DyBluRF: Dynamic Deblurring Neural Radiance Fields for Blurry Monocular Video

Minh-Quan Viet Bui[1*] Jongmin Park[1*] Jihyong Oh[2*] Munchurl Kim[1†]
[1]KAIST [2]Chung-Ang University
{bvmquan, jm.park, mkimee}@kaist.ac.kr jihyongoh@cau.ac.kr
https://kaist-viclab.github.io/dyblurf-site/

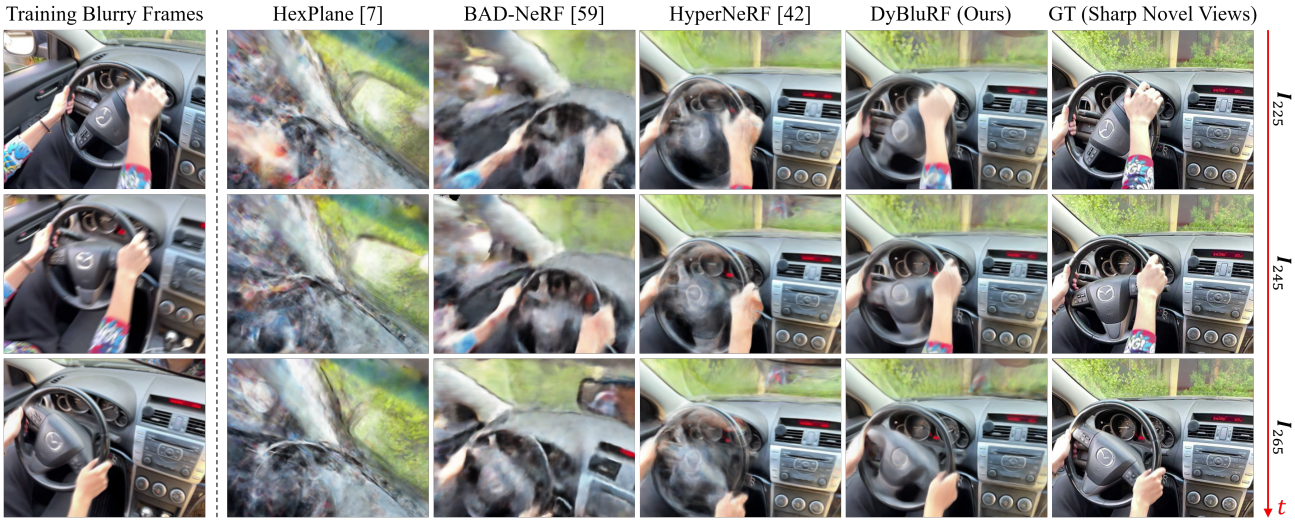


Figure 1. **Dynamic deblurring novel view synthesis results.** We *first* propose a novel dynamic deblurring NeRF for *blurry* monocular videos with *inaccurate* camera poses, called DyBluRF, which significantly outperforms previous SOTA NeRF methods for deblurring static scenes (BAD-NeRF [59]) and monocular videos (HexPlane [7], HyperNeRF [42]), trained on the newly synthesized Blurry iPhone Dataset.

Abstract

Video view synthesis, allowing for the creation of visually appealing frames from arbitrary viewpoints and times, offers immersive viewing experiences. Neural radiance fields, particularly NeRF, initially developed for static scenes, have spurred the creation of various methods for video view synthesis. However, the challenge for video view synthesis arises from motion blur, a consequence of object or camera movement during exposure, which hinders the precise synthesis of sharp spatio-temporal views. In response, we propose a novel dynamic deblurring NeRF framework for blurry monocular video, called DyBluRF, consisting of an Interleave Ray Refinement (IRR) stage and a Motion Decomposition-based Deblurring (MDD) stage. Our DyBluRF is the first that addresses and handles the novel view synthesis for blurry monocular video. The IRR stage jointly reconstructs dynamic 3D scenes and refines the inaccurate camera pose information to combat imprecise pose information extracted from the given blurry frames. The MDD stage is a novel incremental latent sharp-rays prediction (ILSP) approach for the blurry monocular video frames by decomposing the latent sharp rays into global camera motion and local object motion components. Extensive experimental results demonstrate that our DyBluRF outperforms qualitatively and quantitatively the very recent state-of-the-art methods. Our project page including source codes and pretrained model are publicly available at https://kaist-viclab.github.io/dyblurf-site/.

1. Introduction

Free viewpoint rendering for spatio-temporal novel view synthesis has increased lots of interests due to its diverse applications. Especially, video view synthesis can render visu-

*Co-first authors (equal contribution).
†Corresponding author.

ally pleasing frames at arbitrary camera viewpoints and time instances. After the first advent of neural radiance fields for a static scene, called NeRF [35], diverse types of neural rendering methods [26, 47, 58, 62] for video view synthesis have actively been developed. Multi-view-based methods [6, 11, 26, 37, 69, 72] have been traditionally adopted for free viewpoint rendering of dynamic videos. However, they need a synchronized capturing process for multi-camera systems which are impractical for general users. To overcome this, several methods of dynamic view synthesis [2, 17, 18, 27, 28, 44, 51, 56] for a casually captured monocular video have been proposed for easier applications.

On the other hand, motion blur arises as a consequence of either object motion [38, 68] or camera shake [3, 67] caused by the accumulation of light during video acquisition [20, 21, 36, 55]. Therefore, synthesizing *sharp* novel spatio-temporal views from monocular video is faced with several challenges when *blurriness* presents in the given frames due to the camera capture process: (i) One straightforward solution is to apply 2D video deblurring [40, 71] as a preprocessing step to the given blurry frames before optimizing video NeRFs. However, this naive approach has a significant drawback as independently deblurring frames in the pixel domain can introduce inconsistent geometry in 3D space [24, 25], which cannot be corrected through video NeRF optimization; (ii) Although several deblurring NeRFs [24, 25, 33, 59] have been developed to address blurriness in *static* multi-view images, they encounter difficulties in capturing temporal information when extended to blurry monocular videos. This is due to the absence of a motion-aware deblurring module along the temporal dimension. Additionally, the existing state-of-the-art (SOTA) monocular video NeRF methods [7, 15, 42] cannot be directly applied to deblur NeRFs from given blurry frames because they lack an effective deblurring component; (iii) The accuracy of camera poses extracted by Structure-from-Motion (SfM) algorithms from blurry monocular videos, including deformable objects, is particularly low, leading to challenges in detecting and matching salient keypoints [25]. However, the previous bundle-adjustment for NeRFs [30, 43, 61] have primarily focused on rigid scenes.

To address these challenges, we *first* propose a novel dynamic deblurring NeRF for blurry monocular video, called DyBluRF, and our contributions are as follows:

- We *firstly* point out the issue of handling *blurry video of dynamic scenes* for 3D reconstruction, and propose a novel framework, called DyBluRF, which can effectively render the sharp novel spatio-temporal views from blurry monocular videos with imprecise camera poses;
- We propose a novel *Interleave Ray Refinement* (IRR) stage which simultaneously performs the reconstruction of dynamic 3D scenes and the refinement of camera pose to overcome inaccurate camera pose information;
- We propose a novel *Motion Decomposition-based Deblurring* (MDD) stage which includes a novel incremental latent sharp-rays prediction (ILSP) approach to effectively handle the blurriness due to global camera and local object motions in the monocular videos;
- For experiments, we synthesize a new blurry version of iPhone dataset [19] which is used to train the DyBluRF and other methods under fair comparison. The experimental results demonstrate that our DyBluRF achieves superior performance compared to the previous SOTA methods qualitatively (Fig.1) and quantitatively. Notably, the DyBluRF trained using the blurry dataset *even shows comparable results* with the SOTA methods trained on sharp dataset with accurate camera pose information.

2. Related Work

2.1. Conventional Video Deblurring

Motion blur can be attributed to either objects in motion [38, 68] or camera shake [3, 67], both of which stem from the gathering of light during video capture over the exposure time [20, 21, 36, 55]. Various deep learning methods [12, 29, 39, 60, 66, 70] have been developed for video deblurring. However, naively applying conventional video deblurring as a preprocessing step to the given blurry frames before optimizing video NeRFs induces inconsistent geometry in 3D space [24, 25], which cannot be corrected through video NeRF optimization.

2.2. Deblurring NeRFs for Static Scenes

To produce visually appealing frames with consistent 3D geometry from blurry multi-view *static* images, several NeRF [35]-based methods have emerged. DeblurNeRF [33] employs an end-to-end volume rendering framework [13] to estimate spatial blur kernels at the pixel level and the latent sharp radiance fields. BAD-NeRF [59] jointly predicts the virtual camera trajectories during the image exposure capture time. DP-NeRF [24] introduces a rigid blurring kernel to maintain 3D consistency by leveraging physical constraints. ExBluRF [25] introduces an MLP-based framework for reducing the dimensionality of 6-DOF camera poses and employing a voxel-based radiance field [8, 16]. Nonetheless, none of the above methods *can be applicable for non-rigid video view synthesis* due to the lack of motion-aware deblurring for the temporal dimension.

2.3. NeRFs for Dynamic Scenes

Recent methods for video view synthesis have expanded upon the static NeRF framework [35]. They represent dynamic NeRFs by incorporating scene flow-based frameworks [18, 27, 28] or canonical fields [1, 15, 22, 32, 41, 42, 47, 54, 56, 63, 65] to model non-rigid deformable transformations or 4D spatio-temporal radiance fields [2, 7, 14, 17, 18, 26, 27, 51, 57, 64]. The methods such as NSFF [27], DynamicNeRF [18], and DynIBaR [28] typi-

cally combine two types of NeRFs: time-invariant and time-variant, to generate novel spatio-temporal views for monocular videos. However, they rely heavily on pretrained motion mask extraction for moving objects and various regularization losses for 3D scene flows, which makes them less effective in deblurring video view synthesis. The methods of D-NeRF [47], HyperNeRF [42], and TiNeuVox [15] initially learn deformation or offset fields that transform a ray in an observation space to a bent ray in a canonical space. However, none of the existing SOTA monocular video NeRF methods mentioned above can be readily applied for deblurred neural radiance fields from the given blurry frames due to the lack of an effective deblurring component and robust pose optimization in the presence of corrupted pose information due to blurriness.

2.4. NeRFs with Pose Optimization

In order to accurately capture fine details, NeRF requires precise camera poses and stationary scenes during the capture process. However, in real-world scenarios, the camera poses obtained through SfM algorithms inherently contain pixel-level inaccuracies [23, 31, 49]. These inaccuracies can be further exacerbated when dealing with blurry frames due to challenges in detecting and matching distinctive keypoints [25], especially in the presence of both global camera and local object movements. Consequently, the rendered output of NeRF often exhibits significant misalignment with the ground truth images. Several NeRF methods [9, 30, 34, 43, 61] have been developed to jointly optimize NeRF parameters and pose information. However, these methods primarily concentrate on scenarios with rigid characteristics, which considerably differ from the non-rigid properties frequently observed in blurry monocular videos.

3. Proposed Method: DyBluRF

3.1. Design Considerations

We *firstly* propose a novel dynamic deblurring NeRF, called DyBluRF, which aims to represent *sharp* dynamic neural radiance fields from *blurry* monocular videos. DyBluRF consists of two main procedures which are *Interleave Ray Refinement* (IRR) stage (Sec. 3.3) and *Motion Decomposition-based Deblurring* (MDD) stage (Sec. 3.4), as shown in Fig. 2 and Algo. 2.

The accuracy of pose information extracted by SfM algorithms [50] from blurry monocular videos with deformable objects is especially compromised due to challenges in detecting and matching salient keypoints [25]. However, the prior bundle-adjustment approaches [30, 43, 61] only focus on static environments. To solve this issue, in the IRR stage (Sec. 3.3), we coarsely reconstruct the dynamic radiance fields and optimize the imprecise camera poses. Specifically, we propose an interleave optimization strategy (Algo. 1) based on factorizing radiance fields into static and dynamic components. The interleave optimization enhances our DyBluRF's training stability while jointly learning the 3D dynamic reconstruction and refining the camera poses.

On the other hand, the existing SOTA methods for deblurring NeRFs [24, 25, 33, 59] only consider for *static* images so they encounter difficulties in capturing temporal information when extended for blurry *monocular* videos due to the absence of a motion-aware deblurring module. Also, the existing SOTA methods for monocular video view synthesis [7, 15, 42] are not capable of handling the input blurry frames due to the lack of deblurring component. To overcome these limitations, we introduce the MDD stage (Sec. 3.4), with a novel incremental latent sharp-rays prediction (ILSP) method which effectively synthesizes the physical blur process considering global camera motion and local object motion in a progressive manner along temporal axis.

3.2. Preliminaries

Dynamic Neural Radiance Fields. We extend the static NeRF model [35] to our DyBluRF for the monocular video which consists of one frame per time t. Our DyBluRF learns to represent the continuous radiance of a video scene using neural networks, taking into account a set of N_f frames from the monocular video, denoted as $\{\boldsymbol{\mathcal{I}}_t\}_{t=1}^{N_f}$, and the corresponding camera poses $\{\boldsymbol{\mathcal{P}}_t\}_{t=1}^{N_f}$. Following [18, 27, 28, 32], we decompose our radiance representation into Static Net F_{θ^s} and Dynamic Net F_{θ^d}. Given a 3D position $\mathbf{x} = (x, y, z)$ of each sampling point and a viewing direction $\mathbf{d}$ of each ray, $F_{\theta^s} : \gamma(\mathbf{x}, \mathbf{d}) \rightarrow (\boldsymbol{c}^s, \sigma^s, b)$ estimates a color $\boldsymbol{c}^s$, a volume density σ^s of a static scene component and a volume density blending factor b from the spatial positional encoded inputs $\gamma(\mathbf{x}, \mathbf{d})$. On the other hand, $F_{\theta^d} : (\gamma(\mathbf{x}, \mathbf{d}), l(t)) \rightarrow (\boldsymbol{c}^d, \sigma^d)$ maps a time-varying embedding to a color $\boldsymbol{c}^d$ and a volume density σ^d of a dynamic scene component where γ is the positional encoding [35] and l is Generative Latent Optimization (GLO) [5] to encode t. Let $\boldsymbol{r}_{\boldsymbol{p};t}(s) = \mathbf{o}_t + s\mathbf{d}_{\boldsymbol{p};t}$ be the casted ray from a camera origin $\mathbf{o}_t$ through a given pixel $\boldsymbol{p}$ of the image plane at the time t where s and $\mathbf{d}_{\boldsymbol{p};t}$ denote a sampling ray distance and a viewing direction through the pixel $\boldsymbol{p}$ at time t, respectively. We separately estimate the rendered colors $\hat{\boldsymbol{C}}^s(\boldsymbol{r}_{\boldsymbol{p};t})$ of the static scene component and $\hat{\boldsymbol{C}}^d(\boldsymbol{r}_{\boldsymbol{p};t})$ of the dynamic scene component via continuous volume rendering [13] by computing the integral on N piecewise constant segments $\{[s_n, s_{n+1}]\}_{n=1}^{N}$ along the ray $\boldsymbol{r}_{\boldsymbol{p};t}$ as:

$$\hat{\boldsymbol{C}}^s(\boldsymbol{r}_{\boldsymbol{p};t}) = \sum_{n=1}^{N} \mathcal{T}_n^s \alpha_n^s \boldsymbol{c}_n^s, \; \hat{\boldsymbol{C}}^d(\boldsymbol{r}_{\boldsymbol{p};t}) = \sum_{n=1}^{N} \mathcal{T}_n^d \alpha_n^d \boldsymbol{c}_n^d, \quad (1)$$

where $\mathcal{T}_n$ is the accumulated transmittance and α_n is the alpha-compositing weight, which are defined as:

$$\alpha_n = 1 - \exp(-\sigma_n \delta_n), \quad \mathcal{T}_n = \prod_{k=1}^{n-1} 1 - \alpha_k, \quad (2)$$

where $\delta_n = s_{n+1} - s_n$ is the segment length. To predict the full rendered color $\hat{\boldsymbol{C}}^{full}(\boldsymbol{r}_{\boldsymbol{p};t})$ of pixel $\boldsymbol{p}$ with camera pose

