Supplementary Material. EvShutter: Transforming Events for Unconstrained Rolling Shutter Correction

1. Implementation Details

1.1. Training

We train the model by using the training set of the proposed RS-ERGB dataset using an Adam optimizer [3] with a batch size equal to 8 and a learning rate of $8e-4$. In the case of the optical flow module, we set the learning rate to $2e-4$. We use the ReduceLROnPlateau learning rate scheduler with a patience of 10 epochs and an update factor of 0.5. We train the model for a total of 200 epochs, which took approximately 24 hours on an NVIDIA V100. We use the optical flow module proposed by [6] and fine-tune it from their pre-trained weights.

To compare to other state-of-the-art methods on our RS-ERGB dataset, we fine-tuned the image-based methods DSUN [4] and RSCD [10] starting from their publicly available checkpoints for the Fastec-RS [4] dataset. We also tried to fine-tune RSCD [10] starting from their BS-RSCD checkpoint, but achieved worse results compared to starting from the Fastec-RS checkpoint. For our comparisons we used the weights obtained from finetuning on Fastec-RS. For EvUnroll [11], we only initialize the weights of the flow network from their checkpoint, which we then fine-tuned on our dataset with the procedure described in their paper. We were not able to make their flow network converge when initialized from scratch. All the methods have been trained using the training procedure and hyperparameters as described in their respective papers.

1.2. Losses

We train our model with exception of the optical flow network using a combination of the Charbonnier loss [1] $\mathcal{L}_C$ and the LPIPS loss [2] $\mathcal{L}_P$ using the features from the VGG-net [9].

$$\mathcal{L}_{RS} = \mathcal{L}_P + \lambda_1 \mathcal{L}_C \tag{1}$$

We observed that training the optical flow network separately leads to better overall performance. Therefore, during training, we fine-tune the optical flow network in an unsupervised way by warping the RS-image to the corresponding GS-image, at each scale (1, $1/2$ and $1/4$), using the predicted optical flow. As loss, we use a combination of the Structural Similarity Metric (SSIM) [8] $\mathcal{L}_{SSIM}$ and Charbonnier loss [1] $\mathcal{L}_C$, which are averaged over all image location at each each scale. The warping loss is then defined as

$$\mathcal{L}_{flow} = \sum_{s=1}^{3} \lambda_2 \mathcal{L}_{SSIM,s} + \lambda_3 \mathcal{L}_{C,s} \tag{2}$$

with $\lambda_2 = 0.85$ and $\lambda_3 = 0.15$ respectively as in [6]. During the fine-tuning of the optical flow we do not propagate the gradient of the other losses through the warped features.

1.3. Runtime and capacity

Tab. 1 shows the runtime and parameters comparison for Rolling Shutter (RS) correction methods. Our algorithm achieves the best runtime with 327 ms (more than two times quicker than the second best method). On the other hand, our method has more parameters than other methods, most of them come from the Flow Net module (19.8 M), this could be easily fixed by using a lighter module.

Method	№Params [M]↓	Runtime [ms]↓
DeepUnrollNet [4]	**18.6**	2080
RSCD [10]	22.2	2822
EvUnroll [11]	27.5	783
Ours	36.0	**327**

Table 1. **Comparison of model capacity and runtime** on fHD images (1920×1080) for RS correction methods. We measure the runtime on NVIDIA V100 GPU. Shortest runtime and lowest model capacity are shown in bold.

2. Ablation Study

Complementarity of synthesis and warping encoder. In Tab. 2 we show the benefits of combining our synthesis and warping encoders compared to versions of our algorithm that uses either synthesis or warping for RS correction. The combination of the two encoders improves results by 2.42 dB in PSNR and 0.0117 in LPIPS compared to a purely warping-based approach and by 1.79 dB in PSNR and 0.0357 in LPIPS compared to a purely synthesis-based approach. This result shows that synthesis and warping encoders complement each other for the task of RS-correction.

Besides taking care of image areas with brightness changes, in the case of RS correction the synthesis module also helps to fill occluded areas that can not be resolved by using just the warping-based module relying on a single image.

Method	PSNR [dB]↑	LPIPS↓	SSIM↑
Synthesis	29.79	0.1857	0.8668
Warping	29.16	0.1617	0.8828
Both	**31.58**	**0.1500**	**0.8975**

Table 2. **Complementarity of synthesis and warping encoders** shown on validation set of RS-ERGB. The combination of synthesis and warping encoder (in bold) boosts performance compared to only using one of them.

Decoder improvements. Our fusion decoder shown in Fig. 3 is inspired by [10], however we improved the original version in two ways. First, we found that in the Fusion Block, it is sub-optimal to place 1×1 convolution that compresses features between SE and Deformable Convolution blocks, and instead place it after the Deformable Convolution block. We presume this is because the convolution mixes the features coming from different encoders and precludes the Deformable Convolution block to perform channel- and location-wise attenuation. In Tab. 3 we show that our design, which we call Late Compression, boosts the performance by 5.24dB in PSNR and 0.0453 in LPIPS compared to the original design that we call Early Compression. Furthermore, we found that adding the warped images $I^{rs}_{\rightarrow}$ and the input RS-images I^{rs} to the fusion decoder on all scales improves the results by 2.59 dB in PSNR and 0.0154 in LPIPS as shown in Tab. 3. The shortcuts help the network to recover static parts of the image without RS artifacts.

Method	PSNR↑	LPIPS↓	SSIM↑
Position of 1×1 compressing convolution			
Early Compression	28.04	0.1890	0.849
Late Compression (Ours)	**33.28**	**0.1437**	**0.903**
Multi-scale image shortcut connections			
No image shortcuts	30.69	0.1591	0.881
Image shortcuts (Ours)	**33.28**	**0.1437**	**0.903**

Table 3. **Fusion decoder architecture improvement** ablations on validation set of RS-ERGB. The late compression and image shortcuts in the decoder (in bold) both give a boost in the performance.

Visualization of FnF transformed events is shown in Fig. 1. In this example the raw event stream $E_{0\rightarrow t_H}$ and transformed events $E_{gs\rightarrow rs}$ can be compared. It can be seen that FnF transformed events only encode brightness changes from GS to RS frame. For example, the filtered event stream has no events in the middle row of the image as mid-exposure of the latent GS image corresponds to mid-exposure of the middle row of the RS image. Furthermore, the polarity of events in the top half of the image is flipped.

