

# Fast Event-based Double Integral for Real-time Robotics

Shijie Lin<sup>1,5</sup>, Yingqiang Zhang<sup>1</sup>, Dongyue Huang<sup>2,5</sup>, Bin Zhou<sup>3,5</sup>, Xiaowei Luo<sup>4</sup> and Jia Pan<sup>1,†</sup>

**Abstract**—Motion deblurring is a critical ill-posed problem that is important in many vision-based robotics applications. The recently proposed event-based double integral (EDI) provides a theoretical framework for solving the deblurring problem with the event camera and generating clear images at high frame-rate. However, the original EDI is mainly designed for offline computation and does not support real-time requirement in many robotics applications. In this paper, we propose the fast EDI, an efficient implementation of EDI that can achieve real-time online computation on single-core CPU devices, which is common for physical robotic platforms used in practice. In experiments, our method can handle event rates at as high as 13 million event per second in a wide variety of challenging lighting conditions. We demonstrate the benefit on multiple downstream real-time applications, including localization, visual tag detection, and feature matching.

## I. INTRODUCTION

Clear image acquisition is the prerequisite of various vision-based algorithms to operate in robotics systems. However, due to the inherent blurry effect of the active pixel sensor, motion artifacts can heavily degrade the image when relative motion exists between the camera and scenes. This effect is further intensified in low light conditions due to the demand for longer exposure time. Considering only frames, motion deblurring is an ill-posed problem [1] and poorly addressed by existing methods [2]–[4]. Luckily, a novel type of bio-inspired neuromorphic vision sensor, *i.e.*, the dynamic vision sensor (DVS) [5], [6], can encode light variation in high temporal resolution events, allowing the motion deblurring being effectively addressed [7]–[11].

Pan *et al.* [7] proposed the event-based double integral (EDI) model that bridges the formation of events and images, showing the effective capability in deblurring and high-rate video reconstruction. EDI was later adopted by various works [12]–[14] for offline processing. However, for real-time robotics, existing EDI implementations are too slow (1.5s per image [7]). To employ the promising deblurring capability brought by EDI for various robotics applications, we need to run it fast, making millions of events to be pro-

cessed online without jamming the whole system. However, this is still quite challenging for the following reasons.

- **High Computational Complexity.** Vanilla EDI [7], [15] is formulated using latent images. Without modification, the computation complexity is coupled with the image size and requires multiple redundant frame-wise operations, significantly limiting its efficiency.
- **Infeasible to Real-time.** When running EDI for real-time robotic systems, millions of events must be processed within frame intervals to ensure continuous captured data won't jam the whole system. However, the original implementation of EDI was targeted for offline processing and its code is difficult to be adapted for online and onboard robotic system.
- **Unknown Contrast Parameters.** To deal with the contrast parameter, related works using EDI for image deblurring usually involve time-consuming optimization [7], [16] or learning-based inference [8], [17], both of which are highly time-consuming and infeasible for real-time processing with CPU-only devices.

In this work, we propose fast EDI, an efficient reformulated version of EDI that achieves real-time deblurring with an event rate up to 13 Million Ev/s in a single core CPU. To achieve this performance, we propose to evenly distribute the workload for online processing and attain the goal by reformulating the EDI model and implementing it using a novel list-based container. We further introduce a method to estimate the contrast from the hardware parameters allowing robust and efficient parameter determination. In experiments, we compare our methods with existing approaches in terms of perceptual quality and runtime, showing state-of-the-art performance. Our fast EDI allows real-time deblurring and benefits various downstream applications, including visual tag detection in highly dynamic scenes, SLAM and feature matching in low light conditions. We fully release the real-time processing code and dataset with detailed hardware parameters.

**Contribution:** We propose the fast EDI, an efficient way of implementing the event-based double integral (EDI), unlocking the high-speed sensing capability of event cameras for clear image acquisition. We demonstrate that the EDI can be implemented in real-time using a single core CPU, enabling the use of various applications like localization, mapping, tracking, *etc.* We release an efficient C++ implementation and dataset with hardware settings. Webpage: [github.com/eleboss/fast EDI](https://github.com/eleboss/fast EDI)

<sup>1</sup>S. Lin, Y. Zhang, and J. Pan are with the Department of Computer Science, The University of Hong Kong, Hong Kong SAR, China. {lsj2048, zyg507, panj}@connect.hku.hk

<sup>2</sup>D. Huang is with The Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong SAR, China. dyhuang@mae.cuhk.edu.hk

<sup>3</sup>B. Zhou is with the School of Computer Science and Engineering, Beihang University, Beijing, China. zhoubin@buaa.edu.cn

<sup>4</sup>X. Luo is with the Department of Architecture and Civil Engineering, City University of Hong Kong, Hong Kong SAR, China. xiaowluo@cityu.edu.hk

<sup>5</sup> Peng Cheng Laboratory, Shenzhen, Guangdong, China.

<sup>†</sup> Corresponding author.

