EvShutter: Transforming Events for Unconstrained Rolling Shutter Correction

Julius Erbach Stepan Tulyakov Patricia Vitoria Alfredo Bochicchio
Yuanyou Li
Huawei Technologies, Zurich Research Center

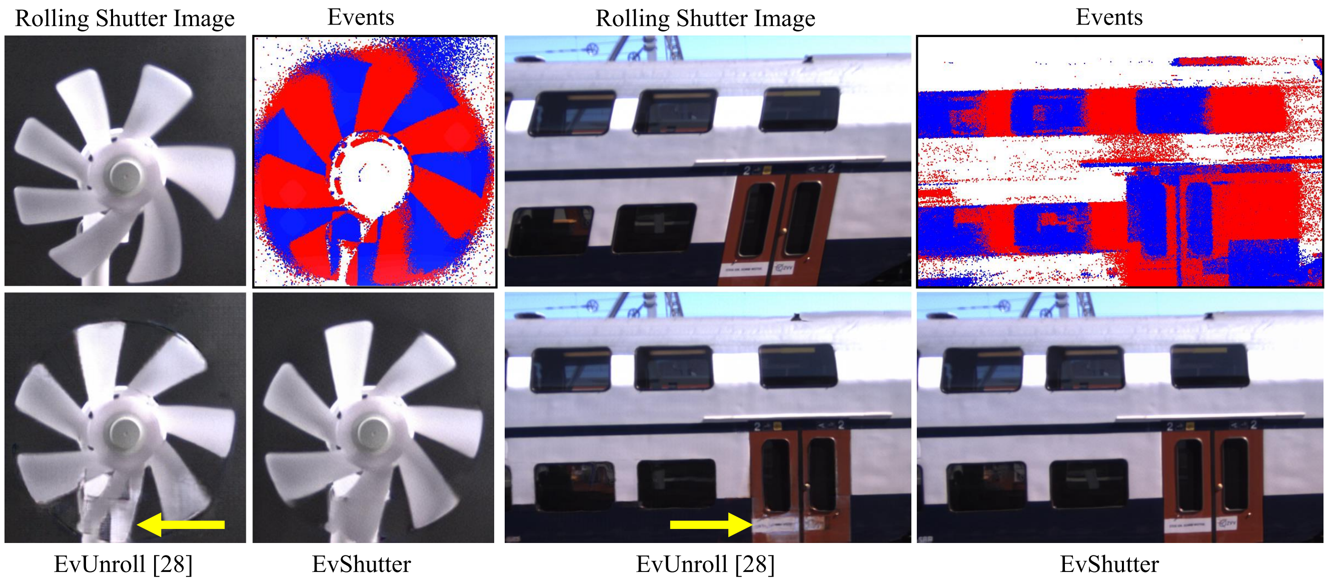

Figure 1. **Comparison of EvShutter to state-of-the-art event-based RS correction method [28].** Our method introduces a novel event transformation, which allows us to correct RS artifacts in the presence of non-linear motion or large displacements.

Abstract

Widely used Rolling Shutter (RS) CMOS sensors capture high resolution images at the expense of introducing distortions and artifacts in the presence of motion. In such situations, RS distortion correction algorithms are critical. Recent methods rely on a constant velocity assumption and require multiple frames to predict the dense displacement field. In this work, we introduce a new method, called Eventful Shutter (EvShutter)[1], that corrects RS artifacts using a single RGB image and event information with high temporal resolution. The method firstly removes blur using a novel flow-based deblurring module and then compensates RS using a double encoder hourglass network. In contrast to previous methods, it does not rely on a constant velocity assumption and uses a simple architecture thanks to an event transformation dedicated to RS, called Filter and Flip (FnF), that transforms input events to encode only the changes between GS and RS images. To evaluate the proposed method and facilitate future research, we collect the first dataset with real events and high-quality RS images with optional blur, called RS-ERGB. We generate the RS images from GS images using a newly proposed simulator based on adaptive interpolation. The simulator permits the use of inexpensive cameras with long exposure to capture high-quality GS images. We show that on this realistic dataset the proposed method outperforms the state-of-the-art image- and event-based methods by 9.16 dB and 0.75 dB respectively in terms of PSNR and an improvement of 23 % and 21 % in LPIPS.

1. Introduction

Most consumer cameras like cell phones or action cameras use a *rolling shutter (RS)* sensor which instead of capturing the whole frame in a single shot as in a *global shutter (GS)* camera, it acquires each row sequentially as shown in Fig. 2. Specifically, it obtains each row y at time

[1]The evaluation code and the dataset can be found here https://github.com/juliuserbach/EvShutter

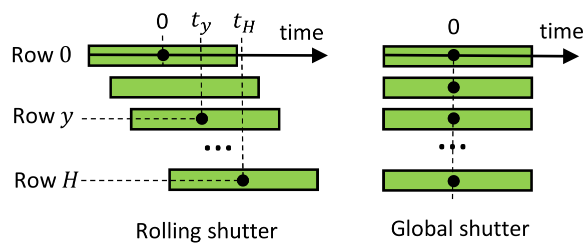

Figure 2. **Rolling Shutter Explained**. While recording with a RS camera (left), image rows are exposed sequentially, while in the case of a GS camera (right) they are exposed all at the same time.

$t_y = y \cdot t_H / H$, where H is the height of an image and t_H is the mid-exposure time of the last row. RS cameras are often prefered due to their high spatial resolution and high frame rates at low cost. Global shutter sensors have a more complicated circuitry, and therefore a smaller resolution than rolling shutter sensors of the same physical size. A larger sensor size in turn increases the cost of the lens. However, in the presence of fast camera or object movement, RS sensors can produce skew distortion in images (see Fig. 1) and jello artifacts in videos.

