32] and the DyBluRF motion blur dataset [47].** Note, "w/o Unseen." refers to no unseen view information utilized.

Method	Defocus Blur			Motion Blur		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
w/o $\mathcal{L}_{spa}$	29.03	0.842	0.086	26.63	0.854	0.072
w/o Shortcut	29.12	0.845	0.086	26.74	0.852	0.078
w/o DGD	29.19	0.843	0.085	26.53	0.847	0.109
w/o Unseen.	29.09	0.836	0.090	26.66	0.853	0.075
Full method	**29.39**	**0.859**	**0.078**	**27.01**	**0.876**	**0.056**

BP-Net. The learning rate for the Gaussians is consistent with that of the original 3DGS [15]. We train each scene for 40,000 iterations and introduce unseen view information to constrain the scene starting from iteration 3,000. We set the iteration interval N_u for using unseen view information to 5. Dynamic Gaussians densification is performed at N_d = 2,500 iterations. We introduce the unified blur synthesis model at iteration 3,500 and incorporate the blur-aware sparsity constraint at N_{spa} = 5,500 iterations. Training on a sequence of 512×288 resolution takes approximately 1 hour on an NVIDIA RTX 3090 GPU, with a rendering speed of 65.143 fps for the same resolution.

4.1 Comparison with State-of-the-Art Methods

Compared methods. We compare our method with various state-of-the-art methods including, DeformableGS [63], Shape-of-Motion [52], D2RF [32], DyBluRF [47], and Deblur4DGS [58], where the method of D2RF [32] is designed for defocus blur, while the methods of DyBluRF [47] and Deblur4DGS [58] are tailored for motion blur. For fair comparison, we produce their results using publicly-available implementation or trained models provided by the authors with the recommended parameter setting.

Comparison on defocus blur. Tables 1 and 2 report the quantitative results on the defocus blur dataset, where we can see that our method outperforms other methods, even existing methods that are fed with deblurred images produced by a state-of-the-art video deblurring method. The reason is that video deblurring methods cannot effectively ensure 3D scene consistency in the deblurred images. Figures 3 and 4 further present a visual comparison of novel-view synthesis. As shown, our method exhibits a clear advantage in producing high-quality novel views in both dynamic and static regions, while the compared methods struggle to maintain structural details and provide sharp results. Please see the supplementary material for additional comparison results, including both image and video results of defocus blur.

Comparison on motion blur. As shown in Tables 1 and 2, our method quantitatively outperforms the compared methods on all three metrics, manifesting its effectiveness in dealing with motion blurred monocular videos. We also show the visual comparison in Figure 5. As can be seen, our method produces results with more incomplete structure details and sharper appearance. Please see the supplementary material for additional comparison results, including both image and video results of motion blur.

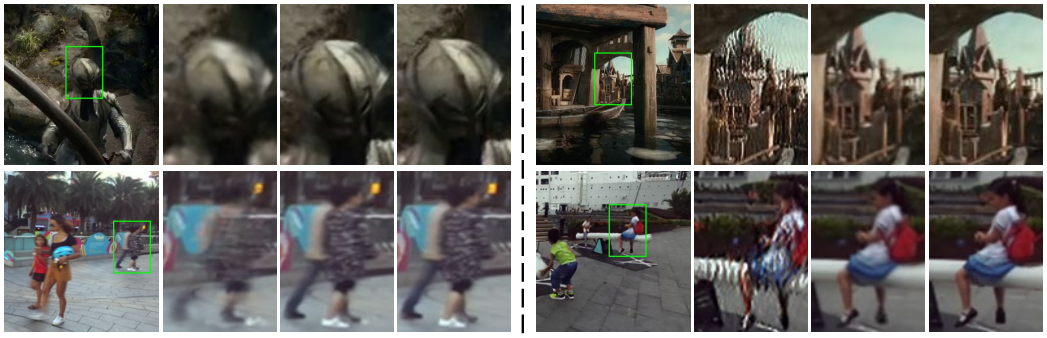

Figure 6: **Left:** Effect of dynamic Gaussian densification (DGD). **Right:** Effect of leveraging unseen view information. Note, "w/o Unseen." refers to no unseen view information utilized.

Figure 7: **Left:** Effect of blur-aware sparsity Constraint $\mathcal{L}_{spa}$ and the shortcut in BP-Net. **Right:** Effect of varying blur kernel size K.

4.2 More Analysis

Ablation study. We conduct an ablation study to evaluate the contribution of each component in our model. Specifically, we evaluate the effect of (i) removing blur-aware sparsity constraint (w/o $\mathcal{L}_{spa}$), (ii) removing the shortcut in BP-Net (w/o Shortcut), (iii) initializing the dynamic Gaussians using only the visible 2D tracking points in the canonical frame, without utilizing dynamic Gaussian densification (w/o DGD), (iv) removing unseen view information during training (w/o Unseen.). We report the quantitative results in Table 3, where we can see that each of our design has a clear contribution. In addition, we in Figures 6 and 7 also qualitatively validate the necessity of each component in our model. Besides, we also assess the influence of different blur kernel size in Figure 7. As shown, a larger blur kernel (K) helps produce better results, but this trend becomes less obvious when $K > 9$.