## II. RELATED WORK

**Motion deblurring and reconstruction.** The problems of motion deblurring and image reconstruction have been widely researched over decades. Early approaches are mostly based on the assumption of the static scene and leverages gradient [18]–[20] or non-gradient priors [21]–[23]. Recent works improved the performance through learning-based methods [2]–[4]. But their performance is generally not guaranteed under all conditions. Challenging motions or non-informative scenes could lead to degraded performance. With events, the motion deblurring becomes much easier to achieve. Early works fuse events and images using the asynchronous complementary filter [24], manifold regularisation [25], or direct integration [26]. But as the image formation model is not considered, their performance is generally worse than EDI [7], which takes into account the event-frame generation model and shows promising performance in image deblurring and high-rate video reconstruction. Based on EDI, later works improved their performance using learning frameworks. Songnan *et al.* [27] leveraged the discrete EDI as a physical insight when designing the learning framework, which was later improved by Xu *et al.* [11] by employing the optical flow of the event and frame for learning and training on the real data. Recently, Xiang and Yu [8] leveraged the EDI model to design a learnable double integral network, which can provide a generalizable model for deblurring and frame interpolation. However, previous works that rely on EDI require time-consuming optimization or learning-based inference, making real-time processing infeasible for robots with only CPU mounted. Even worse, according to our knowledge, there is no solution existing for efficient online processing of EDI. These issues significantly limit the usage of EDI in real-time robotics like localization, feature tracking, visual tag detection, *etc.*

**EDI related applications.** As an important model for event-frame relation, EDI has been adopted in various applications, including image denoising [16], high dynamic range imaging [28], event-based super-resolution [14], human-pose estimation [12], event-based optical flow estimation [29], and depth estimation [30]. However, existing works primarily work in an offline manner that does not require real-time processing. Whenever the number of events for processing within each second easily exceeds millions, EDI computation could consume tremendous time and bottleneck the whole system. Our fast EDI can overcome this limitation and allow the use of the EDI in a wide variety of applications.

## III. PRELIMINARIES

We first briefly review the formation model of events and images, and then formulate the event-based double integral (EDI) model.

*Event generation:* The event camera works asynchronously in responding to changes of log intensity and triggers timestamped events whenever the log-scale intensity change exceeds the contrast parameter  $c > 0$ , *i.e.*,  $\log(L(\mathbf{x}, t)) - \log(L(\mathbf{x}, \tau)) = p \cdot c$ , where  $L(\mathbf{x}, t)$  and  $L(\mathbf{x}, \tau)$  indicate the instantaneous latent image at time  $t$

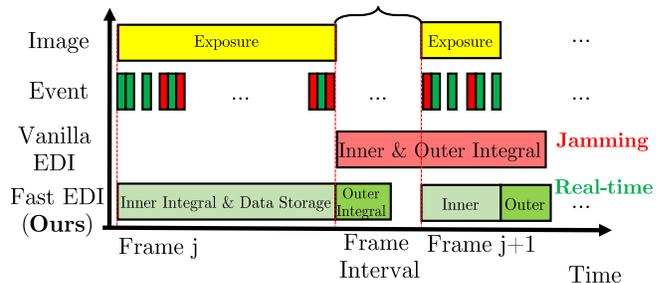


Fig. 1: Workload distribution. Our fast event-based double integral (EDI) can distribute the overall workload during acquisition time, achieving real-time processing. Vanilla EDI processes all events after the acquisition, which cannot complete before the next image is ready, leading to system jamming.

and  $\tau$  at pixel position  $\mathbf{x}$ , and the polarity  $p \in \{+1, -1\}$  denotes the direction of intensity changes. Thus the event is a 4-dimensional tuple  $\mathbf{e}_k \triangleq (\mathbf{x}_k, t_k, p_k)$ , where  $\mathbf{x}_k = (x_k, y_k)^T$  is the pixel position,  $t_k$  is the triggering time and  $p_k$  is the polarity of  $k$ -th events, respectively. Finally, the brightness motion can be encode in the events set  $\mathcal{E} = \{\mathbf{e}_k \mid k = 1, \dots, N_{ev}\}$ , where  $N_{ev}$  is the number of events. Leveraging events, we can give the relationship between latent images  $L(f)$  [7] (omitting the pixel positions  $\mathbf{x}$  for readability):

$$L(t) = L(f) \exp \left( c \int_f^t e(s) ds \right), \quad (1)$$

where  $e(t) \doteq p \cdot \delta(t - \tau)$  is the continuous sampling function of events with the Dirac function  $\delta(\cdot)$ .

*EDI model:* On the other hand, the blurry images are formulated using latent images  $L(t)$  within the exposure time  $T$ :

$$B = \frac{1}{T} \int_{t \in \mathcal{T}} L(t) dt, \quad (2)$$

where the blurry image is averaged results of latent images with the exposure duration  $\mathcal{T}$ . Then the latent images is reformulated as [7]:  $L(f) = \frac{B}{E(f, \mathcal{T})}$ , with

$$E(f, \mathcal{T}) = \frac{1}{T} \int_{t \in \mathcal{T}} \exp \left( c \int_f^t e(s) ds \right) dt \quad (3)$$

indicating the physical relation between latent images and blurry images from concurrent events, which is also refer to the event-based double integral (EDI) model.