To correct such artifacts, RS correction methods are widely used. Conventional methods attempt to recover the corresponding GS image from the RS output by first estimating the camera motion or dense displacement field during the image acquisition and sequentially applying motion compensation to align all the pixels to a fixed timestamp. However, these methods make strong assumptions on the motion type and are prone to fail in the presence of complex motion patterns like local or non-linear motion, which no longer satisfy the assumptions.

In this work, we address the fundamental problem of modeling scene changes during image acquisition by using the high temporal information provided by an auxiliary *event camera*. Instead of measuring synchronous frames of absolute brightness, event cameras only measure asynchronous brightness changes at each pixel with a high temporal resolution in the order of microseconds [3]. Unlike image-based methods, event cameras allow to estimate the motion without relying on a constant speed assumption and synthesizing images by directly adding the brightness changes registered by the events to an image. This enables exciting computational photography applications such as event-guided image deblurring [12, 18, 24], video interpolation [8, 22, 23] and high dynamic range imaging [6, 16].

In this paper, we propose a RS compensation method, called *Eventful Shutter (EvShutter)*, that only requires a single RS image and the corresponding events. The method firstly removes blur using a novel flow-based deblurring module and then compensates RS using an hourglass network with two encoders relying on complementary geometric and synthesis-based interpolation approaches. The proposed method is the first method that does not rely on constant velocity assumption and uses a simple architecture thanks to the newly introduced event transformation, that we call *Filter and Flip (FnF)*. FnF transforms the input events to encode just changes between GS and RS images. To train and evaluate the proposed method, we collect the first dataset with real events and high quality RS images (optionally with blur), called *RS-ERGB*. The RS images are generated from GS images using our new simulator based on adaptive interpolation. This pipeline allows the use of an inexpensive high speed camera and the use of long exposures while capturing the GS images.

Contributions of this work are as follows

1. *Eventful Shutter (EvShutter)* : the first event-assisted RS distortion compensation and deblurring method that avoids constant speed motion assumptions.
2. *Filter and Flip (FnF)*: an event transformation module that transforms the input event stream to encode only the brightness change from GS to RS image and simplifies the architecture of downstream modules.
3. *Rolling Shutter Events and RGB (RS-ERGB)* dataset and simulator: the first dataset with real events and RS images with optional blur generated using a new realistic RS-simulator, based on adaptive interpolation.

2. Related Work

Previous works on RS correction attempt to recover the GS image by estimating the camera motion or dense displacement field during image acquisition and applying motion compensation to align all the pixels to a fixed timestamp. They can be classified according to their motion estimation method into following categories: *gyro-based*, *single image-based*, *multiple image-based* and *event-based*. Below we describe each category in detail.

Gyroscope-based methods [7, 13] utilize a gyro sensor to estimate the rotational motion of the camera. These methods are widely used in mobile devices, due to their speed and reliability. However, they assume a static scene and a simplified rotational camera motion. hence, they can not correct RS distortions due to moving objects (e.g. see Fig. 1), or in the presence of more complex camera motion where the purely rotational assumption is violated.

Single image-based methods infer the motion from a single RS image by relying on a prior model of the scene. Methods in [14,19] assume the presence of multiple straight lines in the scenes (Manhattan world assumption) for motion estimation. These hand-crafted priors fail when the scene deviates from the assumed model. Recent methods [21, 31] rely on deep learning to automatically learn scene priors that are used to estimate the camera motion. While these methods are more robust, they are still limited to certain scene types with static scene and constant depth (except [31]).

Multiple image-based methods estimate the motion between sequential RS images and use it to approximate the motion during image exposure relying on a parametric motion model. While early methods [2, 5] use image feature matching to estimate a rotational or planar homography, more recent methods [1, 15, 27] compute the dense optical flow between RS images and can correct RS distortion due to object motion. However, they suffer from limitations of image-based motion estimation: they do not work with a single image, image blur and illumination changes. Moreover, they lack the exact information about the motion during exposure and can fail in the presence of large displacements in between frames and highly non-linear motion.

Event-based method. Recently, [28] developed an event-assisted RS compensation method based on the work of [23]. To ensure that interpolation in a given row is performed from mid-exposure time of that same row to mid-exposure time of the latent GS image their method uses additional conditioning mechanisms and assumptions in the sub-modules, notably constant velocity assumption similar to images-based methods [1, 15, 27]. In contrast, we eliminate the need of additional strong assumptions and conditioning in downstream modules by introducing a simple yet effective event transformation that modifies the input events to encode the transformation from GS to RS image.

RS Deblur. Besides RS distortion, in presence of fast motion RS images often suffers from blur. In those cases, RS restoration methods needs to simultaneously perform deblur and distortion correction. [27] directly incorporate a deblurring stream in the RS correction network, while [28] adds an optional deblur module before the RS correction.

Datasets. *BS-RSCD* [27] dataset is recorded by using a beamsplitter setup with temporally and geometrically synchronized GS and RS cameras. Other datasets, such as *Fastec-RS* [15] or *Gev-RS* [28] are collected by first using a high speed GS camera and later synthesizing RS images by sequentially copying rows from the high frame rate GS images. For Gev-RS events are simulated from high speed GS images using V2E [10]. Unfortunately, training with synthetic events often does not generalize well to real event data. Moreover, RS images synthesized by the above method suffer from noise, low contrast and have small resolution, due to very short exposure time in the high speed cameras. We address two of the above problems by capturing a new dataset with real events using GS RGB and event cameras arranged in a beamsplitter setup and introducing new high fidelity simulator that generates high quality RS images with optional blur from GS images.