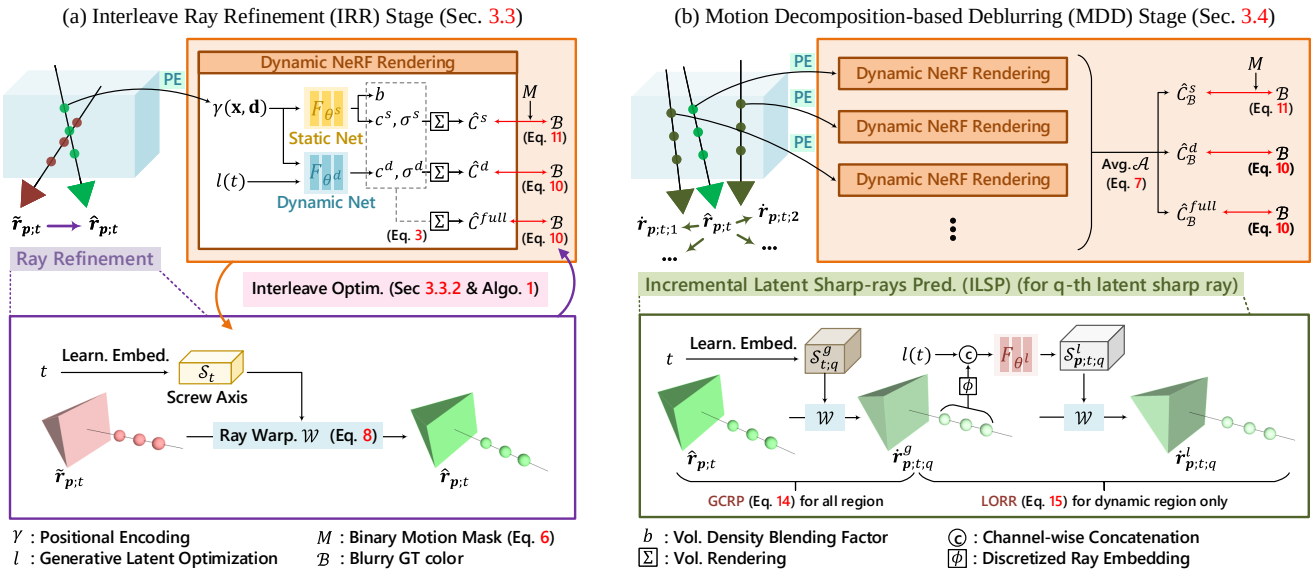


Figure 2. **Overview of our DyBluRF framework.** To effectively optimize the sharp radiance field with the imprecise camera poses extracted from blurry video frames, we design our DyBluRF consisting of two main procedures (Algo. 2) of (a) *Interleave Ray Refinement Stage* (Sec. 3.3 and Algo. 1) and (b) *Motion Decomposition-based Deblurring Stage* (Sec. 3.4).

$\boldsymbol{\mathcal{P}}_t$, our DyBluRF combines the outputs of F_{θ^s} and F_{θ^d} via the volume density blending factor b as:

$$\hat{\boldsymbol{C}}^{full}(\boldsymbol{r}_{\boldsymbol{p};t}) = \sum_{n=1}^{N} \mathcal{T}_n^{full}(\alpha_n^s b_n \boldsymbol{c}_n^s + \alpha_n^d (1-b_n)\boldsymbol{c}_n^d), \quad (3)$$

where the full accumulated transmittance $\mathcal{T}_n^{full}$ is:

$$\mathcal{T}_n^{full} = \prod_{k=1}^{n-1} (1-\alpha_k^s b_k)(1-\alpha_k^d(1-b_k)). \quad (4)$$

Binary Motion Mask Prediction. Learning the motion decomposition has been widely adopted in previous works [18, 26, 28] to stabilize the reconstruction of static scene components in the dynamic NeRFs. In our DyBluRF, motion decomposition is vital for both the IRR stage and the MDD stage. We compute the motion uncertainty $\boldsymbol{M}_{uncert}(\boldsymbol{r}_{\boldsymbol{p};t})$ by accumulating the alpha-compositing weight α_n^d of the dynamic scene component by $(1-b_n)$ along $\boldsymbol{r}_{\boldsymbol{p};t}$ as:

$$\boldsymbol{M}_{uncert}(\boldsymbol{r}_{\boldsymbol{p};t}) = \sum_{n=1}^{N} \mathcal{T}_n^{full} \alpha_n^d (1-b_n). \quad (5)$$

The final binary motion mask $\boldsymbol{M}(\boldsymbol{r}_{\boldsymbol{p};t})$ is obtained by thresholding $\boldsymbol{M}_{uncert}(\boldsymbol{r}_{\boldsymbol{p};t})$ as:

$$\boldsymbol{M}(\boldsymbol{r}_{\boldsymbol{p};t}) = \begin{cases} 1, & \text{if } \boldsymbol{M}_{uncert}(\boldsymbol{r}_{\boldsymbol{p};t}) > 0.5 \\ 0, & \text{otherwise} \end{cases}, \quad (6)$$

Deblurring Neural Radiance Fields. To solve the blur problem for reconstructing sharp radiance fields, we predict the pixel-wise blur kernels and sharp pixel colors to simulate the physical blur process similar to the existing deblurring static NeRF methods [24, 33, 59]. The physical blur process which generates a blurry color $\boldsymbol{\mathcal{B}}_{\boldsymbol{p};t}$ of pixel $\boldsymbol{p}$ at time t by applying an unknown motion blur kernel $k_{\boldsymbol{p};t}$ to the sharp pixel color $\boldsymbol{\mathcal{I}}_{\boldsymbol{p};t}$ is formulated as $\boldsymbol{\mathcal{B}}_{\boldsymbol{p};t} = k_{\boldsymbol{p};t} * \boldsymbol{\mathcal{I}}_{\boldsymbol{p};t}$ where $*$ indicates the convolution operation. We train our DyBluRF with the given blurry monocular video frames $\{\boldsymbol{\mathcal{B}}_t\}_{t=1}^{N_f}$ with inaccurate camera poses $\{\tilde{\boldsymbol{\mathcal{P}}}_t\}_{t=1}^{N_f}$. To optimize our DyBluRF by using the blurry frames, we model the blur process for monocular dynamic radiance fields by predicting the set of latent sharp rays $\{\dot{\boldsymbol{r}}_{\boldsymbol{p};t;q}\}_{q=1}^{N_b}$ casting based on the target ray $\boldsymbol{r}_{\boldsymbol{p};t}$. Then, we average the corresponding volume rendered pixel colors to generate a blurry pixel color where q is the index and N_b is the number of latent sharp rays, respectively. We denote this blur process as:

$$\begin{aligned} \hat{\boldsymbol{C}}_{\mathcal{B}}(\boldsymbol{r}_{\boldsymbol{p};t}) &= \mathcal{A}(\hat{\boldsymbol{C}}(\boldsymbol{r}_{\boldsymbol{p};t}), \{\hat{\boldsymbol{C}}(\dot{\boldsymbol{r}}_{\boldsymbol{p};t;q})\}_{q=1}^{N_b}) \\ &= \frac{1}{N_b+1}\Big(\hat{\boldsymbol{C}}(\boldsymbol{r}_{\boldsymbol{p};t}) + \sum_{q=1}^{N_b} \hat{\boldsymbol{C}}(\dot{\boldsymbol{r}}_{\boldsymbol{p};t;q})\Big), \end{aligned} \quad (7)$$

where $\hat{\boldsymbol{C}}_{\mathcal{B}}(\boldsymbol{r}_{\boldsymbol{p};t})$ is a blurry rendered color of the ray $\boldsymbol{r}_{\boldsymbol{p};t}$ and $\mathcal{A}(\cdot,\cdot)$ is an average function of the rendered color $\hat{\boldsymbol{C}}(\boldsymbol{r}_{\boldsymbol{p};t})$ of the ray $\boldsymbol{r}_{\boldsymbol{p};t}$ and the set of rendered colors $\{\hat{\boldsymbol{C}}(\dot{\boldsymbol{r}}_{\boldsymbol{p};t;q})\}_{q=1}^{N_b}$ of the latent sharp rays $\{\dot{\boldsymbol{r}}_{\boldsymbol{p};t;q}\}_{q=1}^{N_b}$.

3.3. Interleave Ray Refinement Stage

In the IRR stage, we coarsely train the Static and Dynamic Nets and optimize the imprecise camera poses via a learnable embedding for a screw axis $\mathcal{S}_t$ in Eq. 8 from the blurry frames, as shown in Fig. 2-(a) and Algo. 1. We describe our ray refinement and interleave optimization strategy to jointly learn the 3D dynamic reconstruction and adjust the camera pose estimates in the following subsections.

3.3.1 Ray Refinement

Let $\tilde{\boldsymbol{r}}_{\boldsymbol{p};t}$ be an inaccurate ray shooted from pixel $\boldsymbol{p}$ with an inaccurate camera pose $\tilde{\boldsymbol{\mathcal{P}}}_t$ estimate at time t, we refine the resulting inaccurate ray $\tilde{\boldsymbol{r}}_{\boldsymbol{p};t}$ to more accurate ray $\hat{\boldsymbol{r}}_{\boldsymbol{p};t}$ by a ray warping $\mathcal{W}$ as:

$$\hat{\boldsymbol{r}}_{\boldsymbol{p};t} = \mathcal{W}(\tilde{\boldsymbol{r}}_{\boldsymbol{p};t}, \mathcal{S}_t) = e^{[\boldsymbol{\omega}_t]_\times}\tilde{\boldsymbol{r}}_{\boldsymbol{p};t} + \boldsymbol{G}_t\boldsymbol{v}_t, \tag{8}$$

where $\mathcal{S}_t = (\boldsymbol{\omega}_t; \boldsymbol{v}_t) \in \mathbb{R}^6$ is a learnable screw axis with the corresponding rotation encoding $\boldsymbol{\omega}_t \in \mathfrak{so}(3)$ and translation encoding $\boldsymbol{v}_t$ at time t. Similar to the existing methods [24, 41], $e^{[\boldsymbol{\omega}_t]_\times}$ and $\boldsymbol{G}_t\boldsymbol{v}_t$ are the residual rotation and translation matrix, respectively, which are derived as:

$$\begin{aligned} e^{[\boldsymbol{\omega}_t]_\times} &= \boldsymbol{I} + \frac{\sin\theta}{\theta}[\boldsymbol{\omega}_t]_\times + \frac{1-\cos\theta}{\theta^2}[\boldsymbol{\omega}_t]_\times^2, \\ \boldsymbol{G}_t &= \boldsymbol{I} + \frac{1-\cos\theta}{\theta^2}[\boldsymbol{\omega}_t]_\times + \frac{\theta-\sin\theta}{\theta^3}[\boldsymbol{\omega}_t]_\times^2, \end{aligned} \tag{9}$$

where $[\mathbf{w}]_\times$ is the cross-product matrix of a vector $\mathbf{w}$ and $\theta = \|\boldsymbol{\omega}_t\|$ is the angle of rotation at time t. We model $\mathcal{S}_t$ as a learnable embedding of time t, as shown in Fig. 2-(a).

3.3.2 Interleave Optimization

Jointly optimizing dynamic radiance fields and camera poses is highly ill-posed and can lead to bad local-minima. To handle this, we propose a novel interleave optimization strategy that alternatively optimizes the ray refinement and Static and Dynamic Nets as described in Algo. 1 with diverse losses in the followings:

Photometric Loss. To optimize the DyBluRF architecture stably, we minimize the photometric loss of our model for the rendered colors. Given an input ray $\boldsymbol{r}_{\boldsymbol{p};t}$ that can be whether $\hat{\boldsymbol{r}}_{\boldsymbol{p};t}$ or $\dot{\boldsymbol{r}}_{\boldsymbol{p};t;q}$ in Sec. 3.4, we render the color $\hat{\boldsymbol{C}}^d(\boldsymbol{r}_{\boldsymbol{p};t})$ of dynamic scene component and the full rendered color $\hat{\boldsymbol{C}}^{full}(\boldsymbol{r}_{\boldsymbol{p};t})$. Then, we calculate the L2 loss between each rendered color $\hat{\boldsymbol{C}}(\boldsymbol{r}_{\boldsymbol{p};t})$ with the blurry GT color $\boldsymbol{\mathcal{B}}_{\boldsymbol{p};t}$ as:

$$\mathcal{L}_{photo}(\hat{\boldsymbol{C}}(\boldsymbol{r}_{\boldsymbol{p};t})) = \sum_{\boldsymbol{r}_{\boldsymbol{p};t}} \|\hat{\boldsymbol{C}}(\boldsymbol{r}_{\boldsymbol{p};t}) - \boldsymbol{\mathcal{B}}_{\boldsymbol{p};t}\|_2^2, \tag{10}$$

where $\hat{\boldsymbol{C}}(\boldsymbol{r}_{\boldsymbol{p};t})$ can be $\hat{\boldsymbol{C}}^d(\boldsymbol{r}_{\boldsymbol{p};t})$ or $\hat{\boldsymbol{C}}^{full}(\boldsymbol{r}_{\boldsymbol{p};t})$. For the rendered color $\hat{\boldsymbol{C}}^s(\boldsymbol{r}_{\boldsymbol{p};t})$ of static scene component, we adopt a masked photometric loss to prevent learning the dynamic scene component by using predicted $\boldsymbol{M}(\boldsymbol{r}_{\boldsymbol{p};t})$ (Eq. 6) as:

$$\begin{aligned} &\mathcal{L}_{mphoto}(\hat{\boldsymbol{C}}^s(\boldsymbol{r}_{\boldsymbol{p};t})) = \\ &\sum_{\boldsymbol{r}_{\boldsymbol{p};t}} \|(\hat{\boldsymbol{C}}^s(\boldsymbol{r}_{\boldsymbol{p};t})) - \boldsymbol{\mathcal{B}}_{\boldsymbol{p};t}) \cdot (1 - \boldsymbol{M}(\boldsymbol{r}_{\boldsymbol{p};t}))\|_2^2. \end{aligned} \tag{11}$$

Surface Normal Loss. DynIBaR [28] applied a pre-trained monocular depth estimation network to regularize NeRF. To mitigate the difficulty of scale-ambiguous depth estimation,

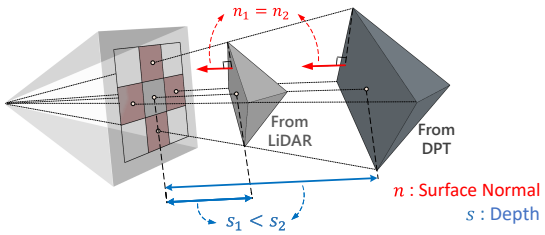


Figure 3. **Surface Normal Loss.** We compute the surface normal by first-order finite differences. As surface normal is invariant to depth scale, we can use the surface normal loss to regularize NeRF without the need of depth alignment.

