# TRAJECTORY-AWARE DIFFERENTIABLE BLUR AUGMENTATION FOR IMAGE DEBLURRING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Image deblurring aims to recover sharp images degraded by motion blur caused by camera shake and object motion. Since collecting blur–sharp pairs covering diverse motion patterns is costly, blur augmentation provides an effective way to expand the training distribution, particularly under limited training data. However, existing methods primarily diversify blur patterns through random perturbations of blur magnitude and orientation, resulting in limited exploration of diverse motion trajectories underlying real-world blur formation. To overcome this limitation, we propose a differentiable trajectory-aware blur augmentation framework that explicitly expands motion-space coverage through trajectory modeling. Specifically, we introduce a Global Trajectory-aware Augmentation (GTA) module that parameterizes camera motion using motion dynamics and trajectory primitives to synthesize diverse motion trajectories. Furthermore, we propose a Local Trajectory-aware Augmentation (LTA) module to model spatially varying local motion, thereby capturing diverse local blur variations and enriching blur diversity. The differentiable formulation enables trajectory augmentation to be optimized through the deblurring objective. Experimental results demonstrate improved blur diversity and deblurring performance across both standard and limited-data training settings.

## 1 INTRODUCTION

Image deblurring aims to recover a latent sharp image from a blurred observation degraded by camera or object motion during exposure (Nah et al., 2017; Tao et al., 2018). Owing to the spatially varying nature of motion blur and the loss of fine image details, it remains an ill-posed restoration problem. Recent deep learning-based methods have substantially improved deblurring performance on synthetic and real-world benchmarks (Zamir et al., 2022; Chen et al., 2022; Kuo et al., 2025). Nevertheless, practical deblurring remains challenging because real-world blur exhibits substantial variations in camera motion, object movement, and scene dynamics (Rim et al., 2020; 2022). Such diverse motion patterns are difficult to fully cover with a fixed set of training blur–sharp pairs.

Blur augmentation has therefore emerged as an effective strategy for improving deblurring generalization by expanding the diversity of motion patterns observed during training (Wu et al., 2024; Kim et al., 2025). This is particularly beneficial under limited training data, where the observed blur distribution covers only a small portion of possible motion patterns. Blur synthesis provides a practical means of expanding this distribution without additional data collection. Recent approaches achieve this through pixel-wise blur condition modeling (Wu et al., 2024) or geometry-aware motion synthesis (Kim et al., 2025).

Despite these advances, existing blur augmentation methods primarily diversify blur patterns by perturbing blur magnitude and orientation (Wu et al., 2024; Kim et al., 2025). While these attributes provide intuitive control over overall motion strength and direction, they offer only a coarse description of the underlying camera motion and do not explicitly capture how the trajectory evolves throughout the exposure interval. As a result, existing augmentation methods provide limited variation in motion dynamics and trajectory geometry. Moreover, existing augmentation pipelines are typically optimized separately from the deblurring network, preventing the synthesis process from adapting to motion patterns that are difficult for the current model. Consequently, the augmentation



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

cannot adapt its trajectory perturbations according to the deblurring objective, and the synthesized samples tend to remain concentrated within a limited region of the trajectory space.

To address these limitations, we propose a differentiable trajectory-aware blur augmentation framework that directly augments the underlying motion trajectories while allowing the augmentation process to be guided by the deblurring objective. Specifically, we introduce a Global Trajectory-aware Augmentation (GTA) module that models camera motion through temporal dynamics and trajectory primitives, enabling diverse trajectories with temporal and geometric variations. We further introduce a Local Trajectory-aware Augmentation (LTA) module that captures spatially varying residual motion, including motion from independently moving objects, complementing global camera motion with diverse local trajectories. Owing to its differentiable formulation, the trajectory parameters can be adaptively perturbed according to the deblurring model, encouraging the synthesis of increasingly challenging and informative blur patterns during training. The proposed method broadens the range of motion patterns encountered during training and improves deblurring generalization.

Our contributions are as follows:

- We propose a differentiable trajectory-aware blur augmentation framework that enables trajectory perturbations to be optimized through the deblurring objective, synthesizing informative blur patterns beyond predefined random augmentations.
- We introduce a Global Trajectory-aware Augmentation (GTA) module that parameterizes camera motion through temporal dynamics and trajectory primitives, enabling diverse trajectory generation beyond conventional magnitude- and orientation-based perturbations.
- We introduce a Local Trajectory-aware Augmentation (LTA) module that models spatially varying residual motion, including motion induced by independently moving objects, complementing global camera motion with diverse local trajectories.
- Extensive experiments across multiple deblurring architectures and datasets demonstrate consistent performance improvements, including under limited training-data settings.

## 2 Related Work

### 2.1 Image Deblurring

Image deblurring aims to recover a sharp image from a blurred observation degraded by camera shake or object motion. Early deep learning-based methods employed convolutional neural networks for blur kernel estimation and end-to-end restoration (Nah et al., 2017; Gong et al., 2017), followed by recurrent, multi-scale, and multi-stage architectures that further improved deblurring performance (Tao et al., 2018; Zamir et al., 2021). More recently, Transformer-based methods have further improved deblurring performance through advanced attention mechanisms and motion-aware feature modeling (Chen et al., 2025; Kuo et al., 2025). Despite these advances, collecting blur–sharp pairs that sufficiently cover diverse real-world motion patterns remains costly and labor-intensive, resulting in limited blur diversity within practical training sets.

### 2.2 Blur Augmentation

Data augmentation has been widely used to improve model generalization by exposing networks to variations beyond the original training distribution (Zhang et al., 2017; Cubuk et al., 2020). Following this principle, recent blur augmentation methods synthesize additional motion blur patterns from blur–sharp pairs (Wu et al., 2024; Kim et al., 2025). ID-Blau (Wu et al., 2024) represents blur using pixel-wise magnitude and orientation and generates controllable blurred images through diffusion-based reblurring, while GeoSyn (Kim et al., 2025) employs 3D-aware motion estimation to synthesize diverse blur by manipulating motion magnitude and direction.

Despite these advances, existing methods largely control blur through magnitude and orientation, providing only limited flexibility over the underlying motion trajectory. Moreover, blur augmentation is typically optimized independently of the deblurring objective, preventing it from adapting to motion patterns challenging for the current model. Existing approaches also have limited ability to model independently moving objects, restricting the diversity of spatially varying local blur.

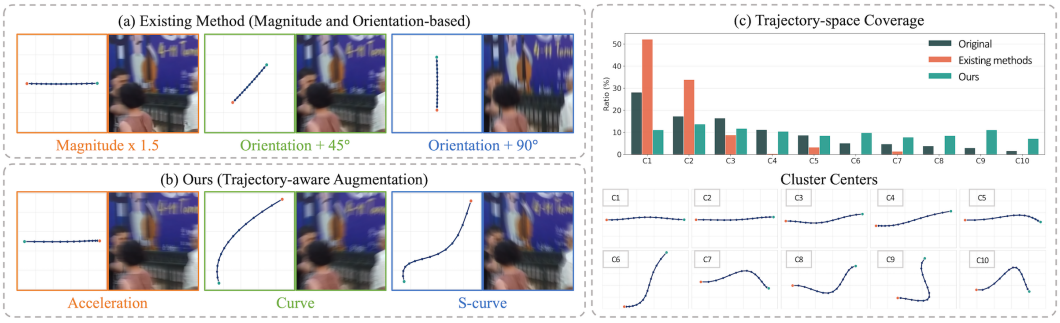



Figure 1: Motivation for trajectory-aware blur augmentation. (a) Existing magnitude/orientation methods yield limited trajectory variation. (b) Our method synthesizes diverse trajectories with temporal and geometric variations. (c) Distribution of the original and augmented motion trajectories over clusters obtained from the GoPro training set, together with representative cluster centers. For all compared data, trajectories are estimated from the corresponding blur images using the same trajectory estimator and clustering protocol.

# 3 METHOD

## 3.1 MOTIVATION

Existing magnitude- and orientation-based blur augmentation methods (Wu et al., 2024; Kim et al., 2025) mainly alter coarse motion attributes and therefore produce only limited trajectory variations, as shown in Figure 1(a). To examine motion diversity in the original GoPro training set (Nah et al., 2017), we estimate a representative 2D motion trajectory for each training sample and cluster the trajectories by geometric shape after the normalization described in Appendix B.3. As shown in Figure 1(c), the resulting clusters reveal substantial geometric variation in motion trajectories, which cannot be sufficiently represented by changes in magnitude and orientation alone.

These observations motivate augmenting trajectory geometry directly rather than relying solely on coarse motion attributes such as magnitude and orientation. Accordingly, we introduce Trajectory-aware Augmentation (TA), which explicitly perturbs the underlying motion trajectories and adapts these perturbations to the deblurring objective through a differentiable formulation.

## 3.2 OVERVIEW

Our framework comprises reblurring and deblurring stages. We first train a trajectory-aware reblurring network to reconstruct the observed blur from blur–sharp pairs, then use it to synthesize diverse blur augmentations for deblurring training.

## 3.3 GLOBAL TRAJECTORY-AWARE BLUR AUGMENTATION

**Preliminaries** Image-plane motion is represented using a backward-sampling convention over the normalized exposure interval $t \in [0, 1]$. Translation is expressed in pixels and in-plane rotation in radians. Let $\mathbf{v}(t) \in \mathbb{R}^2$ denote the image-plane translational velocity at time $t$. The corresponding translational trajectory is defined by temporal integration:

$$\mathbf{T}(t) = \int_0^t \mathbf{v}(\xi)\, d\xi, \qquad \mathbf{T}(0) = \mathbf{0}. \tag{1}$$

All motion quantities follow this backward-sampling convention and are initially defined relative to the exposure start, then re-centered at the sharp-image reference time $t_c = 0.5$ for rendering.