3. Method

3.1. Eventful Shutter (EvShutter) Network

At first, we perform event-assisted deblurring of the input RS image I^{rs}_{blur} image using a pre-processing *deblurring module* inspired by [24]. Next, we compensate RS distortion to obtain at a GS estimate at reference time t_{ref} using the pipeline shown in Fig. 3. As shown in the Fig. 3, instead of directly feeding all the raw events $E_{0\rightarrow t_H}$ to the network as in [28], we firstly pre-process them by using the proposed *Filter and Flip (FnF)* transformation, described in details in Sec. 3.2. Next, we encode the transformed events $E_{gs\rightarrow rs}$ into a *voxel grid* representation $V_{gs\rightarrow rs}$ as described in [32]. Then we feed the voxel grid and the RS image I^{rs} to an hourglass network composed of a *warping encoder* and a *synthesis encoder* using complementary interpolation approaches as in [22, 23]. Lastly, the *fusion decoder* progressively combines multi-scale images and warping and synthesis features to produce the final GS image I^{gs}. Below we explain the main modules of the proposed method.

Deblurring. Besides the RS distortion, RS images can suffer from motion blur due to camera or object motion during the exposure time. Therefore, we propose to deblur RS images using events before passing them to our RS distortion compensation method. The overall pipeline of the deblurring is shown in Fig. 4. First, we filter out all events triggered not during the exposure time of the corresponding image rows and then, we shift them according to the row-wise delay, similar to [28]. This allows the use of GS event-based deblurring methods, such as [24]. The original method estimates a deblur operator directly from events using offsets and masks of deformable convolutions [29]. However, we found that it is possible to estimate better deblur operators from more a explicit motion representation, such as optical flow. To estimate flow, we re-use the same network and weights as in the main RS distortion compensation pipeline.

Warping Encoder. The warping-based encoder takes the voxel grid $V_{gs\rightarrow rs}$ and estimates optical flow $\mathcal{F}_{gs\rightarrow rs}$ relating the latent GS image to RS image using a *flow network* from [23]. Note that because the transformed events only encode changes from GS to RS image we can directly estimate optical flow between them without relying on a constant speed assumption. In other words, we can directly estimate row-wise flow, where each row y represents the pixel displacement from time t_{ref} to time $y \cdot t_H/H$. In contrast in [1, 15, 26, 28], the authors predict optical flow between frames or during the exposure time of the frame using events and approximate the displacement between RS and GS by multiplying with a row dependent coefficient, which is only accurate for linear motions. The estimated flow is used to warp multi-scale features extracted from the RS image I^{rs} using the *image encoder* resulting in multi-scale *warping interpolation features* $\mathbf{F^w_{\rightarrow gs}} = (F^{w(0)}_{\rightarrow gs}, F^{w(1)}_{\rightarrow gs}, F^{w(2)}_{\rightarrow gs})$. Similarly, a multi-scale image pyramid computed from the RS image is warped to get $\mathbf{I^{rs}_{\rightarrow gs}} = (I^{rs(0)}_{\rightarrow gs}, I^{rs(1)}_{\rightarrow gs}, I^{rs(2)}_{\rightarrow gs})$. The warping is performed by using

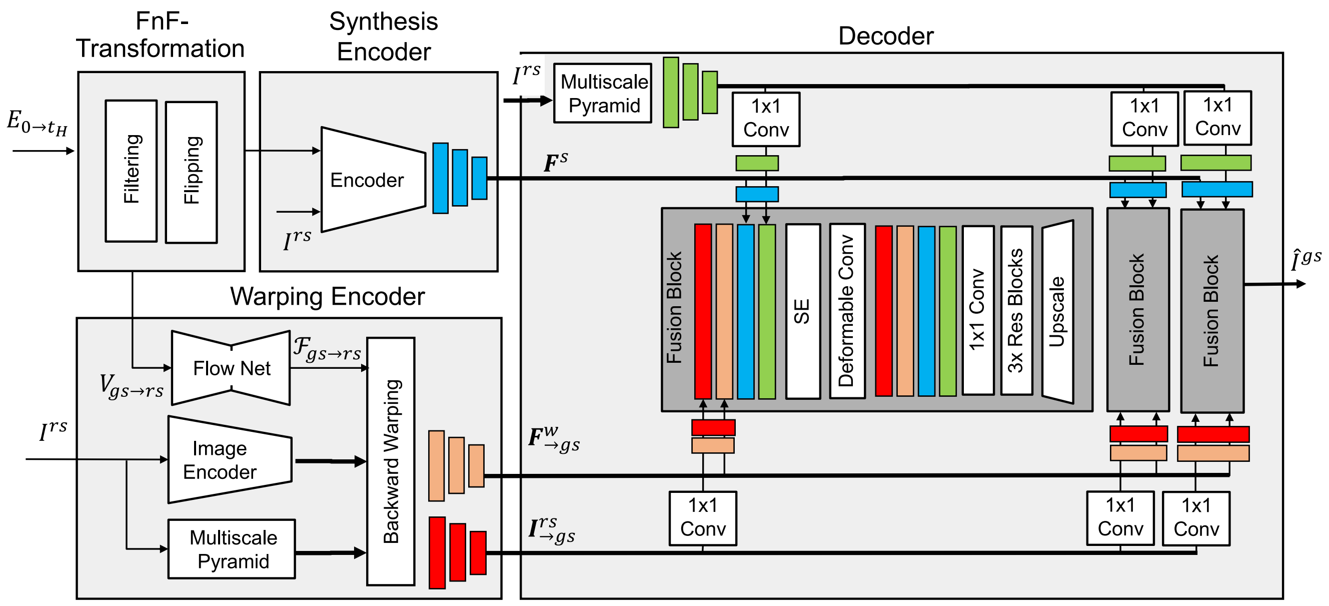

Figure 3. **RS distortion compensation pipeline.** At first the proposed method pre-processes input events with *FnF transformation* to encode only the brightness change between GS and RS frames. The transformed events are then passed to double encoder hourglass network with complementary *warping* and *synthesis* encoders relying on different interpolation principles. Then multi-scale features and images from both encoders are fused in the *decoder* to produce a GS image. The first fusion block of the decoder is shown in detail.