DynIBaR [28] suggested aligning the pre-trained monocular disparity with the SfM point cloud. However, the SfM reconstruction from the blurry video frames without motion segmentation priors may even produce unreliable results. Different from DynIBaR [28], as shown in Fig. 3, we adopt a surface normal supervision to regularize the density prediction of Dynamic Net F_{θ^d}. We compute the ground truth surface normal $\boldsymbol{n}_{\boldsymbol{p};t}$ of pixel $\boldsymbol{p} = (p_u, p_v)$ at time t from the pre-trained depth estimation model [48] using the first-order finite differences. Then, we calculate the predicted surface normal $\hat{\boldsymbol{n}}_{\boldsymbol{p};t}$ as:

$$\hat{\boldsymbol{n}}_{\boldsymbol{p};t} = \overline{\left(\frac{\partial \boldsymbol{r}_{\boldsymbol{p};t}(s*)}{\partial p_u} \times \frac{\partial \boldsymbol{r}_{\boldsymbol{p};t}(s*)}{\partial p_v}\right)}, \tag{12}$$

where $\overline{(\mathbf{w})}$ is the normalization operation of a vector $\mathbf{w}$ and $s* = \sum_{n=1}^{N} \mathcal{T}_n^d \alpha_n^d s_n^d$ is the rendered ray distance for each ray. Our surface normal loss $\mathcal{L}_{sn}$ is the mean square error between $\hat{\boldsymbol{n}}_{\boldsymbol{p};t}$ and $\boldsymbol{n}_{\boldsymbol{p};t}$ weighted by a constant λ_{sn} as:

$$\mathcal{L}_{sn} = \lambda_{sn} \sum_{\boldsymbol{r}_{\boldsymbol{p};t}} \|\hat{\boldsymbol{n}}_{\boldsymbol{p};t} - \boldsymbol{n}_{\boldsymbol{p};t}\|_2^2 \tag{13}$$

Static Blending Loss. Unlike the existing dynamic NeRF methods [18, 27, 28, 32] which rely on the pre-trained segmentation models to supervise the moving object region, our DyBluRF optimizes the binary motion mask prediction in an *unsupervised* manner. Since the Static and Dynamic Nets exhibit disparate convergence speeds, the scene decomposition often tends to favor the dynamic components. To tackle this issue, we introduce a static blending loss weighted by λ_{sb} as $\mathcal{L}_{sb} = -\lambda_{sb} \sum_{\boldsymbol{r}_{\boldsymbol{p};t}} \log(b)$ to encourage the blending factor b to maintain static characteristics.

Distortion Loss. We adopt the distortion loss $\mathcal{L}_{dist}$ in Mip-NeRF 360 [4]. The distortion loss is applied to the density estimations of static, dynamic and blending renderings, which are denoted as $\mathcal{L}_{dist}^s$, $\mathcal{L}_{dist}^d$ and $\mathcal{L}_{dist}^{full}$, respectively.

3.4. Motion Decomposition-based Deblurring Stage

In the MDD stage, we *first* propose a novel incremental latent sharp-rays prediction (ILSP) approach for the *blurry monocular video frames* as shown in Fig. 2-(b) combined

Algorithm 1 Interleave Ray Refinement Stage (Fig. 2-(a))

procedure IRR(F_{θ^s}, F_{θ^d}, $\mathcal{S}_t$)
  for it $= 0$ to 2×10^5 **do**
    Sample random rays $\tilde{\boldsymbol{r}}_{\boldsymbol{p};t}$
    $\hat{\boldsymbol{r}}_{\boldsymbol{p};t} \leftarrow \mathcal{W}(\tilde{\boldsymbol{r}}_{\boldsymbol{p};t}, \mathcal{S}_t)$ (Eq. 8)
    if it$|2$ **then** ▷ **Static Net** and **Ray Refine.**
      Compute $\hat{\boldsymbol{C}}^s(\hat{\boldsymbol{r}}_{\boldsymbol{p};t})$ (Eq. 1)
      $loss \leftarrow \mathcal{L}_{mphoto}(\hat{\boldsymbol{C}}^s(\hat{\boldsymbol{r}}_{\boldsymbol{p};t})) + \mathcal{L}^s_{dist}$ (Eq. 11)
      Update **Static Net** F_{θ^s}, $\mathcal{S}_t$
    else ▷ **Dynamic Net** and **Static Net**
      Compute $\hat{\boldsymbol{C}}^s(\hat{\boldsymbol{r}}_{\boldsymbol{p};t}), \hat{\boldsymbol{C}}^d(\hat{\boldsymbol{r}}_{\boldsymbol{p};t})$ (Eq. 1)
      Compute $\hat{\boldsymbol{C}}^{full}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t})$ (Eq. 3)
      $loss \leftarrow \mathcal{L}_{mphoto}(\hat{\boldsymbol{C}}^s(\hat{\boldsymbol{r}}_{\boldsymbol{p};t}))$
        $+\mathcal{L}_{photo}(\hat{\boldsymbol{C}}^{d,full}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t}))$
        $+\mathcal{L}^{s,d,full}_{dist} + \mathcal{L}_{sn}$ (Eq. 10, 11, 13)
      Update **Static Net** F_{θ^s}, **Dynamic Net** F_{θ^d}

with Algo. 1, which is fully described in Algo. 2. This approach comprises a global camera-motion-aware ray prediction and a local object-motion-aware ray refinement.

Global Camera-motion-aware Ray Prediction (GCRP). To model the camera motion blur process which occurs in both static and dynamic scene components, we estimate multiple latent sharp rays $\{\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}\}^{N_b}_{q=1}$ (Fig. 2-(b)) based on the refined ray $\hat{\boldsymbol{r}}_{\boldsymbol{p};t}$ (Eq. 8 and Fig. 2-(a)) as:

$$\{\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}\}^{N_b}_{q=1} = \{\mathcal{W}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t}, \mathcal{S}^g_{t;q})\}^{N_b}_{q=1}, \tag{14}$$

where $\mathcal{W}$ is defined in Eq. 8 and $\mathcal{S}^g_{t;q}$ is a global camera-motion-aware screw axis which is a learnable embedding of t for the q-th latent sharp ray $\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}$. The global camera-motion-aware ray prediction (GCRP) maps the refined ray $\hat{\boldsymbol{r}}_{\boldsymbol{p};t}$ that is the output of the IRR stage to N_b predicted latent sharp rays $\{\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}\}^{N_b}_{q=1}$ considering the global camera motion (one-to-N_b mapping).

Local Object-motion-aware Ray Refinement (LORR). If only a single motion-aware ray prediction, i.e., GCRP, is adopted for estimating latent sharp rays, the model tends to learn the outer mixture of diverse motions, combining global camera motion and local object motions. This may result in unnatural artifacts such as afterimage spread over the blurry training images. To delicately handle detailed motions, we further decompose the blurry rays into local object motions along the global camera motion by refining the q-th predicted latent sharp ray $\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}$ considering pixel-wise local object motion as:

$$\dot{\boldsymbol{r}}^l_{\boldsymbol{p};t;q} = \mathcal{W}(\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}, \mathcal{S}^l_{\boldsymbol{p};t;q}), \tag{15}$$

where $\mathcal{S}^l_{\boldsymbol{p};t;q} = F_{\theta^l}(\lceil \phi(\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}), l(t) \rfloor)$ is a local object-motion-aware screw axis learned by a local object-motion MLP F_{θ^l} which takes a discretized ray embedding $\phi(\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q})$ [46] and the encoded time $l(t)$ as inputs. $\lceil \cdot \rfloor$ refers to channel-wise concatenation. The local object-motion-aware ray refinement (LORR) maps each predicted latent sharp ray $\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}$ to a single corresponding refined latent sharp ray $\dot{\boldsymbol{r}}^l_{\boldsymbol{p};t;q}$ considering the local object motion (one-to-one mapping). Specifically, the LORR is only applied to the dynamic scene components which are indicated by the binary motion mask $\boldsymbol{M}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t})$ (=1) as in Line 12 of Algo. 2.

To obtain the blurry color $\hat{\boldsymbol{C}}_{\mathcal{B}}$, we apply Eq. 7 to predicted latent sharp rays from Eq. 14 for the static scene components or from Eq. 15 for the dynamic scene components as described in Algo. 2.

Algorithm 2 Overall DyBluRF Framework (Fig. 2)

Init F_{θ^s}, F_{θ^d}, F_{θ^l}, $\mathcal{S}_t$, $\mathcal{S}^g_{t;q}$
Do IRR(F_{θ^s}, F_{θ^d}, $\mathcal{S}_t$) (Fig. 2-(a) and Algo. 1)
for it $= 0$ to 10^5 **do** ▷ MDD (Sec. 3.4 and Fig. 2-(b))
  Sample random rays $\tilde{\boldsymbol{r}}_{\boldsymbol{p};t}$
  $\hat{\boldsymbol{r}}_{\boldsymbol{p};t} \leftarrow \mathcal{W}(\tilde{\boldsymbol{r}}_{\boldsymbol{p};t}, \mathcal{S}_t)$ (Eq. 8) ▷ Freeze $\mathcal{S}_t$
  Compute $\hat{\boldsymbol{C}}^{s,d,full}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t})$, $\boldsymbol{M}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t})$ (Eq. 1, 3, 6)
  $\{\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}\}^{N_b}_{q=1} \leftarrow \{\mathcal{W}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t}, \mathcal{S}^g_{t;q})\}^{N_b}_{q=1}$ (Eq. 14) ▷ GCRP
  if $\boldsymbol{M}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t}) = 0$ **then** ▷ Only global camera motion
    for $q = 1$ to N_b **do**
      $\dot{\boldsymbol{r}}_{\boldsymbol{p};t;q} \leftarrow \dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}$
      Compute $\hat{\boldsymbol{C}}^{s,d,full}(\dot{\boldsymbol{r}}_{\boldsymbol{p};t;q})$
  else if $\boldsymbol{M}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t}) = 1$ **then** ▷ Combine local object motion
    for $q = 1$ to N_b **do**
      $\mathcal{S}^l_{\boldsymbol{p};t;q} \leftarrow F_{\theta^l}(\lceil \phi(\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}), l(t) \rfloor)$
      $\dot{\boldsymbol{r}}^l_{\boldsymbol{p};t;q} \leftarrow \mathcal{W}(\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}, \mathcal{S}^l_{\boldsymbol{p};t;q})$ (Eq. 15) ▷ LORR
      $\dot{\boldsymbol{r}}_{\boldsymbol{p};t;q} \leftarrow \dot{\boldsymbol{r}}^l_{\boldsymbol{p};t;q}$
      Compute $\hat{\boldsymbol{C}}^{s,d,full}(\dot{\boldsymbol{r}}_{\boldsymbol{p};t;q})$
  $\hat{\boldsymbol{C}}^{s,d,full}_{\mathcal{B}}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t})$
    $\leftarrow \mathcal{A}(\hat{\boldsymbol{C}}^{s,d,full}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t}), \{\hat{\boldsymbol{C}}^{s,d,full}(\dot{\boldsymbol{r}}_{\boldsymbol{p};t;q})\}^{N_b}_{q=1})$ (Eq. 7)
  $loss \leftarrow \mathcal{L}_{mphoto}(\hat{\boldsymbol{C}}^s_{\mathcal{B}}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t})) + \mathcal{L}_{photo}(\hat{\boldsymbol{C}}^{d,full}_{\mathcal{B}}(\hat{\boldsymbol{r}}_{\boldsymbol{p};t}))$
    $+\mathcal{L}^{s,d,full}_{dist} + \mathcal{L}_{sn}$
  Update **Static Net** F_{θ^s}, **Dynamic Net** F_{θ^d}, F_{θ^l}, $\mathcal{S}^g_{t;q}$

4. Experiments

Implementation Details. We refer readers to the *Supplemental* for the details of our implementation.

Blurry iPhone Dataset. To evaluate our DyBluRF for deblurring novel view synthesis when only given blurry frames with imprecise camera poses, we synthesize a new blurry dataset with inaccurate camera information, called Blurry iPhone Dataset. This dataset includes synthetic *blurry* RGB video frames $\{\boldsymbol{\mathcal{B}}_t\}^{N_f}_{t=1}$ with *inaccurate* camera pose information $\{\tilde{\boldsymbol{\mathcal{P}}}_t\}^{N_f}_{t=1}$ for the training dataset, and the original *sharp* RGB video frames with *accurate* camera pose information for the evaluation dataset, derived from the iPhone dataset [19]. To generate these blurry RGB training video frames, we firstly employ a VFI method [53] to interpolate seven intermediate frames ($\times 8$) between each pair of adjacent sharp video frames from the origi-

Method	Apple	Block	Paper-windmill	Space-out
TiNeuVox [15]	13.53 / 0.680 / 0.723 / 1.704	10.79 / 0.558 / 0.676 / 1.705	14.15 / 0.273 / 0.781 / 4.108	14.18 / 0.557 / 0.587 / 1.385
HexPlane [7]	16.80 / 0.715 / 0.523 / 1.239	15.58 / 0.604 / 0.459 / 0.820	17.11 / 0.352 / 0.422 / 0.318	14.73 / 0.558 / 0.511 / 1.270
T-NeRF [35]	17.34 / 0.720 / 0.547 / 0.620	16.48 / 0.644 / 0.411 / 0.795	16.83 / 0.338 / 0.424 / 0.569	16.23 / 0.561 / 0.436 / 1.329
HyperNeRF [42]	14.31 / 0.681 / 0.663 / 1.411	16.12 / 0.642 / 0.416 / 0.958	16.59 / 0.335 / 0.365 / 0.666	17.79 / 0.631 / 0.332 / 0.402
DP-NeRF [24]	11.97 / 0.665 / 0.717 / 2.072	9.96 / 0.514 / 0.729 / 1.602	12.66 / 0.241 / 0.713 / 1.482	13.15 / 0.532 / 0.628 / 0.639
BAD-NeRF [59]	12.29 / 0.668 / 0.744 / 1.743	9.61 / 0.517 / 0.736 / 1.290	5.98 / 0.033 / 0.961 / 0.978	12.57 / 0.508 / 0.643 / 0.437
DyBluRF (Ours)	**18.03 / 0.737 / 0.505 / 0.479**	**17.35 / 0.660 / 0.361 / 0.735**	**18.24 / 0.410 / 0.300 / 0.242**	**18.99 / 0.646 / 0.328 / 0.276**

Method	Spin	Teddy	Wheel	**Average**
TiNeuVox [15]	11.13 / 0.411 / 0.726 / 2.239	10.28 / 0.496 / 0.834 / 1.304	9.48 / 0.312 / 0.717 / 3.556	11.93 / 0.470 / 0.721 / 2.286
HexPlane [7]	16.02 / 0.482 / 0.563 / 1.253	12.84 / 0.497 / 0.587 / 1.220	12.87 / 0.409 / 0.521 / 1.336	15.14 / 0.517 / 0.512 / 1.065
T-NeRF [35]	17.16 / 0.503 / 0.534 / 1.162	14.07 / 0.562 / 0.464 / 1.094	14.93 / 0.499 / 0.379 / 1.360	16.15 / 0.547 / 0.456 / 0.990
HyperNeRF [42]	16.39 / 0.498 / 0.499 / 1.277	13.77 / 0.567 / **0.420** / 1.143	12.11 / 0.393 / 0.435 / 1.739	15.30 / 0.535 / 0.447 / 1.085
DP-NeRF [24]	10.65 / 0.404 / 0.730 / 1.956	10.40 / 0.480 / 0.760 / 1.482	9.26 / 0.299 / 0.736 / 2.561	11.15 / 0.448 / 0.716 / 1.685
BAD-NeRF [59]	10.59 / 0.404 / 0.741 / 1.722	9.77 / 0.457 / 0.758 / 1.537	9.23 / 0.303 / 0.748 / 2.544	10.00 / 0.413 / 0.762 / 1.464
DyBluRF (Ours)	**18.20 / 0.534 / 0.415 / 1.011**	**14.61 / 0.569** / 0.451 / **0.855**	**15.09 / 0.501 / 0.316 / 1.095**	**17.22 / 0.580 / 0.382 / 0.670**

Table 1. **Dynamic deblurring novel view synthesis evaluation on Blurry iPhone Dataset.** **Red** and blue denote the best and second best performances, respectively. Each block element of 4-performance denotes (**mPSNR↑ / mSSIM↑ / mLPIPS↓ / tOF↓**).