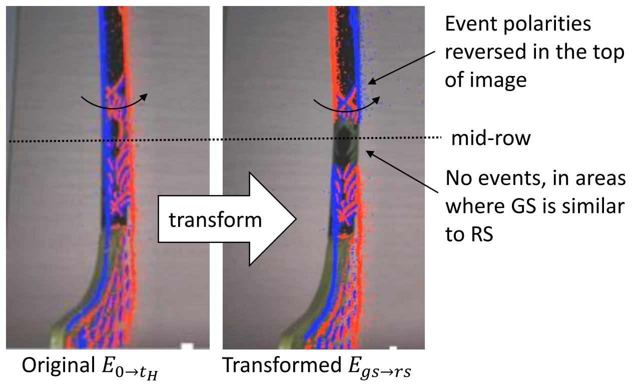

Event polarities reversed in the top of image
mid-row
No events, in areas where GS is similar to RS
transform
Original $E_{0\rightarrow t_H}$
Transformed $E_{gs\rightarrow rs}$

Figure 1. **Visualization of FnF transformed events.** For better illustration we only visualize a temporal slice of events.

3. Dataset and RS simulator

Recording setup shown in Fig. 2 consist in a 50R/50T beamsplitter with a global shutter RGB Flir Blackfly S 1440x1080 and a Prophesee Gen4M 1280x720 event camera. The two devices are hardware time synchronized: every time the RGB camera starts and ends an exposure it sends a trigger signal to the event camera that records the timestamps. Using this mechanism we can obtain the timing of start and end of exposure in the event camera clock, which we then use as reference.

Geometric alignment. To geometrically align events and images we physically place the cameras to approximately intersect their optical axis to avoid parallax, adjust focal length to match field of view and focus both camera at the same distance. Next, we calibrate each camera separately to estimate the lens distortion parameters

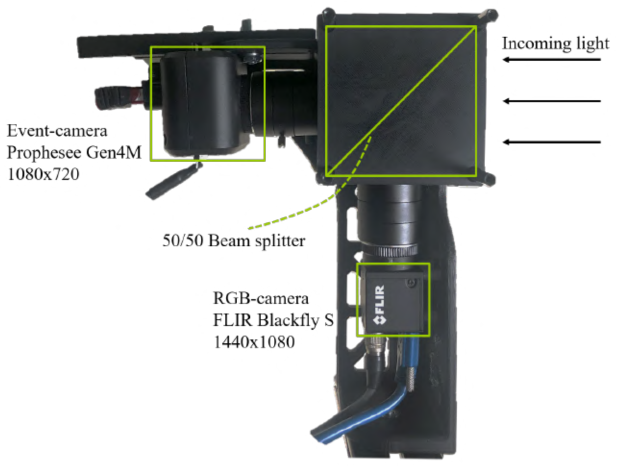


Figure 2. **Data acquisition setup.**

and focal lengths using images produced from events using E2VID [5]. After removing the lens distortion, we estimate a homography that compensates for the misalignment between events and images.

Dataset details. We collect 34 sequences, each 10 seconds long, and set aside 22 sequences for training, 6 for validation and 6 for test. For capturing data we used 500-2000 microseconds exposure, depending on lighting conditions. Note that this exposure is significantly higher than 175 microseconds exposure used in Gev-RS [11] and it allows us capturing images of higher quality. Our advance simulation procedure, described below, allows generating frames required for simulating realistic RS image with blur even from low frame rate videos. We captured sequences that are particularly challenging for a RS sensor such as fast moving and rotating objects, fast panning or tilting motions of the camera in scenes with varying depth.

Importance of adaptive interpolation in RS simulator In Fig. 3 and Fig. 4 we show more comparisons of our RS-simulator to a version without adaptive interpolation. For the simulator with a fixed temporal upscale rate selecting appropriate upscale rate can be tricky since even single scene often contains motion with different speed. Selecting too low upscale rate leads to aliasing artifacts in simulated RS images (e.g. fan), while selecting too high upscale rate is prohibitively expensive. Our adaptive interpolation scheme solves this issue and ensures a sufficiently high temporal resolution regardless of the speed of the motion.

4. Qualitative comparison to SOTA

Fastec-RS. On the public Fastec-RS dataset, we only compare our method qualitatively to the image based methods DSUN [4] and RSCD [10] as there is no public checkpoints available for EvUnroll [11]. Examples are shown in Fig. 5 and Fig. 6.

Data with real events and RS [11]. In Fig. 7, we show additional comparisons on the dataset published in [11] with real RS images and real events. Notice, that using the method proposed in [11] but fine-tuned in our proposed dataset, it improves the overall performance in real data. Additionally, our model trained on the proposed RS-ERGB dataset outperform the proposed method by [11] train on their dataset and on RS-ERGB.

RS-ERGB. In Fig. 8 and Fig. 9, we show additional examples from the proposed RS-ERGB dataset and compare our method to RSCD [10], DSUN [4] and EvUnroll [11]. All methods were fine-tuned on this dataset.

