Event-based Vision: A Survey

Preprint · April 2019

CITATIONS
0

READS
1,819

11 authors, including:

Guillermo Gallego
Technische Universität Berlin
78 PUBLICATIONS **2,994** CITATIONS
SEE PROFILE

Tobi Delbruck
ETH Zurich
252 PUBLICATIONS **12,148** CITATIONS
SEE PROFILE

Garrick Orchard
Intel
74 PUBLICATIONS **1,749** CITATIONS
SEE PROFILE

Chiara Bartolozzi
Istituto Italiano di Tecnologia
93 PUBLICATIONS **2,603** CITATIONS
SEE PROFILE

Event-based Vision: A Survey

Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew Davison, Jörg Conradt, Kostas Daniilidis, Davide Scaramuzza

Abstract— Event cameras are bio-inspired sensors that differ from conventional frame cameras: Instead of capturing images at a fixed rate, they asynchronously measure per-pixel brightness changes, and output a stream of events that encode the time, location and sign of the brightness changes. Event cameras offer attractive properties compared to traditional cameras: high temporal resolution (in the order of μs), very high dynamic range ($140\,\mathrm{dB}$ vs. $60\,\mathrm{dB}$), low power consumption, and high pixel bandwidth (on the order of kHz) resulting in reduced motion blur. Hence, event cameras have a large potential for robotics and computer vision in challenging scenarios for traditional cameras, such as low-latency, high speed, and high dynamic range. However, novel methods are required to process the unconventional output of these sensors in order to unlock their potential. This paper provides a comprehensive overview of the emerging field of event-based vision, with a focus on the applications and the algorithms developed to unlock the outstanding properties of event cameras. We present event cameras from their working principle, the actual sensors that are available and the tasks that they have been used for, from low-level vision (feature detection and tracking, optic flow, etc.) to high-level vision (reconstruction, segmentation, recognition). We also discuss the techniques developed to process events, including learning-based techniques, as well as specialized processors for these novel sensors, such as spiking neural networks. Additionally, we highlight the challenges that remain to be tackled and the opportunities that lie ahead in the search for a more efficient, bio-inspired way for machines to perceive and interact with the world.

Index Terms—Event Cameras, Bio-Inspired Vision, Asynchronous Sensor, Low Latency, High Dynamic Range, Low Power.

1 Introduction and Applications

"The *brain is imagination, and that was exciting to me; I wanted to build a chip that could imagine something*[1]." that is how Misha Mahowald, a graduate student at Caltech in 1986, started to work with Prof. Carver Mead on the stereo problem from a joint biological and engineering perspective. A couple of years later, in 1991, the image of a cat in the cover of Scientific American [1], acquired by a novel "Silicon Retina" mimicking the neural architecture of the eye, showed a new, powerful way of doing computations, igniting the emerging field of neuromorphic engineering. Today, we still pursue the same visionary challenge: understanding how the brain works and building one on a computer chip. Current efforts include flagship billion-dollar projects, such as the Human Brain Project and the Blue Brain Project in Europe and the U.S. BRAIN (Brain Research through Advancing Innovative Neurotechnologies) Initiative.

This paper provides an overview of the bio-inspired technology of silicon retinas, or "event cameras", such as [2], [3], [4], [5], with a focus on their application to solve classical as well as new computer vision and robotic tasks. Sight is, by far, the dominant sense in humans to perceive the world, and, together with the brain, learn new things. In recent years, this technology has attracted a lot of attention from both academia and industry. This is due to the availability of prototype event cameras and the advantages that these devices offer to tackle problems that are difficult with standard frame-based image sensors (that provide stroboscopic synchronous sequences of 2D pictures), such as high-speed motion estimation [6], [7] or high dynamic range (HDR) imaging [8].

Event cameras are *asynchronous* sensors that pose a *paradigm shift* in the way visual information is acquired. This is because they sample light based on the scene dynamics, rather than on a clock that has no relation to the viewed scene. Their advantages are: very high temporal resolution and low latency (both in the order of microseconds), very high dynamic range ($140\,\mathrm{dB}$ vs. $60\,\mathrm{dB}$ of standard cameras), and low power consumption. Hence, event cameras have a large potential for robotics and wearable applications in challenging scenarios for standard cameras, such as high speed and high dynamic range. Although event cameras have become commercially available only since 2008 [2], the recent body of literature on these new sensors[2] as well as the recent plans for mass production claimed by companies, such as Samsung [5] and Prophesee[3], highlight that there is a big commercial interest in exploiting these novel vision sensors for mobile robotic, augmented and virtual reality (AR/VR), and video game applications. However, because event cameras work in a fundamentally different way from standard cameras, measuring per-pixel brightness changes