**Motion Dynamics Modeling** We model temporal motion dynamics using a linearly varying forward velocity:

$$v_{\text{fwd}}(t) = m_0 + at, \tag{2}$$

where $m_0$ denotes the initial forward velocity and $a$ denotes the constant forward acceleration.

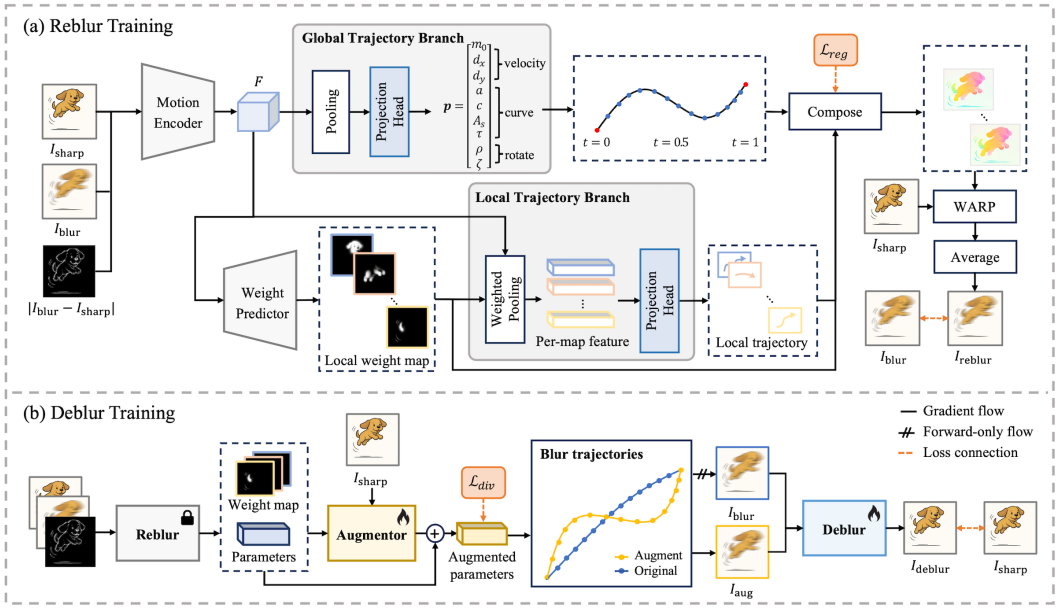



Figure 2: (a) Overview of the proposed trajectory-aware reblurring framework. GTA reconstructs global camera motion through trajectory dynamics and geometry modeling, while LTA models spatially varying local residual motion. (b) Deblurring training pipeline using the pretrained reblurring network to synthesize trajectory-aware augmented blur–sharp pairs for deblurring optimization.

**Trajectory-aware Curve Modeling** We parameterize motion with continuous velocity, acceleration, and trajectory primitives rather than independent displacement samples, producing coherent trajectories through temporal integration. Inspired by continuous-time trajectory estimation (Mueggler et al., 2015; Ovrén & Forssén, 2019), we further model lateral motion relative to the initial motion direction. Specifically, the network predicts an unnormalized direction vector $(d_x, d_y)$, from which we define the initial direction $\mathbf{d}_0$ and its perpendicular direction $\mathbf{n}$:

$$\mathbf{d}_0 = \begin{bmatrix} d_{0,x} \\ d_{0,y} \end{bmatrix} = \frac{1}{\sqrt{d_x^2 + d_y^2 + \epsilon}} \begin{bmatrix} d_x \\ d_y \end{bmatrix}, \qquad \mathbf{n} = \begin{bmatrix} -d_{0,y} \\ d_{0,x} \end{bmatrix}, \tag{3}$$

where $\epsilon > 0$ is a small constant for numerical stability.

We model the lateral component using two trajectory primitives: a curve primitive with constant lateral acceleration $c$ and an S-curve primitive with acceleration amplitude $A_s$, transition time $\tau$, and a fixed transition-sharpness parameter $\eta$:

$$v_{\text{curve}}(t) = ct, \qquad a_{\text{scurve}}(t) = A_s \tanh\left(\eta(t - \tau)\right). \tag{4}$$

Here, $\tau \in [0, 1]$ specifies the time at which the S-curve lateral acceleration crosses zero, while $\eta > 0$ controls the sharpness of the acceleration transition and is fixed throughout training.

$$v_{\text{scurve}}(t) = \int_0^t a_{\text{scurve}}(\xi)\, d\xi, \qquad v_{\text{lat}}(t) = v_{\text{curve}}(t) + v_{\text{scurve}}(t). \tag{5}$$

The forward and lateral components are combined and integrated to obtain the predicted global translational trajectory $\tilde{\mathbf{T}}(t)$:

$$\tilde{\mathbf{v}}(t) = v_{\text{fwd}}(t)\mathbf{d}_0 + v_{\text{lat}}(t)\mathbf{n}, \qquad \tilde{\mathbf{T}}(t) = \int_0^t \tilde{\mathbf{v}}(\xi)\, d\xi. \tag{6}$$

**Trajectory-aware Rotation Modeling** In addition to translation, camera shake can involve in-plane rotation. We model the accumulated rotation angle over the exposure interval as

$$\phi(t) = \rho t + \frac{1}{2}\zeta t^2, \tag{7}$$

where $\rho$ and $\zeta$ denote the initial angular velocity and constant angular acceleration, respectively.

**Trajectory Parameter Prediction** As shown in Figure 2(a), the blurred and sharp images, together with their absolute difference, are encoded into motion-aware features:

$$\boldsymbol{F} = f_e\left(\boldsymbol{I}_{\text{blur}}, \boldsymbol{I}_{\text{sharp}}, |\boldsymbol{I}_{\text{blur}} - \boldsymbol{I}_{\text{sharp}}|\right). \tag{8}$$

The feature map is globally averaged and fed to the global trajectory projection head $f_{p1}(\cdot)$:

$$\boldsymbol{z} = \text{GAP}(\boldsymbol{F}), \qquad \boldsymbol{p} = f_{p1}(\boldsymbol{z}). \tag{9}$$

The resulting global trajectory parameters are

$$\boldsymbol{p} = [m_0,\, d_x,\, d_y,\, a,\, c,\, A_s,\, \tau,\, \rho,\, \zeta]^\top. \tag{10}$$

### 3.4 Local Trajectory-Aware Blur Augmentation

Real-world blur often exhibits spatially varying motion due to local scene dynamics, such as independently moving objects. To model these variations, we introduce LTA, which separates the spatial contribution of local motion components from their trajectory dynamics. From the latent feature $\mathbf{F}$ in Eq. 8, LTA predicts $K$ spatial mixing weights:

$$w_k(\mathbf{x}) = \frac{\exp(\hat{w}_k(\mathbf{x})/T_w)}{\sum_{j=1}^{K} \exp(\hat{w}_j(\mathbf{x})/T_w)}, \tag{11}$$

where $\hat{w}_k(\mathbf{x})$ is the unnormalized logit predicted from $\mathbf{F}$ for the $k$-th local motion component, $T_w = 0.5$ is the softmax temperature used throughout training, and $w_k(\mathbf{x})$ represents the contribution of the component at location $\mathbf{x}$.

For each learned component $k = 2, \ldots, K$, we use its spatially pooled feature $\mathbf{f}_k$ and spatial moment $\boldsymbol{\mu}_k$ to predict local trajectory parameters:

$$\mathbf{f}_k = \frac{\sum_{\mathbf{x}} w_k(\mathbf{x})\mathbf{F}(\mathbf{x})}{\sum_{\mathbf{x}} w_k(\mathbf{x})}, \qquad \mathbf{q}_k = f_{p2}\left([\mathbf{f}_k; \boldsymbol{\mu}_k]\right), \tag{12}$$

where $\boldsymbol{\mu}_k$ encodes the spatial moments of $w_k(\mathbf{x})$, and $f_{p2}$ is the local trajectory projection head.

The predicted local trajectory parameter vector is

$$\mathbf{q}_k = [m_{0,k}, d_{x,k}, d_{y,k}, a_k, c_k, A_{s,k}, \tau_k]^\top. \tag{13}$$

Each $\mathbf{q}_k$, for $k = 2, \ldots, K$, defines a local translational trajectory $\mathbf{b}_k(t)$ using the GTA parameterization without rotation, while the first component is fixed as $\mathbf{b}_1(t) \equiv \mathbf{0}$. These trajectories are combined using the predicted spatial mixing weights:

$$\Delta\mathbf{T}(t, \mathbf{x}) = \sum_{k=1}^{K} w_k(\mathbf{x})\mathbf{b}_k(t). \tag{14}$$

The global trajectory captures scene-wide camera motion, whereas $\Delta\mathbf{T}(t, \mathbf{x})$ represents spatially varying local motion. The local components do not predict separate rotations; rotation is shared globally across the image.

We sample $N$ time points $\{t_j\}_{j=1}^{N}$ over $[0, 1]$, using $t_c = 0.5$ as the sharp-image reference time. The global translation, local trajectories, and global rotation are re-centered at $t_c$ and combined to construct the displacement fields $\mathbf{U}_j$. The sharp image is then warped at each sampled time and temporally averaged:

$$\boldsymbol{I}_{\text{reblur}} = \frac{1}{N}\sum_{j=1}^{N} \mathcal{W}(\boldsymbol{I}_{\text{sharp}}, \mathbf{U}_j), \tag{15}$$

where $\mathcal{W}$ denotes the image warping operator.

Further details on the reblurring network, reference centering, displacement-field construction, temporal sampling, and training procedure are provided in Appendix A.3.