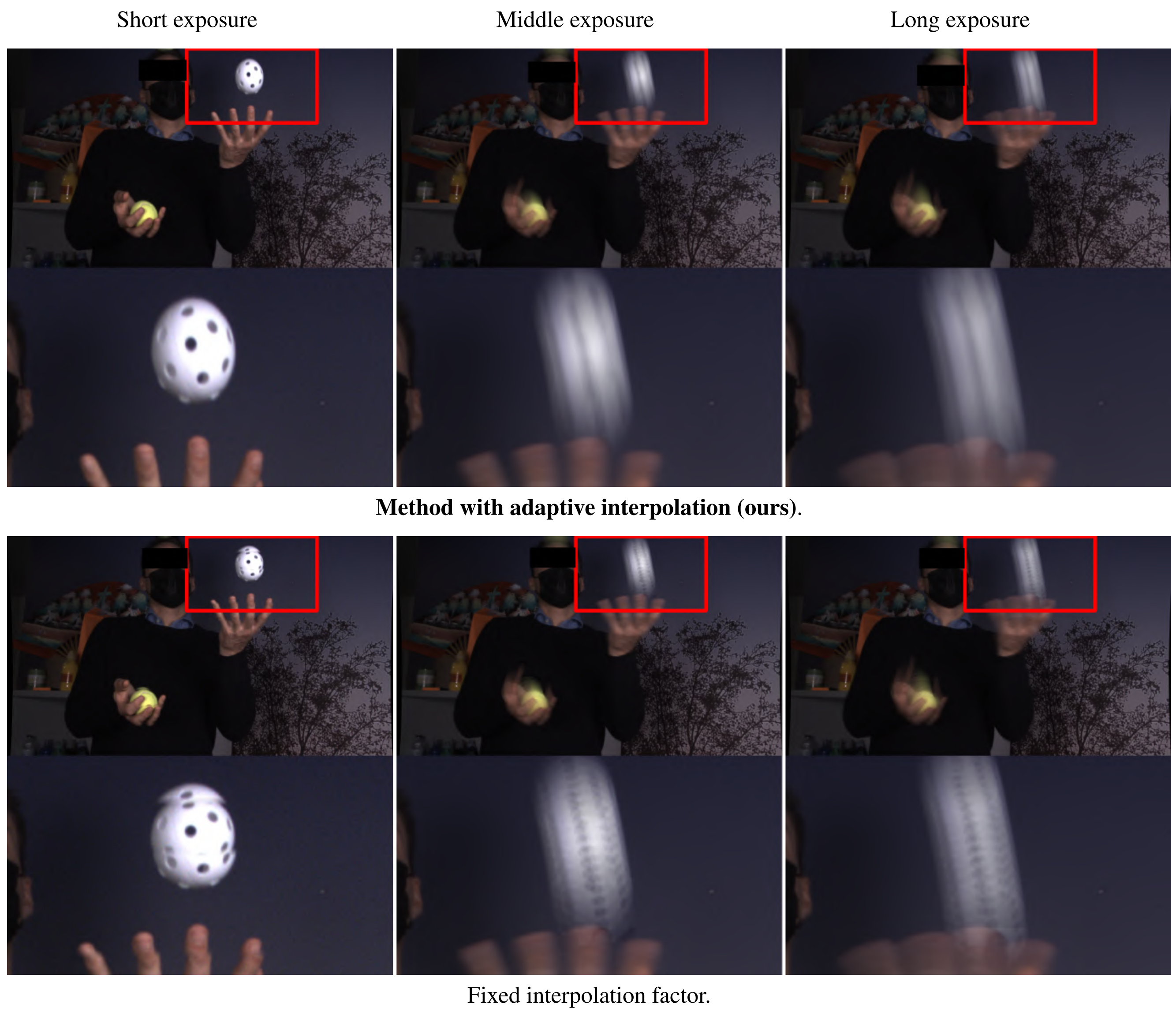

Short exposure
Middle exposure
Long exposure
Method with adaptive interpolation (ours).
Fixed interpolation factor.

Figure 3. **Importance of adaptive interpolation** in the proposed RS image simulator. Note that with fixed factor video interpolation it is very hard to create realistic RS image with blur.

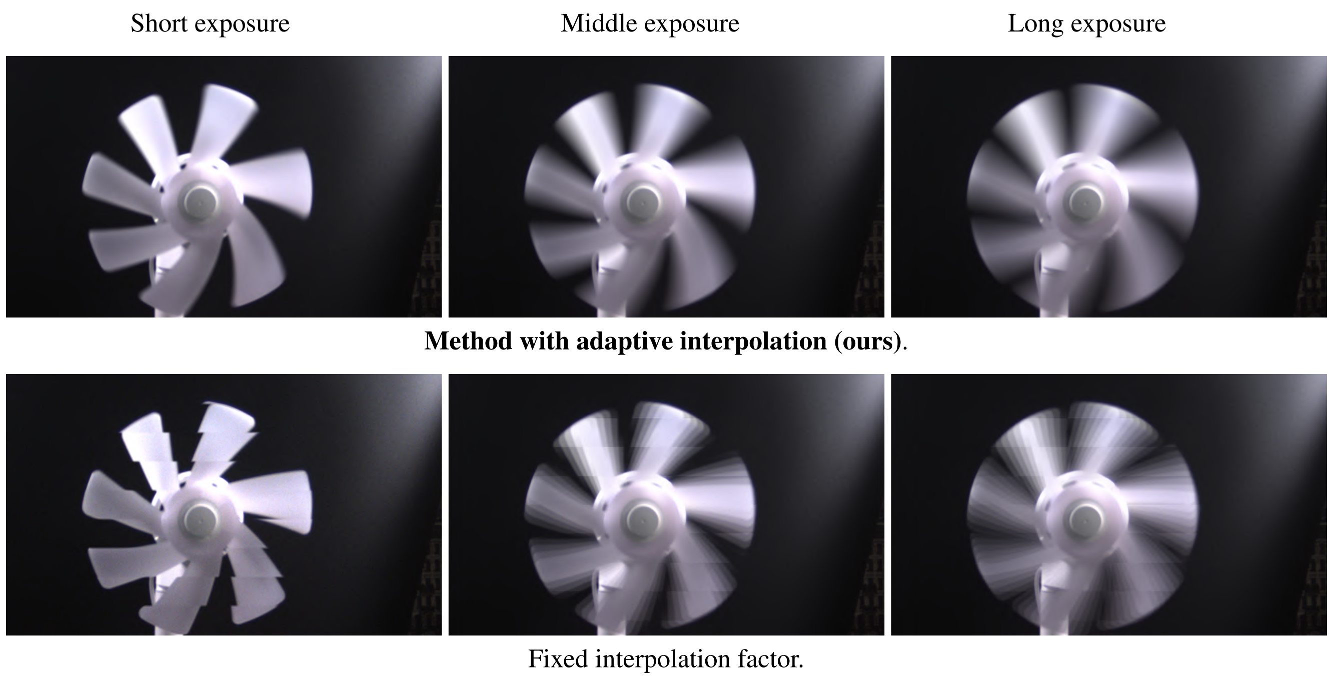

Short exposure
Middle exposure
Long exposure
Method with adaptive interpolation (ours).
Fixed interpolation factor.

Figure 4. **Importance of adaptive interpolation** in the proposed RS image simulator. Note that with fixed factor video interpolation it is very hard to create realistic RS image with blur.