• *G. Gallego is with the Technische Universität Berlin, Berlin, Germany. Tobi Delbruck is with the Dept. of Information Technology and Electrical Engineering, ETH Zurich, at the Inst. of Neuroinformatics, University of Zurich and ETH Zurich, Zurich, Switzerland. Garrick Orchard is with Intel Corp., CA, USA. Chiara Bartolozzi is with the Italian institute of Technology, Genoa, Italy. Brian Taba is with IBM Research, CA, USA. Andrea Censi is with the Dept. of Mechanical and Process Engineering, ETH Zurich, Switzerland. Stefan Leutenegger and Andrew Davison are with Imperial College London, London, UK. Jörg Conradt is with KTH Royal Institute of Technology, Stockholm, Sweden. Kostas Daniilidis is with University of Pennsylvania, PA, USA. D. Scaramuzza is with the Dept. of Informatics University of Zurich and Dept. of Neuroinformatics, University of Zurich and ETH Zurich, Switzerland. February 27, 2020*

1. https://youtu.be/FKemf6Idkd0?t=67

2. https://github.com/uzh-rpg/event-based_vision_resources [9]
3. http://rpg.ifi.uzh.ch/ICRA17_event_vision_workshop.html

(called "*events*") asynchronously rather than measuring "absolute" brightness at constant rate, novel methods are required to process their output and unlock their potential.

Applications of Event Cameras: Typical scenarios where event cameras offer advantages over other sensing modalities include real-time interaction systems, such as robotics or wearable electronics [10], where operation under uncontrolled lighting conditions, latency, and power are important [11]. Event cameras are used for object tracking [12], [13], surveillance and monitoring [14], and object/gesture recognition [15], [16], [17]. They are also profitable for depth estimation [18], [19], structured light 3D scanning [20], optical flow estimation [21], [22], HDR image reconstruction [8], [23], [24] and Simultaneous Localization and Mapping (SLAM) [25], [26], [27]. Event-based vision is a growing field of research, and other applications, such as image deblurring [28] or star tracking [29], [30], will appear as event cameras become widely available.

Outline: The rest of the paper is organized as follows. Section 2 presents event cameras, their working principle and advantages, and the challenges that they pose as novel vision sensors. Section 3 discusses several methodologies commonly used to extract information from the event camera output, and discusses the biological inspiration behind some of the approaches. Section 4 reviews applications of event cameras, from low-level to high-level vision tasks, and some of the algorithms that have been designed to unlock their potential. Opportunities for future research and open challenges on each topic are also pointed out. Section 5 presents neuromorphic processors and embedded systems. Section 6 reviews the software, datasets and simulators to work on event cameras, as well as additional sources of information. The paper ends with a discussion (Section 7) and conclusions (Section 8).

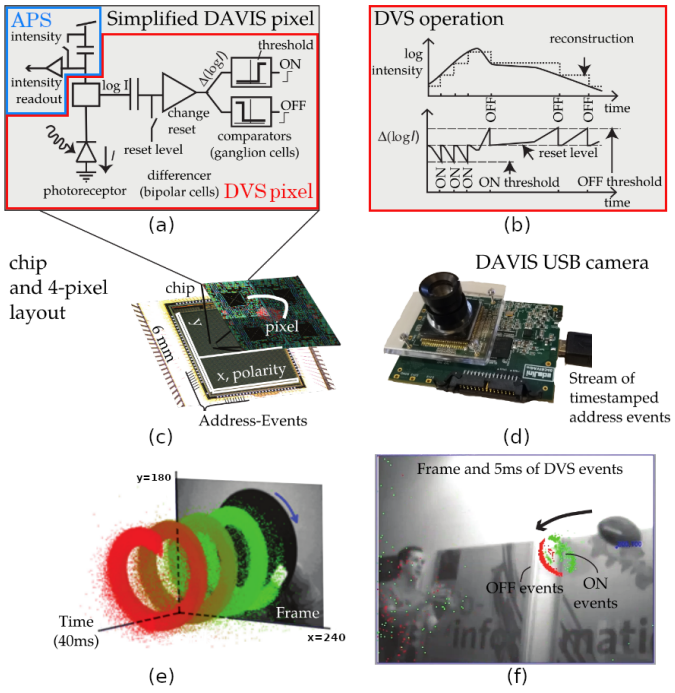


Figure 1. Summary of the DAVIS camera [4], comprising an event-based dynamic vision sensor (DVS [2]) and a frame-based active pixel sensor (**APS**) in the same pixel array, sharing the same photodiode in each pixel. **(a)** Simplified circuit diagram of the DAVIS pixel (DVS pixel in red, APS pixel in blue). **(b)** Schematic of the operation of a DVS pixel, converting light into events. **(c)-(d)** Pictures of the DAVIS chip and USB camera. **(e)** A white square on a rotating black disk viewed by the DAVIS produces grayscale frames and a spiral of evens in space-time. Events in space-time are color-coded, from green (past) to red (present). **(f)** Frame and overlaid events of a natural scene; the frames lag behind the low-latency events (colored according to polarity). Images adapted from [4], [35]. A more in-depth comparison of the DVS, DAVIS and ATIS pixel designs can be found in [36].