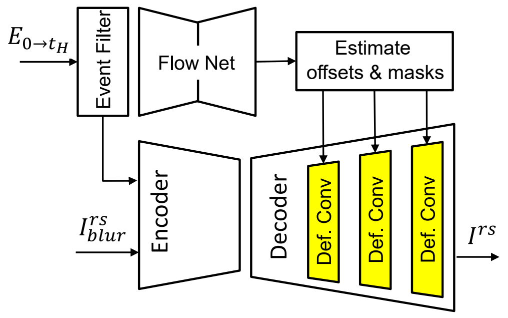

Figure 4. **RS deblurring pipeline**. First we filter out all the events which are not triggered during the exposure time, then we deblur the RS image using an improved version of [24], that relies on optical flow estimation to predict deblur kernels.

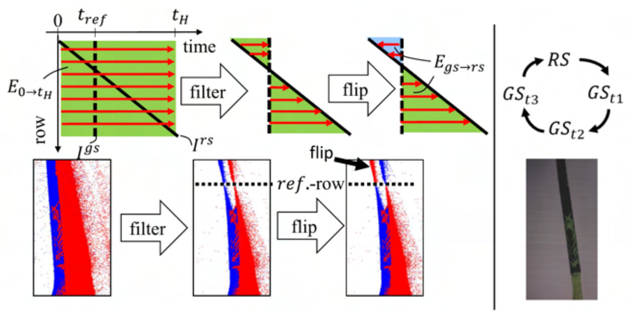

Figure 5. **Event transformation workflow.** Top row: Event stream is transformed by first applying and event filtering followed by a flipping step around an arbitrary reference time. Bottom row: An example of events. Right: Animation (best viewed with Acrobat Reader) between RS-input and GS estimations at different reference times.

differentiable *backward interpolation*. To downscale images and optical flow we perform average pooling and in case of flow we additionally divide the flow by the downscale factor.

Synthesis-based Encoder. The synthesis-based encoder takes both voxel grid $V_{gs\to rs}$ and RS image I^{rs} as inputs and computes multi-scale synthesis interpolation features $\mathbf{F}^s$. Intuitively, this encoder integrates brightness changes from the event stream and adds them to the RS image. Notice that, in contrast to the warping encoder, this module can handle illumination changes and fill motion occlusions.

Fusion Decoder. The fusion decoder progressively combines synthesis $\mathbf{F}^s$ and warping-based $\mathbf{F}^w_{\to gs}$ features, original RS image I^{rs} and warped RS image $I^{rs}_{\to gs}$ on multiple scales to reconstruct the GS image I^{gs}. We use shortcut connections from the original RS image to help the decoder to reconstruct static parts of the image. To select the most informative features on each scale the decoder relies on Squeeze-and-excitation (SE) blocks [9] attenuating channels, and modulated deformable convolutions [29], attenuating the feature maps spatially.

3.2. Filter and Flip (FnF) Transformation

The auxiliary event camera records an event stream $E_{0\to t_H}$ encoding all brightness changes from time 0 to t_H. However, to recover the GS image I^{gs} corresponding to a given RS image I^{rs}, we are only interested on those events $E_{gs\to rs}$ encoding brightness changes between latent GS and observed RS image. In this work, we propose a well-suited RS event transformation called *Filter and Flip (FnF)*, which transforms the event stream $E_{0\to t_H}$ to $E_{gs\to rs}$ through two steps: *filtering* and *flipping* steps as shown in Fig. 5 and described below.

Filtering step. For each row y, we only keep those events triggered between time t_y and the reference time t_{ref}, where t_y is mid-exposure time of row y in observed RS image.

Flipping step. We reverse all events before the reference time t_{ref} by inverting their polarities and recomputing their timestamps such that they are flipped around t_{ref} as shown in Fig. 5. This process ensures that the events encode changes in the direction from GS-frame to RS-frame and that the transformed events start at the time, which matches the GS-frame and end at the RS-time.

Applying the event transformation eliminates the need in time conditioning mechanisms and additional assumptions in the downstream modules. For example, in contrast to [28], the proposed method does not require constant speed assumption in the flow estimation module and attention mechanisms in the synthesis-based encoder, which allows re-using already pretrained modules on widely used GS images. Also, notice, that the FnF transformation can be adapted to arbitrary reference times.

In order to allow a good comparison to other works, we set $t_{ref} = t_H/2$, which corresponds to the mid-exposure time of the central row.

3.3. RS-ERGB Dataset

To bridge the sim-to-real gap from synthetic to real events we build a hybrid imaging system of a high speed GS RGB-camera and an event camera in a beam splitter setup. We use hardware temporal synchronization and geometrically align events and images as described in the Supp. Mat. With this setup we capture the first RS dataset with real events and high quality images. We generate high quality RS images with optional blur using our novel simulator based on adaptive interpolation described below.

High quality RS simulator. Our proposed pipeline allows to record GS sequences at lower speed than previous methods (160 fps compared to [15, 28] that record GS sequence at 2400 or 5700 fps). This allows us to use a longer exposure and as a result, acquire images of higher quality. Then we adaptively interpolate captured images using *Super Slomo* [11], so that the maximum displacement between interpolated frames is at most 1 pixel similar to

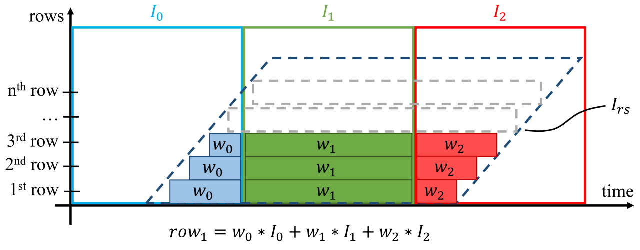

Figure 6. **Synthetic RS-images from a sequence of GS-Images I_i.** Each row is constructed separately by computing the weighted average of the GS-frames, which are assumed to be a close approximation of the latent GS-frames.