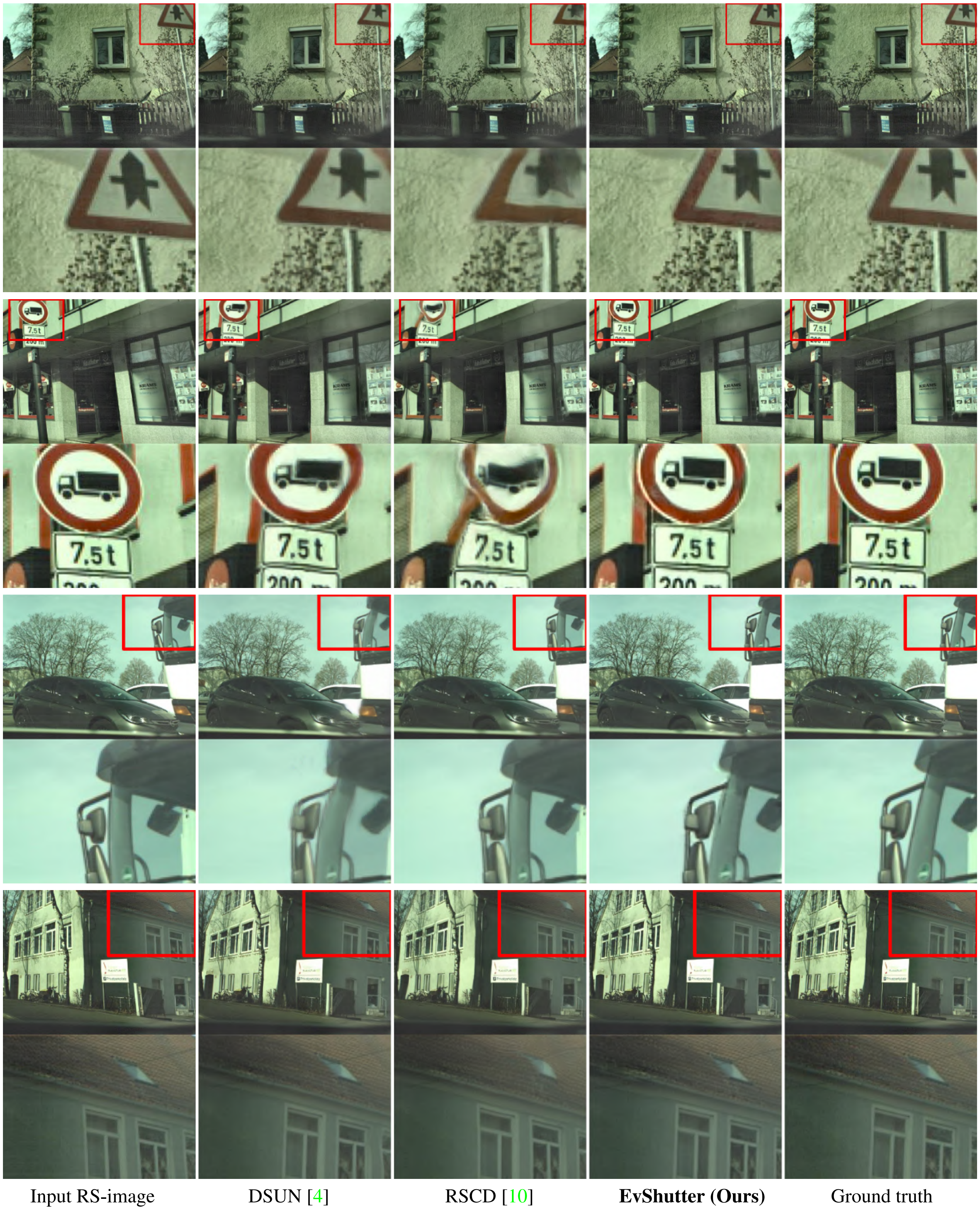


Figure 5. **Comparisons on Fastec-RS dataset.**

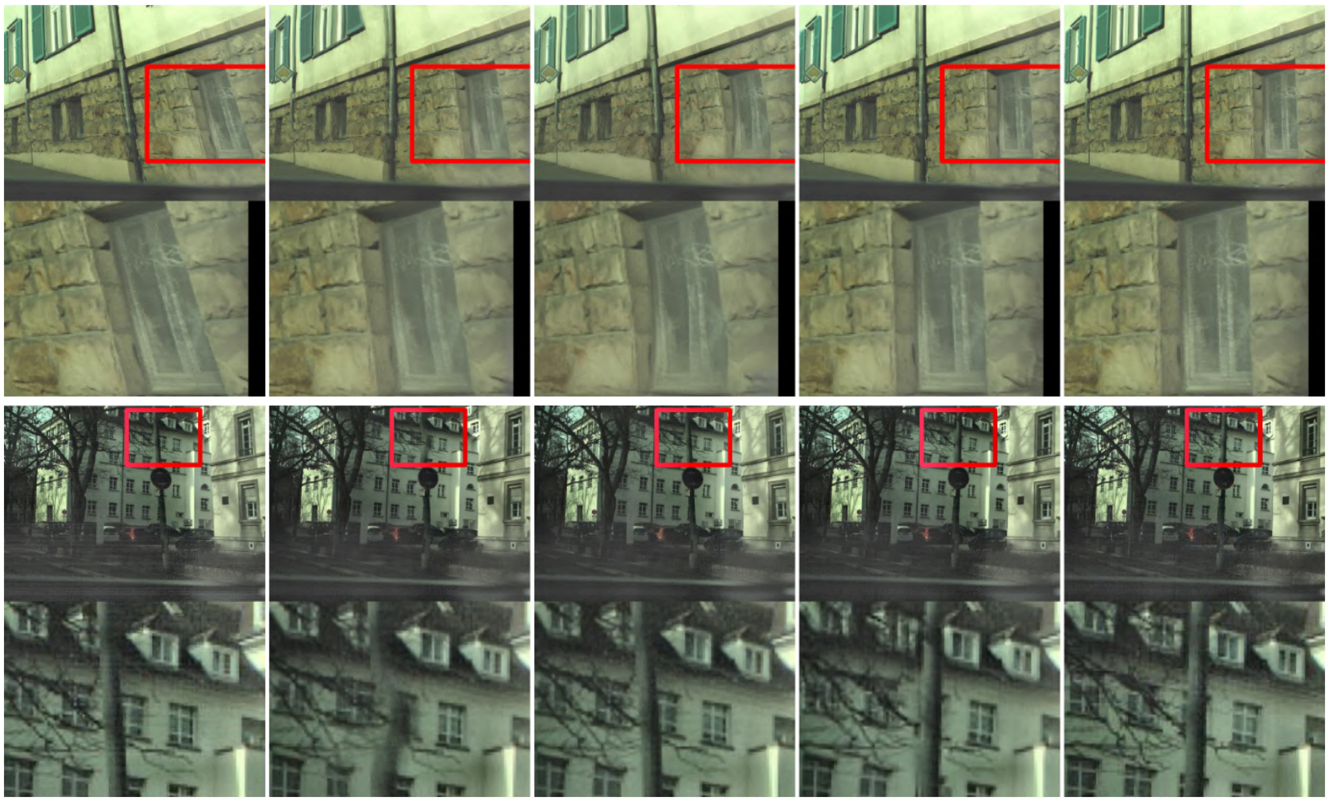


Figure 6. **Comparisons on Fastec-RS dataset.**