*Current issues:* Although the continuous EDI model is given in previous literature [7], [15], efficient online implementation was never specified. To achieve this goal, we need to overcome the following difficulties:

- **Workload imbalance.** The EDI computation requires an accurate selection to separate events within the exposure interval. However, for online processing, the camera generates data sequentially (Fig. 1), meaning the selection cannot be made before the image's exposure is completed. Thus, vanilla EDI starts processing after the exposure is completed. However, the exposure time of

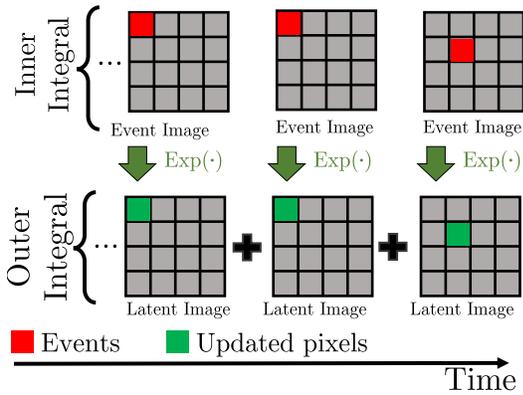


Fig. 2: EDI implemented with conventional array-like container requires frame-wise additions for the outer integral and thus is inefficient.

an image could exceed 30 ms, and millions of events could be triggered during this time. In a real-time system, all data need to be processed before subsequent data is ready. If the computation of vanilla EDI cannot be done within the frame interval, the subsequent data will cause jamming, slowing down the system and leading to an out-of-memory crash.

- **Redundant operations.** Unfortunately, the vanilla EDI usually cannot complete all computations within the frame interval due to redundant operations. Specifically, computing Eq. (3) requires repeatedly frame-like reconstruction of the latent image (inner integral) over the whole exposure interval (outer integral), making the real-time processing highly challenging.
- **Unknown contrast parameter.** Previous works [7], [15] use time-consuming optimization or inaccurate manual setting to handle the contrast parameters  $c$ . Both of them cannot satisfy the need for real-time robotics systems.

Therefore, we need to develop the whole computation pipeline from scratch to implement the EDI for efficient online processing.

#### IV. METHODS

To improve the EDI for real-time processing, we developed the fast EDI, whose implementation is based on a novel list-based container to achieve a balanced workload distribution. We further introduce the method of contrast estimation using hardware parameter [31], allowing a robust and efficient determination.

##### A. Fast event-based double integral (F-EDI)

1) *Workload distribution:* As the events are generated asynchronously from the sensor, one key to boosting the processing speed is to make each event processed right after it is triggered. However, for robotics systems, various applications are needed to consume the processing power of the CPU, and cores in the low-end processor are limited. Thus it is best if we can complete all computation using a single thread. To do that, we first compute the inner integral

and exponentiation for each event *i.e.*  $\exp\left(c \int_f^t e(s) ds\right)$ , and record the integral result for quick retrieval. Once the exposure is completed, the outer integral only needs to retrieve all containers and calculate the outer integral. This workload distribution allows us to fully leverage the CPU’s computing resource to process millions of events in real-time (Fig. 1), thereby alleviating the jamming problem.

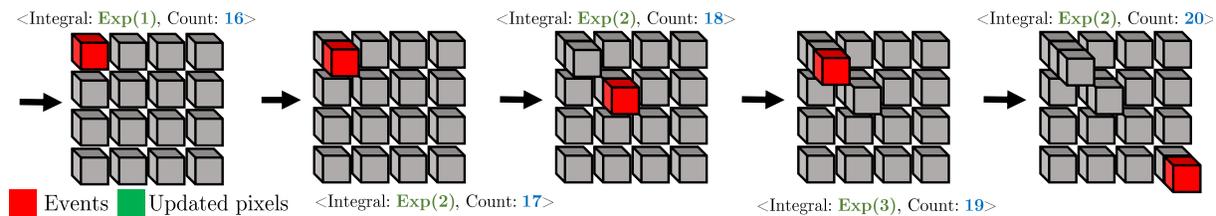
2) *Efficient list-based EDI:* However, even with the desired workload, the EDI computation is still inefficient for real-time processing. The vanilla EDI is formulated as a latent image, and the inner integral is responsible for rendering the latent image. Then the outer integral needs to sum up all latent images repetitively. However, since exponentiation exists between the inner and outer integral, frame-wise additions are required for summing up all latent images. As shown in Fig. 2, given the latest triggered events marked in red, the inner integral renders the event images, then exponentiation transforms the event image into the latent image, and the outer integral need to sum up all latent images. The first two operations, *i.e.* inner integral and exponentiation, are efficient as we can use the pixel-wise operation. But after the exponentiation, the latent image is nonlinear. Thus the outer integral needs to repetitively sums up all latent images in a frame-wise manner. Meaning for each event, equal to frame size  $N_x$  additions are required for the outer integral leading to  $\mathcal{O}(N_{ev} \times N_x)$  complexity.

We find this process could be simplified by redesigning the container. Since each time an event is generated, only one pixel alters its value in the latent image, most pixels for the additions in outer integral remain unchanged. Thus, we can temporally record each pixel’s inner integral results and the number of additions during the exposure time. Then, we can convert the repetitive additions to summing several multiplications in a single retrieval after the exposure time. Therefore, we propose the list-based container, as shown in Fig. 3. With this container, the fast EDI can be done in two steps. **Step 1:** During the exposure time, it conducts the inner integral using the latest activated events, taking its exponentiation, and increasing the global addition counter by one. Then two values (*i.e.*,  $\langle \text{exp value, counter} \rangle$ ) will be pushed back in a list marked by specific pixels positions (the red cube in Fig. 3). **Step 2:** After exposure, our fast EDI loops over all lists at specific pixels to retrieve the integral values and compute the outer integral in a each list by summing multiplications between the latent pixel value (green exp value in Fig. 3) and the difference of two consecutive counter numbers (blue value in Fig. 3).

