

# Event-Based Feature Tracking in Continuous Time with Sliding Window Optimization

Jason Chui<sup>[0000-0001-8188-9734]</sup>, Simon Klenk, and  
Daniel Cremers<sup>[0000-0002-3079-7984]</sup>

Technical University of Munich, Germany  
{jason.chui, simon.klenk, cremers}@tum.de

**Abstract.** We propose a novel method for continuous-time feature tracking in event cameras. To this end, we track features by aligning events along an estimated trajectory in space-time such that the projection on the image plane results in maximally sharp event patch images. The trajectory is parameterized by  $n^{\text{th}}$  order B-splines, which are continuous up to  $(n - 2)^{\text{th}}$  derivative. In contrast to previous work, we optimize the curve parameters in a sliding window fashion. On a public dataset we experimentally confirm that the proposed sliding-window B-spline optimization leads to longer and more accurate feature tracks than in previous work.

**Keywords:** Dynamic vision sensor · Continuous-time feature tracking · Sliding window · B-splines · SE2 warping

Please add the paper submission id and the track name to the paper.

## 1 Introduction

Event cameras (dynamic vision sensors) are imaging devices which asynchronously measure per-pixel brightness changes. These sensors are suited for robotics and virtual reality applications, since they offer lower latency, lower power consumption as well as higher dynamic range and higher temporal resolution compared to frame-based cameras. In order to actually tap into these benefits, computer vision algorithms for event-based sensors need to be developed. However, since event sensors are based on fundamentally different measurement principles than standard frame-based cameras, traditional computer vision algorithms cannot simply be applied to event data, but rather need to be developed from scratch.

Event cameras report per-pixel brightness changes of the observed logarithmic brightness. Each event  $\mathbf{e}_i = \{t_i, \mathbf{x}_i, p_i\}$  is a tuple of a micro-resolution timestamp  $t_i$ , image plane coordinates  $\mathbf{x}_i = (x_i, y_i)$  and the respective polarity change  $p_i \in \{-1, 1\}$ . The data stream is asynchronous and sparse because an event is transmitted only if the logarithmic brightness changes by a predefined, usually unknown threshold. This is in contrast to frame-based cameras, where each pixel

is illuminated during a shared exposure time interval, resulting in an absolute brightness measurement at a fixed frequency.

A common approach is to accumulate events over a fixed time interval into a frame-like structure and apply traditional computer vision methods on them. The drawback of a naive accumulation is that moving edges in the scene result in blurred edges in the image plane. One popular way to correct for this error is by estimating constant optical flow in a certain space-time window, i.e. at a predefined patch location and during a time interval. Each event in the window is then warped to a common reference time using the estimated optical flow. The goal is to create maximally sharp event patch image. This approach is known as motion-compensation [4]. It is applied to a variety of event-based vision tasks, such as feature tracking [5], [10], ego-motion estimation [16], [14] or motion segmentation [12].

In case of feature tracking, there remains one main drawback with motion-compensation: Usually, the estimated optical flow is assumed to be constant in a certain (short) time interval, hence the trajectory of events in the x-y-t space-time volume is piece-wise linear. To resolve this problem, we propose to track features with a continuous curve and optimize the curve in the manner of sliding window optimization. In this paper, we propose to employ B-splines as a curve representation in sliding window optimization for two reasons: (1) To account for feature tracks of variable length, we can build n-th order continuous trajectories by adding any number of knots to the curve. (2) Adding knots or changing knot values only changes the curve *locally*, so we can reduce the problem size by setting old knots which are out of scope to a constant value. Our approach is compared to a state-of-the-art event feature tracking algorithm and shows significant improvements in terms of error and feature age. The contributions of this paper can be summarized as follows:

1. We introduce the first event feature tracking algorithm that uses continuous B-spline functions and employs SE2 warping of events.
2. We optimize the B-spline parameters of the trajectory in a sliding-window manner.
3. We experimentally confirm significant improvements in tracking precision and feature age over existing event tracking algorithms.

## 2 Related Work

One line of work performs hybrid tracking by combining frames with events. The advantage of hybrid approaches is that during certain degenerate motions, such as movements parallel to an intensity edge (when no events are triggered), the frames can still provide useful intensity information for tracking. The work by Gehrig et al. [5] detects features in standard frames and tracks those using events. Their approach employs an event generation model, which is based on frames and estimated optical flow to predict the observed events. To model larger

variations in appearance, the feature patches are additionally parameterized by a rigid warp function in the image plane. This method achieves accurate results on a variety of datasets. However, it relies on specialized hardware, such as the Dynamic and Active-pixel Vision Sensor [2], which captures frames and events within the same pixel array, or alternatively requires beam splitting techniques.