Limitations. Since our method relies on 2D image priors, errors arising from 2D prediction such as depth estimation and segmentation may degrade the overall performance of our method. Moreover, for dynamic scenes with large non-rigid motion blur, our method, as well as other state-of-the-art methods, would fail to produce high-quality novel-view results free of visual artifacts, as demonstrated in Figure 8. Finally, similar to the vanilla 3DGS, our method has to be optimized for each scene separately.

Figure 8: **Failure case.** Our method may fail to handle a dynamic scene with large non-rigid motion blur. Note, the left column demonstrates results for defocus blur, while the right column presents motion blur outcomes.

5 Conclusion

We have presented a unified framework for generating high-quality novel views from defocused and motion-blurred monocular videos. In contrast to previous methods, which are either tailored for defocus blur or motion blur, we propose to model both blur types using blur kernel-based convolution. To this end, we develop a blur prediction network exploiting blur-related scene and camera information to estimate reliable blur kernels and pixel-wise intensity. Besides, we introduce a dynamic Gaussian densification strategy to mitigate the lack of Gaussians for incomplete regions and boost the performance of novel view synthesis by incorporating unseen view information to constrain scene optimization. Extensive experiments demonstrate that our method outperforms the state-of-the-art methods in generating photorealistic novel view synthesis from defocused and motion-blurred monocular videos.

Acknowledgement. This work was supported by the National Natural Science Foundation of China (62471499), the Guangdong Basic and Applied Basic Research Foundation (2023A1515030002).

References

[1] Abuolaim , A. & Brown , M. S. (2020) Defocus deblurring using dual-pixel data. In *ECCV*

[2] Bae , J., Kim , S., Yun , Y., Lee , H., Bang , G., & Uh , Y. (2024) Per-gaussian embedding-based deformation for deformable 3d gaussian splatting. In *ECCV*

[3] Bui , M.-Q. V., Park , J., Oh , J., & Kim , M. (2023) Dyblurf: Dynamic deblurring neural radiance fields for blurry monocular video. *arXiv preprint arXiv:2312.13528*

[4] Bui , M.-Q. V., Park , J., Bello , J. L. G., Moon , J., Oh , J., & Kim , M. (2025) Mobgs: Motion deblurring dynamic 3d gaussian splatting for blurry monocular video. *arXiv preprint arXiv:2504.15122*

[5] Cao , A. & Johnson , J. (2023) Hexplane: A fast representation for dynamic scenes. In *CVPR*

[6] Cao , Z., Xu , L., Zhang , J., Yang , B., Chen , K., & Zhang , R. (2025) Dbdb: de-bimodal defocus blur in joint infrared-visible imaging. *Visual Intelligence*

[7] Cho , W. O., Cho , I., Kim , S., Bae , J., Uh , Y., & Kim , S. J. (2024) 4d scaffold gaussian splatting for memory efficient dynamic scene reconstruction. *arXiv preprint arXiv:2411.17044*

[8] Chu , W.-H., Ke , L., & Fragkiadaki , K. (2024) Dreamscene4d: Dynamic multi-object scene generation from monocular videos. *arXiv preprint arXiv:2405.02280*

[9] Dai , P., Zhang , Y., Yu , X., Lyu , X., & Qi , X. (2023) Hybrid neural rendering for large-scale scenes with motion blur. In *CVPR*

[10] Doersch , C., Yang , Y., Vecerik , M., Gokay , D., Gupta , A., Aytar , Y., Carreira , J., & Zisserman , A. (2023) Tapir: Tracking any point with per-frame initialization and temporal refinement. In *CVPR*

[11] Duan , Y., Wei , F., Dai , Q., He , Y., Chen , W., & Chen , B. (2024) 4d gaussian splatting: Towards efficient novel view synthesis for dynamic scenes. *SIGGRAPH*

[12] Guo , Z., Zhou , W., Li , L., Wang , M., & Li , H. (2024) Motion-aware 3d gaussian splatting for efficient dynamic scene reconstruction. *IEEE Transactions on Circuits and Systems for Video Technology*

[13] Huang , Y.-H., Sun , Y.-T., Yang , Z., Lyu , X., Cao , Y.-P., & Qi , X. (2024) Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *CVPR*

[14] Katsumata , K., Vo , D. M., & Nakayama , H. (2023) An efficient 3d gaussian representation for monocular/multi-view dynamic scenes. *arXiv preprint arXiv:2311.12897*

[15] Kerbl , B., Kopanas , G., Leimkühler , T., & Drettakis , G. (2023) 3d gaussian splatting for real-time radiance field rendering. *SIGGRAPH*

[16] Kim , M., Lim , J., & Han , B. (2024) 4d gaussian splatting in the wild with uncertainty-aware regularization. *NeurIPS*

[17] Kingma , D. P. & Ba , J. (2014) Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*

[18] Kirillov , A., Mintun , E., Ravi , N., Mao , H., Rolland , C., Gustafson , L., Xiao , T., Whitehead , S., Berg , A. C., Lo , W.-Y., & others (2023) Segment anything. In *ICCV*