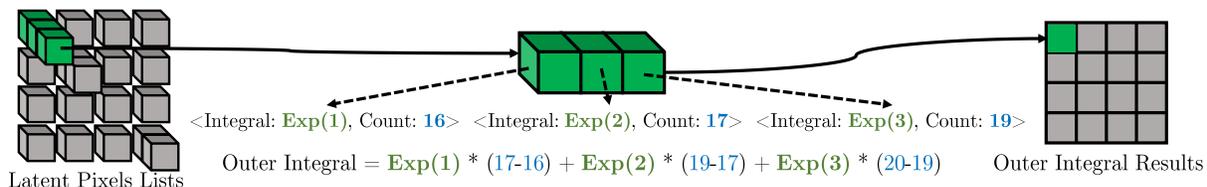
In this way, we can turn millions of additions into simply hundreds of multiplications and additions, thereby reducing the processing time. Finally, the time complexity is purely linear to the number of events, *i.e.*,  $\mathcal{O}(N_{ev})$ .

##### B. Event Contrast from Hardware Parameters

Vanilla EDI [7] uses an energy function based on image sharpness for optimization to find the optimal contrast. But such an optimization algorithm has two drawbacks. First, if the highly blurred image lacks enough high-contrast edges,



(a) Step 1: inner integral (during exposure)



(b) Step 2: outer integral (after exposure).

Fig. 3: EDI implemented with the proposed list-based container can significantly reduce the number of addition operations and decouple it from the image size, thus achieving efficient real-time processing. In step 1, the inner integral operates pixel-wise and pushes back the latest latent pixel value, *i.e.*,  $\text{Exp}(\cdot)$  and the counter value in the latent pixels lists. In step 2, the outer integral could be computed by summing the multiplication between latent pixel value (green exp value) and the difference of two consecutive counter numbers.

TABLE I: Average scores of deblurring (Best:  $\downarrow$ )

Metric	Method			
	[34]	[35]	[7]	Ours
PIQE [36]	62.92	54.71	42.55	<b>38.12</b>
SSEQ [37]	31.64	37.58	32.04	<b>24.01</b>
BRISQUE [38]	32.53	37.39	35.04	<b>28.75</b>

TABLE II: Average matched inliers of feature tracking (Better:  $\uparrow$ )

Feature	Trajectory 1		Trajectory 2		Trajectory 3	
	Blur	Ours	Blur	Ours	Blur	Ours
SURF [39]	83.5	<b>92.1</b>	90.2	<b>100.4</b>	95.5	<b>106.8</b>
FAST [40]	39.3	<b>41.6</b>	42.7	<b>46.1</b>	71.2	<b>73.6</b>
MSER [41]	39.2	<b>42.3</b>	48.8	<b>51.6</b>	82.0	<b>84.6</b>
BRISK [42]	68.3	<b>67.8</b>	73.2	<b>73.3</b>	112.2	<b>112.4</b>

the optimization is likely to converge to a local optimum. Second, the optimization is inefficient, making it difficult to deploy on real-time robotic systems. To address these issues, we introduce an estimation method based on hardware parameters. From hardware settings, we can directly calculate its contrast:

$$C_{\text{OFF}} = \frac{\kappa_n C_2}{\kappa_p^2 C_1} \ln\left(\frac{I_{\text{OFF}}}{I_d}\right), C_{\text{ON}} = \frac{\kappa_n C_2}{\kappa_p^2 C_1} \ln\left(\frac{I_{\text{ON}}}{I_d}\right), \quad (4)$$

where  $\kappa_n = \kappa_p = 0.7$  are the back gate coefficients of n and p FET transistors.  $C_1/C_2$  is the capacitor ratio of DVS (130/6 for our DAVIS346).  $I_d$ ,  $I_{\text{ON}}$ , and  $I_{\text{OFF}}$  are the bias current set by the user in a coarse-fine bias generator [32], which can be computed with jAER toolbox [33]. In this way, we can greatly simplify the optimization and increase the efficiency.

## V. EXPERIMENTS

In experiments, we first evaluate the primary performance of the fast EDI, *i.e.*, runtime and deblurring capability. Then we evaluate the fast EDI in three real-time downstream applications, *i.e.*, feature tracking, visual tag detection, and SLAM. We encourage the reader to view the supplementary video for more information.

### A. Experimental Setups

1) *Hardware*: We conducted all the experiments using Intel NUC mini PC with Intel i7-10710H@1.1 GHz CPU, and DAVIS346 [6] with contrast parameter set as  $C_{\text{ON}} = 0.26$  and  $C_{\text{OFF}} = -0.26$  to fit the single contrast model used

in [7]. The NUC mini PC is a common choice for various robotics systems and has multi-CPU cores for running other robotics algorithms like recognition, planning, and localization. We use C++ to implement our algorithms on the DV Platform with a ROS node to publish the real-time results.