There have been adoptions of frame-based corner detectors to event streams, such as the event-FAST corner detector by Muggler et al. [9] or the event-Harris detector by Vasco et al. [13]. However, those trackers are not robust to changes in motion direction [7]. In our work, we circumvent this issue by keeping most information from the past in the form of a template, see section 3.

The work by Ignacio et al. [1] proposes an event-by-event tracking approach which models different hypotheses per tracked feature. While this work tracks features at a very high rate of up to 12500 events per second, it is still formulated in discrete time and thus does not allow for simple derivative calculations of the trajectory, which can be useful in some applications. The approach by Manderscheid et al. [7] also performs tracking on an event-by-event basis. They train a random forest which extracts only the corner features from the event stream. The main drawback is that they rely on absolute intensity information during training time. The work by Zhu et al. [15] first builds feature templates by accumulating the event stream over a short time interval. A batch of new event is then aligned to those templates by probabilistic, soft association. The optical flow is computed as an expectation over all associations and the patch can undergo affine deformations during tracking. The work by Seok et al. [10] is the first approach to formulate event tracking in continuous time. However, adding more knots to the existing Bézier curve changes all previous knots, so the feature trajectory has to be formed by concatenating many short Bézier curves and is only zero order continuous.

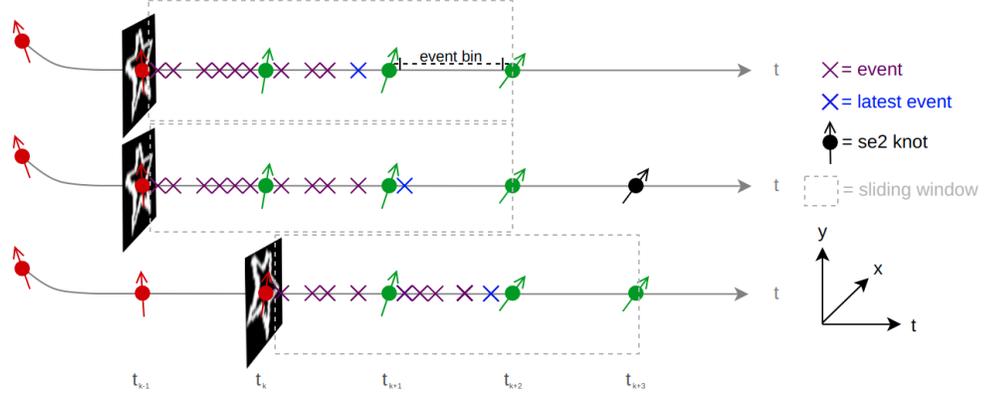
To the best of our knowledge, we are the first to formulate event tracking with the concept of sliding window optimization.

### 3 Method

We define a B-spline curve  $B(t; \Theta, \Delta t)$  which returns a transformation  $T_{r,t}$  to transform a 2D-point from current frame at timestamp  $t$  to reference frame  $r$ . All knots in the spline are denoted by  $\Theta$  and the time difference between two knots is denoted by  $\Delta t$ , which is a pre-determined constant and we will ignore it in our notation later in the paper. If an event  $e_n$  is within the region of a patch, it satisfy the condition  $B(t_n; \Theta)x_{start} - \mathbf{x}_n < R$ , where  $x_{start}$  is the starting position of the patch and  $R$  is the radius of the patch. The position of a warped event  $e_n$  is defined as

$$\mathbf{x}'_n = B(t_n; \Theta, \Delta t)\mathbf{x}_n \quad (1)$$

We create a patch image by warped back all events in the patch to reference time (here we set the reference time as the starting time of the feature). Since the warped positions of events are not guaranteed to be integers, a bi-linear kernel



**Fig. 1.** Process of updating the sliding window with optimizing 3 knots using SE(2) warping. The arrows on the knots indicate the rotation angles, green knots are the knots inside the sliding window, red knots are fixed knots. The image patches containing the star shape show the History patches. Top: The latest event is in the second last event. Middle: The latest event is in the last event bin, we add an additional knot for the trajectory. Down: After we add a new knot, we update the History patch and the location of the sliding window to fix the problem size.