[19] Kupyn , O., Budzan , V., Mykhailych , M., Mishkin , D., & Matas , J. (2018) Deblurgan: Blind motion deblurring using conditional adversarial networks. In *CVPR*

[20] Kupyn , O., Martyniuk , T., Wu , J., & Wang , Z. (2019) Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *CVPR*

[21] Lee , B., Lee , H., Sun , X., Ali , U., & Park , E. (2024) Deblurring 3d gaussian splatting. In *ECCV*

[22] Lee , D., Lee , M., Shin , C., & Lee , S. (2023.) Dp-nerf: Deblurred neural radiance field with physical scene priors. In *CVPR*

[23] Lee , D., Oh , J., Rim , J., Cho , S., & Lee , K. M. (2023.) Exblurf: Efficient radiance fields for extreme motion blurred images. In *ICCV*

[24] Lei , J., Weng , Y., Harley , A., Guibas , L., & Daniilidis , K. (2025) Mosca: Dynamic gaussian fusion from casual videos via 4d motion scaffolds. *CVPR*

[25] Li , L., Pan , J., Lai , W.-S., Gao , C., Sang , N., & Yang , M.-H. (2020) Dynamic scene deblurring by depth guided model. *IEEE Transactions on Image Processing*

[26] Li , Z., Chen , Z., Li , Z., & Xu , Y. (2024) Spacetime gaussian feature splatting for real-time dynamic view synthesis. In *CVPR*

[27] Liang , Y., Khan , N., Li , Z., Nguyen-Phuoc , T., Lanman , D., Tompkin , J., & Xiao , L. (2023) Gaufre: Gaussian deformation fields for real-time dynamic novel view synthesis. *arXiv preprint arXiv:2312.11458*

[28] Liang , Y., Xu , T., & Kikuchi , Y. (2025) Himor: Monocular deformable gaussian reconstruction with hierarchical motion representation. *CVPR*

[29] Liu , Q., Liu , Y., Wang , J., Lyv , X., Wang , P., Wang , W., & Hou , J. (2024) Modgs: Dynamic gaussian splatting from causually-captured monocular videos. *arXiv preprint arXiv:2406.00434*

[30] Lu , Y., Zhou , Y., Liu , D., Liang , T., & Yin , Y. (2025) Bard-gs: Blur-aware reconstruction of dynamic scenes via gaussian splatting. *CVPR*

[31] Luiten , J., Kopanas , G., Leibe , B., & Ramanan , D. (2024) Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *3DV*

[32] Luo , X., Sun , H., Peng , J., & Cao , Z. (2024) Dynamic neural radiance field from defocused monocular video. In *ECCV*

[33] Luthra , A., Gantha , S. S., Song , X., Yu , H., Lin , Z., & Peng , L. (2024) Deblur-nsff: Neural scene flow fields for blurry dynamic scenes. In *WACV*

[34] Ma , L., Li , X., Liao , J., Zhang , Q., Wang , X., Wang , J., & Sander , P. V. (2022) Deblur-nerf: Neural radiance fields from blurry images. In *CVPR*

[35] Mildenhall , B., Srinivasan , P. P., Tancik , M., Barron , J. T., Ramamoorthi , R., & Ng , R. (2021) Nerf: Representing scenes as neural radiance fields for view synthesis. *ECCV*

[36] Nah , S., Hyun Kim , T., & Mu Lee , K. (2017) Deep multi-scale convolutional neural network for dynamic scene deblurring. In *CVPR*

[37] Niu , M., Zhan , Y., Zhu , Q., Li , Z., Wang , W., Zhong , Z., Sun , X., & Zheng , Y. (2024) Bundle adjusted gaussian avatars deblurring. *arXiv preprint arXiv:2411.16758*

[38] Oh , J., Chung , J., Lee , D., & Lee , K. M. (2024) Deblurgs: Gaussian splatting for camera motion blur. *arXiv preprint arXiv:2404.11358*

[39] Pan , L., Chowdhury , S., Hartley , R., Liu , M., Zhang , H., & Li , H. (2021) Dual pixel exploration: Simultaneous depth estimation and image restoration. In *CVPR*

[40] Park , J., Bui , M.-Q. V., Bello , J. L. G., Moon , J., Oh , J., & Kim , M. (2025) Splinegs: Robust motion-adaptive spline for real-time dynamic 3d gaussians from monocular video. *CVPR*

[41] Peng , C., Tang , Y., Zhou , Y., Wang , N., Liu , X., Li , D., & Chellappa , R. (2024) Bags: Blur agnostic gaussian splatting through multi-scale kernel modeling. In *ECCV*

[42] Peng , J., Cao , Z., Luo , X., Lu , H., Xian , K., & Zhang , J. (2022) Bokehme: When neural rendering meets classical rendering. In *CVPR*

[43] Ruan , L., Chen , B., Li , J., & Lam , M. (2022) Learning to deblur using light field generated and real defocus images. In *CVPR*

[44] Srinivasan , P. P., Ng , R., & Ramamoorthi , R. (2017) Light field blind motion deblurring. In *CVPR*