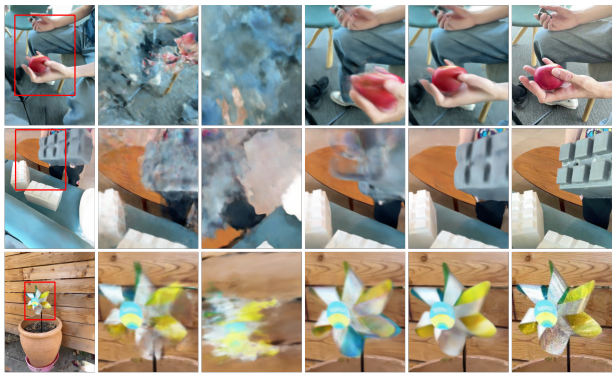


Figure 4. **Qualitative comparisons on Blurry iPhone Dataset.**

nal iPhone dataset [19]. Following that, we average seven consecutive frames with an eight-frame stride, incorporating the original frame and three interpolated frames from both preceding and succeeding moments. This process results in the creation of each blurry RGB frame, assuming the camera motion is in constant velocity within exposure time [36, 52, 59]. To calculate the corresponding camera rotation for each blurry frame, we utilize the spherical linear interpolation [45] to generate interpolated rotation matrices for each interpolated frame. These matrices are then converted into quaternions for averaging the corresponding quaternions of the averaged RGB frames. The resulting averaged quaternions are then reconverted into rotation matrices. To calculate the corresponding camera translation, we linearly interpolate the original translation vector and average the interpolated translation vectors.

Metrics. To evaluate our DyBluRF and compare it with other SOTA methods in the monocular video settings, we utilize the co-visibility masked image metrics, including mPSNR, mSSIM, and mLPIPs, following the approach introduced by Dycheck [19]. These metrics mask out the regions of the test video frames which are not observed by the training camera. We further utilize tOF [10] to measure the temporal consistency of reconstructed video frames.

4.1. Comparisons with State-of-the-Art Methods

To validate the quality of deblurring monocular video novel view synthesis of our DyBluRF, we compare it with the existing dynamic novel view synthesis methods including TiNeuVox [15], HexPlane [7], T-NeRF [35] and HyperNeRF [42] as well as the existing deblurring novel view synthesis methods DP-NeRF [24] and BAD-NeRF [59]. All methods are optimized using the newly synthesized Blurry iPhone Dataset. For the existing deblurring novel view synthesis methods [24, 59] which are originally designed solely for static novel view synthesis, we incorporate time instances as additional inputs to make them synthesize dynamic components for a fair comparison. Table 1 shows quantitative comparison for various methods. As shown, our DyBluRF significantly outperforms the existing SOTA methods for all metrics. Specifically, our model shows significantly better structural quality via deep metrics, e.g. perceptual quality (mLPIPS) and temporal consistency (tOF), than other methods. Figs. 1 and 4 show the superior visualization of our DyBluRF compared to the SOTA methods.

To further demonstrate the effectiveness of the video deblurring ability of DyBluRF, we compare the dynamic novel view synthesis results of our DyBluRF with the existing monocular video NeRF methods [27, 32, 35, 41, 42] which are optimized using the sharp video frames and accurate camera poses, in Table 2. It should be noted that

Method	Trained w/	mPSNR↑	mSSIM↑	mLPIPS↓	tOF↓
NSFF [27]	Sharp	15.46	0.569	0.396	-
Nerfies [41]	Sharp	16.45	0.569	0.339	0.961
T-NeRF [35]	Sharp	16.96	0.577	0.379	0.843
HyperNeRF [42]	Sharp	16.81	0.550	**0.332**	0.869
RoDynRF [32]	Sharp	17.09	0.534	0.517	0.979
DyBluRF (Ours)	Blurry	**17.22**	**0.580**	0.382	**0.670**

Table 2. **Dynamic novel view synthesis evaluation**. Our DyBluRF trained with the Blurry iPhone Dataset ('Trained w/ Blurry') with inaccurate camera poses even shows comparable results with the SOTA methods trained with the original iPhone Dataset [19] ('Trained w/ Sharp') with accurate camera poses.

we train our DyBluRF with the new Blurry iPhone dataset, whereas other dynamic novel view synthesis methods are trained with the original iPhone dataset [19]. As shown in Table 2, our DyBluRF trained on the *blurry* dataset of *inaccurate* poses achieves comparable results, even rivaling SOTA methods trained on *sharp* datasets with *accurate* camera pose information.

4.2. Ablation Study

We conduct an ablation study for three components including the IRR stage, the surface normal loss $\mathcal{L}_{sn}$ and the MDD stage. Table 3 presents detailed quantitative results for the average performance across all seven scenes.

IRR Stage. The effectiveness of our IRR stage (Sec. 3.3 and Fig. 2-(a)) can be observed by comparing variant (c) to variants (a) and (b) in Table 3. Pose optimization with the IRR stage is crucial for effectively handling novel view synthesis with deblurring, especially when trained with the given blurry frames that come with inaccurate pose information. It is worth noting that naively adopting pose optimization [61] (b) for the given blurry frames, i.e., without interleave optimization, results in a more noisy radiance field compared to having no optimization at all (a).

Surface Normal Loss $\mathcal{L}_{sn}$. Table 3 also emphasizes the improvement of surface normal loss (e) (Sec. 3.3.2) compared to the common depth supervision (d) for geometry regularizing NeRF. Since the pre-trained monocular depth estimator is scale-ambiguous, it is difficult to match the scale of the pre-trained depth maps and the radiance fields. Although we apply median scaling to the depth ground truth before supervision for stability, variant (d) trained with depth supervision still produces unreasonable rendering results. Finally, the variant trained with $\mathcal{L}_{sn}$ (e) exhibits better results in terms of mPSNR and mSSIM compared to variant (c) without any geometry regularization. Visual analysis for our $\mathcal{L}_{sn}$ compared to the depth supervision is provided in *Supplemental*.

MDD Stage. By comparing the final DyBluRF (f) to variant (e), the MDD stage (Sec. 3.4 and Fig. 2-(b)) especially increases perceptual quality (mLPIPS). In addition, Fig. 5

Variant	IRR	$\mathcal{L}_{sn}$	MDD	mPSNR↑	mSSIM↑	mLPIPS↓	tOF↓
(a)	-	-	-	16.15	0.547	0.456	0.990
(b)	$\mathcal{N}$	-	-	16.06	0.545	0.449	0.806
(c)	✓	-	-	17.00	0.580	0.414	0.662
(d)	✓	$\mathcal{D}$	-	8.92	0.376	0.768	1.244
(e)	✓	✓	-	17.19	**0.584**	0.414	**0.653**
(f)	✓	✓	✓	**17.22**	0.580	**0.382**	0.670

Table 3. **Ablation Study**. The '$\mathcal{N}$' indicates the naive pose optimization [61] and the '$\mathcal{D}$' indicates the depth supervision [48]. Variant (f) denotes our final DyBluRF.

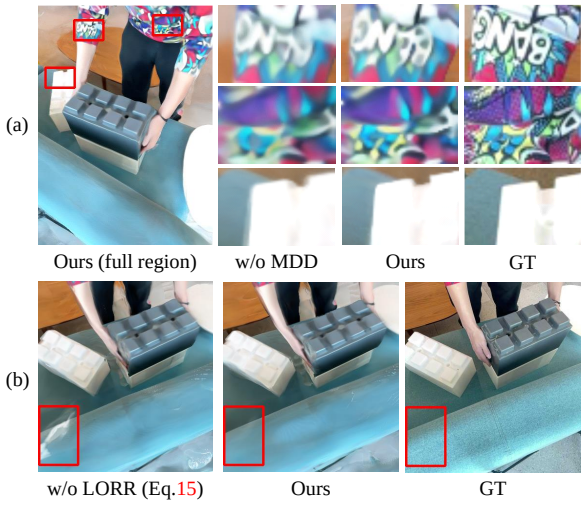


Figure 5. **Ablation Study on the MDD stage.** (a) The effectiveness of MDD stage (Fig. 2-(b)). (b) The effectiveness of decomposition of both global (GCRP) and local (LORR) motions-aware ray predictions.

shows the visual comparison results of the ablation study on our deblurring approach for the 'Block' scene. Specifically, Fig. 5-(a) demonstrates the effectiveness of our final DyBluRF in terms of novel view synthesis for video deblurring, attributed to the MDD stage. The MDD stage effectively handles the blurriness at the edges of objects or the regions of complex textures. In addition, as shown in Fig. 5-(b), our model better delicately decomposes the mixture of global camera motion and local object (i.e. white brick) motions across the training time indices. This results in robust novel view synthesis for a region where the object passes during training, compared to our variant not trained with LORR (Eq. 15).

5. Conclusion

We *first* propose a novel dynamic deblurring NeRF for blurry monocular video, called DyBluRF, which can effectively render the sharp novel spatio-temporal views from blurry monocular frames with inaccurate camera poses. DyBluRF includes an Interleave Ray Refinement (IRR) stage and a Motion Decomposition-based Deblurring (MDD) stage. The IRR stage simultaneously reconstructs dynamic

3D scenes and optimizes inaccurate camera pose information. The MDD stage introduces a novel Incremental Latent Sharp Rays Prediction (ILSP) approach to decompose latent sharp rays into global camera motion and local object motion components. Extensive experimental results demonstrate that DyBluRF outperforms recent state-of-the-art methods both qualitatively and quantitatively.

References

[1] ShahRukh Athar, Zexiang Xu, Kalyan Sunkavalli, Eli Shechtman, and Zhixin Shu. Rignerf: Fully controllable neural 3d portraits. In *CVPR*, 2022. 2

[2] Benjamin Attal, Jia-Bin Huang, Christian Richardt, Michael Zollhoefer, Johannes Kopf, Matthew O'Toole, and Changil Kim. Hyperreel: High-fidelity 6-dof video with ray-conditioned sampling. In *CVPR*, pages 16610–16620, 2023. 2

[3] Yuval Bahat, Netalee Efrat, and Michal Irani. Non-uniform blind deblurring by reblurring. In *ICCV*, pages 3286–3294, 2017. 2

[4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, pages 5470–5479, 2022. 5

[5] Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. Optimizing the latent space of generative networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, 2018. 3

[6] Michael Broxton, John Flynn, Ryan Overbeck, Daniel Erickson, Peter Hedman, Matthew Duvall, Jason Dourgarian, Jay Busch, Matt Whalen, and Paul Debevec. Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics (TOG)*, 39(4):86–1, 2020. 2

[7] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *CVPR*, pages 130–141, 2023. 1, 2, 3, 7, 8, 9, 10

[8] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *ECCV*, pages 333–350. Springer, 2022. 2

[9] Shin-Fang Chng, Sameera Ramasinghe, Jamie Sherrah, and Simon Lucey. Garf: gaussian activated radiance fields for high fidelity reconstruction and pose estimation. *arXiv e-prints*, pages arXiv–2204, 2022. 3

[10] Mengyu Chu, You Xie, Jonas Mayer, Laura Leal-Taixé, and Nils Thuerey. Learning temporal coherence via self-supervision for gan-based video generation. *ACM Transactions on Graphics (TOG)*, 39(4):75–1, 2020. 7

[11] Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. High-quality streamable free-viewpoint video. *ACM Transactions on Graphics (ToG)*, 34(4):1–13, 2015. 2

[12] Senyou Deng, Wenqi Ren, Yanyang Yan, Tao Wang, Fenglong Song, and Xiaochun Cao. Multi-scale separable network for ultra-high-definition video deblurring. In *ICCV*, pages 14030–14039, 2021. 2

[13] Robert A Drebin, Loren Carpenter, and Pat Hanrahan. Volume rendering. *ACM Siggraph Computer Graphics*, 22(4): 65–74, 1988. 2, 3

[14] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *ICCV*, pages 14304–14314. IEEE Computer Society, 2021. 2

[15] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–9, 2022. 2, 3, 7, 1

[16] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, pages 5501–5510, 2022. 2

[17] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *CVPR*, pages 12479–12488, 2023. 2

[18] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *ICCV*, pages 5712–5721, 2021. 2, 3, 4, 5

[19] Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Dynamic novel-view synthesis: A reality check. In *NeurIPS*, 2022. 2, 6, 7, 8, 1, 5

[20] Ankit Gupta, Neel Joshi, C Lawrence Zitnick, Michael Cohen, and Brian Curless. Single image deblurring using motion density functions. In *ECCV*, pages 171–184. Springer, 2010. 2

[21] Stefan Harmeling, Hirsch Michael, and Bernhard Schölkopf. Space-variant single-image blind deconvolution for removing camera shake. *NeurIPS*, 23:829–837, 2010. 2

[22] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. Neuman: Neural human radiance field from a single video. In *ECCV*, 2022. 2

[23] Yifan Jiang, Peter Hedman, Ben Mildenhall, Dejia Xu, Jonathan T Barron, Zhangyang Wang, and Tianfan Xue. Alignerf: High-fidelity neural radiance fields via alignment-aware training. In *CVPR*, pages 46–55, 2023. 3