$k_b$  is used to construct differentiable patches. If event  $e_n$  is received at pixel  $\mathbf{x}_n$  at time  $t_n$ , the patch image from events received in the time domain  $[t : t']$  is defined as

$$I_{[t:t']}(\mathbf{x}) = \sum_{\{n;t_n \in [t:t']\}} k_b(\mathbf{x}, \mathbf{x}'_n) \quad (2)$$

$$k_b(\mathbf{a}, \mathbf{b}) = \max(0, 1 - |a_1 - b_1|) \cdot \max(0, 1 - |a_2 - b_2|) \quad (3)$$

The advantage of using bi-linear kernel instead of using Dirac-delta function is that the patch image is also well-defined in sub-pixel level, which enables us to calculate well-defined  $\frac{\partial I}{\partial \mathbf{x}'}$ .

In principle, we can optimize a continuous B-spline trajectory by optimizing all knots on the spline and taking all events into account, but this is expensive and inefficient. Inspired by [10], we make use of a History patch  $H$  which compresses the information of previous knots and events in order to speed up the algorithm. The History patch  $H$  is built in recurrent relation.  $H$  and the modified patch image  $I^*(t)$  at timestamp  $t$  are defined as

$$H_{t_k}(\mathbf{x}; \Theta) = I_{[t_{k-1}:t_k]}(\mathbf{x}; \Theta) + \rho H_{t_{k-1}}(\mathbf{x}; \Theta) \quad (4)$$

$$I^*(\mathbf{x}, t; \Theta) = I_{[t_k:t]}(\mathbf{x}; \Theta) + H_{t_k}(\mathbf{x}; \Theta) \quad (5)$$

where  $t_k$  is the timestamp of knot  $k$ ,  $H_{t_0}(\mathbf{x}) = \mathbf{0}$  and  $0 \leq \rho \leq 1$  is the decaying parameter.

The best B-spline curve is the one which maximize the variance (sharpness) of a patch image from active event with using History patch, see Fig. 1 for an equivalent visualization of the problem using SE(2) warping. The modified variance  $\sigma^*$  of the patch at timestamp  $t$  is defined as

$$\sigma^*(P(t); \Theta) = \frac{1}{N} \sum_{\mathbf{x}} (I^*(\mathbf{x}, t; \Theta) - \langle I^*(\mathbf{x}, t; \Theta) \rangle)^2 \quad (6)$$

where  $N$  is the total number of pixels in the patch,  $\mathbf{x}$  is the image coordinate,  $\langle I^*(\mathbf{x}, t; \Theta) \rangle$  is the mean value of the modified patch image  $I^*$ . The work in [3] shows that among 22 possibilities of measuring sharpness in event images, the variance is often a suitable choice.

To maximize  $I^*$ , we need the Jacobian of the variance function w.r.t warping parameters  $\Theta$ :

$$\frac{\partial I^*(\mathbf{x}, t; \Theta)}{\partial \Theta} = \sum_{n=1}^{N_c^i} \frac{\partial k_b(\mathbf{x}, \mathbf{x}'_n)}{\partial \mathbf{x}'_n} \frac{\partial \mathbf{x}'_n}{\partial B} \frac{\partial B(t_n; \Theta)}{\partial \Theta} \quad (7)$$

To use SE(2) B-spline as an example, the parameter  $\Theta$  with  $N_k$  knots is defined as

$$\Theta = [k_1, \dots, k_{N_k}] \quad (8)$$

with  $k_i = [x_1^i, x_2^i, \theta^i]$

$$\frac{\partial \mathbf{x}'}{\partial B} = - [R_{r,t} \mid \sigma_x \mathbf{x}']_{2 \times 3} \quad (9)$$

with  $\sigma_x = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$ ,  $R_{r,t}$  is the rotation matrix of the feature relative to its original orientation

$$\frac{\partial B(t_n; \Theta)}{\partial k_i} = \lambda(t, \Delta t_{knot}) \mathbb{I}_{3 \times 3} \quad (10)$$

We refer the reader to [11] for the deviation of  $\lambda(t, \Delta t_{knot})$ . The optimal solution  $\Theta^*$  for Equation 6 is calculated through maximizing  $\sigma^*$  by line search.