Figure 3: Qualitative comparison of deblurring results on representative benchmark images.

### 3.5 Training Procedure

**Reblurring Network Training** Using the predicted trajectories, the reblurring network reconstructs the observed blur $\boldsymbol{I}_{\text{blur}}$ from the sharp image $\boldsymbol{I}_{\text{sharp}}$ and is trained with the following loss:

$$\mathcal{L}_{\text{reblur}} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{flow-reg}} + \mathcal{L}_{\text{local-reg}}. \tag{16}$$

Here, $\mathcal{L}_{\text{rec}}$ is the reconstruction loss, while $\mathcal{L}_{\text{flow-reg}}$ and $\mathcal{L}_{\text{local-reg}}$ denote the weighted sums of the displacement-field and local-motion regularization terms, respectively. The local-motion regularization is omitted during global-motion pretraining.

**Deblurring Network Training** We introduce a learnable augmentation network (Augmentor) that perturbs trajectory parameters to synthesize challenging blur patterns for the deblurring model.

Given a blur–sharp pair, the pretrained reblurring network predicts $\boldsymbol{p}$, $\{\mathbf{q}_k\}_{k=2}^{K}$, and $\mathbf{W} = \{w_k(\mathbf{x})\}_{k=1}^{K}$. Conditioned on these predictions and image-region features, Augmentor predicts normalized perturbations $\boldsymbol{\Delta}_g$ and $\{\boldsymbol{\Delta}_{l,k}\}_{k=2}^{K}$ through a shared motion encoder. These perturbations use the estimated sample-specific motion as an anchor while exploring controlled trajectory variations.

We scale the normalized perturbations by per-parameter standard deviations $\boldsymbol{\sigma}_g$ and $\boldsymbol{\sigma}_l$, estimated from the pretrained reblurring network's predictions on the training set and fixed thereafter:

$$\hat{\boldsymbol{p}} = \boldsymbol{p} + \alpha_g \boldsymbol{\sigma}_g \odot \boldsymbol{\Delta}_g, \qquad \hat{\mathbf{q}}_k = \mathbf{q}_k + \alpha_l \boldsymbol{\sigma}_l \odot \boldsymbol{\Delta}_{l,k}, \qquad k = 2, \ldots, K, \tag{17}$$

where $\alpha_g$ and $\alpha_l$ control the global and local perturbation scales, respectively, and $\odot$ denotes element-wise multiplication. We keep $m_0$ and $m_{0,k}$ fixed and clip $\hat{\tau}$ and $\hat{\tau}_k$ to $[0, 1]$.

Instead of perturbing pixels, Augmentor perturbs structured trajectory parameters. The normalized perturbations are bounded and scaled by empirical variation, enabling controlled motion deviations under the deblurring objective. We render the perturbed trajectories to obtain augmented blur:

$$\boldsymbol{I}_{\text{aug}} = \mathcal{B}\left(\boldsymbol{I}_{\text{sharp}}; \hat{\boldsymbol{p}}, \{\hat{\mathbf{q}}_k\}_{k=2}^{K}, \mathbf{W}\right), \tag{18}$$

where $\mathcal{B}(\cdot)$ denotes the differentiable blur renderer defined by the warping and temporal averaging process in Eq. 15. Let $\mathcal{D}(\cdot)$ denote the deblurring network and $\ell_{\text{rec}}(\cdot, \cdot)$ its reconstruction loss.

We define the losses on the original and augmented blur as

$$\mathcal{L}_{\text{orig}} = \ell_{\text{rec}}\left(\mathcal{D}(\boldsymbol{I}_{\text{blur}}), \boldsymbol{I}_{\text{sharp}}\right), \qquad \mathcal{L}_{\text{aug}} = \ell_{\text{rec}}\left(\mathcal{D}(\boldsymbol{I}_{\text{aug}}), \boldsymbol{I}_{\text{sharp}}\right). \tag{19}$$

The deblurring network is trained on both original and augmented samples:

$$\mathcal{L}_{\text{deb}} = \frac{1}{2}\left(\mathcal{L}_{\text{orig}} + \mathcal{L}_{\text{aug}}\right). \tag{20}$$

Table 1: Deblurring performance with varying amounts of GoPro training data. Results are evaluated on the GoPro and HIDE test sets. Best results are shown in **bold**.

| Backbone | Data (%) | Aug. | GoPro | | | | HIDE | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | MUSIQ$^{\uparrow}$ | LPIPS$^{\downarrow}$ | PSNR$^{\uparrow}$ | DISTS$^{\downarrow}$ | MUSIQ$^{\uparrow}$ | LPIPS$^{\downarrow}$ | PSNR$^{\uparrow}$ | DISTS$^{\downarrow}$ |
| Concertormer (Kuo et al., 2025) | 10% | – | 38.87 | 0.136 | 30.47 | 0.091 | 45.70 | 0.160 | 28.82 | 0.094 |
| | | ID-Blau | 39.16 | 0.133 | 30.52 | 0.090 | 47.17 | 0.158 | 28.77 | 0.092 |
| | | Ours | **41.12** | **0.116** | **31.39** | **0.083** | **48.92** | **0.148** | **29.61** | **0.086** |
| | 50% | – | 42.38 | 0.105 | 32.08 | 0.079 | 50.71 | 0.126 | 30.40 | 0.078 |
| | | ID-Blau | 42.81 | 0.102 | 32.24 | 0.079 | 49.92 | 0.128 | 30.42 | 0.081 |
| | | Ours | **43.69** | **0.094** | **32.70** | **0.076** | **51.91** | **0.125** | **30.58** | **0.075** |
| | 100% | – | 43.15 | 0.098 | 32.63 | 0.076 | 51.72 | 0.118 | 30.80 | 0.074 |
| | | ID-Blau | 43.42 | 0.096 | 32.73 | 0.077 | 51.59 | 0.118 | 30.77 | 0.078 |
| | | Ours | **44.01** | **0.089** | **33.10** | **0.073** | **53.41** | **0.114** | **30.95** | **0.073** |
| MIMO-UNet+ (Cho et al., 2021) | 10% | – | 38.49 | 0.155 | 28.92 | 0.097 | 43.64 | 0.192 | 26.71 | 0.113 |
| | | ID-Blau | 41.49 | 0.123 | 30.64 | 0.085 | 48.37 | 0.157 | 28.50 | 0.090 |
| | | Ours | **43.42** | **0.099** | **31.37** | **0.074** | **51.37** | **0.142** | **28.98** | **0.081** |
| | 50% | – | 43.43 | 0.097 | 31.93 | 0.073 | 51.79 | 0.132 | 29.50 | 0.076 |
| | | ID-Blau | 43.99 | 0.092 | 32.60 | 0.074 | 52.19 | 0.124 | 30.35 | 0.076 |
| | | Ours | **44.65** | **0.085** | **32.71** | **0.069** | **53.49** | **0.117** | **30.42** | **0.071** |
| | 100% | – | 43.99 | 0.090 | 32.44 | 0.072 | 52.13 | 0.124 | 30.00 | 0.073 |
| | | ID-Blau | 43.98 | 0.088 | 32.93 | 0.072 | 52.48 | 0.117 | 30.68 | 0.073 |
| | | Ours | **45.02** | **0.082** | **33.10** | **0.069** | **53.89** | **0.112** | **30.71** | **0.070** |

Augmentor is optimized to generate challenging yet controlled and diverse trajectory perturbations:

$$\mathcal{L}_{\text{augmentor}} = -\mathcal{L}_{\text{aug}} + \lambda_{\text{mag}}\mathcal{L}_{\text{mag}} - \lambda_{\text{div}}\mathcal{L}_{\text{div}}. \tag{21}$$

Here, $\mathcal{L}_{\text{mag}}$ limits perturbation magnitude, while $\mathcal{L}_{\text{div}}$ encourages variation in perturbation directions across the mini-batch. Augmentor and the deblurring network are optimized alternately; further details are provided in Appendix A.4.

# 4 Experiments

## 4.1 Experimental Setup

We conduct experiments on GoPro (Nah et al., 2017), HIDE (Shen et al., 2019), and RSBlur (Rim et al., 2022). Following previous works (Wu et al., 2024; Kim et al., 2025), models trained on GoPro are evaluated on both GoPro and HIDE. To study data efficiency, we construct 10% and 50% GoPro training subsets by sampling the corresponding proportion of blur–sharp pairs from each scene, with 100% denoting the full training set. We evaluate deblurring performance using PSNR, LPIPS (Zhang et al., 2018), DISTS (Ding et al., 2020), and MUSIQ (Ke et al., 2021). Model-specific architectures, optimization settings, and training schedules are provided in Appendix A.

## 4.2 Performance Comparison

### 4.2.1 Quantitative Results

Table 1 evaluates the effectiveness of blur augmentation under varying amounts of training data. Our method consistently outperforms existing methods (Wu et al., 2024) across 10%, 50%, and 100% of the GoPro training data, demonstrating its effectiveness in both data-scarce and full-data settings.

Table 2: Comparison results on RSBlur.

| Methods | GMACs$^{\downarrow}$ | PSNR$^{\uparrow}$ |
|---|---|---|
| Restormer (Zamir et al., 2022) | 141.00 | 33.69 |
| ConvIR-L (Cui et al., 2024) | 71.22 | 34.06 |
| NAFNet (Chen et al., 2022) | 63.64 | 33.97 |
| + ID-Blau (Wu et al., 2024) | 63.64 | 34.01 |
| + GeoSyn (Kim et al., 2025) | 63.64 | 34.23 |
| + Ours | 63.64 | **34.34** |

We further evaluate our method on RSBlur in terms of performance and computational cost. As shown in Table 2, our method improves the NAFNet baseline without increasing inference GMACs.