2 Principle of Operation of Event Cameras

In contrast to standard cameras, which acquire full images at a rate specified by an external clock (e.g., 30 fps), event cameras, such as the Dynamic Vision Sensor (**DVS**) [2], [31], [32], [33], [34], respond to *brightness changes* in the scene *asynchronously* and *independently* for every pixel (Fig. 1b). Thus, the output of an event camera is a variable data-rate sequence of digital "events" or "spikes", with each event representing a change of brightness (log intensity)[4] of predefined magnitude at a pixel at a particular time[5] (Fig. 1b) (Section 2.4). This encoding is inspired by the spiking nature of biological visual pathways (Section 3.3). Each pixel memorizes the log intensity each time it sends an event, and continuously monitors for a change of sufficient magnitude from this memorized value (Fig. 1a). When the change exceeds a threshold, the camera sends an event, which is transmitted from the chip with the x, y location, the time t, and the 1-bit polarity p of the change (i.e., brightness increase ("ON") or decrease ("OFF")). This event output is illustrated in Figs. 1b, 1e and 1f.

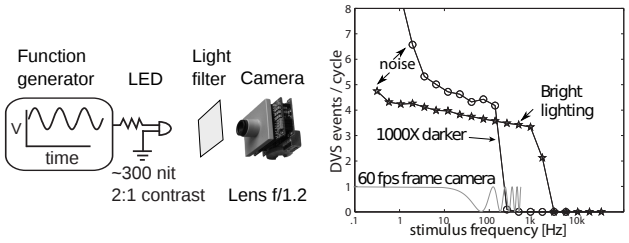
The events are transmitted from the pixel array to periphery and then out of the camera using a shared digital output bus, typically by using address-event representation (**AER**) readout [37], [38]. This bus can become saturated, which perturbs the times that events are sent. Event cameras have readout rates ranging from 2 MHz [2] to 1200 MHz [39], depending on the chip and type of hardware interface.

Event cameras are data-driven sensors: their output depends on the amount of motion or brightness change in the scene. The faster the motion, the more events per second are generated, since each pixel adapts its delta modulator sampling rate to the rate of change of the log intensity signal that it monitors. Events are timestamped with microsecond resolution and are transmitted with sub-millisecond latency, which make these sensors react quickly to visual stimuli.

The incident light at a pixel is a product of scene illumination and surface reflectance. Thus, a log intensity change in the scene generally signals a reflectance change (because usually the illumination is constant and the log of a product is the sum of the logs). These changes in reflectance are mainly the result of the movement of objects in the field

4. *Brightness* is a perceived quantity; for brevity we use it to refer to log intensity since they correspond closely for uniformly-lighted scenes.

5. Nomenclature: "Event cameras" output data-driven events that signal a place and time. This nomenclature has evolved over the past decade: originally they were known as address-event representation (AER) silicon retinas, and later they became event-based cameras. In general, events can signal any kind of information (intensity, local spatial contrast, etc.), but over the last five years or so, the term "event camera" has unfortunately become practically synonymous with the particular representation of brightness change output by DVS's.



(a) Measurement setup (b) Measured responses for two DC levels of illumination.

Figure 2. "Event transfer function" from a single DVS pixel in response to sinusoidal LED stimulation. The background events cause additional ON events at very low frequencies. The 60 fps camera curve shows the transfer function including aliasing from frequencies above the Nyquist frequency. Figure adapted from [2].

of view. That is why the DVS brightness change events have a built-in invariance to scene illumination [2].