[45] Stearns , C., Harley , A., Uy , M., Dubost , F., Tombari , F., Wetzstein , G., & Guibas , L. (2024) Dynamic gaussian marbles for novel view synthesis of casual monocular videos. In *SIGGRAPH Asia 2024*

[46] Su , S., Delbracio , M., Wang , J., Sapiro , G., Heidrich , W., & Wang , O. (2017) Deep video deblurring for hand-held cameras. In *CVPR*

[47] Sun , H., Li , X., Shen , L., Ye , X., Xian , K., & Cao , Z. (2024.) Dyblurf: Dynamic neural radiance fields from blurry monocular video. In *CVPR*

[48] Sun , J., Jiao , H., Li , G., Zhang , Z., Zhao , L., & Xing , W. (2024.) 3dgstream: On-the-fly training of 3d gaussians for efficient streaming of photo-realistic free-viewpoint videos. In *CVPR*

[49] Tao , X., Gao , H., Shen , X., Wang , J., & Jia , J. (2018) Scale-recurrent network for deep image deblurring. In *CVPR*

[50] Torres , G. F., Kalliola , J., Tripathy , S., Acar , E., & Kämäräinen , J.-K. (2024) Davide: Depth-aware video deblurring. *arXiv preprint arXiv:2409.01274*

[51] Wang , P., Zhao , L., Ma , R., & Liu , P. (2023) Bad-nerf: Bundle adjusted deblur neural radiance fields. In *CVPR*

[52] Wang , Q., Ye , V., Gao , H., Austin , J., Li , Z., & Kanazawa , A. (2024.) Shape of motion: 4d reconstruction from a single video. *arXiv preprint arXiv:2407.13764*

[53] Wang , S., Yang , X., Shen , Q., Jiang , Z., & Wang , X. (2025) Gflow: Recovering 4d world from monocular video. *AAAI*

[54] Wang , Y., Yang , Y., Yang , Z., Zhao , L., Wang , P., & Xu , W. (2018) Occlusion aware unsupervised learning of optical flow. In *CVPR*

[55] Wang , Y., Chakravarthula , P., & Chen , B. (2024.) Dof-gs: Adjustable depth-of-field 3d gaussian splatting for refocusing, defocus rendering and blur removal. *arXiv preprint arXiv:2405.17351*

[56] Wang , Z., Bovik , A. C., Sheikh , H. R., & Simoncelli , E. P. (2004) Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*

[57] Wu , G., Yi , T., Fang , J., Xie , L., Zhang , X., Wei , W., Liu , W., Tian , Q., & Wang , X. (2024.) 4d gaussian splatting for real-time dynamic scene rendering. In *CVPR*

[58] Wu , R., Zhang , Z., Chen , M., Fan , X., Yan , Z., & Zuo , W. (2024.) Deblur4dgs: 4d gaussian splatting from blurry monocular video. *arXiv preprint arXiv:2412.06424*

[59] Wu , Z., Li , X., Peng , J., Lu , H., Cao , Z., & Zhong , W. (2022) Dof-nerf: Depth-of-field meets neural radiance fields. In *ACM MM*

[60] Xu , W., Gao , H., Shen , S., Peng , R., Jiao , J., & Wang , R. (2024) Mvpgs: Excavating multi-view priors for gaussian splatting from sparse input views. In *ECCV*

[61] Yan , Y., Zhou , Z., Wang , Z., Gao , J., & Yang , X. (2024) Dialoguenerf: Towards realistic avatar face-to-face conversation video generation. *Visual Intelligence*

[62] Yang , L., Kang , B., Huang , Z., Xu , X., Feng , J., & Zhao , H. (2024.) Depth anything: Unleashing the power of large-scale unlabeled data. In *CVPR*

[63] Yang , Z., Gao , X., Zhou , W., Jiao , S., Zhang , Y., & Jin , X. (2024.) Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *CVPR*

[64] Yang , Z., Yang , H., Pan , Z., & Zhang , L. (2024.) Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *ICLR*

[65] Zamir , S. W., Arora , A., Khan , S., Hayat , M., Khan , F. S., & Yang , M.-H. (2022) Restormer: Efficient transformer for high-resolution image restoration. In *CVPR*

[66] Zhang , H., Xie , H., & Yao , H. (2024) Blur-aware spatio-temporal sparse transformer for video deblurring. In *CVPR*

[67] Zhang , R., Isola , P., Efros , A. A., Shechtman , E., & Wang , O. (2018) The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*

[68] Zhao , L., Wang , P., & Liu , P. (2024) Bad-gaussians: Bundle adjusted deblur gaussian splatting. In *ECCV*

[69] Zhong , Z., Gao , Y., Zheng , Y., & Zheng , B. (2020) Efficient spatio-temporal recurrent neural network for video deblurring. In *ECCV*

[70] Zhou , S., Zhang , J., Zuo , W., Xie , H., Pan , J., & Ren , J. S. (2019) Davanet: Stereo deblurring with view aggregation. In *CVPR*