It also outperforms ID-Blau under our evaluation protocol and the published GeoSyn result, demonstrating its effectiveness on realistic blur.

#### 4.2.2 QUALITATIVE RESULTS

Figure 3 presents qualitative comparisons using MIMO-UNet+ trained on the full GoPro training set. Our method better restores fine structures, text, and object boundaries across diverse scenes and blur patterns, consistent with the quantitative improvements. Additional comparisons are provided in Appendix C.1.

#### 4.2.3 TRAJECTORY-AWARE BLUR AUGMENTATIONS

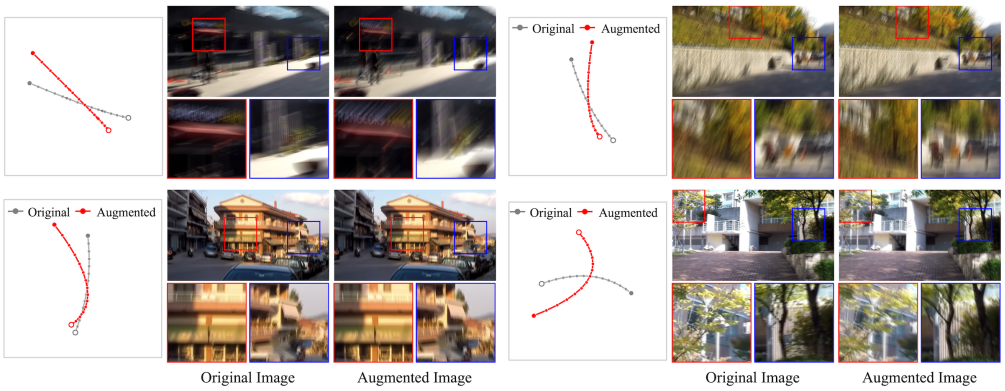



Figure 4: Visualization of trajectory-aware augmentation and the corresponding blur variations.

Figure 4 visualizes the augmented camera trajectories synthesized by our method and their corresponding blur augmentation results. The trajectory examples include acceleration, curve, and S-curve motions, illustrating diverse temporal and geometric variations from the original camera motion. The bottom examples show blur augmentations produced from the curve and S-curve trajectories, where the enlarged regions highlight the resulting changes in blur patterns.

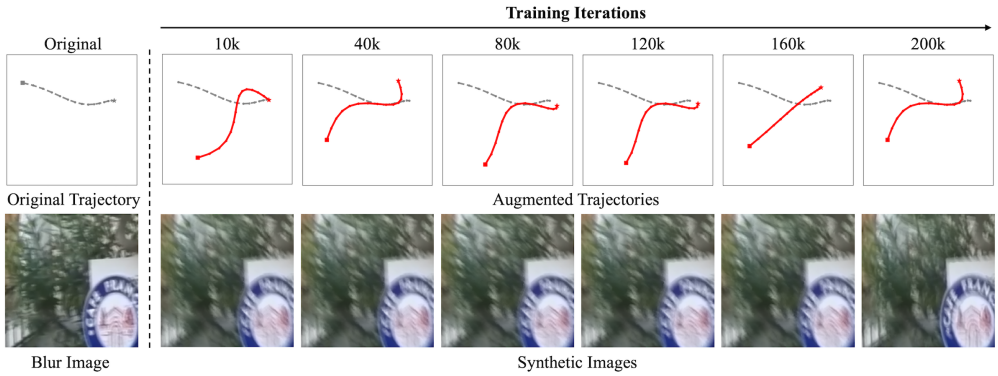



Figure 5: Evolution of learnable trajectory augmentation during deblurring training.

Figure 5 shows how the augmented trajectory evolves during deblurring training. Since our augmentation is differentiable and optimized for the current deblurring objective, the trajectory adapts with the deblurring network, producing diverse and challenging blur patterns throughout training.

#### 4.2.4 LOCAL TRAJECTORY-AWARE BLUR AUGMENTATIONS

Figure 6 visualizes the local trajectory components and spatial weights. We use $K = 4$, with one zero-motion background and three learned motion components, enabling spatially varying blur augmentation. Additional results are provided in Appendix C.3.

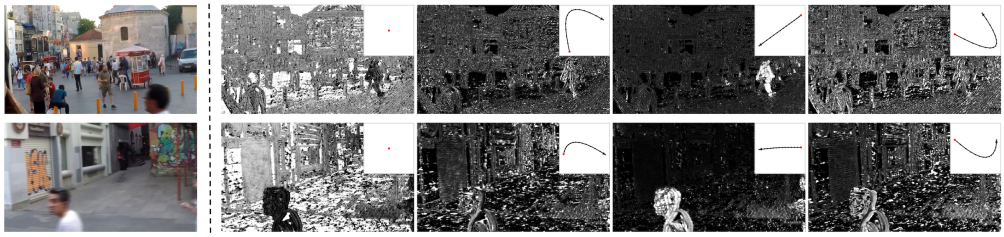



Figure 6: Visualization of local trajectory components and their spatial weights.

### 4.3 Ablation Study

Table 3 evaluates trajectory-aware modeling and differentiable augmentation using 10% of the GoPro training set. For a fair comparison, random and differentiable perturbations use the same augmentation ratio and training schedule, differing only in how trajectory perturbations are generated. Global and local trajectory modeling improves over the baseline, while differentiable augmentation consistently outperforms random perturbation on both GoPro and HIDE, demonstrating the benefit of adapting trajectory perturbations to the deblurring objective.

Table 3: Ablation study of the proposed components on GoPro and HIDE.

| GTA | LTA | Differentiable | GoPro | | | | HIDE | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | MUSIQ$^{\uparrow}$ | LPIPS$^{\downarrow}$ | PSNR$^{\uparrow}$ | DISTS$^{\downarrow}$ | MUSIQ$^{\uparrow}$ | LPIPS$^{\downarrow}$ | PSNR$^{\uparrow}$ | DISTS$^{\downarrow}$ |
| | | | 38.49 | 0.155 | 28.92 | 0.097 | 43.64 | 0.192 | 26.71 | 0.113 |
| ✓ | | ✓ | 43.20 | 0.104 | 31.24 | 0.077 | 50.64 | 0.150 | 28.69 | 0.085 |
| ✓ | ✓ | | 42.39 | 0.106 | 31.19 | 0.077 | 50.76 | 0.147 | 28.78 | 0.083 |
| ✓ | ✓ | ✓ | **43.42** | **0.099** | **31.37** | **0.074** | **51.37** | **0.142** | **28.98** | **0.081** |

## 5 Limitations

Our framework has several limitations. First, the current blur synthesis focuses on motion blur represented by parameterized trajectories and does not explicitly model other real-world degradations, such as saturation, sensor noise, or rolling-shutter effects. Second, since the differentiable blur augmentation is guided by the deblurring objective, its signal may be less informative during the early stages of training, before the deblurring network has sufficiently converged. Future work will explore these directions to improve the realism and adaptability of the proposed framework.

## 6 Conclusion

Motion blur in real-world scenes arises from diverse camera and object motions, making image deblurring dependent on the diversity of blur patterns encountered during training. However, collecting blur–sharp pairs that sufficiently cover such motion variations remains costly, while existing blur augmentation methods provide only limited control over the underlying motion trajectories. In this work, we proposed a differentiable trajectory-aware blur augmentation framework that directly broadens the motion space through global camera trajectory modeling and spatially varying local motion modeling. By explicitly representing temporal dynamics and trajectory geometry, the proposed framework captures a wider range of structured blur patterns beyond coarse magnitude- and orientation-based perturbations. Moreover, its differentiable formulation allows trajectory perturbations to be optimized with respect to the deblurring objective, enabling the synthesis of challenging and informative blur patterns throughout training. Extensive experiments across multiple datasets and deblurring architectures demonstrate consistent performance improvements under both standard and limited-data settings. These results show the effectiveness of trajectory-aware blur augmentation for enriching motion diversity and improving deblurring generalization.

### AI USE STATEMENT

In this work, we used generative AI tools to obtain feedback on our research methodology and experimental evaluation. Additionally, we used generative AI tools for grammar checking and minor language editing of the manuscript. We critically reviewed all AI-generated feedback and manually verified the resulting edits to ensure that they preserved the intended meaning and technical accuracy of the manuscript. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

### ETHICS STATEMENT

This work does not involve human subjects, sensitive personal data, or other ethical issues requiring special consideration. We are not aware of any specific concerns related to privacy, fairness, harmful applications, conflicts of interest, or research integrity under the ICLR Code of Ethics.

### REPRODUCIBILITY STATEMENT

We provide detailed implementation and training settings in the appendix, including model architecture, training schedule, optimization parameters, and other experimental details required to reproduce our results. Additional details on the datasets, evaluation settings, and experimental protocols are also provided in the appendix.

## REFERENCES

Duosheng Chen, Shihao Zhou, Jinshan Pan, Jinglei Shi, Lishen Qu, and Jufeng Yang. A polarization-aided transformer for image deblurring via motion vector decomposition. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 28061–28070. IEEE, 2025.

Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. In *European conference on computer vision*, pp. 17–33. Springer, 2022.

Sung-Jin Cho, Seo-Won Ji, Jun-Pyo Hong, Seung-Won Jung, and Sung-Jea Ko. Rethinking coarse-to-fine approach in single image deblurring. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4621–4630. IEEE, 2021.

Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 702–703, 2020.

Yuning Cui, Wenqi Ren, Xiaochun Cao, and Alois Knoll. Revitalizing convolutional network for image restoration. *IEEE transactions on pattern analysis and machine intelligence*, 46(12):9423–9438, 2024.