2) *Dataset*: Since existing datasets do not include details of hardware settings for contrast estimation, during the real-time experiments, we recorded the online data as an evaluation dataset, which contains different motions (*e.g.*, shaking, random move) and scene illuminations (*e.g.*, low lighting conditions, sunlight, artificial light). Details are in the appendix.

### B. Runtime Performance

The vanilla EDI is implemented using Matlab with C++ warper and cannot support further modification. Its performance also can not support real-time processing (1.5 seconds per image, according to [7]). Thus, we compare our fast EDI with EDI implemented with the conventional array-like container using C++ and the same setup. During experiments, we record the average events rate of both methods. Our fast EDI could exceed 13 million Ev/s event rates using a single core CPU. In comparison, vanilla EDI can only achieve 49 thousand Ev/s. This result shows our fast EDI is 260 $\times$  faster than vanilla EDI, which demonstrates the efficiency of our method.

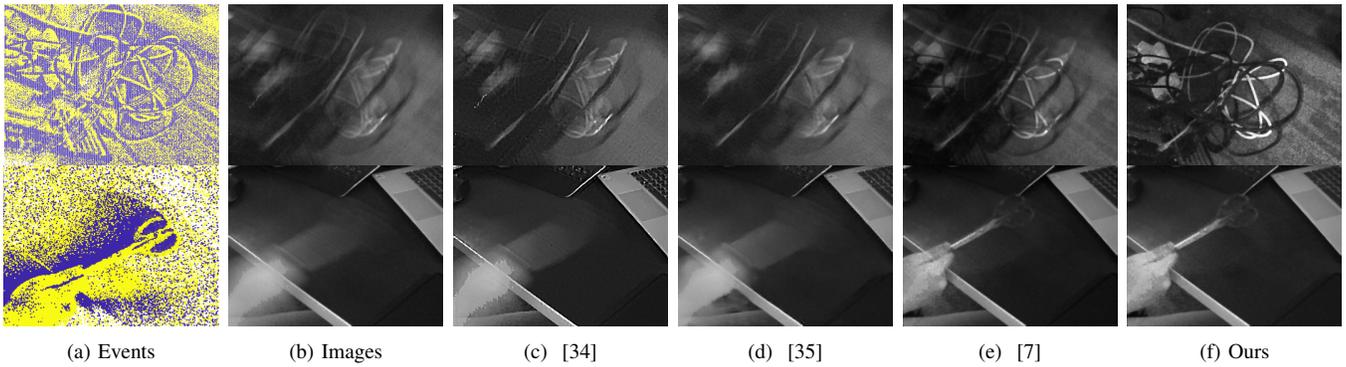


Fig. 4: Image deblurring results in two sequences, *line* (first row) and *scissor* (second row). (c) Conventional method [34] and (d) learning-based method [35] fail to recover (a) the heavily blur images. (e) [7] works well when the background is clean but is unable to tackle heavily blurred images. In contrast, (f) our method can stably generate clean images.

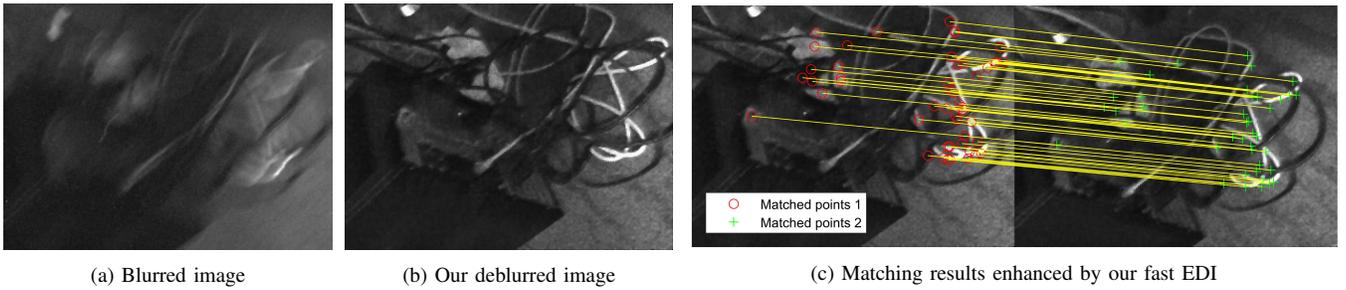


Fig. 5: Results of SURF [39] feature matching. Our method can effectively deblur (a) the blurry image and generate clear images for (b) accurate feature matching.