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We describe the motivation and main contributions of this work in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: We discuss the limitations of this work and do a visualization in our paper.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We show the necessary theoretical proof and experimental validation in the main PDF and supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide full implementation details inside the supplementary material, while our code and trained model will be made publicly available.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The current code implementation requires additional refactoring for optimal readability. A refined open-source release is planned following code reorganization.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide data splits, setting of hyper-parameters, details of optimizer in the experimental data presentation and implementation details sections.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We didn't experiment with statistical significance.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We describe the computer resources required to reproduce the experiment in the implementation details.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have read the NeurIPS Code of Ethics and strictly adhere to it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Our work is oriented towards theoretical research and does not have a negative social impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This work involves no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All third-party assets (code, data, models) are explicitly cited in the paper, with original authors and sources clearly credited.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: Our code assets have accompanying documentation, and we will provide the appropriate documentation when we open source the code.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: Our core method development in this work does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Implementation Details

Network Architecture. The detailed architecture of Scene Feature Extractor Network and Blur Prediction Network in our framework are illustrated in Figure 9. We enhance the ability to capture high-frequency details by using positional encoding p for pixel coordinates x and a discrete variable embedding module e (implemented with PyTorch) for camera indices i.

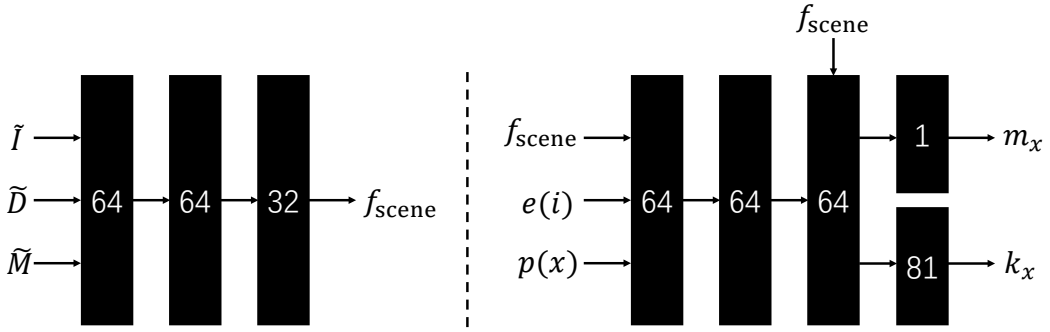

Figure 9: **Scene Feature Extractor Network (Left).** Scene Feature Extractor Network takes rendered image $\tilde{I}$, rendered depth $\tilde{D}$ and rendered mask $\tilde{M}$ as input, and outputs scene feature f_{scene}. Besides the last layer, each layer outputs 64-dimensional features with ReLU activations. **Blur Prediction Network (Right).** Blur Prediction Network takes scene feature f_{scene}, camera embedding vector $e(i)$ and pixel positional encoding $p(x)$ as input, and outputs blur kernel k_x and blur intensity m_x. Each layer outputs 64-dimensional features with ReLU activations, except for the last layer. Note, in the last layer m_x is obtained via Sigmoid activations, while k_x is obtained via Softmax activations.

B Additional Quantitative and Qualitative Results

We compare our method with dynamic scene reconstruction methods [63, 52] that use video-deblurred images as input. Figure 10 presents a visual comparison of novel view synthesis on the motion blur dataset, where we can see that our method outperforms existing methods that are fed with deblurred images produced by a state-of-the-art video deblurring method. The reason is that video deblurring methods cannot effectively ensure 3D scene consistency in the deblurred images. Please see the video supplementary material for additional novel view synthesis comparison results.

B.1 Novel View Synthesis Comparison

D2RF and DyBluRF Dataset. We compare our approach against BAGS [41] and De3DGS [21], two methods designed to reconstruct sharp static scenes from blurred static images, and evaluate them on the D2RF [32] and DyBluRF [47] datasets. Table 4 and Figure 11 present the comparison results. Clearly, our method demonstrates significant advantages over other methods, producing sharper novel view images while better preserving realistic motion details.

D2RF-v2 and DyBluRF-v2 Dataset. We evaluate our method on two datasets (DyBluRF-v2 and D2RF-v2) with both motion and defocus blur occurring simultaneously. Note that we obtain the DyBluRF-v2 dataset by applying depth-of-field (DoF) rendering technique in Bokehme [42] to the original DyBluRF dataset [47] to simulate defocus blur, and create the D2RF-v2 dataset with motion blur by processing the original D2RF dataset [32] using the motion blur generation method in Davanet [70]. Table 5 present the comparison results. As shown, our method clearly outperforms all the compared methods on the two datasets, verifying the advantage of our method in handling cases with motion and defocus blur occurring jointly.

Deblur-NeRF Dataset. We compare our approach against De3DGS [21], which is designed to reconstruct sharp static scenes from blurred static images, and evaluate them on the Deblur-NeRF [34] dataset. Table 6 and Figure 12 present the comparison results. Clearly, our method demonstrates significant advantages over other methods, producing sharper novel view images.