Keyan Ding, Kede Ma, Shiqi Wang, and Eero P Simoncelli. Image quality assessment: Unifying structure and texture similarity. *IEEE transactions on pattern analysis and machine intelligence*, 44(5):2567–2581, 2020.

Dong Gong, Jie Yang, Lingqiao Liu, Yanning Zhang, Ian Reid, Chunhua Shen, Anton Van Den Hengel, and Qinfeng Shi. From motion blur to motion flow: A deep learning solution for removing heterogeneous motion blur. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2319–2328, 2017.

Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. Musiq: Multi-scale image quality transformer. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5128–5137. IEEE, 2021.

Insoo Kim, Hana Lee, Hyong-Euk Lee, and Jinwoo Shin. Controllable blur data augmentation using 3d-aware motion estimation. In *The Thirteenth International Conference on Learning Representations*, 2025.

Pin-Hung Kuo, Jinshan Pan, Shao-Yi Chien, and Ming-Hsuan Yang. Efficient concertormer for image deblurring and beyond. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14665–14675. IEEE, 2025.

Elias Mueggler, Guillermo Gallego, and Davide Scaramuzza. Continuous-time trajectory estimation for event-based vision sensors. In *Robotics: Science and Systems XI*, 2015.

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

Hannes Ovrén and Per-Erik Forssén. Trajectory representation and landmark projection for continuous-time structure from motion. *The International Journal of Robotics Research*, 38(6): 686–701, 2019.

Jaesung Rim, Haeyun Lee, Jucheol Won, and Sunghyun Cho. Real-world blur dataset for learning and benchmarking deblurring algorithms. In *European conference on computer vision*, pp. 184–201. Springer, 2020.

Jaesung Rim, Geonung Kim, Jungeon Kim, Junyong Lee, Seungyong Lee, and Sunghyun Cho. Realistic blur synthesis for learning image deblurring. In *European conference on computer vision*, pp. 487–503. Springer, 2022.

Ziyi Shen, Wenguan Wang, Xiankai Lu, Jianbing Shen, Haibin Ling, Tingfa Xu, and Ling Shao. Human-aware motion deblurring. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5572–5581, 2019.

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

Jia-Hao Wu, Fu-Jen Tsai, Yan-Tsung Peng, Chung-Chi Tsai, Chia-Wen Lin, and Yen-Yu Lin. Id-blau: Image deblurring by implicit diffusion-based reblurring augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 25847–25856, 2024.

Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *2021 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 14816–14826. IEEE, 2021.

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

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

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

# A Training and Experimental Details

## A.1 Experimental Scope

Our experiments are designed to evaluate whether the proposed trajectory-aware blur augmentation improves deblurring under limited training data and across different network architectures. We therefore consider 10%, 50%, and 100% of the GoPro training set and evaluate both convolution-based and Transformer-based deblurring models, including MIMO-UNet+ (Cho et al., 2021) and the recent Concertormer (Kuo et al., 2025). This allows us to examine both data efficiency and architectural generality of the proposed augmentation.

### A.2 Comparison with Existing Blur Augmentation

**Comparison Protocol** For a fair comparison, all reproducible competing augmentation methods are trained using the same baseline training pipeline and evaluated across the same network architectures and experimental settings. GeoSyn (Kim et al., 2025) is the only exception, since its official code and pretrained blur-synthesis model are not publicly available. Similar to ID-Blau (Wu et al., 2024), GeoSyn primarily diversifies blur by manipulating motion magnitude and direction. Without access to its code or pretrained model, we are unable to retrain GeoSyn under different backbones and experimental settings or conduct further quantitative and qualitative comparisons. Therefore, we report the performance on RSBlur published in the original GeoSyn paper in Table 2. Using the reported GeoSyn result as a reference, our method achieves higher deblurring performance on RSBlur. Except for GeoSyn, all reproducible augmentation methods are trained and evaluated under the same experimental setup. The Restormer and ConvIR-L results in Table 2 are taken from their respective original papers.

**Dataset Subsets.** For the limited-data experiments, we construct scene-balanced GoPro (Nah et al., 2017) subsets by independently sampling the specified proportion of blur–sharp image pairs from each training scene. All crops originating from the same image are kept together. We use a fixed random seed of 0, and the resulting image lists are shared across all compared methods to ensure identical training subsets.

For each of the 10%, 50%, and 100% data regimes, we train our trajectory-aware reblurring network and the deblurring network using the corresponding GoPro training subset. We also retrain the ID-Blau reblurring model using the same subset for each data regime. Thus, both augmentation methods use reblurring models trained on the same GoPro training images available in the corresponding deblurring experiment.

**Evaluation Metrics.** We evaluate full-resolution RGB outputs using PSNR, LPIPS (Zhang et al., 2018), DISTS (Ding et al., 2020), and MUSIQ (Ke et al., 2021). All metrics are computed using the `pyiqa` implementation. No border cropping or Y-channel conversion is applied. PSNR, LPIPS, and DISTS are computed between the restored image and its ground-truth sharp image, whereas MUSIQ is computed on the restored image. For RSBlur, we use the official training and test splits and follow the evaluation protocol established by the original RSBlur benchmark (Rim et al., 2022).

### A.3 Reblurring Network Training

**Network Architecture.** The reblurring network uses a lightweight NAFNet (Chen et al., 2022) as its backbone. It takes the concatenation of the blur image, sharp image, and their absolute difference as a 9-channel input. We use a channel width of 8, encoder block configuration $[1, 1, 1, 28]$, one middle block, and decoder block configuration $[1, 1, 1, 1]$. The backbone produces a 128-channel motion feature map $\mathbf{F}$, from which a $K$-channel output head predicts the spatial mixing logits.

The global projection head $f_{p1}$ consists of global average pooling followed by a two-layer MLP with dimensions $128 \to 128 \to 9$. The local projection head $f_{p2}$ takes the concatenation of the component-wise pooled feature and its five-dimensional spatial moment vector, and uses a three-layer MLP with dimensions $133 \to 128 \to 128 \to 7$. ReLU activations are applied between successive linear layers.

**Spatial Moments.** Unless stated otherwise, all numerical-stability constants denoted by $\epsilon$ in the main paper and appendix are set to $10^{-8}$. The spatial mixing weights $\{w_k\}_{k=1}^{K}$ are obtained using softmax normalization with a fixed temperature $T_w = 0.5$.

The first local component represents the background, with its trajectory fixed at zero:

$$\mathbf{b}_1(t) \equiv \mathbf{0}, \qquad t \in [0, 1]. \tag{22}$$

The remaining components $k = 2, \ldots, K$ model spatially varying residual motion. Pixels with large $w_1(\mathbf{x})$ primarily follow global camera motion, with little local displacement.

For each learned component $k = 2, \ldots, K$, we summarize the spatial distribution of its weight map using five moments. Let $\mathbf{r} = (r_x, r_y)^\top$ denote coordinates normalized to $[-1, 1]$. The total weight

and weighted spatial center of component $k$ are

$$Z_k = \sum_{\mathbf{r}} w_k(\mathbf{r}) + \epsilon, \qquad \bar{\mathbf{r}}_k = \frac{1}{Z_k} \sum_{\mathbf{r}} w_k(\mathbf{r})\mathbf{r}. \tag{23}$$

The corresponding weighted covariance matrix is

$$\mathbf{C}_k = \frac{1}{Z_k} \sum_{\mathbf{r}} w_k(\mathbf{r})(\mathbf{r} - \bar{\mathbf{r}}_k)(\mathbf{r} - \bar{\mathbf{r}}_k)^\top. \tag{24}$$

Here, $\epsilon = 10^{-8}$ suppresses unstable spatial moments when a component has negligible total weight. The spatial moment vector is

$$\boldsymbol{\mu}_k = [\bar{r}_{k,x}, \bar{r}_{k,y}, C_{k,xx}, C_{k,yy}, C_{k,xy}]^\top, \tag{25}$$

which encodes the spatial center, spread, and covariance of the $k$-th weight map.

**Temporal Sampling.** We uniformly sample $N$ time points including both endpoints of the normalized exposure interval:

$$t_j = \frac{j-1}{N-1}, \qquad j = 1, \ldots, N. \tag{26}$$

Throughout the paper, we use the exposure center $t_c = 0.5$ as the sharp-image reference time. With $N = 15$, the reference time coincides with the central temporal sample, i.e., $t_c = t_8$.

The global translation, local trajectories, and global rotation are initially defined relative to the start of exposure. We convert them into reference-centered quantities as

$$\tilde{\mathbf{T}}_c(t) = \tilde{\mathbf{T}}(t) - \tilde{\mathbf{T}}(t_c), \qquad \mathbf{b}_{k,c}(t) = \mathbf{b}_k(t) - \mathbf{b}_k(t_c), \qquad \phi_c(t) = \phi(t) - \phi(t_c). \tag{27}$$

The reference-centered local displacement residual is

$$\Delta\mathbf{T}_c(t, \mathbf{x}) = \sum_{k=1}^{K} w_k(\mathbf{x})\mathbf{b}_{k,c}(t), \qquad \mathbf{b}_{1,c}(t) \equiv \mathbf{0}. \tag{28}$$

Thus, the first component contributes no additional local displacement, while the remaining components describe spatially varying residual motion relative to the global camera trajectory.