[24] Dogyoon Lee, Minhyeok Lee, Chajin Shin, and Sangyoun Lee. Dp-nerf: Deblurred neural radiance field with physical scene priors. In *CVPR*, pages 12386–12396, 2023. 2, 3, 4, 5, 7, 1, 8, 9, 10

[25] Dongwoo Lee, Jeongtaek Oh, Jaesung Rim, Sunghyun Cho, and Kyoung Mu Lee. Exblurf: Efficient radiance fields for extreme motion blurred images. In *ICCV*, pages 17639–17648, 2023. 2, 3

[26] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, and Zhaoyang Lv. Neural 3d video synthesis from multi-view video. In *CVPR*, pages 5521–5531, 2022. 2, 4

[27] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *CVPR*, pages 6498–6508, 2021. 2, 3, 5, 7, 8

[28] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering. In *CVPR*, 2023. 2, 3, 4, 5

[29] Jingyun Liang, Yuchen Fan, Xiaoyu Xiang, Rakesh Ranjan, Eddy Ilg, Simon Green, Jiezhang Cao, Kai Zhang, Radu Timofte, and Luc Van Gool. Recurrent video restoration transformer with guided deformable attention. In *NeurIPS*, 2022. 2

[30] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *ICCV*, pages 5741–5751, 2021. 2, 3

[31] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-perfect structure-from-motion with featuremetric refinement. In *ICCV*, pages 5987–5997, 2021. 3

[32] Yu-Lun Liu, Chen Gao, Andreas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust dynamic radiance fields. In *CVPR*, pages 13–23, 2023. 2, 3, 5, 7, 8

[33] 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 *CVPR*, pages 12861–12870, 2022. 2, 3, 4

[34] Quan Meng, Anpei Chen, Haimin Luo, Minye Wu, Hao Su, Lan Xu, Xuming He, and Jingyi Yu. Gnerf: Gan-based neural radiance field without posed camera. In *ICCV*, pages 6351–6361, 2021. 3

[35] 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 *ECCV*, 2020. 2, 3, 7, 8, 1, 5

[36] Jihyong Oh and Munchurl Kim. Demfi: deep joint deblurring and multi-frame interpolation with flow-guided attentive correlation and recursive boosting. In *ECCV*, pages 198–215. Springer, 2022. 2, 7

[37] Martin Ralf Oswald, Jan Stühmer, and Daniel Cremers. Generalized connectivity constraints for spatio-temporal 3d reconstruction. In *ECCV*, pages 32–46. Springer, 2014. 2

[38] Jinshan Pan, Deqing Sun, Hanspeter Pfister, and Ming-Hsuan Yang. Blind image deblurring using dark channel prior. In *CVPR*, pages 1628–1636, 2016. 2

[39] Jinshan Pan, Haoran Bai, and Jinhui Tang. Cascaded deep video deblurring using temporal sharpness prior. In *CVPR*, pages 3043–3051, 2020. 2

[40] Jinshan Pan, Boming Xu, Jiangxin Dong, Jianjun Ge, and Jinhui Tang. Deep discriminative spatial and temporal network for efficient video deblurring. In *CVPR*, pages 22191–22200, 2023. 2

[41] 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 *ICCV*, pages 5865–5874, 2021. 2, 5, 7, 8

[42] 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. *ACM Trans. Graph.*, 40(6), 2021. 1, 2, 3, 7, 8, 5, 9, 10

[43] Keunhong Park, Philipp Henzler, Ben Mildenhall, Jonathan T. Barron, and Ricardo Martin-Brualla. Camp: Camera preconditioning for neural radiance fields. *CoRR*, 2023. 2, 3

[44] Sungheon Park, Minjung Son, Seokhwan Jang, Young Chun Ahn, Ji-Yeon Kim, and Nahyup Kang. Temporal interpolation is all you need for dynamic neural radiance fields. In *CVPR*, pages 4212–4221, 2023. 2

[45] Xavier Pennec. *Computing the mean of geometric features application to the mean rotation*. PhD thesis, INRIA, 1998. 7, 1

[46] Martin Piala and Ronald Clark. Terminerf: Ray termination prediction for efficient neural rendering. In *3DV*, pages 1106–1114, 2021. 6

[47] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *CVPR*, pages 10318–10327, 2021. 2, 3

[48] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, pages 12159–12168, 2021. 5, 8

[49] Vincent Raoult, Sarah Reid-Anderson, Andreas Ferri, and Jane E Williamson. How reliable is structure from motion (sfm) over time and between observers? a case study using coral reef bommies. *Remote Sensing*, 9(7):740, 2017. 3

[50] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, pages 4104–4113, 2016. 3

[51] Ruizhi Shao, Zerong Zheng, Hanzhang Tu, Boning Liu, Hongwen Zhang, and Yebin Liu. Tensor4d: Efficient neural 4d decomposition for high-fidelity dynamic reconstruction and rendering. In *CVPR*, pages 16632–16642, 2023. 2

[52] Wang Shen, Wenbo Bao, Guangtao Zhai, Li Chen, Xiongkuo Min, and Zhiyong Gao. Blurry video frame interpolation. In *CVPR*, pages 5114–5123, 2020. 7

[53] Hyeonjun Sim, Jihyong Oh, and Munchurl Kim. Xvfi: extreme video frame interpolation. In *ICCV*, pages 14489–14498, 2021. 6, 1

[54] Liangchen Song, Anpei Chen, Zhong Li, Zhang Chen, Lele Chen, Junsong Yuan, Yi Xu, and Andreas Geiger. Nerfplayer: A streamable dynamic scene representation with decomposed neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2732–2742, 2023. 2

[55] Jacob Telleen, Anne Sullivan, Jerry Yee, Oliver Wang, Prabath Gunawardane, Ian Collins, and James Davis. Synthetic shutter speed imaging. In *Computer Graphics Forum*, pages 591–598. Wiley Online Library, 2007. 2

[56] 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 *ICCV*, pages 12959–12970, 2021. 2

[57] Basile Van Hoorick, Purva Tendulkar, Didac Suris, Dennis Park, Simon Stent, and Carl Vondrick. Revealing occlusions with 4d neural fields. In *CVPR*, pages 3011–3021, 2022. 2

[58] Liao Wang, Jiakai Zhang, Xinhang Liu, Fuqiang Zhao, Yanshun Zhang, Yingliang Zhang, Minye Wu, Jingyi Yu, and

Lan Xu. Fourier plenoctrees for dynamic radiance field rendering in real-time. In *CVPR*, pages 13524–13534, 2022. 2

[59] Peng Wang, Lingzhe Zhao, Ruijie Ma, and Peidong Liu. Bad-nerf: Bundle adjusted deblur neural radiance fields. In *CVPR*, pages 4170–4179, 2023. 1, 2, 3, 4, 7

[60] Yusheng Wang, Yunfan Lu, Ye Gao, Lin Wang, Zhihang Zhong, Yinqiang Zheng, and Atsushi Yamashita. Efficient video deblurring guided by motion magnitude. In *ECCV*, 2022. 2

[61] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. Nerf-: Neural radiance fields without known camera parameters. *CoRR*, 2021. 2, 3, 8

[62] Chung-Yi Weng, Brian Curless, Pratul P Srinivasan, Jonathan T Barron, and Ira Kemelmacher-Shlizerman. Humannerf: Free-viewpoint rendering of moving people from monocular video. In *CVPR*, pages 16210–16220, 2022. 2

[63] Chung-Yi Weng, Brian Curless, Pratul P. Srinivasan, Jonathan T. Barron, and Ira Kemelmacher-Shlizerman. HumanNeRF: Free-viewpoint rendering of moving people from monocular video. In *CVPR*, pages 16210–16220, 2022. 2

[64] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *CVPR*, pages 9421–9431, 2021. 2

[65] Gengshan Yang, Minh Vo, Neverova Natalia, Deva Ramanan, Vedaldi Andrea, and Joo Hanbyul. Banmo: Building animatable 3d neural models from many casual videos. In *CVPR*, 2022. 2

[66] Huicong Zhang, Haozhe Xie, and Hongxun Yao. Spatio-temporal deformable attention network for video deblurring. In *ECCV*, 2022. 2

[67] Kaihao Zhang, Wenhan Luo, Yiran Zhong, Lin Ma, Wei Liu, and Hongdong Li. Adversarial spatio-temporal learning for video deblurring. *IEEE Transactions on Image Processing*, 28(1):291–301, 2018. 2

[68] Kaihao Zhang, Wenhan Luo, Yiran Zhong, Lin Ma, Bjorn Stenger, Wei Liu, and Hongdong Li. Deblurring by realistic blurring. In *CVPR*, pages 2737–2746, 2020. 2

[69] Li Zhang, Brian Curless, and Steven M Seitz. Spacetime stereo: Shape recovery for dynamic scenes. In *CVPR*, pages II–367. IEEE, 2003. 2

[70] Shangchen Zhou, Jiawei Zhang, Jinshan Pan, Haozhe Xie, Wangmeng Zuo, and Jimmy Ren. Spatio-temporal filter adaptive network for video deblurring. In *ICCV*, pages 2482–2491, 2019. 2

[71] Qi Zhu, Man Zhou, Naishan Zheng, Chongyi Li, Jie Huang, and Feng Zhao. Exploring temporal frequency spectrum in deep video deblurring. In *ICCV*, pages 12428–12437, 2023. 2

[72] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM transactions on graphics (TOG)*, 23(3):600–608, 2004. 2

DyBluRF: Dynamic Deblurring Neural Radiance Fields for Blurry Monocular Video

Supplementary Material

A. List of Notations

Table 4 provides an organized list of notations used in the main paper, designed to improve readers' comprehension of our text and equations.

B. Implementation Details

Our DyBluRF is implemented using JAX and built upon Dycheck [19] codebase. Similar to Dycheck [19], we adopt a coarse-to-fine sampling strategy for NeRF, employing $N = 128$ samples at each sampling step. Our model undergoes training for 2×10^5 iterations in the IRR stage and 1×10^5 in the MDD stage. The hyperparameters for the loss weighting coefficients, denoted as λ_{sn}, λ_{sb} and λ_{dist}, are empirically set to 0.075, 0.002 and 0.01, respectively. For all scenes, our best model is configured with a fixed number of latent sharp rays $N_b = 6$. We utilize the Adam optimizer with exponential scheduling for training our networks. Notably, the learning rates for the screw axis embeddings $\mathcal{S}_t$ and $\mathcal{S}^g_{t;q}$ are in the range $[10^{-4}, 10^{-6}]$, while the learning rates for the remaining MLPs are in the range $[10^{-3}, 10^{-4}]$. For the implementation of the Static Net F_{θ^s} and Dynamic Net F_{θ^d} of the main paper, we adopt the NeRF architecture from the Dycheck [19] codebase to ensure a fair comparison. This NeRF architecture comprises a trunk MLP, a sigma MLP and an rgb MLP. The sigma and rgb MLPs take the output feature of the trunk MLP as their respective inputs. We configure the number of layers for the trunk, sigma, and rgb MLPs to be 9, 1, and 2, respectively. The dimensions of the hidden layer for the trunk and rgb MLPs are set to 256 and 128, respectively. For the Static Net, we introduce a blending MLP that consists of a single layer. This MLP takes the output of the trunk MLP as input to produce the volume density blending factor b. For the local object-motion MLP F_{θ^l}, we set the number of layers and dimension of hidden layers to 8 and 128, respectively. We set the number of frequencies for each positional encoding γ of 3D sample positions $\mathbf{x}$ and their corresponding viewing directions $\mathbf{d}$ to 8 and 4, respectively. We also set the output dimension of the time encoding $l(t)$ to 8 and the number of samples for the discretized ray embedding ϕ to 32. On the other hand, we also recommend readers to refer to our project page at https://kaist-viclab.github.io/dyblurf-site/, which provides source codes and pretrained models for the sake of *reproducibility*.

C. Details for Blurry iPhone Dataset Synthesis

We present the visual process of synthesizing our proposed Blurry iPhone Dataset, which includes blurry RGB frames and inaccurate camera poses, in Fig. 6. The synthesis of the blurry video frame $\boldsymbol{\mathcal{B}}_t$ from the original iPhone Dataset [19] is illustrated in Fig. 6-(a). We employ the VFI [53] method to first generate seven interpolated frames between each pair of original video frames ($\boldsymbol{\mathcal{I}}_{t-1}$ and $\boldsymbol{\mathcal{I}}_t$, $\boldsymbol{\mathcal{I}}_t$ and $\boldsymbol{\mathcal{I}}_{t+1}$). Subsequently, we select seven frames ($\boldsymbol{\mathcal{I}}_{t-\frac{5}{8}}$ to $\boldsymbol{\mathcal{I}}_{t+\frac{3}{8}}$) and average them. To simulate the blurry camera pose information $\tilde{\boldsymbol{\mathcal{P}}}_t = [\tilde{\boldsymbol{\mathcal{R}}}_t | \tilde{\boldsymbol{\mathcal{T}}}_t]$ for our Blurry iPhone Dataset, we maintain the same interpolation concept for the camera pose matrices to reflect the camera motion trajectory, as shown in Fig. 6-(b). For the rotation matrix $\tilde{\boldsymbol{\mathcal{R}}}_t$, we use the Spherical Linear Interpolation (Quaternion Interpolation) [45] method to generate interpolated rotation matrices between each pair of existing rotation matrices. We then choose the seven rotation matrices ($\boldsymbol{\mathcal{R}}_{t-\frac{5}{8}}$ to $\boldsymbol{\mathcal{R}}_{t+\frac{3}{8}}$) and average them. Since directly calculating the average rotation matrix from multiple rotation matrices is challenging, we convert the matrices into quaternions and average these quaternions. For the translation matrix $\tilde{\boldsymbol{\mathcal{T}}}_t$, we adopt linear interpolation to generate interpolated matrices and average the seven matrices ($\boldsymbol{\mathcal{T}}_{t-\frac{5}{8}}$ to $\boldsymbol{\mathcal{T}}_{t+\frac{3}{8}}$). The resulting interpolated camera pose $\tilde{\boldsymbol{\mathcal{P}}}_t$ has different rotation and translation matrices with the accurate camera pose $\boldsymbol{\mathcal{P}}_t$, making it more challenging for existing methods [7, 15, 24, 35, 42, 59] to reconstruct the correct geometry of a scene, except for our DyBluRF.

D. Details for Blurry Color Rendering Process with Latent Sharp Rays

We describe the blurry color rendering process of the MDD stage (Sec. 3.4) in Fig. 7. In Fig. 7-(a), the predicted ray $\hat{\boldsymbol{r}}_{\boldsymbol{p};t}$ and the latent sharp rays $\{\hat{\boldsymbol{r}}_{\boldsymbol{p};t;q}\}_{q=1}^{N_b}$ render the corresponding sharp RGB colors separately. We average the individual sharp RGB colors to predict the blurry RGB color for the photometric loss function. As shown in Fig. 7-(b), our predicted latent sharp frames are surprisingly learned as latent sharp rays by naturally following a blurry process by Eq. 7, where zooming results with red boxes and smaller green and blue boxes in the different latent sharp frames show detailed local object motions to synthesize the given blurry frame. These visualization results support our DyBluRF's superior performance and reliability in dynamic deblurring.