## 4 Experiments

We evaluate all methods on each dataset with the same pre-selected, evenly-distributed Harris corners. We use a circular patch with diameter  $d = 31$ , a decaying parameter of  $\rho = 0.9$  and track up to 60 features in each experiment. We use third order B-spline and create a new spline knot every 50 milliseconds. If the number of events used in the optimization is small, it may lead to a wrong optimal solution. We tackle this problem by optimizing more knots with more events in this case. In the experiments denoted by *ours\**, we optimize three knots

**Table 1.** Quantitative Comparison of [15] against our method with error threshold 3.

dataset	method	mean relative age	mean age[sec]	mean error[pix]	mean common error[pix]	mean common error for ours/ours*[pix]
shapes_translation	ours	<b>0.27</b>	<b>0.71</b>	0.80	<b>0.91</b>	<b>0.95</b>
	ours*	0.26	0.63	<b>0.79</b>	<b>0.91</b>	<b>0.95</b>
	Zhu et al.	0.04	0.08	2.89	2.92	-
shapes_rotation	ours	0.30	0.61	<b>0.81</b>	<b>0.93</b>	<b>0.90</b>
	ours*	<b>0.32</b>	<b>0.67</b>	0.83	<b>0.93</b>	<b>0.90</b>
	Zhu et al.	0.02	0.02	2.81	2.81	-
shapes_6dof	ours	<b>0.32</b>	<b>1.48</b>	<b>1.05</b>	<b>0.61</b>	<b>1.14</b>
	ours*	0.31	1.45	1.07	<b>0.61</b>	1.16
	Zhu et al.	0.05	0.18	2.05	1.97	-
poster_translation	ours	0.47	1.43	<b>0.87</b>	<b>0.71</b>	<b>0.88</b>
	ours*	<b>0.48</b>	<b>1.47</b>	<b>0.87</b>	<b>0.71</b>	<b>0.88</b>
	Zhu et al.	0.33	0.71	1.15	1.10	-
poster_rotation	ours	0.41	1.33	<b>0.79</b>	<b>0.66</b>	0.80
	ours*	<b>0.45</b>	<b>1.54</b>	0.80	<b>0.66</b>	<b>0.79</b>
	Zhu et al.	0.20	0.51	1.51	1.40	-
poster_6dof	ours	<b>0.35</b>	2.57	<b>1.05</b>	<b>0.87</b>	<b>1.04</b>
	ours*	<b>0.35</b>	<b>2.61</b>	<b>1.05</b>	<b>0.87</b>	<b>1.04</b>
	Zhu et al.	0.25	1.37	1.32	1.28	-
boxes_translation	ours	0.35	1.35	<b>0.98</b>	<b>0.88</b>	<b>0.98</b>
	ours*	<b>0.37</b>	<b>1.50</b>	1.01	<b>0.88</b>	1.00
	Zhu et al.	0.31	0.95	1.19	0.94	-
boxes_rotation	ours	0.33	1.22	<b>0.79</b>	<b>0.68</b>	0.81
	ours*	<b>0.36</b>	<b>1.41</b>	0.80	<b>0.68</b>	<b>0.80</b>
	Zhu et al.	0.19	0.59	1.57	1.53	-
boxes_6dof	ours	0.37	1.76	<b>1.07</b>	0.72	1.08
	ours*	<b>0.38</b>	<b>1.85</b>	<b>1.07</b>	<b>0.71</b>	<b>1.07</b>
	Zhu et al.	0.15	0.64	1.88	1.78	-

when there are less than  $\frac{d^2}{4}$  events in the sliding window, otherwise we optimize only two knots to speed up the run-time. The method of always optimizing two knots is denoted *ours*.

Evaluation is performed on the Event Camera Dataset [8], which contains recordings from a DAVIS camera. Ground truth feature tracks are computed from frames of the DAVIS camera using the KLT optical flow method [6]. We compare our methods against Zhu et al. [15]. To allow for a fair comparison against Zhu et al., we use the authors public MATLAB implementation and initialize the tracking with exactly the same feature positions as in our method, disabling the re-detection of new features.

We use four different metrics to do the evaluation. To illustrate the metrics clearly, the error of feature  $f_i$  at time  $t$  with using method  $m$  is denoted by  $e_m^i(t)$ . The lifetime of feature  $f_i$  before the error is larger than threshold  $th$  with using method  $m$  is denoted by  $L_m^{th}(f_i)$ . The definition of each metric with error threshold  $th$  are:

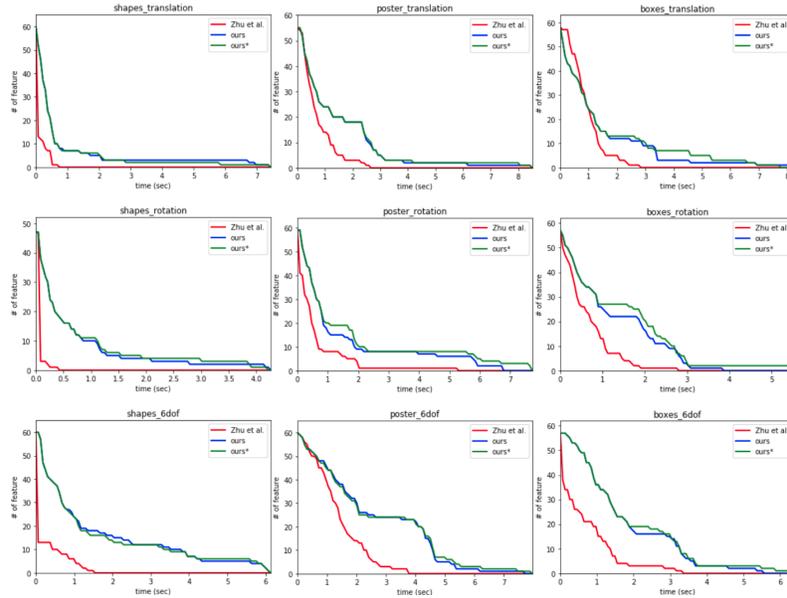
$$\text{mean relative age} = \left\langle \frac{L_m^{th}(f_i)}{L_{gt}(f_i)} \right\rangle_i \quad (11)$$

$$\text{mean age} = \langle L_m^{th}(f_i) \rangle_i \quad (12)$$

$$\text{mean error} = \langle \{e_m^i(t); t \leq L_m^{th}(f_i)\} \rangle_i \quad (13)$$

$$\text{mean common error} = \langle \{e_m^i(t); t \leq t_{min}\} \rangle_i \quad (14)$$

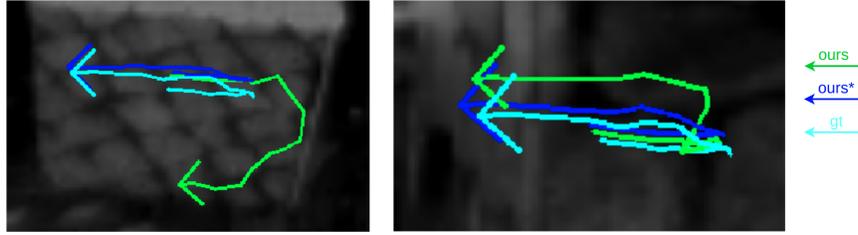
where  $\langle \dots \rangle_i$  takes average measurement of all features  $i$ ,  $t_{min} = \min\{L_{m_j}^{th}(f_i); j = 1, 2, \dots, N_m\}$  is the minimal feature lifetime and  $N_m$  is the number of methods we compare. We set the threshold  $th$  to 3 pixels in all experiments.



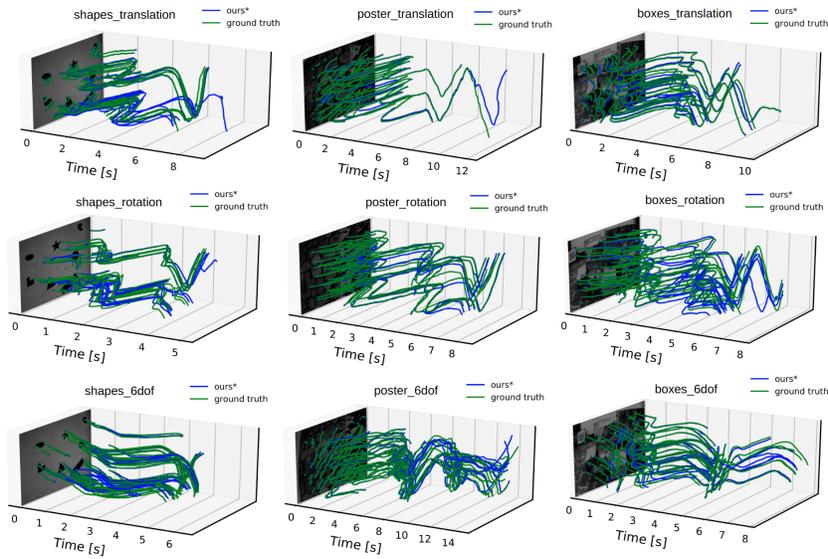
**Fig. 2.** Number of feature tracks over time with error threshold 3 for the Event Camera Dataset.