**Displacement-Field Construction.** At each sampled time $t_j$, we construct the displacement field by combining the reference-centered global rotation, global translation, and local motion residual:

$$\mathbf{U}_j(\mathbf{x}) = \big(\mathbf{R}(\phi_c(t_j)) - \mathbf{I}_2\big)(\mathbf{x} - \mathbf{o}) + \tilde{\mathbf{T}}_c(t_j) + \Delta\mathbf{T}_c(t_j, \mathbf{x}). \tag{29}$$

Here, $\mathbf{x}$ and $\mathbf{U}_j(\mathbf{x})$ denote the pixel coordinate and pixel displacement, respectively, in the rendered image, and

$$\mathbf{o} = \left(\frac{W_{\text{img}} - 1}{2}, \frac{H_{\text{img}} - 1}{2}\right)^\top \tag{30}$$

is the image center, where $W_{\text{img}}$ and $H_{\text{img}}$ denote the image width and height, respectively. The matrix $\mathbf{I}_2$ is the $2 \times 2$ identity matrix. The rotation matrix is

$$\mathbf{R}(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}. \tag{31}$$

Rotation and translation are modeled as additive image-plane displacement components rather than as a composed SE(2) camera pose. Under the backward-sampling convention, $\mathbf{x} + \mathbf{U}_j(\mathbf{x})$ specifies the location sampled from the reference sharp image for output pixel $\mathbf{x}$ at time $t_j$. We evaluate the sharp image at this location using bilinear interpolation, with border replication for coordinates outside the image.

**Training settings.** We train separate reblurring networks on the GoPro (Nah et al., 2017) and RSBlur (Rim et al., 2022) training splits for their respective experiments. During training, we randomly crop aligned $256 \times 256$ patches from each image pair and apply horizontal flips and rotations for data augmentation. We use a batch size of 32 and a fixed random seed of 0 for reproducibility.

The motion network uses $N = 15$ temporal samples and $K = 4$ local components. The first local component is the fixed zero-motion background component, and the three components are learned local-motion components. The softmax temperature for the spatial mixing weights is fixed to $T_w = 0.5$. The transition-sharpness parameter is fixed to $\eta = 6$ for both global and learned local S-curve primitives throughout training. The transition times $\tau$ and $\{\tau_k\}_{k=2}^{K}$ are constrained to $[0, 1]$.

**Two-stage optimization.** For GoPro, training is performed in two stages. We first train the reblurring network for 80 epochs with the local-motion branch disabled. We then enable the local-motion branch and jointly train the global- and local-motion components for an additional 920 epochs, retaining the weights learned during the first stage. We use AdamW with $\beta_1 = \beta_2 = 0.9$ and a weight decay of $10^{-3}$ throughout training. During the first stage, the learning rate is cosine-annealed from $1 \times 10^{-3}$ to $1 \times 10^{-7}$ over 80 epochs. For joint training, we initialize a new optimizer and cosine-annealing scheduler and decay the learning rate from $5 \times 10^{-4}$ to $1 \times 10^{-7}$ over 920 epochs. The scheduler is updated after every optimization step without learning-rate warm-up. For each stage, the cosine-annealing period is set to the total number of optimization steps in that stage. For RSBlur, we train a separate reblurring network using the same training settings, except that the total training duration is reduced to 300 epochs: 80 epochs of global-motion pretraining followed by 220 epochs of joint training.

**Training Objective.** Let $\mathbf{U}_j$ denote the displacement field at temporal sample $j$, and let

$$\mathbf{U}_j^{\mathrm{loc}} = \Delta\mathbf{T}_c(t_j, \cdot) \tag{32}$$

denote the local displacement residual. Let $w_k(\mathbf{r})$ denote the spatial mixing weight of component $k$. We use $\mathrm{mean}(\cdot)$ to average over all dimensions of its argument, including the batch dimension.

The reconstruction loss is

$$\mathcal{L}_{\mathrm{rec}} = \mathrm{mean}\left(|\boldsymbol{I}_{\mathrm{reblur}} - \boldsymbol{I}_{\mathrm{blur}}|\right). \tag{33}$$

We regularize the spatial and temporal variations of the complete displacement fields:

$$\mathcal{L}_{\text{flow-TV}} = \frac{1}{N}\sum_{j=1}^{N}\left[\mathrm{mean}\left(|D_x\mathbf{U}_j|\right) + \mathrm{mean}\left(|D_y\mathbf{U}_j|\right)\right], \tag{34}$$

$$\mathcal{L}_{\text{flow-2nd}} = \frac{1}{N-2}\sum_{j=2}^{N-1}\mathrm{mean}\left(|\mathbf{U}_{j+1} - 2\mathbf{U}_j + \mathbf{U}_{j-1}|\right). \tag{35}$$

Here, $D_x$ and $D_y$ are spatial forward-difference operators. Regularization acts on the full displacement field $\mathbf{U}_j$, including both global rotation and local residual motion.

For local motion, we penalize the residual magnitude and mixing-weight entropy:

$$\mathcal{L}_{\text{local-mag}} = \frac{1}{N}\sum_{j=1}^{N}\mathrm{mean}\left(|\mathbf{U}_j^{\mathrm{loc}}|\right), \tag{36}$$

$$\mathcal{L}_{\text{weight-ent}} = -\,\mathrm{mean}_{b,\mathbf{r}}\left[\sum_{k=1}^{K} w_k^{(b)}(\mathbf{r})\log\left(\max\left(w_k^{(b)}(\mathbf{r}), 10^{-8}\right)\right)\right], \tag{37}$$

where $b$ indexes samples in the batch and $\mathbf{r}$ indexes spatial locations in the weight map. The local-magnitude term limits the strength of local displacement, while the entropy term encourages concentrated component assignments.

The two regularization terms in the main-text training objective are defined as

$$\mathcal{L}_{\text{flow-reg}} = 10^{-3}\mathcal{L}_{\text{flow-TV}} + 10^{-4}\mathcal{L}_{\text{flow-2nd}}, \tag{38}$$

$$\mathcal{L}_{\text{local-reg}} = 10^{-3}\mathcal{L}_{\text{local-mag}} + 10^{-5}\mathcal{L}_{\text{weight-ent}}. \tag{39}$$

The fixed zero-motion component provides a background anchor. Pixels assigned a large value of $w_1(\mathbf{x})$ follow the global camera motion without an additional local displacement. Together with $\mathcal{L}_{\text{local-mag}}$, this encourages the learned local components to represent spatially varying residual motion rather than scene-wide motion.

We use the full objective during joint training. During global-motion pretraining, local motion is disabled and $\mathcal{L}_{\text{local-reg}}$ is omitted.

### A.4 Learnable Trajectory Augmentation

**Augmentor Architecture.** The pretrained reblurring network predicts global parameters $\mathbf{p} \in \mathbb{R}^9$, local parameters $\{\mathbf{q}_k\}_{k=2}^K$ with $\mathbf{q}_k \in \mathbb{R}^7$, and spatial mixing weights $\{w_k\}_{k=1}^K$, while the first local component is fixed as $\mathbf{b}_1(t) \equiv \mathbf{0}$. Augmentor encodes the sharp image using three $3 \times 3$ convolutional layers with channel dimensions $3 \to 32 \to 64 \to 128$, each with stride 2 and ReLU. Global and component-specific image features are obtained by global spatial pooling and spatial-weight-guided pooling, respectively. Each spatial weight map is additionally encoded using three $3 \times 3$ convolutional layers with channel dimensions $1 \to 32 \to 64 \to 64$ and strides 4, 4, and 2, followed by global average pooling. The global and local inputs are first mapped to a common 128-dimensional feature space using separate linear projections followed by ReLU activations. These features are then processed by a shared motion encoder consisting of two 128-dimensional linear layers with ReLU activations. Separate global and local output heads, each consisting of a linear layer followed by $\tanh$, predict 9-dimensional global and 7-dimensional local perturbations in $[-1, 1]$, respectively. No perturbation is predicted for the fixed zero-motion component.

**Parameter-wise Trajectory Perturbation.** Because trajectory parameters have different numerical ranges, we normalize their perturbation scales using empirical statistics from the pretrained reblurring network. Before deblurring training, we estimate the per-parameter standard deviations separately for each data regime using its corresponding training subset and keep them fixed thereafter. For local trajectories, the statistics are computed across both images and the learned local-motion components $k = 2, \ldots, K$.

Let $\boldsymbol{\sigma}_g$ and $\boldsymbol{\sigma}_l$ denote the standard deviations of the global and local trajectory parameters, respectively. The perturbed parameters are

$$\hat{\mathbf{p}} = \mathbf{p} + \alpha_g \boldsymbol{\sigma}_g \odot \boldsymbol{\Delta}_g, \qquad \hat{\mathbf{q}}_k = \mathbf{q}_k + \alpha_l \boldsymbol{\sigma}_l \odot \boldsymbol{\Delta}_{l,k}, \quad k = 2, \ldots, K, \tag{40}$$

where $\alpha_g = 1.0$ and $\alpha_l = 0.3$ control the global and local perturbation scales, respectively, and $\odot$ denotes element-wise multiplication.

The initial forward velocities $m_0$ and $\{m_{0,k}\}_{k=2}^K$ remain unchanged, while the remaining trajectory parameters are adaptively perturbed. Accordingly, the entries of $\boldsymbol{\Delta}_g$ and $\boldsymbol{\Delta}_{l,k}$ corresponding to $m_0$ and $m_{0,k}$ are set to zero. The spatial mixing weights are preserved so that augmentation modifies local trajectory dynamics while retaining the estimated spatial motion regions. The perturbed transition times $\hat{\tau}$ and $\{\hat{\tau}_k\}_{k=2}^K$ are hard-clipped to $[0, 1]$, with zero gradients outside this interval. The first local trajectory remains fixed as $\mathbf{b}_1(t) \equiv \mathbf{0}$.

For the random perturbation baseline in Table 3, normalized perturbations are independently sampled from $\mathcal{U}(-1, 1)$ and scaled by $\alpha_g \boldsymbol{\sigma}_g$ and $\alpha_l \boldsymbol{\sigma}_l$ for the global and learned local trajectory parameters, respectively. The first zero-motion component remains fixed.