Notation	Description
$\boldsymbol{\mathcal{I}}_t$	A sharp frame at time t of a monocular video
$\boldsymbol{\mathcal{I}}_{\boldsymbol{p};t}$	A sharp pixel color of pixel $\boldsymbol{p}$ at time t of a monocular video
$\boldsymbol{\mathcal{B}}_t$	A blurry frame at time t of a monocular video
$\boldsymbol{\mathcal{B}}_{\boldsymbol{p};t}$	A blurry pixel color of pixel $\boldsymbol{p}$ at time t of a monocular video
$\boldsymbol{\mathcal{P}}_t$	A camera pose information at time t
$\tilde{\boldsymbol{\mathcal{P}}}_t$	An inaccurate camera pose information at time t
N_f	The number of frames in a video
$\mathbf{o}_t$	An origin of a ray at time t
$\mathbf{d}_{\boldsymbol{p};t}$	A viewing direction of a ray of pixel $\boldsymbol{p}$ at time t
s	A sampling ray distance
$\boldsymbol{r}_{\boldsymbol{p};t}$	A casted ray from $\mathbf{o}_t$ with $\mathbf{d}_{\boldsymbol{p};t}$
F_{θ^s}	Static Net
F_{θ^d}	Dynamic Net
$\boldsymbol{c}^s$ / $\boldsymbol{c}^d$	An estimated color by Static/Dynamic Net
σ^s / σ^d	An estimated volume density by Static/Dynamic Net
b	A volume density blending factor
$\hat{\boldsymbol{C}}^{s,d,full}$	A volume rendered color for static, dynamic or full scene component, respectively
$\boldsymbol{\mathcal{T}}^{s,d,full}$	An accumulated transmittance for static, dynamic or full scene component, respectively
α^s / α^d	An alpha-compositing weight for static/dynamic scene component
$\boldsymbol{M}$	A predicted binary motion mask
$\hat{\boldsymbol{C}}_{\mathcal{B}}$	A blurry rendered color
$\mathcal{A}$	An average function
$\tilde{\boldsymbol{r}}_{\boldsymbol{p};t}$	An inaccurate ray from $\boldsymbol{p}$ with $\tilde{\boldsymbol{\mathcal{P}}}_t$
$\hat{\boldsymbol{r}}_{\boldsymbol{p};t}$	An refined ray from $\tilde{\boldsymbol{r}}_{\boldsymbol{p};t}$
$\dot{\boldsymbol{r}}_{\boldsymbol{p};t;q}$	The q-th predicted latent sharp ray to generate the blurry pixel color $\hat{\boldsymbol{C}}_{\mathcal{B}}$ which can be $\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}$ or $\dot{\boldsymbol{r}}^l_{\boldsymbol{p};t;q}$
$\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}$	The q-th predicted latent sharp ray to generate the blurry pixel color of static scene component by GCRP (Eq. 14
$\dot{\boldsymbol{r}}^l_{\boldsymbol{p};t;q}$	The q-th predicted latent sharp ray to generate the blurry pixel color of dynamic scene component by LORR (Eq. 15)
N_b	The number of latent sharp rays
$\mathcal{S}_t$	A learnable embedding of screw axis for ray refinement of $\tilde{\boldsymbol{r}}_{\boldsymbol{p};t}$
$\mathcal{S}^g_{t;q}$	A learnable embedding of global camera-motion-aware screw axis for prediction of ray $\dot{\boldsymbol{r}}^g_{\boldsymbol{p};t;q}$
$\mathcal{S}^l_{\boldsymbol{p};t;q}$	A local object-motion-aware screw axis for refinement of ray $\dot{\boldsymbol{r}}^l_{\boldsymbol{p};t;q}$
F_{θ^l}	A local object-motion-aware MLP
$\gamma(\cdot)$	A positional encoding
$l(\cdot)$	Generative Latent Optimization
$\phi(\cdot)$	A discretized ray embedding
$\boldsymbol{n}_{\boldsymbol{p};t}$	A ground-truth surface normal computed from DPT's depth estimation
$\hat{\boldsymbol{n}}_{\boldsymbol{p};t}$	A predicted surface normal computed from volume density
$\mathcal{L}_{photo}$ / $\mathcal{L}_{mphoto}$	A photometric/masked photometric loss function
$\mathcal{L}_{dist}$	A distortion loss function
$\mathcal{L}_{sn}$	A surface normal loss function

Table 4. **List of Notations.**

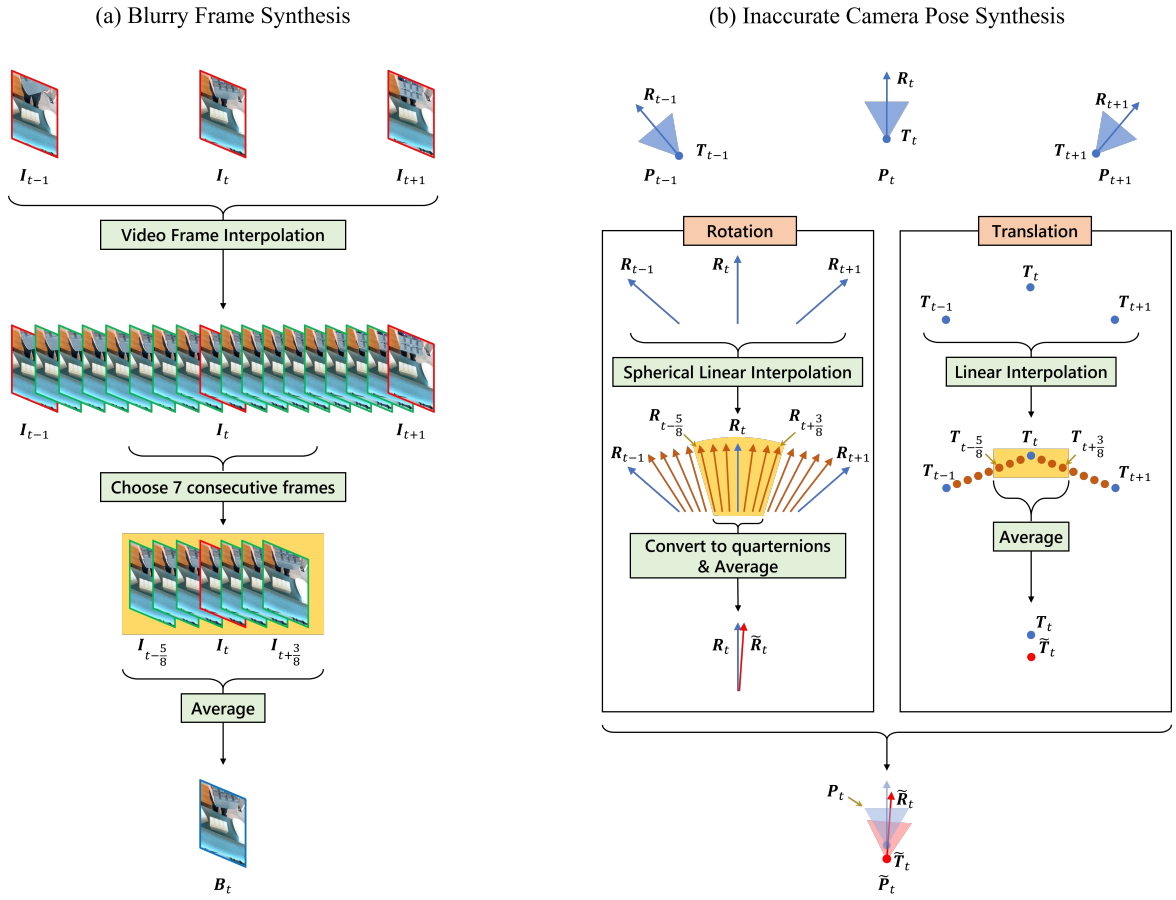


Figure 6. **Simulation of Blurry iPhone Dataset.** (a) Simulation of blurry video frames. (b) Simulation of inaccurate camera poses. Please note that we just visualize arrows for the rotation matrices and dots for the translation matrices in (b) for simplicity.

E. Additional Experimental Results

E.1. Robustness of our DyBluRF on Two Different Qualities of Datasets.

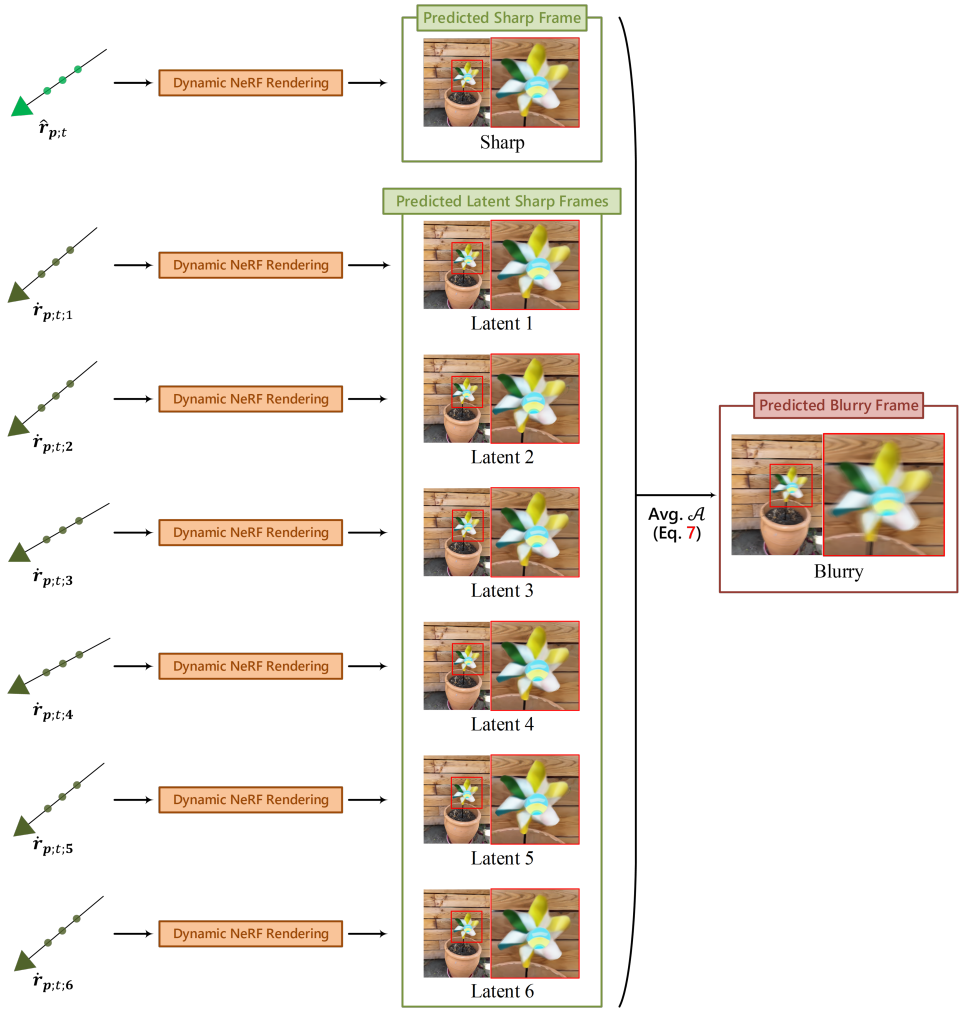
Table 5 shows the robustness of DyBluRF on different dataset capturing quality. In particular, the performance of the existing methods significantly decreases while training low-quality data, i.e., Blurry iPhone Dataset. Especially, there are large quality drops for HyperNeRF of -9% (mPSNR), -3% (mSSIM), -35% (mLPIPS) and -25% (tOF) and for T-NeRF of -5% (mPSNR), -5% (mSSIM), -20% (mLPIPS) and -17% (tOF). Whereas DyBluRF shows much smaller performance decreases in mLPIPS (-3.8%) and tOF (-0.9%) and even shows performance gains in mPSNR (+0.3%) and mSSIM (+0.17%). Note that '-' denotes the performance drop, while '+' is the performance gain.

E.2. Local vs. Global Motion-aware Ray Prediction

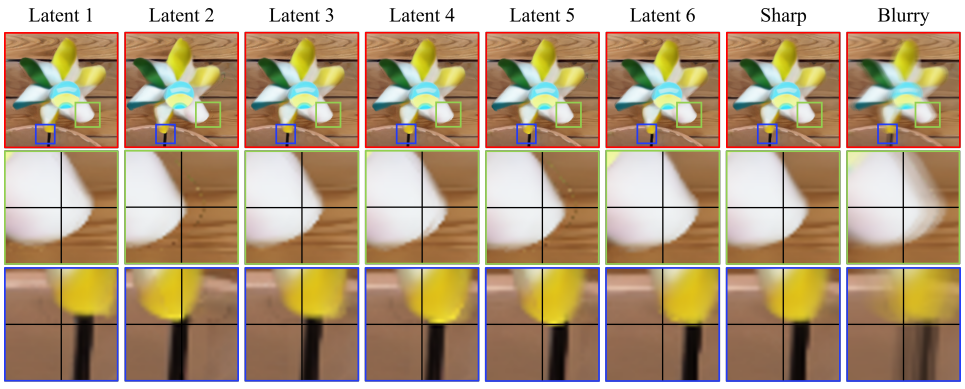
We provide additional visualizations to illustrate the varying rendering quality of the model without local object motion learning and the final DyBluRF in Fig. 8. Our final model excels at finely breaking down the blend of global camera motion and local object motions throughout the training times. This leads to a strong and reliable synthesis of novel views for our final DybluRF, in contrast to our variant that wasn't trained with LORR (Eq. 15).

E.3. Pose Error

We present both quantitative and qualitative results of camera pose error to demonstrate the superiority of our IRR stage compared to naive pose optimization. As shown in Table 6 and Fig. 9, optimizing the dynamic radiance field and the camera poses in the same training iteration can lead to bad local minima. In contrast, our DyBluRF adopts the interleave optimization by utilizing the scene components decomposition. It is worth noting that the naive pose optimization and our interleave optimization leverage simple $\mathfrak{se}(3)$ camera parameterization for pose optimization to be equipped into our effective dynamic deblurring NeRF framework. Please note that even though it is not our main scope to handle the pose optimization algorithm itself, our IRR stage can further incorporate advanced pose optimization approaches such as CamP [43] or BARF [30] in future works.



(a)



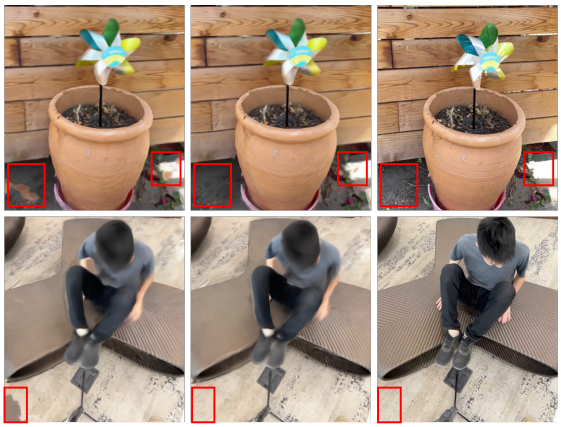
(b)

Figure 7. **Visualization of our blurry color rendering process with latent sharp rays**. (a) Blurry color rendering process in the MDD stage (Sec. 3.4). (b) Our predicted latent sharp frames are remarkably acquired as latent sharp rays by inherently learning a blurring process according to Eq. 7. The zoomed-in outcomes with red and smaller green and blue boxes across distinct latent sharp frames reveal intricate local object motion.