Vid2E [4]. This procedure ensures that a sufficient temporal resolution is achieved to synthesize realistic blur and RS effect. As shown in Fig. 7, using a fixed interpolation factor can give unrealistic results as the amount of motion can differ significantly by region and scene. The resulting interpolated video sequence can be thought of as discrete approximation of latent GS-images I_i of the true latent frames $I_{latent}(t, x, y)$. To synthesize a row y_{rs} of a RS-image the integral of $I_{latent}(t, x, y = y_{rs})$ has to be computed over the exposure time for this row, as

$$I_{rs}(t_{start}, t_{end}, y_i) = \frac{1}{t_{end} - t_{start}} \int_{t_{start}}^{t_{end}} I_{latent}(t, y_i)\, dt. \tag{1}$$

We use a discrete approximation instead and compute a weighted average of the approximated latent frames I_i as shown in Fig. 6. Using this procedure a synthetic exposure time and readout time T are chosen to synthesize corresponding RS frames. Notice that exposure time and readout time can be set separately allowing us to create blurry RS images with sharp RS ground truth for the task of RS deblurring. Video interpolation has already been used in the past to generate more realistic blur from videos [17], but to the best of our knowledge, we are the first to use an adaptive upsampling scheme to get consistently realistic results. Examples of our dataset with different amounts of blur and a comparison to a version without adaptive interpolation can be seen in the Supp. Mat..

Comparison to Gev-RS [28] is shown in Tab. 1. Compared to Gev-RS, our dataset has real events and the RS and GS image have better quality and resolution, despite using a lower end camera [2], since our new synthesis procedure allows using lower frame rate with longer exposure.

4. Experiments

In this section, first, we ablate the key contributions of the proposed EvShutter method. Second, we benchmark

[2]Phantom VEO 640 is used for collecting Gev-RS cost 60k$ while Blackfly S used for collecting our datasets costs just 500$.

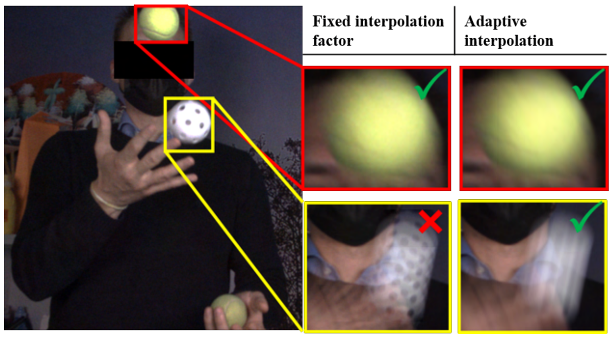

Figure 7. **Benefit of the proposed RS simulator** In contrast to a simulator with fixed interpolation rate, the proposed simulator with an adaptive interpolation does not suffer from aliasing artifacts regardless of the motion speed.

	RS-ERGB (ours)	**Gev-RS** [28]
Event data	**real** (Prophesee Gen4M)	synthetic
GS camera	**160 fps**	5.7k fps
	Blackfly S	Phantom VEO 640
№ sequence	**34**	29
Image quality	**high**	low
Resolution	**970×625**	640×360

Table 1. **Details of our RS-ERGB compared to Gev-RS** [28]. Best-suited properties are shown in bold.

the EvShutter against state-of-the-art image-based [15, 27] and event-based [28] methods on our RS-ERGB dataset, as well as on the public Fastec-RS dataset [15].

Details of the objective loss, training procedure and runtime comparisons together with ablation on additional design choices can be found in the Supp. Mat.

4.1. Ablation Study

FnF event transformation. To evaluate the proposed FnF transformation Tab. 2 we dissect our pipeline in purely warping-based, purely synthesis-based and the full pipeline.

We compare the purely warping-based method to a version that relies on a constant velocity assumption similar to EvUnroll [28] by applying a row-wise scaling to the optical flow computed from t_{GS} to t_0 and from t_{GS} to t_H. Our FnF transformation improves the warping-based method by 0.77 dB in PSNR and 0.0104 in LPIPS [25].

We also test the purely synthesis based module with and without the FnF transformation. As shown in Tab. 2 the synthesis module with the transformation outperforms the version without by 3.92 dB in PSNR and 0.0376 in LPIPS.

The overall architecture also benefits from the FnF transformation by 2.61 dB in PSNR and 0.0203 in LPIPS.

Importance of optical flow in RS deblur module. We show benefits of estimating the deblur operator from optical flow instead of raw events in Tab. 3. By introducing the optical flow explicitly to the deblurring network we observe

Method	**PSNR↑**	**LPIPS↓**	**SSIM↑**
	Warping		
Linearized flow	28.39	0.1721	0.870
FnF-Transform	**29.16**	**0.1617**	**0.883**
	Synthesis		
w/o FnF-Transform	25.87	0.2233	0.807
FnF-Transform	**29.79**	**0.1857**	**0.867**
	Warping & Synthesis		
Linearized flow	28.97	0.1703	0.870
FnF-Transform	**31.58**	**0.1500**	**0.897**

Table 2. **Importance of FnF event transformation**. The transformation improves performance of synthesis encoder and warping encoder. In the warping encoder it allows to avoid constant speed motion assumption. Ablation is made on the RS-ERGB validation set and best method is shown in bold.

an improvement of 1.4 dB in PSNR without effectively increasing the capacity, since we reuse the optical flow module from the RS correction pipeline. Furthermore, we visualized the offsets and masks of the deformable kernels in Fig. 9 and observed that in the case of large motions the kernels did not follow the motion in the original implementation of [24], but with our flow based adaption the deformable kernel is well aligned with the motion. For the sake of completeness we also add the results for deblurring of the EvUnroll [28] deblur module finetuned on our data.