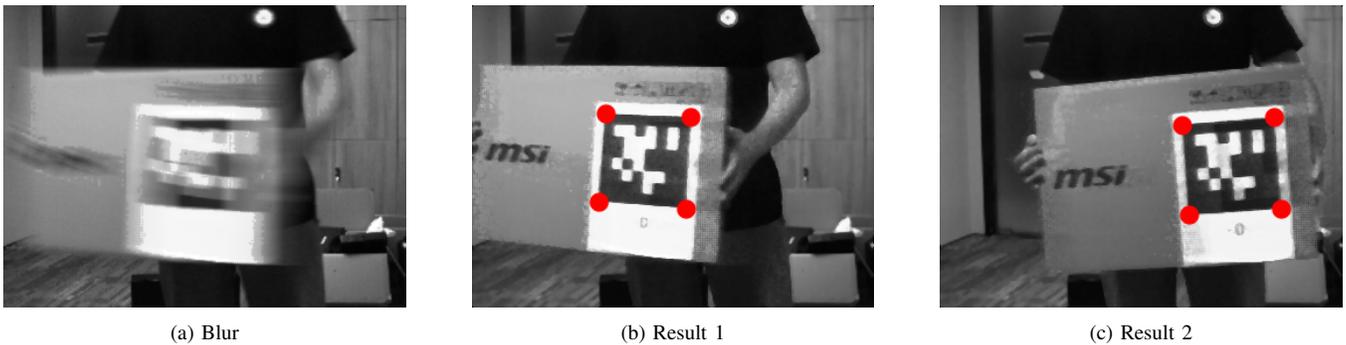


Fig. 6: Results of Apriltag [43] detection. Our method can deblur (a) the blurry image cause by rapid shaking motions and generate (b) (c) clear images, making consecutive successful Apriltag detection possible.

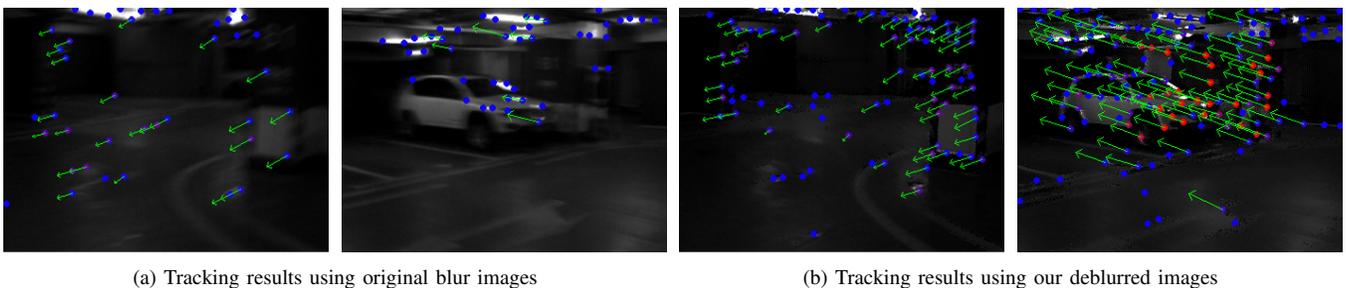


Fig. 7: Tracking result using VINS-mono SLAM [44]. It could be clearly observed that (a) the tracking using blurry images cannot extract effective features. Most of them are in low confidence (blue dots). In contrast, tracking using images deblurred by our methods has much more effective features with high confidence (red dots).

### C. Deblurring

We compare our approach with other state-of-the-art image deblurring methods, including conventional method [34],

learning-based method [35], and event-frame reconstruction method [7]. Qualitative comparisons are shown in Fig. 4. Multiple no-reference metrics, *i.e.*, SSEQ [37], PIQE [36], and BRISQUE [38] are adopted to provide quantitative comparisons, with the average scores on our dataset shown in Table I, where the lower is the score, the better is the performance. According to Table I, our method achieves the best score regarding all metrics. Other methods get suboptimal scores because the heavily blurred image can hardly provide any useful information for reconstruction. From Fig. 4, we can see that both our method and [7] handle the partially blurred images well, *e.g.*, the blur of a moving scissor in Fig. 4b. However, for completely blurred images, only our method can reconstruct clean images (Fig. 4f). [7] converges to sub-optimal contrast parameters, and the other two methods [34], [35] fail to converge. It is worth noting that, with a carefully manually adjusted contrast parameter, the deblurring performance of vanilla EDI is identical to our fast EDI. Thus our method also shares the defects of EDI, like easily affected by noise. Some of these defects have been addressed in recently proposed learning-based methods [8], but such methods rely on GPU for fast inference. As our work aims at introducing an efficient implementation of EDI, comparison with them is out of the scope of this work.

#### D. Real-time Applications

1) *Feature matching and tracking*: Our method can also benefit the feature matching and tracking by improving image quality. As shown in Fig. 5a, images taken in low lighting suffer from motion blur, which can barely provide any information for feature matching. In contrast, our fast EDI recovers a clean image in Fig. 5b, allowing successful SURF [39] feature matching in Fig. 5c. Quantitative comparison results are summarized in Table II, where our fast EDI improves the matching for all three types of features, which again verifies the effectiveness of our method.

2) *Visual tag detection*: Our fast EDI also benefits visual tag detection due to its effectiveness in real-time image deblurring. As shown in Fig. 6a, large motions will blur the visual tag, while our reconstructed clean image can properly preserve important information for successful Apriltag [43] detection in a sequence of 206 Apriltag images, as shown in Fig. 6b and Fig. 6c. In particular, the number of detected Apriltag increases from 73 to 198 after using our method.

3) *SLAM*: Simultaneous Localization and Mapping (SLAM) is an important robotics application that relies on real-time image processing. We use the VINS-Mono ([44], no loop closure) in a dark underground garage with a handheld DAVIS346 camera for evaluation. In experiments, we started and ended from the same position marked by visual tags. As the environment is quite dark ( $< 50$  Lux), the frame-based camera requires a longer exposure time for imaging, introducing motion blur and destabilizing the tracking front-end (Fig. 7a). Our fast EDI can provide real-time deblurred results to facilitate tracking (Fig. 7b), gaining a much higher localization accuracy compared with trajectory using original blur image, as shown in Fig. 8.