Method	Trained w/	Apple	Block	Paper-windmill	Space-out
T-NeRF [35]	Sharp	17.43 / 0.728 / 0.508 / 0.591	**17.52** / **0.669** / **0.346** / 0.741	17.55 / 0.367 / **0.258** / 0.418	17.71 / 0.591 / 0.377 / 0.491
T-NeRF [35]	Blurry	17.34 / 0.720 / 0.547 / 0.620	16.48 / 0.644 / 0.411 / 0.795	16.83 / 0.338 / 0.424 / 0.569	16.23 / 0.561 / 0.436 / 1.329
HyperNeRF [42]	Sharp	16.47 / **0.754** / **0.414** / 0.693	14.71 / 0.606 / 0.438 / 0.722	14.94 / 0.272 / 0.348 / 0.453	17.65 / 0.636 / 0.341 / 0.440
HyperNeRF [42]	Blurry	14.31 / 0.681 / 0.663 / 1.411	16.12 / 0.642 / 0.416 / 0.958	16.59 / 0.335 / 0.365 / 0.666	17.79 / 0.631 / 0.332 / 0.402
DyBluRF (Ours)	Sharp	18.00 / 0.737 / 0.488 / 0.495	17.47 / 0.665 / 0.349 / **0.713**	18.19 / **0.405** / 0.301 / **0.224**	18.72 / 0.640 / **0.323** / **0.273**
DyBluRF (Ours)	Blurry	**18.03** / 0.737 / 0.505 / **0.479**	17.35 / 0.660 / 0.361 / 0.735	**18.24** / **0.410** / 0.300 / 0.242	**18.99** / 0.646 / 0.328 / 0.276

Method	Trained w/	Spin	Teddy	Wheel	**Average**
T-NeRF [35]	Sharp	**19.16** / **0.567** / 0.443 / 1.064	13.71 / 0.570 / 0.429 / 1.142	**15.65** / **0.548** / **0.292** / 1.453	16.96 / 0.577 / 0.379 / 0.843
T-NeRF [35]	Blurry	17.16 / 0.503 / 0.534 / 1.162	14.07 / 0.562 / 0.464 / 1.094	14.93 / 0.499 / 0.379 / 1.360	16.15 / 0.547 / 0.456 / 0.990 -0.81 / -0.030 / +0.077 / +0.147
HyperNeRF [42]	Sharp	17.26 / 0.540 / **0.371** / 1.017	12.59 / 0.537 / 0.527 / 0.940	14.59 / 0.511 / 0.331 / 1.818	16.81 / 0.550 / **0.332** / 0.869
HyperNeRF [42]	Blurry	16.39 / 0.498 / 0.499 / 1.277	13.77 / 0.567 / **0.420** / 1.143	12.11 / 0.393 / 0.435 / 1.739	15.30 / 0.535 / 0.447 / 1.085 -1.51 / -0.015 / +0.115 / +0.216
DyBluRF (Ours)	Sharp	18.11 / 0.535 / 0.386 / **0.992**	**14.61** / **0.572** / 0.425 / 0.867	15.03 / 0.497 / 0.303 / **1.082**	17.16 / 0.579 / 0.368 / **0.664**
DyBluRF (Ours)	Blurry	18.20 / 0.534 / 0.415 / 1.011	**14.61** / 0.569 / 0.451 / **0.855**	15.09 / 0.501 / 0.316 / 1.095	**17.22** / **0.580** / 0.382 / 0.670 +0.06 / +0.001 / +0.014 / +0.006

Table 5. **Dynamic novel view synthesis evaluation.** **Red** and blue denote the best and second best performances, respectively. Each block element of 4-performance denotes (**mPSNR↑ / mSSIM↑ / mLPIPS↓ / tOF↓**). The 'Trained w/ Sharp' indicates the result of each method trained with the original iPhone Dataset [19], whereas the 'Trained w/ Blurry' indicates the results of each method trained with the newly synthesized Blurry iPhone Dataset. We indicate the average performance differences of each method across all metrics when trained with the original iPhone Dataset [19] and the Blurry iPhone Dataset below the performance achieved with the Blurry iPhone Dataset. A **light green** color denotes a **performance gain**, while an **orange** color indicates a **performance loss**. Different from the existing video view synthesis methods [35, 42] which are highly exacerbated from our Blurry iPhone Dataset, our DyBluRF shows significant robustness.



w/o LORR (Eq.15) Ours GT

Figure 8. **Visualization of rendering quality improvement using motion decomposition**.

Variant	Error$_{rot}$ (°)↓	Error$_{trans}$↓
Naive pose optimization	0.00914	0.00899
IRR	**0.00871**	**0.00697**

Table 6. **Average camera pose error**. We compare the average rotation error (Error$_{rot}$) and translation error (Error$_{trans}$) of the naive pose optimization and our IRR stage.

E.4. Visual Analysis of Surface Normal Loss

As depicted in Fig. 10, the depth map estimated by the pre-trained monocular depth estimation model [48] may exhibit a significant scale difference compared to the absolute scale of the target scene (1.3 vs. 55.7 and 1.8 vs. 68.8 for two specific pixel locations) which can be obtained from LiDAR sensors. Adopting the pre-trained monocular depth estimation model for NeRF regularization presents challenges related to the scale ambiguity. Common approaches to address this issue involve aligning depth maps with SfM sparse reconstruction or employing median scaling for supervised loss. For the DyBluRF, we opt for a straightforward yet effective surface normal supervision, denoted as $\mathcal{L}_{sn}$, to enhance the geometric fidelity of our radiance field. As shown in Table 3, the surface normal loss $\mathcal{L}_{sn}$ improves our model performance, while the median scaling depth loss failed to reconstruct the target scene. In addition, we provide a qualitative comparison to show the superiority of $\mathcal{L}_{sn}$ in Fig. 11. As shown in Fig. 11, the depth loss enhances the representation of the surface of the *gray block*, but it struggles to accurately resolve the scale-ambiguity; thus the model is not capable of reconstructing the target scene. In contrast, our model without geometry regularization results in imprecise geometry of the moving block. In contrast, our DyBluRF incorporating $\mathcal{L}_{sn}$ significantly improves the rendering quality, especially in the moving objects regions.

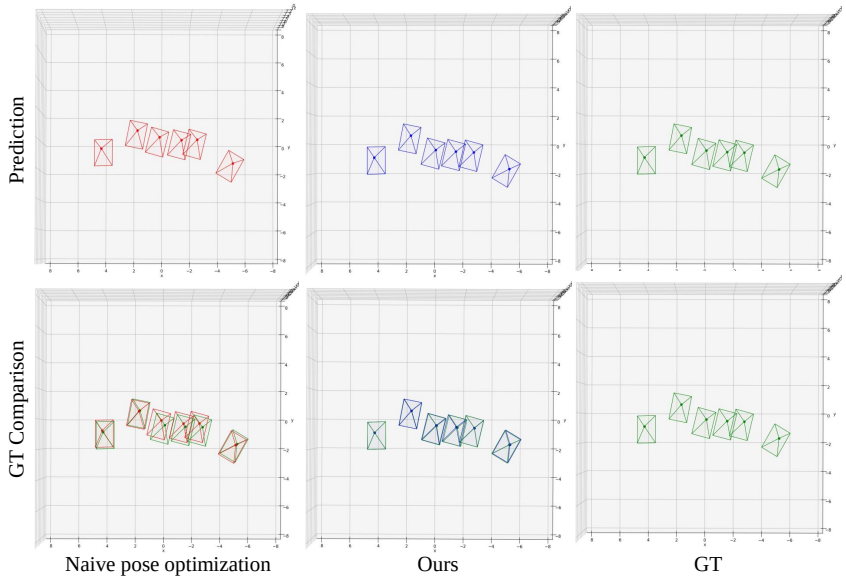


Figure 9. **Visualization of camera pose error.** 'GT Comparisons' shows overlapped GT camera poses extracted from sharp frames and the optimized poses by each method: Naive pose optimization and DyBluRF (Ours).

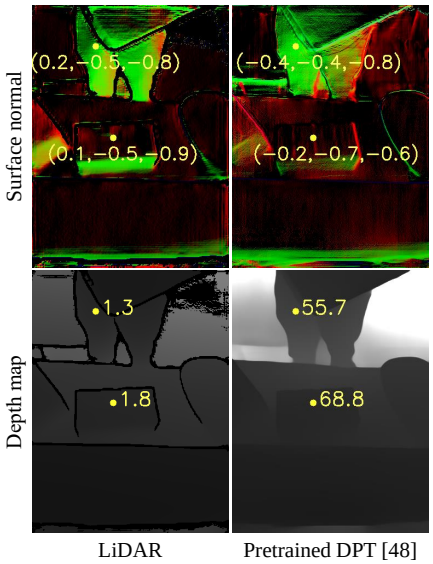


Figure 10. **Visualization of depth maps and the corresponding surface normals.** We compare the depth and surface normal values (yellow dots) obtained from LiDAR sensor and predicted by the pretrained model.

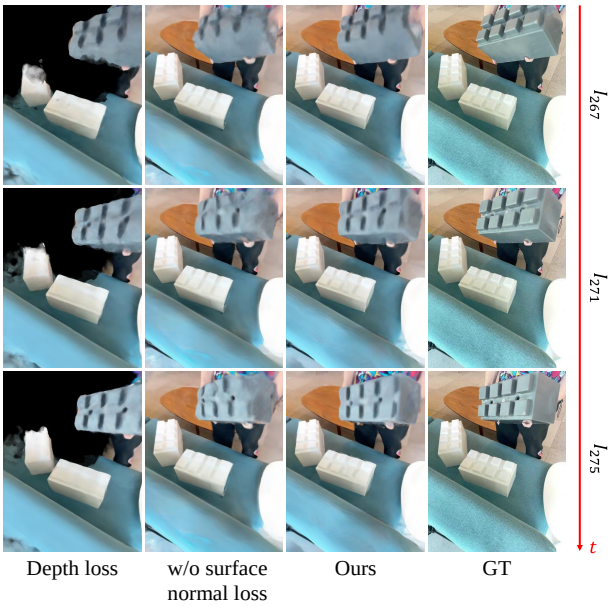


Figure 11. **Qualitative ablation study on surface normal loss.** We provide the visualization to show the effectiveness of $\mathcal{L}_{sn}$ for geometry regularization.

E.5. Per-Scene Ablation Study Results

Table 7 presents the per-scene quantitative results for the six variants of our DyBluRF, corresponding to Table 3 in the main paper. By comparing variant (f) to (e) across all scenes in terms of mLPIPS, our proposed MDD stage stably and significantly increases the perceptual quality of reconstructed frames.

E.6. Additional Qualitative Comparisons

Figs. 13, 14 and 15 show the abundant visual comparisons for dynamic deblurring novel view synthesis. Our DyBluRF generally better synthesizes *temporally consistent* deblurring results, which tend to be failed by all the previous methods.

Variant	IRR	$\mathcal{L}_{sn}$	MDD	Apple	Block	Paper-windmill	Space-out
(a)	-	-	-	17.34 / 0.720 / 0.547 / 0.620	16.48 / 0.644 / 0.411 / 0.795	16.83 / 0.338 / 0.424 / 0.569	16.23 / 0.561 / 0.436 / 1.329
(b)	$\mathcal{N}$	-	-	17.07 / 0.721 / 0.542 / 0.636	16.38 / 0.641 / 0.415 / 0.829	16.38 / 0.329 / 0.436 / 0.493	17.99 / 0.629 / 0.368 / 0.309
(c)	✓	-	-	**18.24** / **0.739** / 0.508 / **0.470**	17.31 / **0.664** / 0.388 / **0.687**	18.15 / 0.405 / 0.346 / 0.285	17.40 / 0.647 / 0.364 / 0.255
(d)	✓	$\mathcal{D}$	-	8.17 / 0.582 / 0.834 / 1.169	7.94 / 0.402 / 0.767 / 1.690	16.76 / 0.382 / 0.432 / 0.429	6.92 / 0.340 / 0.724 / 0.808
(e)	✓	✓	-	18.02 / **0.739** / 0.509 / 0.473	17.23 / 0.660 / 0.391 / 0.699	**18.36** / **0.427** / 0.346 / 0.271	18.56 / **0.649** / 0.352 / **0.251**
(f)	✓	✓	✓	18.03 / 0.737 / **0.505** / 0.479	**17.35** / 0.660 / **0.361** / 0.735	18.24 / 0.410 / **0.300** / **0.242**	**18.99** / 0.646 / **0.328** / 0.276

Variant	IRR	$\mathcal{L}_{sn}$	MDD	Spin	Teddy	Wheel	**Average**
(a)	-	-	-	17.16 / 0.503 / 0.534 / 1.162	14.07 / 0.562 / 0.464 / 1.094	14.93 / 0.499 / 0.379 / 1.360	16.15 / 0.547 / 0.456 / 0.990
(b)	$\mathcal{N}$	-	-	16.07 / 0.478 / 0.557 / 1.156	13.87 / 0.553 / 0.461 / 1.067	14.66 / 0.466 / 0.365 / 1.153	16.06 / 0.545 / 0.449 / 0.806
(c)	✓	-	-	18.14 / 0.522 / 0.481 / 1.026	14.40 / 0.565 / 0.465 / 0.908	15.40 / **0.518** / 0.348 / **1.004**	17.00 / 0.580 / 0.414 / 0.662
(d)	✓	$\mathcal{D}$	-	10.78 / 0.394 / 0.679 / 1.435	6.84 / 0.360 / 0.866 / 1.698	5.04 / 0.173 / 1.073 / 1.479	8.92 / 0.376 / 0.768 / 1.244
(e)	✓	✓	-	18.14 / 0.531 / 0.484 / **0.986**	14.57 / **0.571** / 0.466 / **0.844**	**15.43** / 0.512 / 0.349 / 1.049	17.19 / **0.584** / 0.414 / **0.653**
(f)	✓	✓	✓	**18.20** / **0.534** / **0.415** / 1.011	**14.61** / 0.569 / **0.451** / 0.855	15.09 / 0.501 / **0.316** / 1.095	**17.22** / 0.580 / **0.382** / 0.670

Table 7. **Per-Scene breakdowns of ablation study.** **Red** and blue denote the best and second best performances, respectively. Each block element of 4-performance denotes (**mPSNR↑ / mSSIM↑ / mLPIPS↓ / tOF↓**).