In Table 1, we compare our method to Zhu’s [15] approach. It shows that our method is always better than Zhu. By comparing *ours* and *ours\**, we can see that enlarging the window size during low-events periods can help to improve the mean age around 6% and the mean common error for *ours/ours\** almost remains unchanged. Fig. 3 shows some examples of features which are improved when using method *ours\**. Fig. 2 shows the number of feature tracks over time. The features in *ours\** last slightly longer than in *ours*. Compared to Zhu, our proposed algorithm tracks more features (with lower error) at almost all instances in time.

Since there is no public implementation of [10] available, and we are using different initial positions, we can only compare our result qualitatively. In Fig. 4 we show the 3D trajectories which can be used to compare the results to [10] qualitatively. It shows that our trajectories live longer than theirs.



**Fig. 3.** Example of the features in Boxes dataset which are improved in method *ours\** compared to method *ours*.



**Fig. 4.** Qualitative results of feature tracking for the Event Camera Dataset.

## 5 Conclusion

In this paper we proposed a novel event tracking algorithm that aligns the event stream with a B-spline curve representation in a sliding window fashion. By using a history patch, the locality of B-splines and a sliding window optimization, our algorithm can track features accurately and for a long time. Our experiments show that the proposed algorithm outperforms the state-of-the-art time-continuous event tracking algorithm. We believe that this method can serve as a basis for event-based video analysis and event-based SLAM. Future research aims at extending our algorithm to a Sim(2)-formulation, allowing to track features with scale changes.

## References

1. Alzugaray, I., Chli, M.: Asynchronous multi-hypothesis tracking of features with event cameras. In: 2019 International Conference on 3D Vision (3DV). pp. 269–278 (2019)
2. Brandli, C., Berner, R., Yang, M., Liu, S.C., Delbruck, T.: A  $240 \times 180$  130 db  $3 \mu\text{s}$  latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits* **49**(10), 2333–2341 (2014)
3. Gallego, G., Gehrig, M., Scaramuzza, D.: Focus is all you need: Loss functions for event-based vision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12280–12289 (2019)
4. Gallego, G., Rebecq, H., Scaramuzza, D.: A unifying contrast maximization framework for event cameras, with applications to motion, depth, and optical flow estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3867–3876 (2018)
5. Gehrig, D., Rebecq, H., Gallego, G., Scaramuzza, D.: EKLT: Asynchronous, photometric feature tracking using events and frames. *Int. J. Comput. Vis.* (2019)
6. Lucas, B.D., Kanade, T., et al.: An iterative image registration technique with an application to stereo vision. Vancouver, British Columbia (1981)
7. Manderscheid, J., Sironi, A., Bourdis, N., Migliore, D., Lepetit, V.: Speed invariant time surface for learning to detect corner points with event-based cameras. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10245–10254 (2019)
8. Mueggler, E., Rebecq, H., Gallego, G., Delbruck, T., Scaramuzza, D.: The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *The International Journal of Robotics Research* **36**(2), 142–149 (2017)
9. Müggler, E., Bartolozzi, C., Scaramuzza, D.: Fast event-based corner detection. In: British Machine Vision Conference (BMVC). pp. 1–8. British Machine Vision Conference (BMVC), London, 2017. (September 2017)
10. Seok, H., Lim, J.: Robust feature tracking in dvs event stream using bezier mapping. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) (March 2020)
11. Sommer, C., Usenko, V., Schubert, D., Demmel, N., Cremers, D.: Efficient derivative computation for cumulative b-splines on lie groups. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
12. Stoffregen, T., Gallego, G., Drummond, T., Kleeman, L., Scaramuzza, D.: Event-based motion segmentation by motion compensation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7244–7253 (2019)
13. Vasco, V., Glover, A.J., Bartolozzi, C.: Fast event-based harris corner detection exploiting the advantages of event-driven cameras. 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) pp. 4144–4149 (2016)
14. Ye, C., Mitrokhin, A., Fermüller, C., Yorke, J.A., Aloimonos, Y.: Unsupervised learning of dense optical flow, depth and egomotion from sparse event data. arXiv preprint arXiv:1809.08625 (2018)
15. Zhu, A.Z., Atanasov, N., Daniilidis, K.: Event-based feature tracking with probabilistic data association. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). pp. 4465–4470. IEEE (2017)
16. Zhu, A.Z., Yuan, L., Chaney, K., Daniilidis, K.: Unsupervised event-based learning of optical flow, depth, and egomotion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)