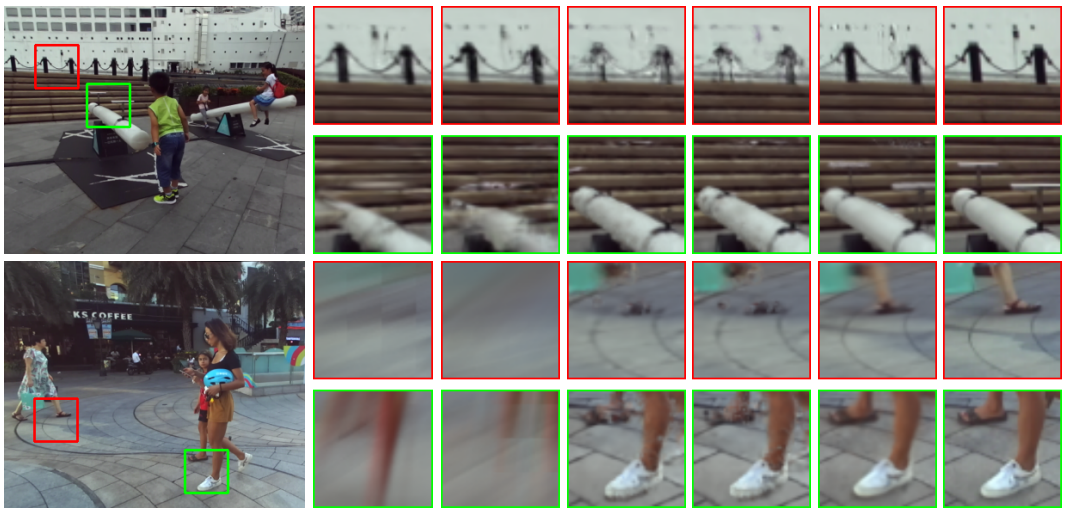

Figure 10: **Visual comparison of novel view synthesis on the DyBluRF motion blur dataset [47].** Here, we also compare with methods fed with deblurred images produced by a state-of-the-art video deblurring method [66] to manifest the effectiveness of our method.

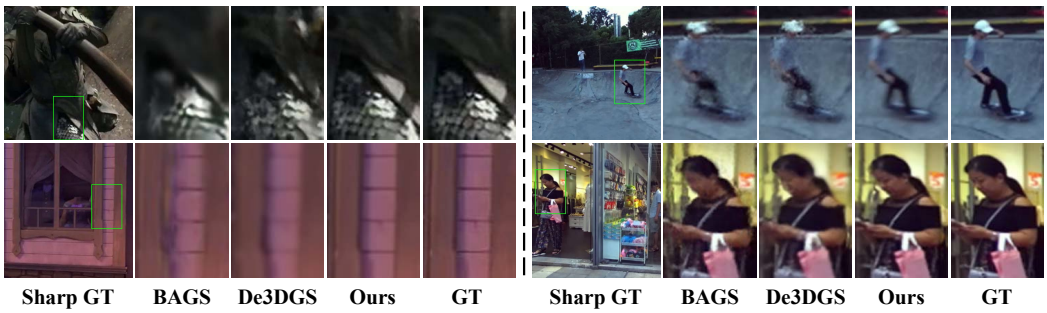

Figure 11: **Visual comparison of novel view synthesis.** Here, we compare our method with methods that are designed to reconstruct sharp static scenes from blurred static scene images. Note, the left column demonstrates results for defocus blur, while the right column presents motion blur outcomes.

Table 4: **Quantitative comparison of novel view synthesis on the D2RF defocus blur dataset [32] and the DyBluRF motion blur dataset [47].**

Method	Defocus Blur			Motion Blur		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
BAGS [41]	24.41	0.730	0.167	24.27	0.723	0.208
De3DGS [21]	23.74	0.716	0.190	22.45	0.689	0.253
Ours	**29.39**	**0.859**	**0.078**	**27.01**	**0.876**	**0.056**

B.2 Deblurring Comparison

We compare the deblur ring capability of our method with a broad range of existing methods, including 3DGS- and NeRF-based methods for both dynamic and static scenes [32, 58, 41, 21], as well as transformer-based video deblurring method [66]. Specifically, we compare the sharp images produced at training views. For 3DGS- and NeRF-based methods, these images are rendered from the trained sharp scene representations using the same training views. Table 7 and Figure 13 present the comparison results. Our method outperforms 3DGS- and NeRF-based deblurring approaches and achieves performance comparable to state-of-the-art video deblurring methods.

De3DGS [63] Ours GT

Figure 12: **Visual comparison of novel view synthesis on the Deblur-NeRF dataset [34].**

Table 5: **Quantitative comparison of novel view synthesis on the D2RF-v2 defocus blur dataset and the DyBluRF-v2 motion blur dataset.** The numerical results of defocus blur are obtained on the Shop and Car scenes of the D2RF-v2 dataset, and the numerical results of motion blur are obtained on the Man and Seesaw scenes of the DyBluRF-v2 dataset.