E.7. Visual Comparisons with Demo Video

We provide a video at https://youtu.be/UF5QTWF3_Dg for comparing our DyBluRF to the existing methods [7, 24, 32, 35, 42]. The demo video consists of three comparisons (Dynamic Deblurring Novel View Synthesis Evaluation on Blurry iPhone Dataset, Table 1) on scenes of *paper-windmill*, *block*, *wheel* and one comparison (Dynamic Novel View Synthesis Evaluation, Table 2) on a scene of *apple* and two ablation studies on the effectiveness of MDD stage and decompositions of GCRP and LORR.

E.8. The Number of Latent Sharp Rays

We provide a quantitative comparison of our DyBluRF model with varying numbers of latent sharp rays for the MDD stage in Table 8. In particular, increasing the number N_b of latent sharp rays can improve the reconstruction performance but it requires more computational cost. For the best quality and computational efficiency, we adopt $N_b = 6$ for all experiments of the final DyBluRF. To measure the training time per iteration, we utilize a single RTX 3090Ti GPU and set the batch size to 128. It is notable that the number of N_b only affects the training time. In the inference stage, where we render a single predicted sharp ray per pixel, the inference time remains constant across all variants, irrespective of the different values of N_b.

N_b	mPSNR↑	mSSIM↑	mLPIPS↓	tOF↓	Training time per iter. (s)
2	16.90	0.571	0.409	0.750	**0.095**
6	**17.22**	**0.580**	**0.382**	**0.670**	0.161
10	17.19	0.579	0.383	0.675	0.226

Table 8. **Ablation Study on the number of latent sharp rays.**

E.9. Limitations

In the Blurry iPhone Dataset, there is one continuous training monocular video for each scene. Additionally, two fixed cameras with novel views are provided for evaluation, following the evaluation protocol of the original iPhone Dataset [19]. This configuration makes it challenging to train deep-learning-based methods on unseen information that may be requested for evaluation. For instance, in some specific scenes, as shown in Fig. 12, due to different lighting environments, the tone of validation views can only be observed in a significantly small number of training samples (View 2). Hence, all kinds of radiance fields are generally overfitted to the major lighting condition of View 1. This challenge arises from the dataset's characteristics, which exhibit extremely diverse training and validation views. Introduction of relighting technique, modeling of reflectance component or balancing training data can be one of an option to handle this issue and we leave it for one of our future work.

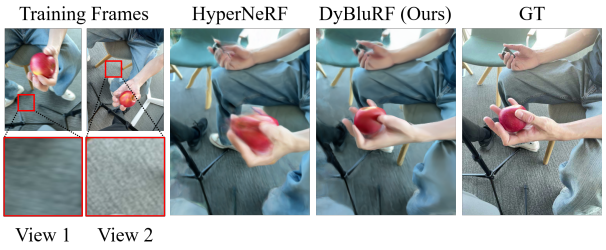


Figure 12. **Limitations of our DyBluRF.** Since the majority of training video frames in the Blurry iPhone Dataset exhibit a similar tone in View 1 (about 92% of the training frames), reconstructing the accurate tone of validation views with a similar tone in View 2 (about 8% of the training frames) becomes challenging. This difficulty arises due to the varying lighting effects present in the Blurry iPhone Dataset.

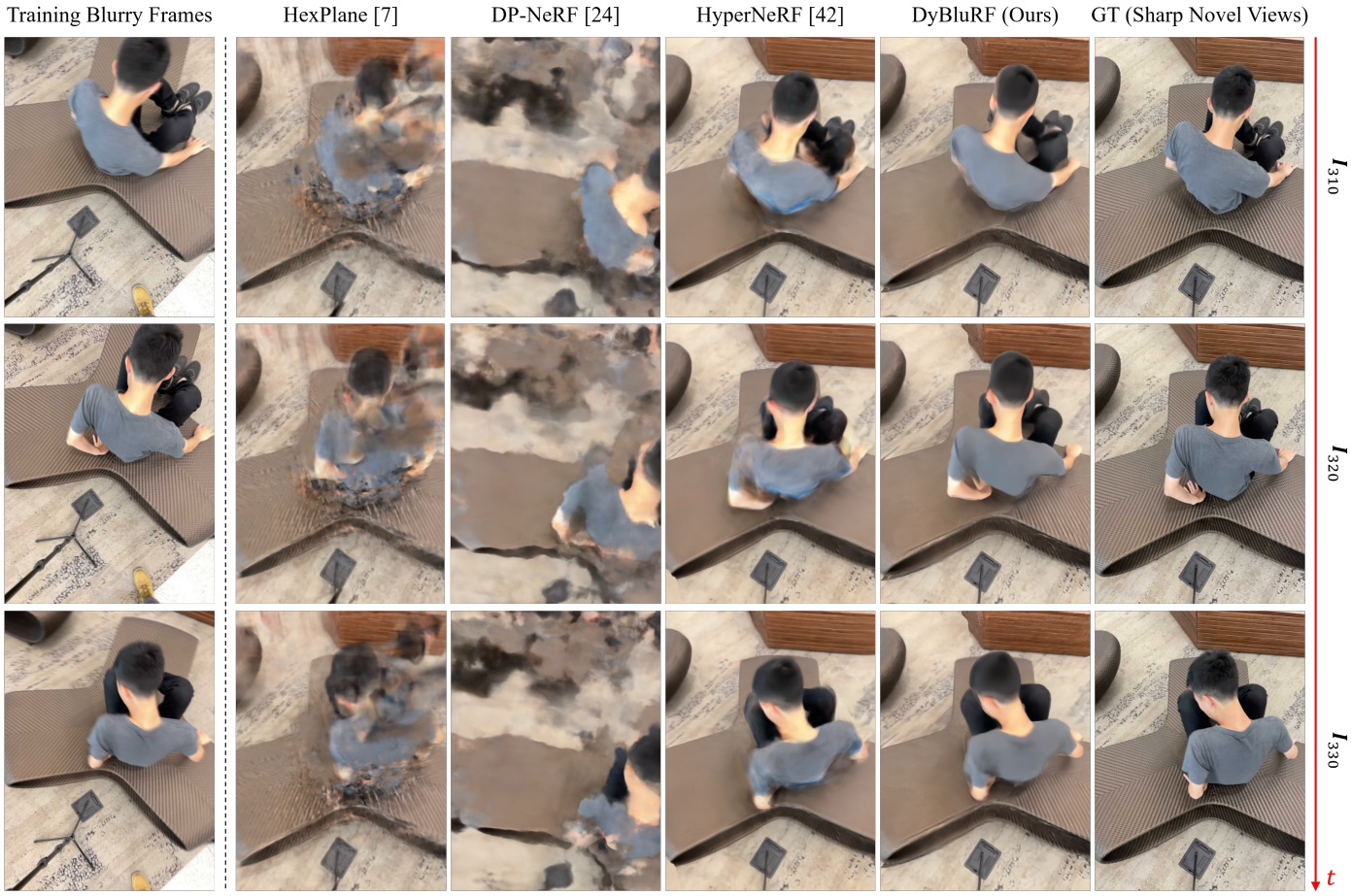


Figure 13. **Dynamic deblurring novel view synthesis results.** We compare the novel view synthesis quality of our DyBluRF for *spin* scene with the existing methods [7, 24, 42]. Each row corresponds to the 310th, 320th and 330th frame, respectively.

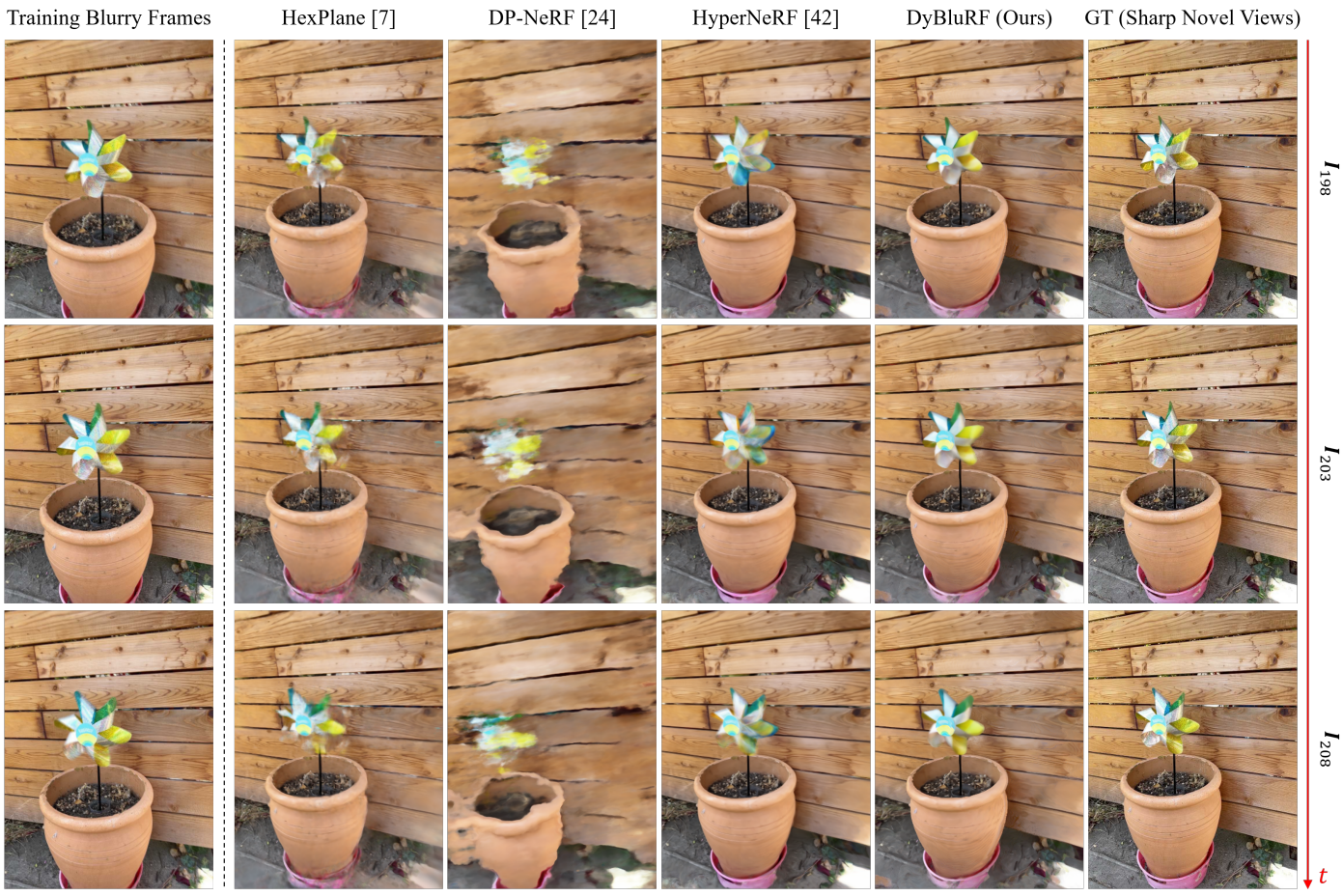


Figure 14. **Dynamic deblurring novel view synthesis results.** We compare the novel view synthesis quality of our DyBluRF for *paper-windmill* scene with the existing methods [7, 24, 42]. Each row corresponds to the 198th, 203th and 298th frame, respectively.

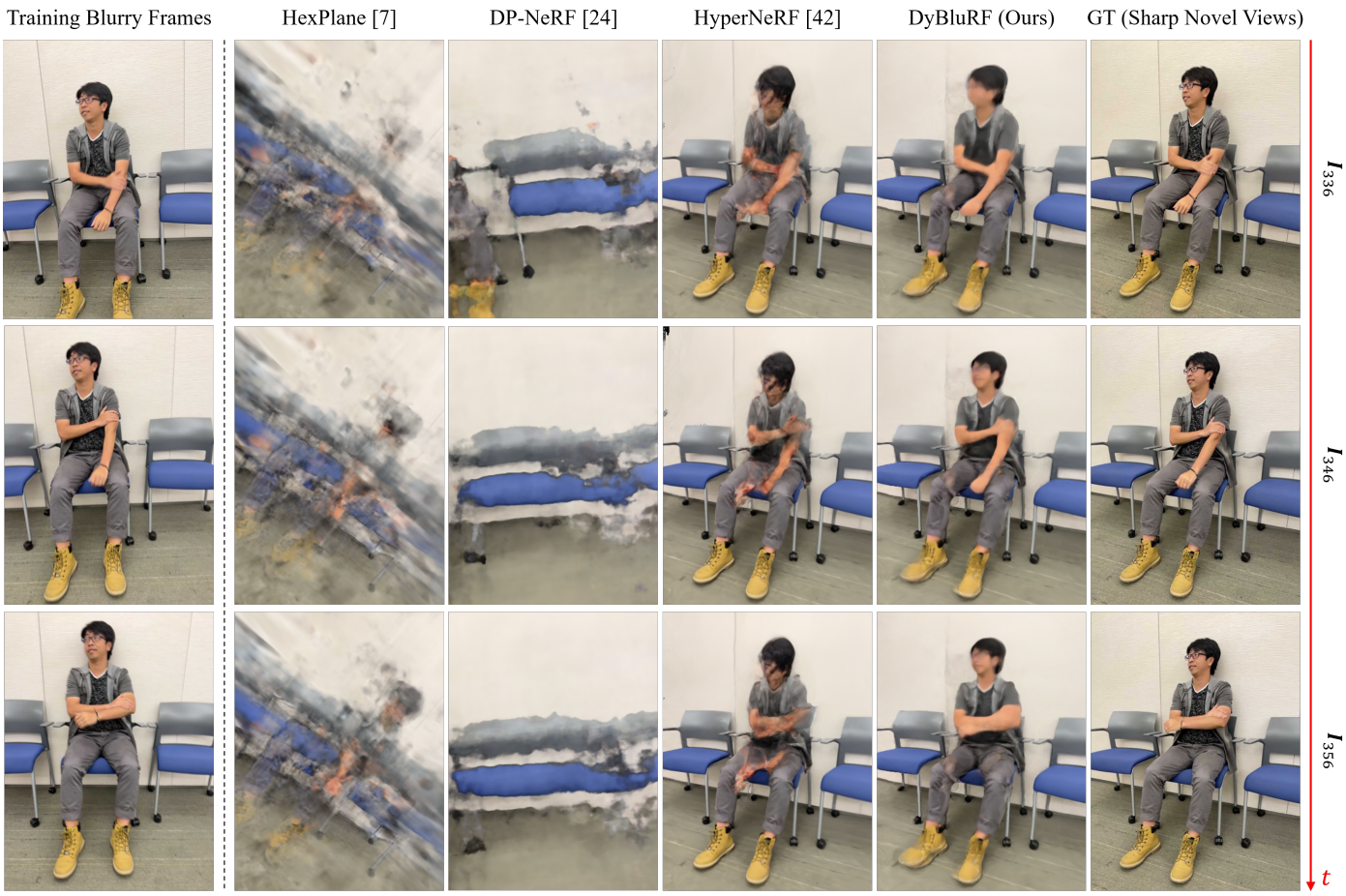


Figure 15. **Dynamic deblurring novel view synthesis results.** We compare the novel view synthesis quality of our DyBluRF for *space-out* scene with the existing methods [7, 24, 42]. Each row corresponds to the 336th, 346th and 356th frame, respectively.