During augmentation, we retain $N = 15$, $K = 4$, and $\eta = 6$, along with the spatial mixing weights $\mathbf{W} = \{w_k(\mathbf{x})\}_{k=1}^K$ predicted using $T_w = 0.5$. We pass the perturbed trajectory parameters through the differentiable blur renderer to synthesize an augmented blurry image:

$$\boldsymbol{I}_{\text{aug}} = \mathcal{B}\left(\boldsymbol{I}_{\text{sharp}}; \hat{\mathbf{p}}, \{\hat{\mathbf{q}}_k\}_{k=2}^K, \mathbf{W}\right). \tag{41}$$

**Augmentation Objective.** Augmentor is optimized to generate trajectory perturbations that are challenging for the current deblurring network while controlling their magnitude and encouraging diversity:

$$\mathcal{L}_{\text{augmentor}} = -\mathcal{L}_{\text{aug}} + \lambda_{\text{mag}} \mathcal{L}_{\text{mag}} - \lambda_{\text{div}} \mathcal{L}_{\text{div}}, \tag{42}$$

where $\mathcal{L}_{\text{aug}}$ is the deblurring reconstruction loss on augmented samples. We use $\lambda_{\text{mag}} = 10^{-4}$ and $\lambda_{\text{div}} = 1.0$.

We stack the learned local-motion perturbations as $\mathbf{\Delta}_l = \mathrm{stack}_{k=2}^{K} \mathbf{\Delta}_{l,k}$, excluding the fixed zero-motion component, which has no learnable trajectory perturbation. The magnitude regularization penalizes the element-wise squared magnitude of the normalized global and local perturbations:

$$\mathcal{L}_{\mathrm{mag}} = \mathrm{mean}\left(\mathbf{\Delta}_g \odot \mathbf{\Delta}_g\right) + \mathrm{mean}\left(\mathbf{\Delta}_l \odot \mathbf{\Delta}_l\right), \tag{43}$$

where $\mathrm{mean}(\cdot)$ averages over all dimensions of its argument, including the batch dimension. The zero entries corresponding to $m_0$ and $m_{0,k}$ are retained in both the mean computations and the subsequent vectorization.

For each sample $b$, we concatenate its global and learned local perturbations into a vector and normalize it:

$$\mathbf{d}_b = \left[\mathbf{\Delta}_g^{(b)}; \mathrm{vec}\left(\mathbf{\Delta}_l^{(b)}\right)\right], \qquad \mathbf{u}_b = \frac{\mathbf{d}_b}{\max(\|\mathbf{d}_b\|_2, \epsilon)}, \tag{44}$$

where vec flattens the learned local perturbations and $\epsilon = 10^{-8}$ ensures numerical stability. For perturbation vectors with $\|\mathbf{d}_b\|_2 \geq \epsilon$, normalization makes the diversity term depend on direction rather than magnitude. We then define the diversity objective as

$$\mathcal{L}_{\mathrm{div}} = \frac{1}{B}\sum_{b=1}^{B} \|\mathbf{u}_b - \bar{\mathbf{u}}\|_2^2, \qquad \bar{\mathbf{u}} = \frac{1}{B}\sum_{b=1}^{B} \mathbf{u}_b, \tag{45}$$

where $B > 1$ is the batch size. Because $\mathcal{L}_{\mathrm{div}}$ is subtracted from the Augmentor objective, minimizing $\mathcal{L}_{\mathrm{augmentor}}$ promotes dispersion of normalized parameter perturbations across samples. When all $\mathbf{u}_b$ have unit norm, this term equals $1 - \|\bar{\mathbf{u}}\|_2^2$; it therefore favors a small batch-mean vector but does not by itself guarantee broad trajectory-space coverage. All Augmentor training configurations use a batch size greater than one.

**Alternating Optimization.** We first train the deblurring network for 5,000 iterations without updating Augmentor, providing a sufficiently informative deblurring objective before learnable augmentation begins. After this warm-up, Augmentor is updated once every four deblurring iterations.

Augmentor is optimized using AdamW with an initial learning rate of $5 \times 10^{-4}$, which is decayed to $1 \times 10^{-6}$ using a cosine annealing scheduler.

During an Augmentor update, the deblurring network's parameters are frozen, but gradients propagate through the network and differentiable blur renderer to Augmentor. The augmented images are then resynthesized using the updated trajectory perturbations and detached from the augmentation graph before the deblurring update.

We train the deblurring network on original and augmented samples with equal weighting:

$$\mathcal{L}_{\mathrm{deb}} = \frac{1}{2}\left(\mathcal{L}_{\mathrm{orig}} + \mathcal{L}_{\mathrm{aug}}\right). \tag{46}$$

The pretrained reblurring network remains frozen throughout the entire deblurring stage. The first local component remains fixed as the zero-motion background component, and all synthesized image intensities are clamped to $[0, 1]$.

### A.5 Deblurring Network Training

**Training Settings.** We follow the augmentation procedure in Appendix A.4, keeping each backbone's architecture, optimizer settings, and data augmentation identical across methods. Across GoPro subsets, Concertormer (Kuo et al., 2025) scales training iterations with subset size, while MIMO-UNet+ (Cho et al., 2021) uses a fixed epoch count, maintaining approximately equal epochs across data regimes for each backbone. NAFNet (Chen et al., 2022) is used only for full-data RSBlur. Reported batch sizes count original blur–sharp pairs; augmentation concatenates their augmented counterparts with equal loss weights, doubling the effective batch size.

**MIMO-UNet+ Training** We train MIMO-UNet+ (Cho et al., 2021) using the original architecture. The network is trained for 3,000 epochs using $256 \times 256$ randomly cropped patches with a batch size of 4. We use the Adam optimizer with an initial learning rate of $1 \times 10^{-4}$ and a cosine annealing scheduler. Following the original implementation, the training objective consists of multi-scale spatial and frequency-domain reconstruction losses. Random horizontal flipping is applied for data augmentation.

**Concertormer Training** For Concertormer (Kuo et al., 2025), we use a lightweight variant with the channel width reduced to 24 and the block depths set to $[2, 2, 7, 2, 7, 2, 2]$, while retaining the original block design. The model is trained for 30K, 150K, and 300K iterations in the 10%, 50%, and 100% data regimes, respectively, using randomly cropped $256 \times 256$ patches with a batch size of 8. These schedules maintain approximately the same number of effective epochs across the three data regimes. We retain the original training objective without modification. We use AdamW with an initial learning rate of $1 \times 10^{-3}$, which is decayed to $1 \times 10^{-7}$ using a cosine annealing scheduler.

**NAFNet Training** We use NAFNet (Chen et al., 2022) with a channel width of 64 and the encoder–decoder configuration of $[1, 1, 1, 28]$ encoder blocks, one middle block, and $[1, 1, 1, 1]$ decoder blocks. The network is trained for 550K iterations using $256 \times 256$ randomly cropped patches with a batch size of 16. We retain the original training objective without modification. We use AdamW with $\beta_1 = \beta_2 = 0.9$. The initial learning rate is set to $1 \times 10^{-3}$ and decayed to $1 \times 10^{-7}$ using a cosine annealing scheduler. Random flipping and rotation are applied for data augmentation.

# B Additional Analysis

## B.1 Reblurring Reconstruction Analysis

Table 4: Reblurring performance for different numbers of local trajectory components $K$ on the GoPro dataset. All models are trained on the full GoPro training set and evaluated on the official GoPro test split. $K$ includes one fixed zero-motion component.

| Local Components | PSNR↑ | LPIPS↓ | DISTS↓ |
|---|---|---|---|
| GTA only (no LTA) | 34.17 | 0.082 | 0.059 |
| $K = 2$ | 38.87 | 0.034 | 0.031 |
| $K = 4$ | 41.55 | 0.024 | 0.023 |
| $K = 8$ | 41.73 | 0.023 | 0.022 |

Table 4 evaluates the effect of the number of local trajectory components on reblurring reconstruction. For this experiment, each reblurring network is trained using the full GoPro training set (Nah et al., 2017) and evaluated on the official GoPro test split. Adding local trajectory components substantially improves reconstruction performance over GTA alone, indicating the importance of modeling spatially varying residual motion. Performance continues to improve as $K$ increases, but $K = 4$ captures most of the reconstruction gain, with only marginal additional improvement at $K = 8$.

## B.2 Trajectory Primitive Ablation

Table 5: Ablation of trajectory primitives under the 10% GoPro training setting. "Motion" denotes perturbations of motion direction, temporal acceleration, and global rotation.

| Motion | Curve | S-curve | GoPro | | | | HIDE | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | MUSIQ↑ | LPIPS↓ | PSNR↑ | DISTS↓ | MUSIQ↑ | LPIPS↓ | PSNR↑ | DISTS↓ |
| | | | 38.49 | 0.155 | 28.92 | 0.097 | 43.64 | 0.192 | 26.71 | 0.113 |
| ✓ | | | 43.04 | 0.106 | 30.94 | 0.076 | 49.71 | 0.153 | 28.58 | 0.084 |
| ✓ | ✓ | | 43.34 | 0.102 | 31.26 | 0.077 | 51.21 | 0.158 | 28.54 | 0.085 |
| ✓ | | ✓ | 42.89 | 0.104 | 31.19 | 0.078 | 50.76 | 0.153 | 28.72 | 0.086 |
| ✓ | ✓ | ✓ | **43.42** | **0.099** | **31.37** | **0.074** | **51.37** | **0.142** | **28.98** | **0.081** |

Table 5 evaluates the contribution of individual trajectory primitives under the 10% GoPro training setting. All variants use the same training subset and optimization settings and are evaluated on the GoPro and HIDE test sets. Here, Motion denotes perturbations of motion direction, temporal acceleration, and global rotation, while Curve and S-curve additionally perturb the corresponding

trajectory primitives. Basic motion perturbation provides a substantial gain over the baseline, while curve and S-curve modeling introduce complementary geometric trajectory variations. Combining all trajectory primitives achieves the best performance across all metrics on both datasets.