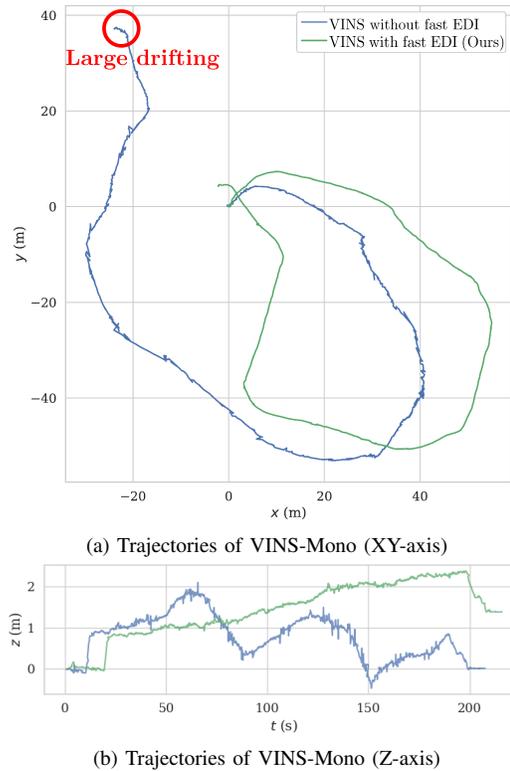


Fig. 8: Result of SLAM using VINS-mono [44]. VINS runs with images deblurred by our fast EDI (green trajectory) has much lower drifting compared with results using original blurred image (red trajectory). The starting and ending position are the same and marked by a calibration board. Best view in color.

Using blurry images, the translation error between starting and ending point is 44.16 meter. This error is reduced to 5.22 meter with the assistance of fast EDI.

#### E. Limitations

The memory consumption of this work increases linear to the number of triggered events. We therefore suggest that a minimum of 8 GB memory be allocated for the system. When using event cameras from manufacturers other than Inivation, it is noteworthy that the jAER toolbox may not yield accurate contrast parameter values. Hence, it is advised that users either contact the manufacturers directly for inquiry or employ calibration methods [45].

#### VI. CONCLUSION

In this paper, we propose the fast EDI, a novel way to efficiently implement the EDI to achieve real-time image deblurring using event cameras. In comparison with the conventional EDI model, we boost the computational speed 260 times on single core CPU and successfully bring the deblurring capability of EDI in improving the performance of various real-time applications, including SLAM, visual tag detection, and feature matching.

#### REFERENCES

[1] K. Purohit, A. Shah, and A. Rajagopalan, “Bringing alive blurred moments,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 6830–6839.