Method	**PSNR↑**	**LPIPS↓**	**SSIM↑**
EDDMA [24]	34.92	0.0588	0.936
EDDMA [24]+Flow	**36.32**	**0.0488**	**0.951**
EvUnroll + Deblur [28]	35.64	0.0499	0.934

Table 3. **Importance of optical flow in RS deblur module**. For this experiment we generate RS images with 50ms exposure and compare RS-Deblurring results without RS correction. Best method is shown in bold. We also show performance of EvUnroll deblur module for reference.

4.2. Comparison with SOTA methods

RS-ERGB dataset. We compare our method to the state-of-the-art image-based methods DSUN [15], RSCD [27] and the recent event-based EvUnroll [28] on RS-ERGB-Sharp dataset without blur and RSCD [27] and EvUnroll [28] on the RS-ERGB-Blurry dataset with blur. All methods were finetuned on our datasets. For more details please refer to the Supp. Mat. The results are reported in Tab. 4 and qualitative comparisons are shown in Fig. 8. On the RS-ERGB dataset, we outperform the best image-based method DSUN [15] by 9.16 dB in terms of PSNR and 0.0966 in LPIPS and the event-based method EvUnroll [28] by 0.75 dB in PSNR and 0.0966 in LPIPS. On the RS-ERGB-Blurry dataset, we outperform EvUnroll [28] by 0.48 dB in PSNR and 0.053 in LPIPS. The image based-

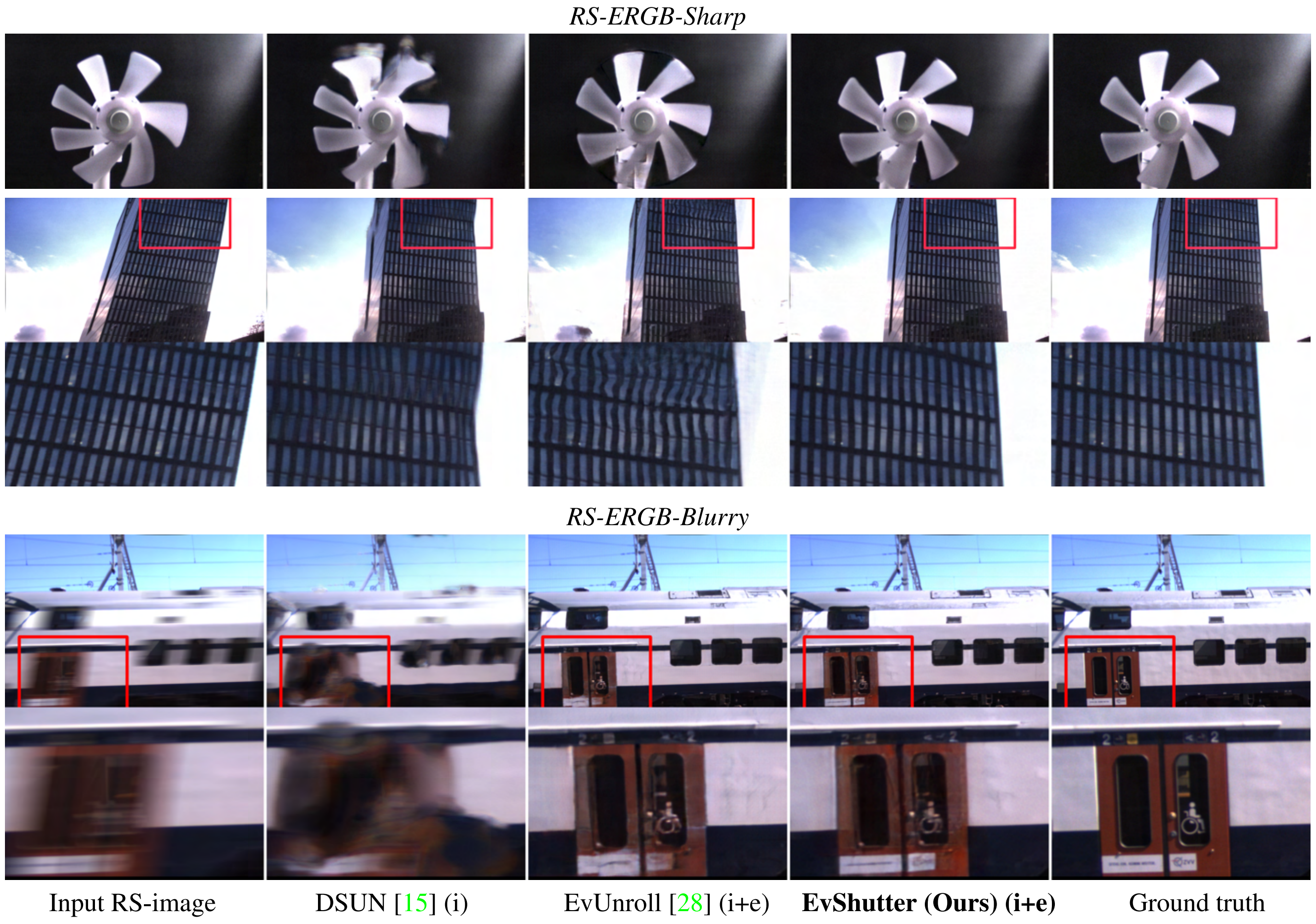

Figure 8. **Quantitative results on RS-ERGB-Sharp and ER-ERGB-Blurry datasets**. Image-based methods are marked by (i) and event and image based methods are marked by (i+e). The best method is shown in bold.