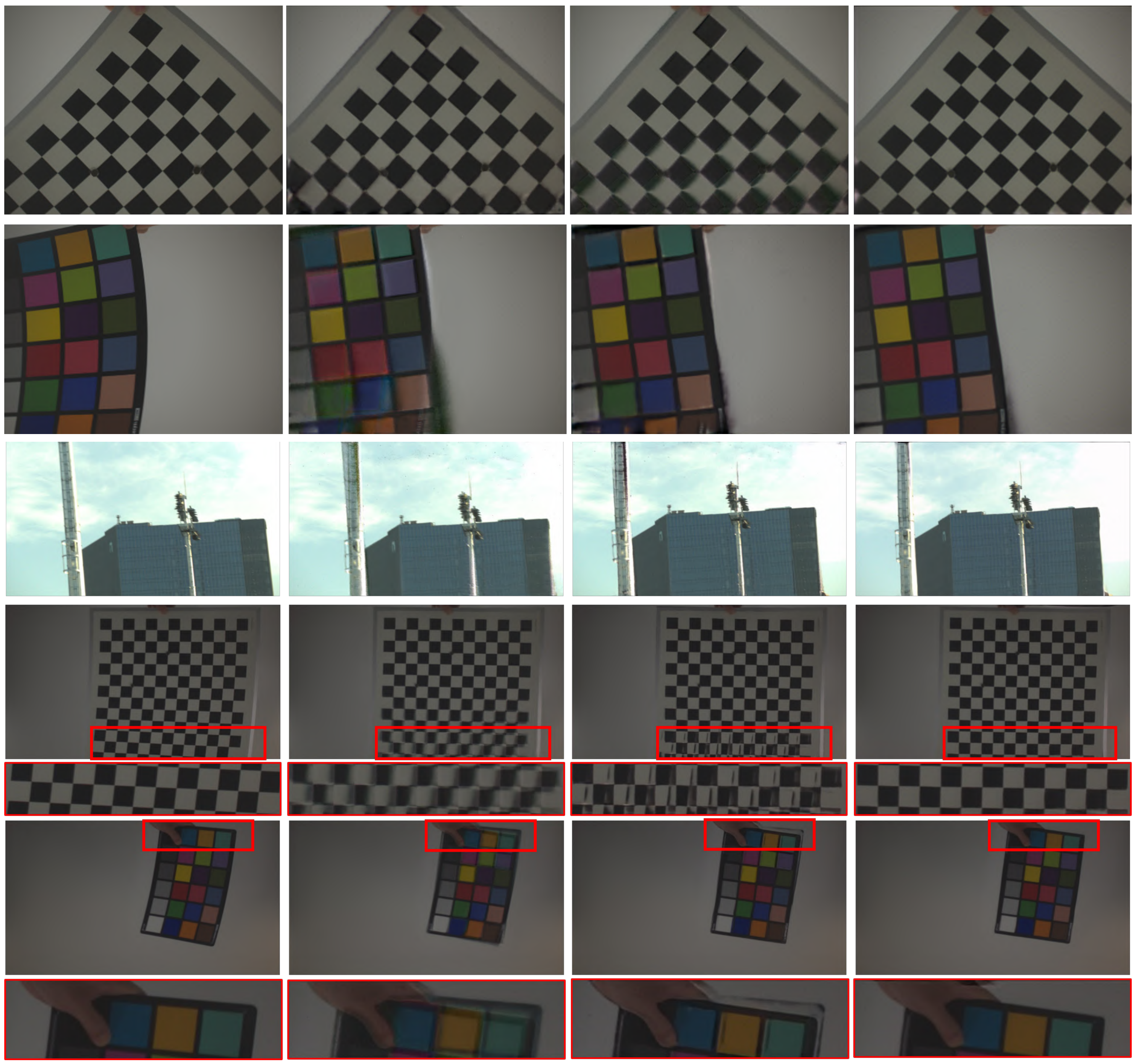


Figure 7. **Comparisons on data with real events and RS [11]**. Note that this dataset does not have ground truth data.