- [2] J. Sun, W. Cao, Z. Xu, and J. Ponce, "Learning a convolutional neural network for non-uniform motion blur removal," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 769–777.
- [3] X. Tao, H. Gao, X. Shen, J. Wang, and J. Jia, "Scale-recurrent network for deep image deblurring," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8174–8182.
- [4] L. Pan, Y. Dai, M. Liu, and F. Porikli, "Depth map completion by jointly exploiting blurry color images and sparse depth maps," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 1377–1386.
- [5] P. Lichtsteiner, C. Posch, and T. Delbruck, "A  $128 \times 128$  120 db  $15 \mu\text{s}$  latency asynchronous temporal contrast vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, 2008.
- [6] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A  $240 \times 180$  130 db  $3 \mu\text{s}$  latency global shutter spatiotemporal vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, 2014.
- [7] L. Pan, C. Scheerlinck, X. Yu, R. Hartley, M. Liu, and Y. Dai, "Bringing a blurry frame alive at high frame-rate with an event camera," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019, pp. 6813–6822.
- [8] X. Zhang and L. Yu, "Unifying motion deblurring and frame interpolation with events," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 17765–17774.
- [9] C. Scheerlinck, H. Rebecq, D. Gehrig, N. Barnes, R. Mahony, and D. Scaramuzza, "Fast image reconstruction with an event camera," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 156–163.
- [10] B. Wang, J. He, L. Yu, G.-S. Xia, and W. Yang, "Event enhanced high-quality image recovery," in *European Conference on Computer Vision*. Springer, 2020, pp. 155–171.
- [11] F. Xu, L. Yu, B. Wang, W. Yang, G.-S. Xia, X. Jia, Z. Qiao, and J. Liu, "Motion deblurring with real events," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 2583–2592.
- [12] L. Xu, W. Xu, V. Golyanik, M. Habermann, L. Fang, and C. Theobalt, "Eventcap: Monocular 3d capture of high-speed human motions using an event camera," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4968–4978.
- [13] D. Gu, J. Li, Y. Zhang, and Y. Tian, "How to learn a domain-adaptive event simulator?" in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 1275–1283.
- [14] L. Wang, T.-K. Kim, and K.-J. Yoon, "Eventsr: From asynchronous events to image reconstruction, restoration, and super-resolution via end-to-end adversarial learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 8315–8325.
- [15] L. Pan, R. Hartley, C. Scheerlinck, M. Liu, X. Yu, and Y. Dai, "High frame rate video reconstruction based on an event camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.
- [16] Z. W. Wang, P. Duan, O. Cossairt, A. Katsaggelos, T. Huang, and B. Shi, "Joint filtering of intensity images and neuromorphic events for high-resolution noise-robust imaging," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 1609–1619.
- [17] S. Lin, J. Zhang, J. Pan, Z. Jiang, D. Zou, Y. Wang, J. Chen, and J. Ren, "Learning event-driven video deblurring and interpolation," in *European Conference on Computer Vision*. Springer, 2020, pp. 695–710.
- [18] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman, "Removing camera shake from a single photograph," in *ACM SIGGRAPH 2006 Papers*, 2006, pp. 787–794.
- [19] D. Krishnan, T. Tay, and R. Fergus, "Blind deconvolution using a normalized sparsity measure," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 233–240.
- [20] L. Sun, S. Cho, J. Wang, and J. Hays, "Edge-based blur kernel estimation using patch priors," in *IEEE international conference on computational photography (ICCP)*. IEEE, 2013, pp. 1–8.
- [21] W.-S. Lai, J.-J. Ding, Y.-Y. Lin, and Y.-Y. Chuang, "Blur kernel estimation using normalized color-line prior," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 64–72.
- [22] J. Pan, D. Sun, H. Pfister, and M.-H. Yang, "Deblurring images via dark channel prior," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 10, pp. 2315–2328, 2017.
- [23] Y. Yan, W. Ren, Y. Guo, R. Wang, and X. Cao, "Image deblurring via extreme channels prior," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4003–4011.
- [24] C. Scheerlinck, N. Barnes, and R. Mahony, "Continuous-time intensity estimation using event cameras," in *Asian Conference on Computer Vision (ACCV)*. Springer, 2018, pp. 308–324.
- [25] G. Munda, C. Reinbacher, and T. Pock, "Real-time intensity-image reconstruction for event cameras using manifold regularisation," *International Journal of Computer Vision*, vol. 126, no. 12, pp. 1381–1393, 2018.
- [26] C. Brandli, L. Muller, and T. Delbruck, "Real-time, high-speed video decompression using a frame- and event-based davis sensor," in *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2014, pp. 686–689.
- [27] S. Lin, J. Zhang, J. Pan, Z. Jiang, D. Zou, Y. Wang, J. Chen, and J. Ren, "Learning event-driven video deblurring and interpolation," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2020, pp. 695–710.
- [28] Z. Wang, Y. Ng, C. Scheerlinck, and R. Mahony, "An asynchronous kalman filter for hybrid event cameras," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 448–457.
- [29] L. Pan, M. Liu, and R. Hartley, "Single image optical flow estimation with an event camera," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2020, pp. 1669–1678.
- [30] D. Gehrig, M. Rügge, M. Gehrig, J. Hidalgo-Carri6, and D. Scaramuzza, "Combining events and frames using recurrent asynchronous multimodal networks for monocular depth prediction," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2822–2829, 2021.
- [31] Y. Nozaki and T. Delbruck, "Temperature and parasitic photocurrent effects in dynamic vision sensors," *IEEE Transactions on Electron Devices*, vol. 64, no. 8, pp. 3239–3245, 2017.
- [32] T. Delbruck, R. Berner, P. Lichtsteiner, and C. Dualibe, "32-bit configurable bias current generator with sub-off-current capability," in *2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2010, pp. 1647–1650.
- [33] "jaer project," <http://jaerproject.org>.
- [34] J. Pan, D. Sun, H. Pfister, and M.-H. Yang, "Deblurring images via dark channel prior," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 10, pp. 2315–2328, 2018.
- [35] M. Jin, G. Meishvili, and P. Favaro, "Learning to extract a video sequence from a single motion-blurred image," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018, pp. 6334–6342.
- [36] N. Venkatanath, D. Praneeth, M. C. Bh, S. S. Channappayya, and S. S. Medasani, "Blind image quality evaluation using perception based features," in *2015 Twenty First National Conference on Communications (NCC)*, 2015, pp. 1–6.
- [37] L. Liu, B. Liu, H. Huang, and A. C. Bovik, "No-reference image quality assessment based on spatial and spectral entropies," *Signal Processing: Image Communication*, vol. 29, no. 8, pp. 856–863, 2014.
- [38] A. Mittal, A. K. Moorthy, and A. C. Bovik, "No-reference image quality assessment in the spatial domain," *IEEE Transactions on Image Processing*, vol. 21, no. 12, pp. 4695–4708, 2012.
- [39] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.
- [40] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking," in *Tenth IEEE International Conference on Computer Vision (ICCV) Volume 1*, vol. 2, 2005, pp. 1508–1515.
- [41] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and Vision Computing*, vol. 22, no. 10, pp. 761–767, 2004.
- [42] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *2011 International Conference on Computer Vision (ICCV)*, 2011, pp. 2548–2555.
- [43] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 3400–3407.

- [44] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.
- [45] Z. Wang, Y. Ng, P. van Goor, and R. Mahony, "Event camera calibration of per-pixel biased contrast threshold," *arXiv preprint arXiv:2012.09378*, 2020.