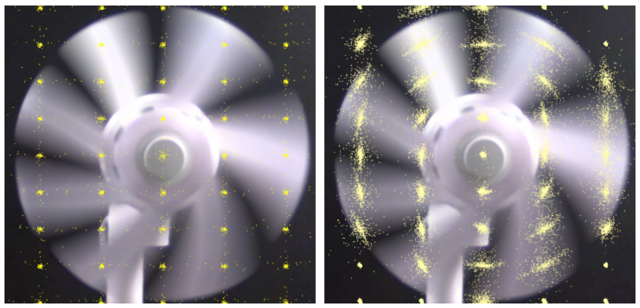

Figure 9. **Visualization of deformable convolutions offsets in deblurring** module for the original method [24](left), and our method (right), where offset are computed from optical flow.

method RSCD [27] seems not able to handle the difficult combination of large RS distortion and blur. Our method improves current image-based SOTA methods especially in areas with large displacements between RS and GS-frame. DSUN [15] is able to reconstruct the overall shape of the object, but produces strong artifacts. EvUnroll [28] is able to recover the structure of the GS images very well, but seems to heavily rely on their synthesis branch in areas with large non linear motion. We hypothesize that the fusion in RGB space makes it more difficult to combine the fine detail and texture from the warping branch with the coarse structurally correct results from the synthesis branch, which often lack precise colors. Additional comparisons can be found in the Supp. Mat..

Fastec-RS dataset To compare the proposed method to other SOTA on the public Fastec-RS dataset [15] we synthesized events from the original high speed sequences shared by the authors using ESIM [20] with the same parameters as in [28] and re-train our method from scratch. We run JCD [27] and DSUN [15] using the publicly available checkpoints. For [26, 30], we extract the metrics from the JCD paper [27] and for EvUnroll [28] and SUNet [1] from their respective papers. The results are reported in Tab. 5. The proposed method outperforms the best image-based approach by 4.07dB in PSNR and the event-based approach by 1.09dB in PSNR, 0.023 in terms of LPIPS and 0.03 in

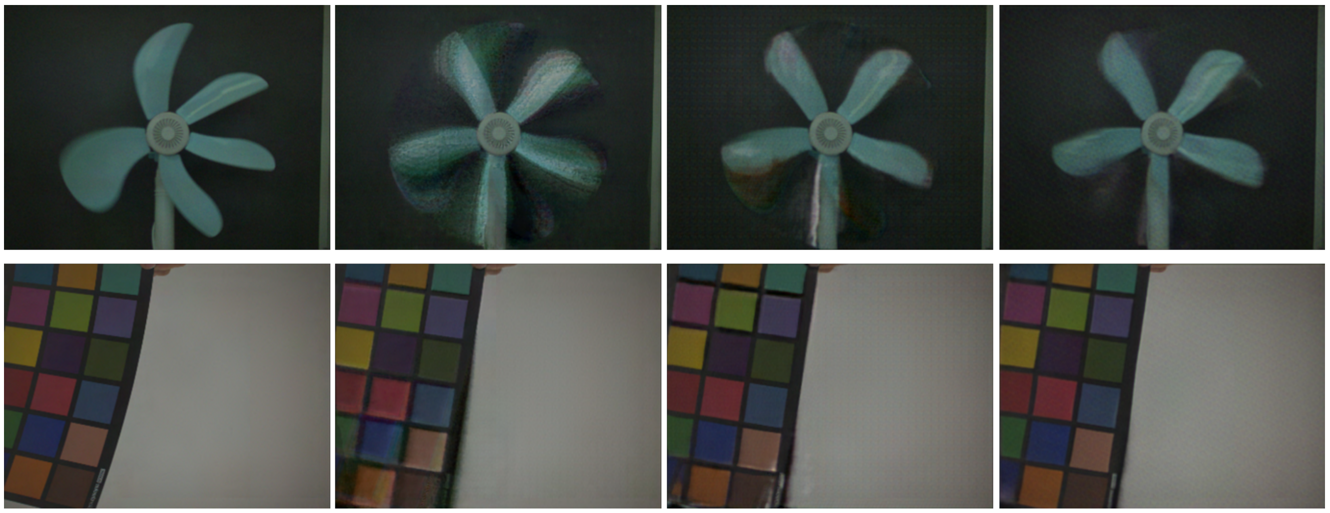

Figure 10. **Qualitative comparison on Real Events and RS images from EvUnroll [28].** Note that no ground truth data is provided for this dataset.

Method	PSNR [dB]↑	LPIPS↓	SSIM↑
RS-ERGB-Sharp			
DSUN [15] (i)	22.97	0.2670	0.7976
RSCD [27] (i)	22.62	0.2759	0.7884
EvUnroll [28] (i+e)	31.38	0.2334	0.8614
EvShutter (Ours) (i+e)	**32.13**	**0.1704**	**0.8866**
RS-ERGB-Blurry			
RSCD [27] (i)	17.87	0.3713	0.7109
EvUnroll [28] (i+e)	31.07	0.2576	0.8592
EvShutter (Ours) (i+e)	**31.55**	**0.2046**	**0.8780**

Table 4. **RS correction on RS-ERGB dataset with real events.** The best method is shown in bold. Image-base methods are marked by (i) and image and event-based methods by (i+e).

SSIM. Additionally, some qualitative examples are shown in the Supp. Mat.. Similarly, to the results on the RS-ERGB dataset we observe that our method performs better at object boundaries, where the displacement is not homogeneous. We obtain sharper and less ”wobbly” results.

Method	PSNR ↑	LPIPS↓	SSIM↑
DSUN [15] (i)	26.52	0.122	0.79
RSCD [27] (i)	24.84	0.107	0.78
SUNet [1] (i)	28.34	-	0.84
ESTRNN [26] (i)	27.41	0.189	0.84
EvUnroll [28] (i+e)	31.32	0.084	0.88
EvShutter (Ours) (i+e)	**32.41**	**0.061**	**0.91**

Table 5. **RS correction on Fastec-RS dataset with synthetic events.** The best method is shown in bold. Image-base methods are marked by (i) and image and event-based methods by (i+e).