Method	Defocus Blur			Motion Blur		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
D3DGS [63]	23.66	0.739	0.257	21.75	0.655	0.289
SoM [52]	29.04	0.820	0.094	27.28	0.791	0.103
D2RF [32]	27.82	0.795	0.132	25.89	0.722	0.133
DyBluRF [47]	27.30	0.771	0.150	26.54	0.753	0.112
De4DGS [58]	29.74	0.856	0.078	27.97	0.824	0.087
Ours	**30.26**	**0.885**	**0.062**	**28.55**	**0.859**	**0.064**

Table 6: **Quantitative comparison of novel view synthesis on the Deblur-NeRF dataset [34].**

Method	Defocus Blur			Motion Blur		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
De3DGS [21]	23.71	0.747	0.110	26.61	0.822	0.108
Ours	**24.22**	**0.768**	**0.095**	**27.14**	**0.835**	**0.096**

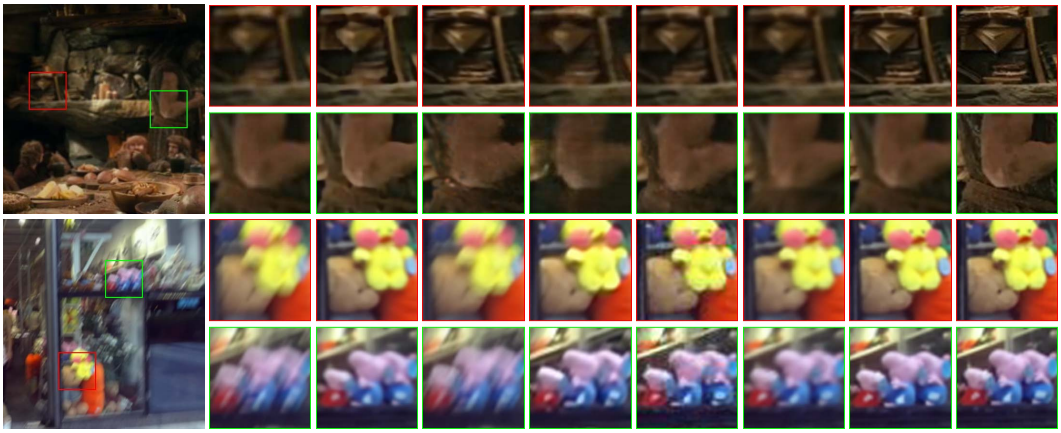

Figure 13: **Visual comparison of deblurring.** Our method enables the synthesis of high-quality deblurring results for videos with defocus blur (top) and motion blur (bottom).

Table 7: **Quantitative comparison of deblurring on the D2RF defocus blur dataset [32] and the DyBluRF motion blur dataset [47].**

Method	Defocus Blur			Motion Blur		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
BSSTNet [66]	33.54	0.961	0.039	**33.71**	**0.965**	**0.030**
D2RF [32]	34.33	0.976	0.028	32.14	0.949	0.049
De4DGS [58]	32.92	0.953	0.044	33.27	0.958	0.035
BAGS [41]	30.69	0.940	0.084	30.55	0.935	0.092
De3DGS [21]	30.36	0.941	0.090	29.84	0.924	0.105
Ours	**34.85**	**0.977**	**0.027**	33.45	0.960	0.036

C Additional Ablation Results

C.1 Ablation on BP-Net

We conduct an ablation study to evaluate the contribution of BP-Net. Specifically, we compare three different blur modeling methods: (i) the motion blur and defocus blur modeling method used in De3DGS [21] (w/ blur modeling in De3DGS [21]), (ii) the motion blur modeling method in De4DGS [58] (w/ blur modeling in Deblur4DGS [58]), and (iii) the defocus blur modeling method in D2RF [32] (w/ blur modeling in D2RF [32]). We report the quantitative results in Table 8, where we can see that our method with the proposed BP-Net produces better results than these alternatives, demonstrating the effectiveness of the BP-Net.

Table 8: **Effect of BP-Net.**

Method	Defocus Blur			Motion Blur		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
w/ blur modeling in De3DGS [21]	28.31	0.812	0.098	26.05	0.823	0.118
w/ blur modeling in De4DGS [58]	28.63	0.829	0.094	26.74	0.859	0.060
w/ blur modeling in D2RF [32]	28.96	0.832	0.094	26.30	0.825	0.109
Ours with BP-Net	**29.39**	**0.859**	**0.078**	**27.01**	**0.876**	**0.056**

C.2 Ablation on Blur Kernel Size

Table 9 further perform quantitative evaluation on how different blur kernel sizes (denoted as K) affect the performance of our method. As shown, a larger blur kernel helps to obtain better results. However, this trend becomes less obvious when K is larger than 9. To balance the performance and the computational cost, we thus choose $K = 9$ as our default choice.

Table 9: **Effect of varying blur kernel size K.** The numerical results of defocus blur are obtained on the Gate and Dock scenes of the D2RF dataset, and the numerical results of motion blur are obtained on the Skating and Man scenes of the DyBluRF dataset.