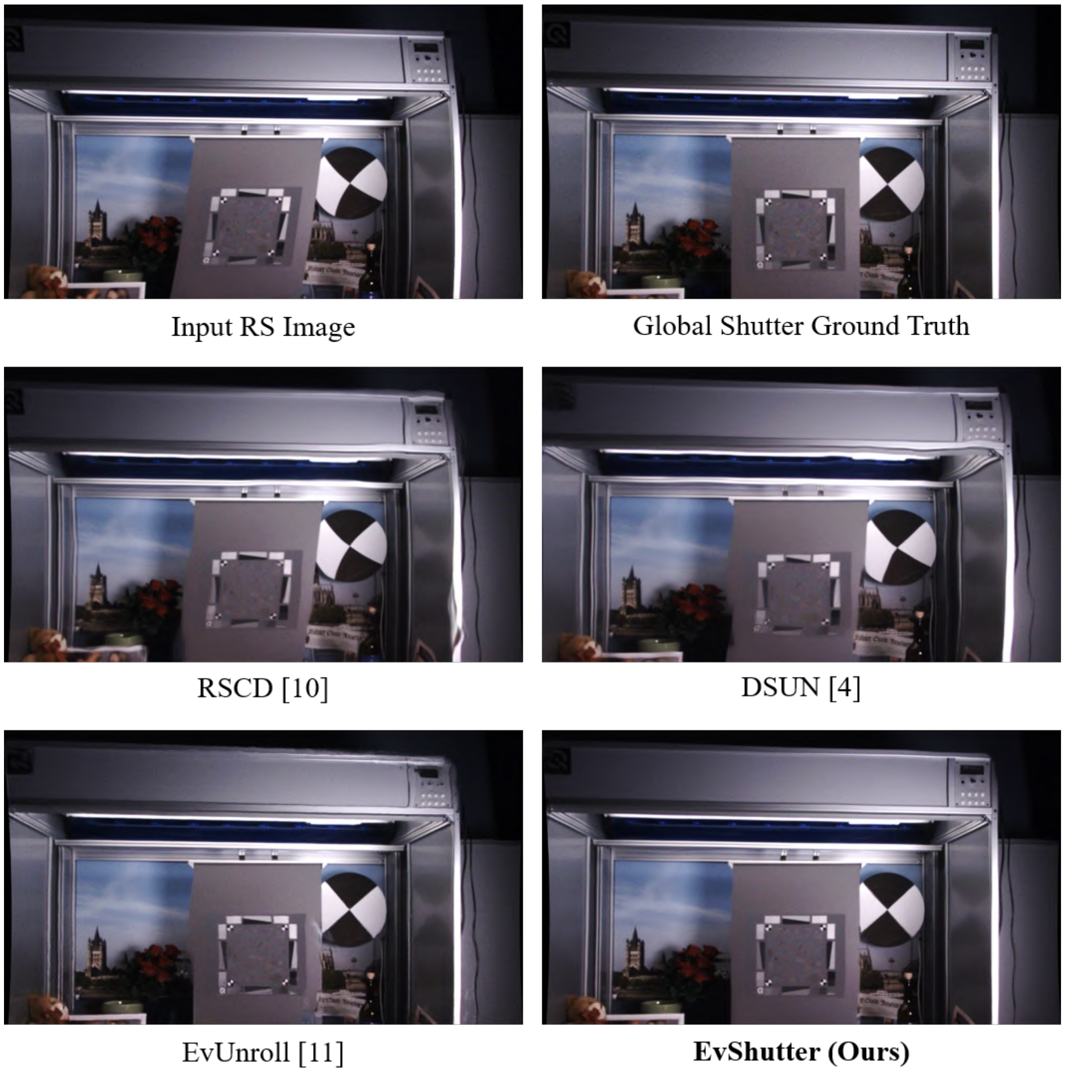


Figure 8. **Comparisons on RS-ERGB dataset.**

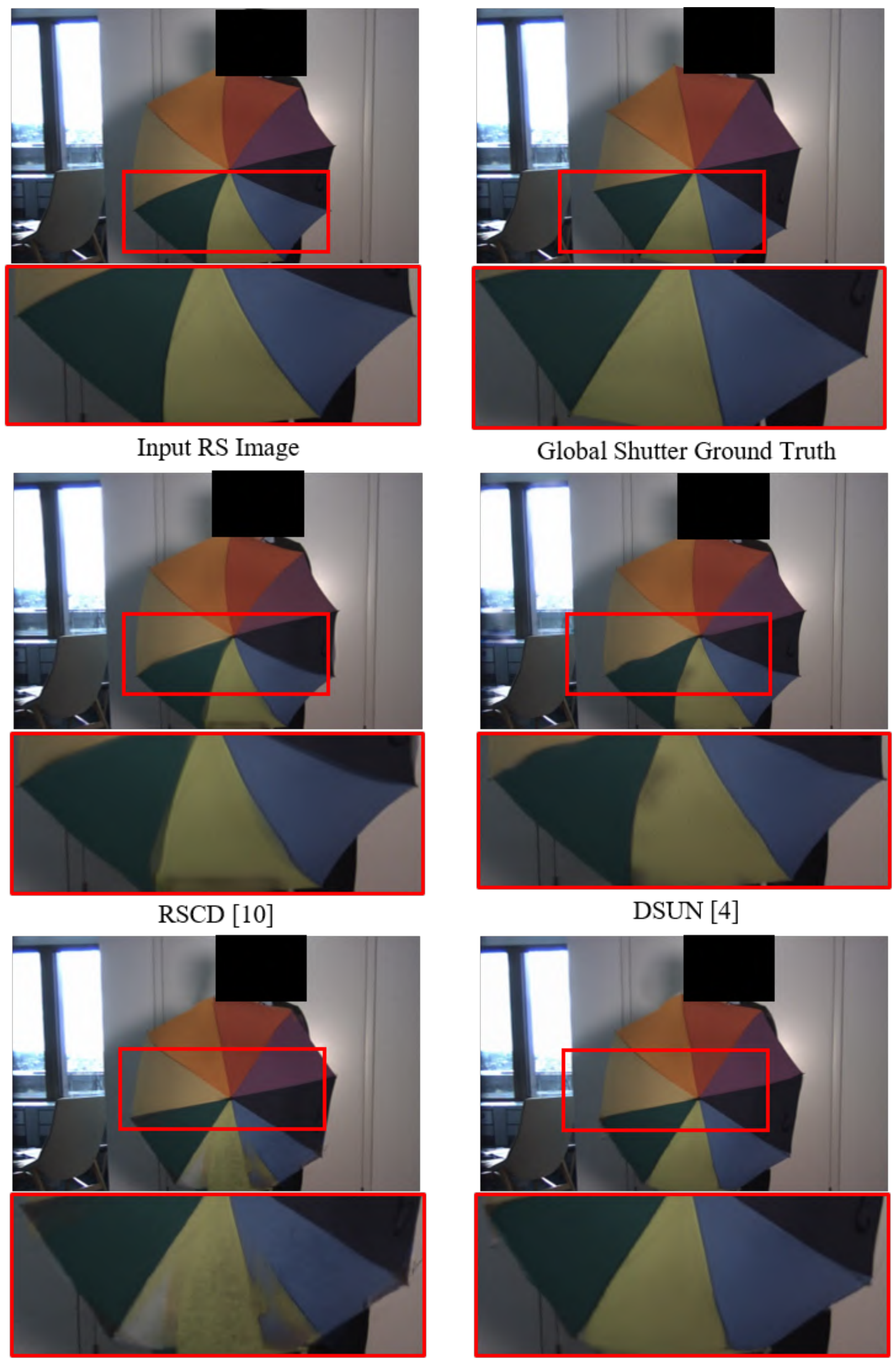
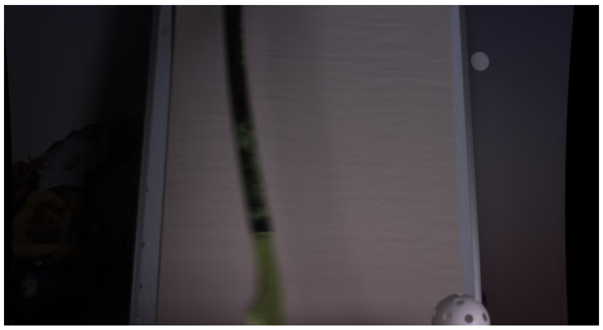
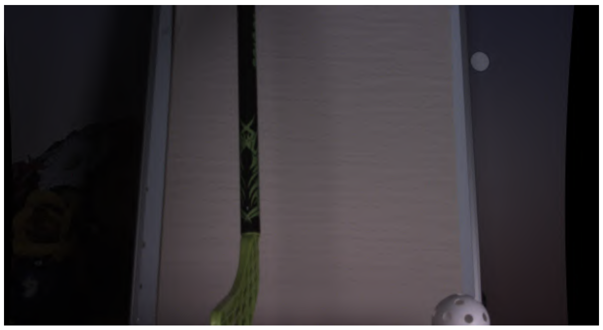
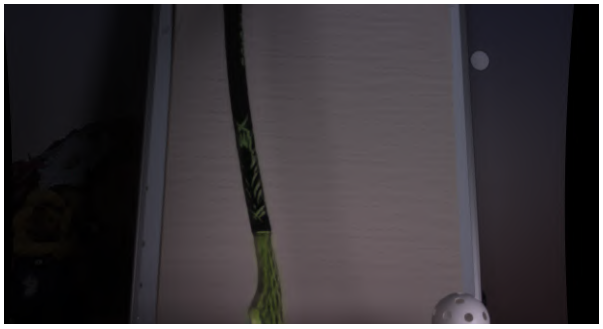
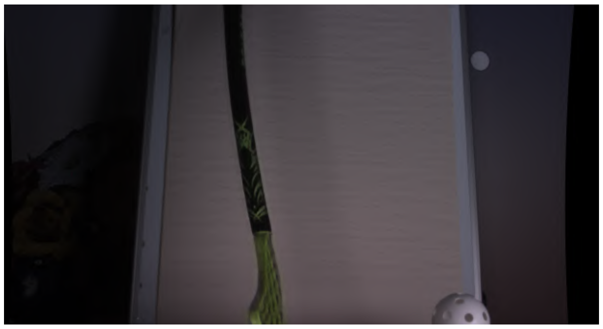