Qualitative comparison on real RS dataset from [28] For qualitative comparison on real data, we evaluated our method on the dataset published in [28], which features real events and real RS images, but no ground truth. In Fig. 10, we compare to the results from EvUnroll [28] trained on their synthetic dataset with synthetic events and their model trained on our RS-ERGB dataset, see Fig. 10. We observe that by training on our RS-ERGB dataset with real events EvUnroll [28] greatly improves their performance. This shows the benefits of the proposed dataset with real events and the novel RS image simulator. Our method performs even better and does not suffer from strong ghosting artifacts as the EvUnroll [28]. Additonal examples can be found in Supp. Mat.

5. Conclusion

This paper proposes a method for RS distortion correction and deblurring. The method estimates deblurring operator from event-based optical flow and then compensates RS using a double encoder hourglass network. In contrast to previous methods, it does not rely on a constant velocity assumption and uses a simple architecture thanks to the proposed event transformation. Additionally, we propose the first RS dataset with optional blur containing real events and simulated high-quality RS images and show that training on this dataset increases performance of methods in real case scenarios. We carefully ablate each of the contributions and demonstrate the qualitative and quantitative performance boost compared to previous state-of-the-art methods.

References

[1] Bin Fan, Yuchao Dai, and Mingyi He. Sunet: symmetric undistortion network for rolling shutter correction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4541–4550, 2021. 3, 7, 8

[2] Per-Erik Forssén and Erik Ringaby. Rectifying rolling shutter video from hand-held devices. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 507–514. IEEE, 2010. 3

[3] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J Davison, Jörg Conradt, Kostas Daniilidis, et al. Event-based vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(1):154–180, 2020. 2

[4] Daniel Gehrig, Mathias Gehrig, Javier Hidalgo-Carrió, and Davide Scaramuzza. Video to events: Recycling video datasets for event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3586–3595, 2020. 5

[5] Matthias Grundmann, Vivek Kwatra, Daniel Castro, and Irfan Essa. Calibration-free rolling shutter removal. In *2012 IEEE international conference on computational photography (ICCP)*, pages 1–8. IEEE, 2012. 3

[6] Jin Han, Chu Zhou, Peiqi Duan, Yehui Tang, Chang Xu, Chao Xu, Tiejun Huang, and Boxin Shi. Neuromorphic camera guided high dynamic range imaging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1730–1739, 2020. 2

[7] Gustav Hanning, Nicklas Forslöw, Per-Erik Forssén, Erik Ringaby, David Törnqvist, and Jonas Callmer. Stabilizing cell phone video using inertial measurement sensors. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1–8. IEEE, 2011. 2

[8] Weihua He, Kaichao You, Zhendong Qiao, Xu Jia, Ziyang Zhang, Wenhui Wang, Huchuan Lu, Yaoyuan Wang, and Jianxing Liao. Timereplayer: Unlocking the potential of event cameras for video interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17804–17813, 2022. 2

[9] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 4

[10] Y Hu, S C Liu, and T Delbruck. v2e: From video frames to realistic DVS events. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2021. 3

[11] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9000–9008, 2018. 5

[12] Zhe Jiang, Yu Zhang, Dongqing Zou, Jimmy Ren, Jiancheng Lv, and Yebin Liu. Learning event-based motion deblurring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3320–3329, 2020. 2

[13] Alexandre Karpenko, David Jacobs, Jongmin Baek, and Marc Levoy. Digital video stabilization and rolling shutter correction using gyroscopes. *CSTR*, 1(2):13, 2011. 2

[14] Yizhen Lao and Omar Ait-Aider. A robust method for strong rolling shutter effects correction using lines with automatic feature selection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4795–4803, 2018. 2

[15] 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. 3, 5, 6, 7, 8

[16] Nico Messikommer, Stamatios Georgoulis, Daniel Gehrig, Stepan Tulyakov, Julius Erbach, Alfredo Bochicchio, Yuanyou Li, and Davide Scaramuzza. Multi-bracket high dynamic range imaging with event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 547–557, 2022. 2

[17] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In *CVPR Workshops*, June 2019. 5

[18] Liyuan Pan, Cedric Scheerlinck, Xin Yu, Richard Hartley, Miaomiao Liu, and Yuchao Dai. Bringing a blurry frame alive at high frame-rate with an event camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6820–6829, 2019. 2

[19] Pulak Purkait, Christopher Zach, and Ales Leonardis. Rolling shutter correction in manhattan world. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 882–890, 2017. 2

[20] Henri Rebecq, Daniel Gehrig, and Davide Scaramuzza. ESIM: an open event camera simulator. *Conf. on Robotics Learning (CoRL)*, Oct. 2018. 7

[21] Vijay Rengarajan, Yogesh Balaji, and AN Rajagopalan. Unrolling the shutter: Cnn to correct motion distortions. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, pages 2291–2299, 2017. 2

[22] Stepan Tulyakov, Alfredo Bochicchio, Daniel Gehrig, Stamatios Georgoulis, Yuanyou Li, and Davide Scaramuzza. Time lens++: Event-based frame interpolation with parametric non-linear flow and multi-scale fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17755–17764, June 2022. 2, 3

[23] 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. 2, 3

[24] 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. 2, 3, 4, 6, 7

[25] 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. 6

[26] Zhihang Zhong, Ye Gao, Yinqiang Zheng, and Bo Zheng. Efficient spatio-temporal recurrent neural network for video deblurring. In *European Conference on Computer Vision*, pages 191–207. Springer, 2020. 3, 7, 8

[27] 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. 3, 6, 7, 8

[28] 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, 5, 6, 7, 8

[29] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9308–9316, 2019. 3, 4

[30] Bingbing Zhuang, Loong-Fah Cheong, and Gim Hee Lee. Rolling-shutter-aware differential sfm and image rectification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 948–956, 2017. 7

[31] Bingbing Zhuang, Quoc-Huy Tran, Pan Ji, Loong-Fah Cheong, and Manmohan Chandraker. Learning structure-and-motion-aware rolling shutter correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4551–4560, 2019. 2

[32] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised event-based optical flow using motion compensation. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018. 3