Blur kernel size	Defocus Blur			Motion Blur		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
K=5	27.84	0.817	0.098	29.53	0.906	0.092
K=7	28.12	0.836	0.074	29.79	0.913	0.067
K=9	28.29	0.842	0.067	30.01	0.921	0.052
K=11	28.30	0.843	0.066	30.02	0.920	0.050
K=13	28.31	0.842	0.067	30.04	0.922	0.051

Figure 14: **Visualization of the blur kernel and blur intensity predicted by BP-Net.** Note that the top row shows an image with defocus blur, while the bottom row shows an image with motion blur. In the blur ground truth (GT) image, blue markers indicate pixels with almost no blur, green markers denote pixels with mild blur, and red markers represent pixels with severe blur. A higher blur intensity value corresponds to a more heavily blurred area. Clearly, BP-Net can accurately predict blur regions in images with different types of blur and estimate the corresponding blur kernels at pixel locations with varying blur levels.

D Robustness to Preprocessing and Segmentation Errors

Our method represents image blurring effects using two components: a blur kernel k and a blur intensity m, as illustrated in Figure 14. The blur intensity m effectively emphasizes the spatial regions affected by blur within each training image. Moreover, the type of blur can be intuitively inferred from the estimated kernels. Specifically, kernels corresponding to motion blur capture structured trajectories that reflect the camera's movement, while those associated with defocus blur present Gaussian-shaped patterns that vary with the distance of the pixel from the focal plane.

To evaluate the accuracy of the blur kernel k predicted by BP-Net, we compare the ground truth blur kernel with the blur kernel predicted by BP-Net on a blurry monocular video dataset with ground truth blur kernel. To this end, we construct two datasets with ground truth blur kernels, referred to as D2RF-v3 and DyBluRF-v3, by randomly sampling two global Gaussian and linear distribution blur kernels of size 9 x 9 and then respectively applying them to the ground truth sharp images in D2RF [32] and DyBluRF [47] to obtain the corresponding blurry images. With the two datasets, we quantitatively compare our estimated blur kernels and the ground truth blur kernels using PSNR and KL divergence as metrics. Table 10 and Figure 15 present the comparison results. Clearly, our estimated blur kernels are highly similar to the ground truth blur kernels in numerical metrics, manifesting the effectiveness of the BP-Net in predicting different types of blur.

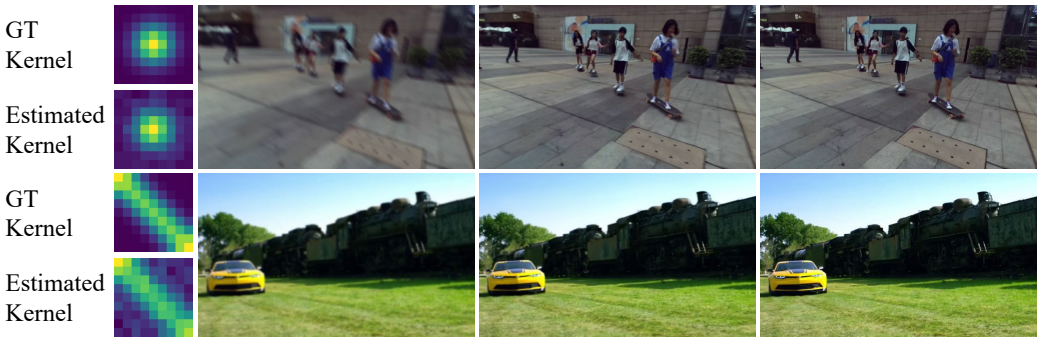

Figure 15: **Visual comparison of estimated kernel on the D2RF-v3 defocus blur dataset and the DyBluRF-v3 motion blur dataset.** Note that the top row shows an image with defocus blur, while the bottom row shows an image with motion blur. Clearly, BP-Net can accurately estimate blur kernels from various types of blurry images and thus recover sharp images.

Table 10: **Comparison of ground truth kernels and estimated kernels.**

	D2RF-v3	DyBluRF-vs
PSNR↑	32.516	29.941
KL Div.↓	0.214	0.247

E Robustness to Preprocessing and Segmentation Errors

In the Table 11, we quantitatively evaluate how errors from external preprocessing steps affect the robustness of our method. To simulate errors from depth estimation, we randomly scale and shift the estimated depth maps within the range of [0.8, 1.2] and [-20, 20], respectively. To simulate errors from 2D point tracking and SAM, we randomly shift the 2D tracking points within the range of [-30, 30], and randomly add or delete five 25 X 25 mask regions towards the mask predicted by SAM. As shown, our results produced with the artificially perturbed depth, tracking points, and motion mask are comparable to those produced with the originally estimated depth, tracking points, and motion mask, indicating that our method has some tolerance to errors from external preprocessing steps.

Table 11: **Analysis on the impact of errors from external preprocessing steps.**

Method	Defocus Blur			Motion Blur		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Ours w/ depth perturbation	29.19	0.844	0.088	26.79	0.862	0.074
Ours w/ tracking perturbation	29.21	0.854	0.084	26.86	0.874	0.060
Ours w/ mask perturbation	29.25	0.854	0.085	26.74	0.865	0.067
Ours	29.39	0.859	0.078	27.01	0.876	0.056