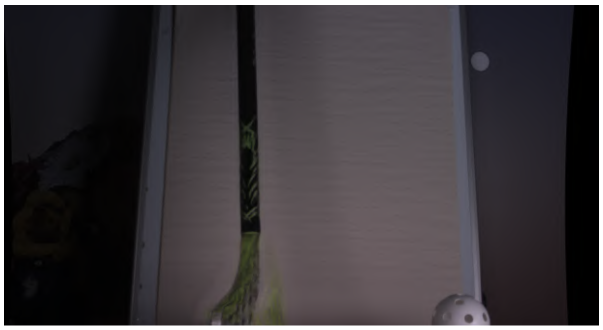
Figure 9. **Comparisons on RS-ERGB dataset.**

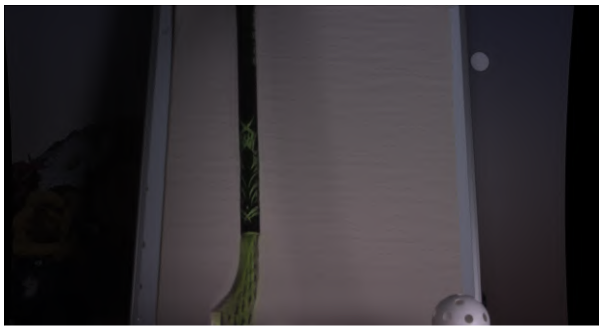

Input blurry RS-image


Ground truth global shutter


EvUnroll [11] deblurring result


Our deblurring result using the flow based adaption of [7]


EvUnroll [11] deblurred and RS corrected


EvShutter (Ours)