### B.3 Trajectory Clustering

To analyze trajectory-shape diversity in Figure 1(c), we compare the original GoPro training blurry images, the publicly released ID-Blau reblurred dataset (Wu et al., 2024), and our augmented blurry images. We use ID-Blau as a representative existing method for this analysis, as both ID-Blau and GeoSyn primarily manipulate motion magnitude and orientation, while the GeoSyn blur-synthesis model is not publicly available. For all three groups, we use our pretrained trajectory-aware reblurring network to predict global trajectory parameters from each blurry image and its corresponding sharp image.

To compare trajectory geometry independently of motion magnitude and orientation, each trajectory is translated to start at the origin, normalized by its total path length, and resampled at uniform arc-length intervals. We then align its initial direction with the horizontal axis and vertically reflect the trajectory when the $y$-coordinate of its endpoint is negative, ensuring a consistent orientation.

The normalized trajectory coordinates are flattened into feature vectors and standardized before applying $K$-means clustering with 10 clusters. The standardization statistics and cluster centers are fitted using trajectories estimated from the original GoPro training blurry images. For both the publicly released ID-Blau reblurred images and our augmented images, the estimated trajectories are processed using the same normalization and standardization procedure and assigned to the nearest cluster center.

We compare the resulting cluster distributions to examine how augmentation changes trajectory-shape coverage. This analysis focuses on geometric trajectory shape rather than temporal dynamics, since arc-length resampling removes the original temporal spacing. The resulting distributions characterize trajectory shapes estimated by our model rather than ground-truth motion trajectories.

## C Additional Results

### C.1 Additional Qualitative Results

We provide additional qualitative comparisons on the GoPro and HIDE datasets.

Figure 7 presents additional results on the GoPro test set using 10% and 50% of the training data in panels (a) and (b), respectively, complementing the quantitative evaluation under limited-data settings. Figure 8 presents additional results on HIDE to examine generalization across datasets. These comparisons focus on the restoration of fine structures and object boundaries, as well as residual blur and artifacts. Our method effectively restores fine structures while suppressing residual blur and undesired artifacts.

### C.2 Additional Trajectory Visualization

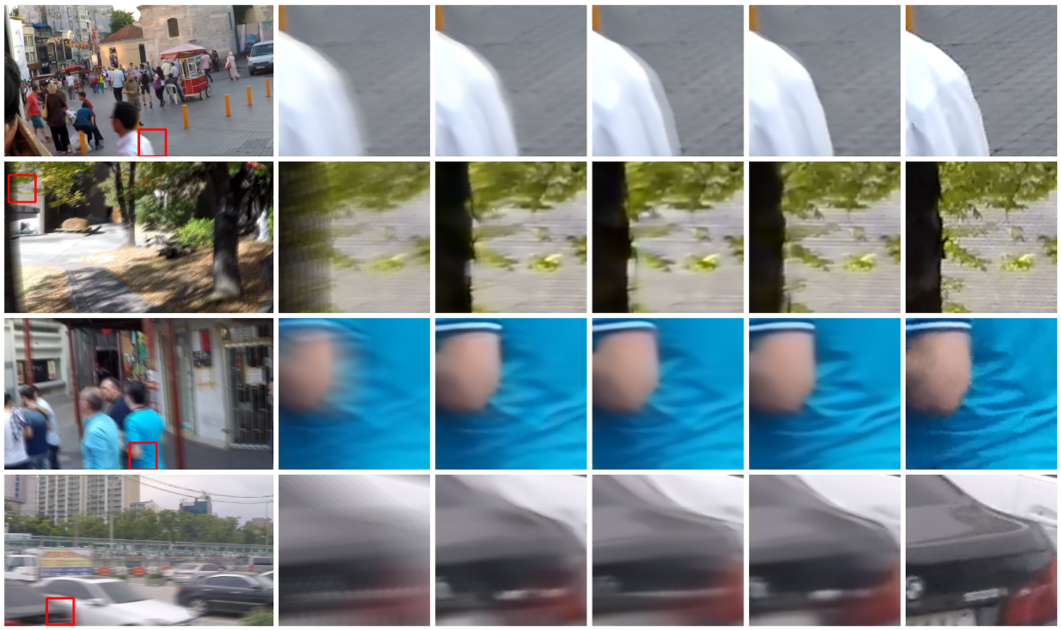
Figure 9 presents additional examples of estimated global motion trajectories and their augmented variants. The comparisons illustrate how trajectory-aware augmentation modifies motion dynamics and path geometry, producing diverse motion patterns from each estimated trajectory. The corresponding synthesized images exhibit natural-looking blur that reflects these trajectory variations while preserving the underlying scene content.

### C.3 Additional Local Trajectory Visualization

Figure 10 provides additional examples of local trajectory augmentation and the corresponding synthesized blurred images. By perturbing local motion trajectories while retaining their spatial mixing weights, our method introduces diverse, spatially varying blur patterns. The resulting images exhibit natural-looking local blur while preserving the underlying scene content.




(a) Additional qualitative results on the GoPro test set using 10% of the training data.




(b) Additional qualitative results on the GoPro test set using 50% of the training data.

Figure 7: Additional qualitative results on the GoPro dataset.

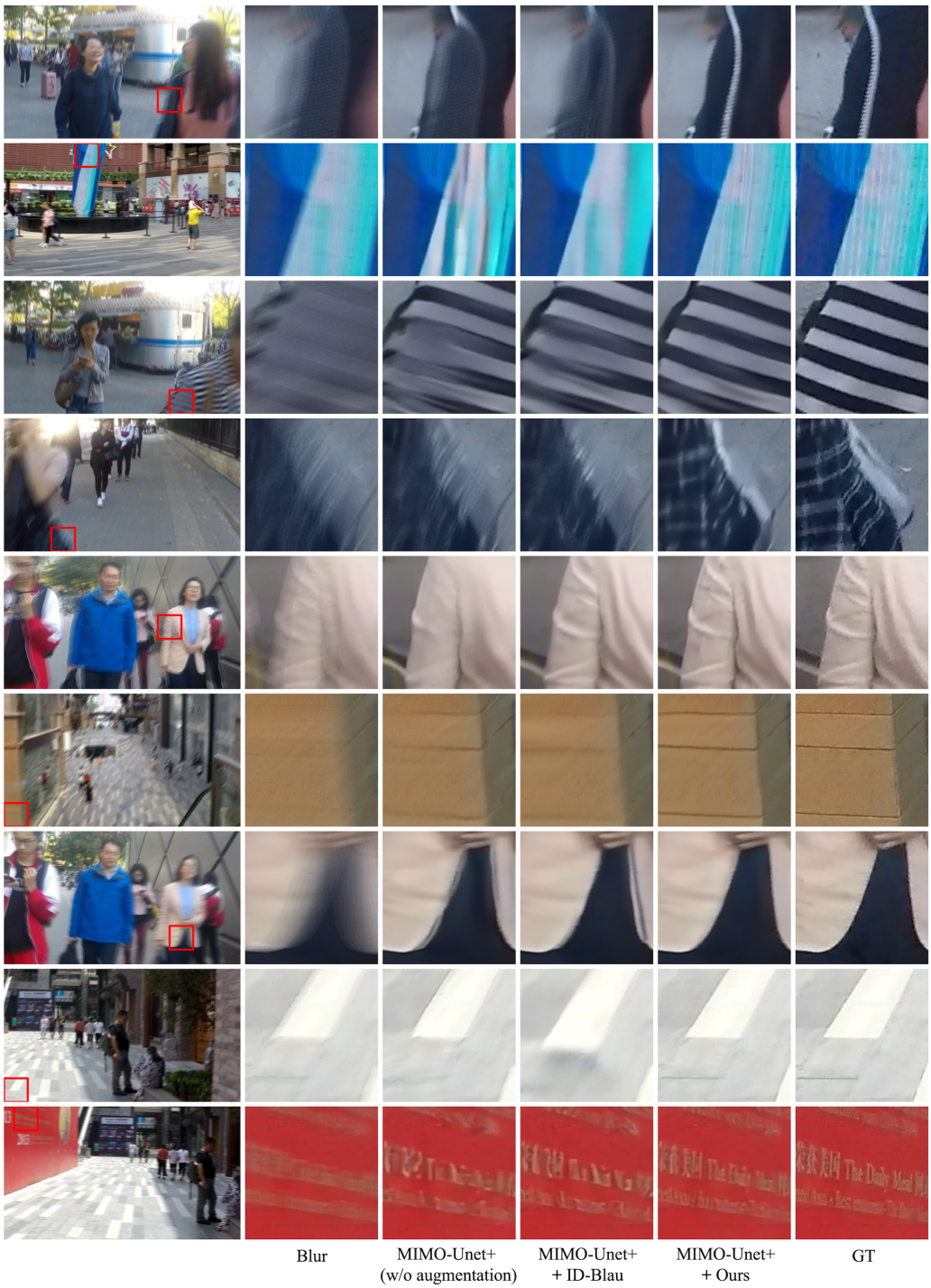



Figure 8: Qualitative results on the HIDE dataset.




Figure 9: Additional visualizations of global motion trajectories and their augmented variants.




Figure 10: Additional local trajectory visualizations with spatial mixing weights (left) and synthesized blur comparisons before and after augmentation (right).