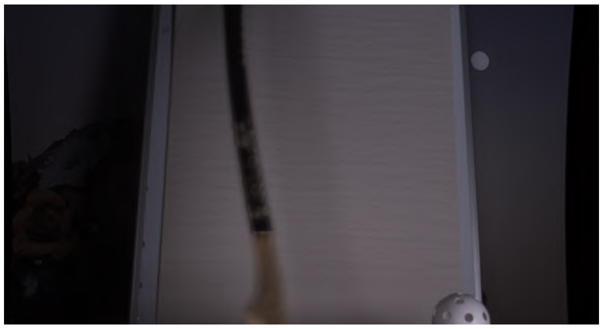
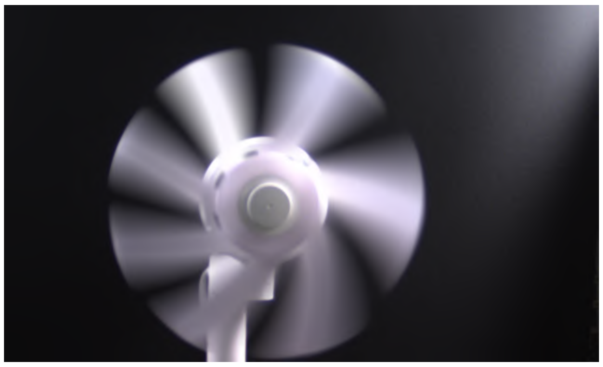
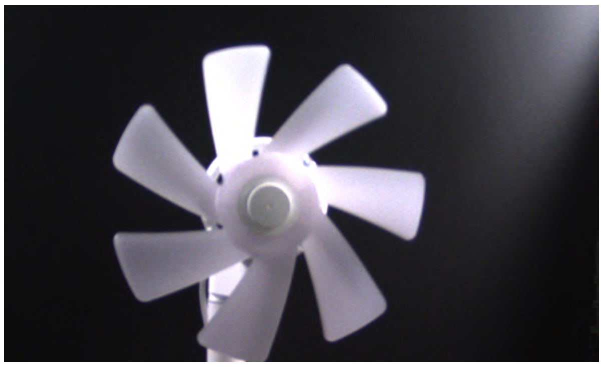
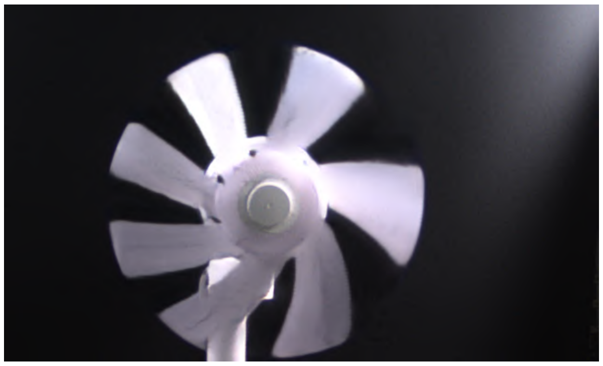
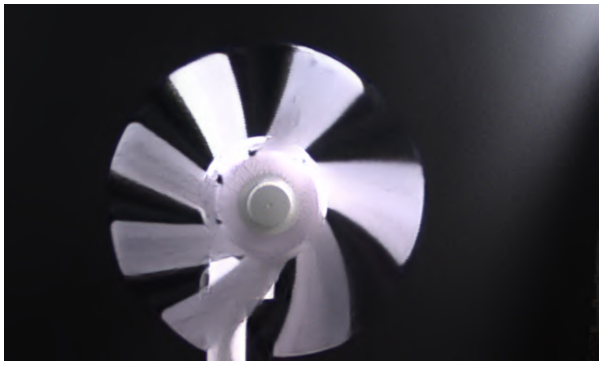

RSCD [10]

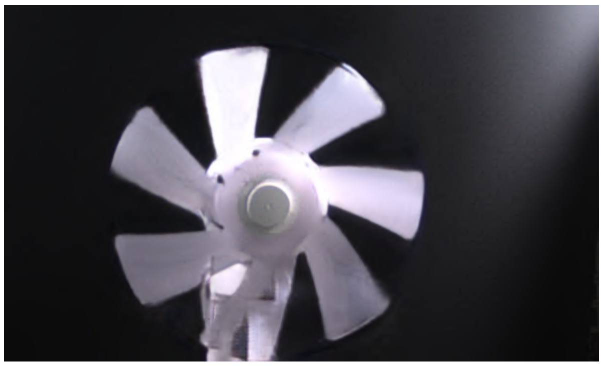
Figure 10. **Comparison on RS-ERGB-Blur dataset.**

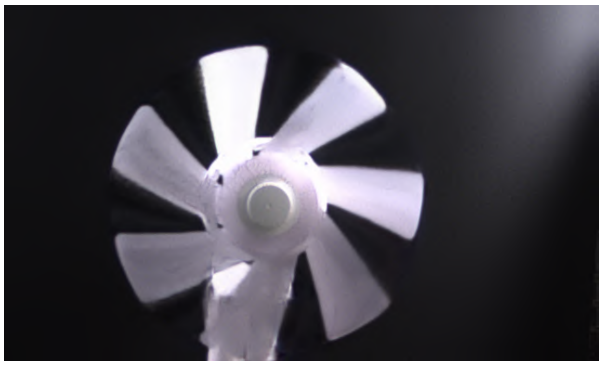

Input blurry RS-image


Ground truth global shutter


EvUnroll [11] deblurring result


Our deblurring result using the flow based adaption of [7]


EvUnroll [11] deblurred and RS corrected


EvShutter (Ours)

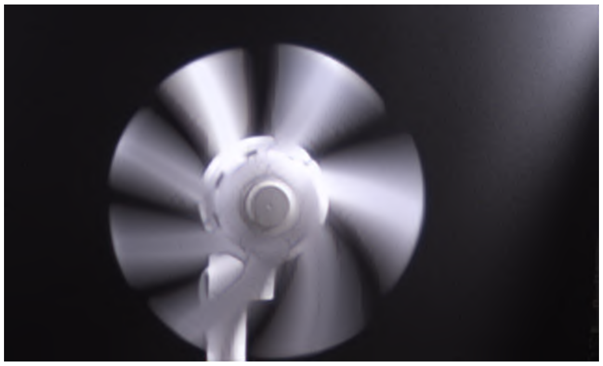

RSCD [10]

Figure 11. **Comparison on RS-ERGB-Blur dataset.**

References

[1] Pierre Charbonnier, Laure Blanc-Feraud, Gilles Aubert, and Michel Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Proceedings of 1st International Conference on Image Processing*, volume 2, pages 168–172. IEEE, 1994. 1

[2] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016. 1

[3] Diederik P. Kingma and Jimmy L. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. 1

[4] Peidong Liu, Zhaopeng Cui, Viktor Larsson, and Marc Pollefeys. Deep shutter unrolling network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5941–5949, 2020. 1, 3, 6, 7

[5] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. High speed and high dynamic range video with an event camera. *IEEE Trans. Pattern Anal. Mach. Intell. (T-PAMI)*, 2019. 3

[6] Stepan Tulyakov, Daniel Gehrig, Stamatios Georgoulis, Julius Erbach, Mathias Gehrig, Yuanyou Li, and Davide Scaramuzza. TimeLens: Event-based video frame interpolation. *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 1

[7] Patricia Vitoria, Stamatios Georgoulis, Stepan Tulyakov, Alfredo Bochicchio, Julius Erbach, and Yuanyou Li. Event-based image deblurring with dynamic motion awareness. *arXiv preprint arXiv:2208.11398*, 2022. 11, 12

[8] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 1

[9] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 1

[10] Zhihang Zhong, Yinqiang Zheng, and Imari Sato. Towards rolling shutter correction and deblurring in dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9219–9228, 2021. 1, 2, 3, 6, 7, 11, 12

[11] Xinyu Zhou, Peiqi Duan, Yi Ma, and Boxin Shi. Evunroll: Neuromorphic events based rolling shutter image correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17775–17784, June 2022. 1, 3, 8, 11, 12