Comparing Bandwidths of DVS Pixels and Frame-based Camera: Although DVS pixels are fast, like any physical transducer, they have a finite bandwidth: if the incoming light intensity varies too quickly, the front-end photoreceptor circuits filter out the variations. The rise and fall time that is analogous to the exposure time in standard image sensors is the reciprocal of this bandwidth. Fig. 2 shows an example of measured DVS pixel frequency response (DVS128 in [2]). The measurement setup (Fig. 2a) uses a sinusoidally-varying generated signal to measure the response. Fig. 2b shows that, at low frequencies, the DVS pixel produces a certain number of events per cycle. Above some cutoff frequency, the variations are filtered out by the photoreceptor dynamics and the number of events per cycle drops. This cutoff frequency is a monotonically increasing function of light intensity. At the brighter light intensity, the DVS pixel bandwidth is about 3 kHz, equivalent to an exposure time of about 300 µs. At 1000× lower intensity, the DVS bandwidth is reduced to about 300 Hz. Even when the LED brightness is reduced by a factor of 1000, the frequency response of DVS pixels is ten times higher than the 30 Hz Nyquist frequency from a 60 fps image sensor. Also, the frame-based camera aliases frequencies above the Nyquist frequency back to the baseband, whereas the DVS pixel does not due to the continuous time response.

2.1 Event Camera Designs

This section presents the most common event camera designs. The actual devices (commercial or prototype cameras such as the DAVIS240) are summarized in Section 2.5.

The first silicon retina was developed by Mahowald and Mead at Caltech during the period 1986-1992, in Ph.D. thesis work [40] that was awarded the prestigious Clauser prize[6]. Mahowald and Mead's sensor had logarithmic pixels, was modeled after the three-layer Kufler retina, and produced as output spike events using the AER protocol. However, it suffered from several shortcomings: each wire-wrapped retina board required precise adjustment of biasing potentiometers; there was considerable mismatch between the responses of different pixels; and pixels were too large to be a device of practical use. Over the next decade the neuromorphic community developed a series of silicon retinas. These developments are summarized in [36], [38], [41], [42].

The ***DVS*** *event camera* [2] had its genesis in a frame-based silicon retina design where the continuous-time photoreceptor was capacitively coupled to a readout circuit that was reset each time the pixel was sampled [43]. More recent event camera technology has been reviewed in the electronics and neuroscience literature [10], [36], [38], [44], [45], [46]. Although surprisingly many applications can be solved by only processing DVS events (i.e., brightness changes), it became clear that some also require some form of static output (i.e., "absolute" brightness). To address this shortcoming, there have been several developments of cameras that concurrently output dynamic and static information.

The *Asynchronous Time Based Image Sensor* (**ATIS**) [3], [47] has pixels that contain a DVS subpixel that triggers another subpixel to read out the absolute intensity. The trigger resets a capacitor to a high voltage. The charge is bled away from this capacitor by another photodiode. The brighter the light, the faster the capacitor discharges. The ATIS intensity readout transmits two more events coding the time between crossing two threshold voltages, as in [48]. This way, only pixels that change provide their new intensity values. The brighter the illumination, the shorter the time between these two events. The ATIS achieves large static dynamic range (>120 dB). However, the ATIS has the disadvantage that pixels are at least double the area of DVS pixels. Also, in dark scenes the time between the two intensity events can be long and the readout of intensity can be interrupted by new events ([49] proposes a workaround to this problem).

The widely-used *Dynamic and Active Pixel Vision Sensor* (**DAVIS**) [4], [50] illustrated in Fig. 1 combines a conventional active pixel sensor (**APS**) [51] in the same pixel with DVS. The advantage over ATIS is a much smaller pixel size since the photodiode is shared and the readout circuit only adds about 5 % to the DVS pixel area. Intensity (APS) frames can be triggered at a constant frame rate or on demand, by analysis of DVS events, although the latter is seldom exploited[7]. However, the APS readout has limited dynamic range (55 dB) and like a standard camera, it is redundant if the pixels do not change.

Since the ATIS and DAVIS pixel designs include a DVS pixel (change detector) [36] we often use the term "DVS" to refer to the binary-polarity event output or circuitry, regardless of whether it is from a DVS, ATIS or DAVIS design.

2.2 Advantages of Event cameras

Event cameras offer numerous potential advantages over standard cameras:

High Temporal Resolution: monitoring of brightness changes is fast, in analog circuitry, and the read-out of the events is digital, with a 1 MHz clock, i.e., events are detected and timestamped with microsecond resolution. Therefore, event cameras can capture very fast motions, without suffering from motion blur typical of frame-based cameras.

Low Latency: each pixel works independently and there is no need to wait for a global exposure time of the frame:

6. http://www.gradoffice.caltech.edu/current/clauser

7. https://github.com/SensorsINI/jaer/blob/master/src/eu/seebetter/ini/chips/davis/DavisAutoShooter.java

as soon as the change is detected, it is transmitted. Hence, event cameras have minimal latency: about $10\,\mu s$ on the lab bench, and sub-millisecond in the real world.

Low Power: Because event cameras transmit only brightness changes, and thus remove redundant data, power is only used to process changing pixels. At the die level, most cameras use about $10\,mW$, and there are prototypes that achieve less than $10\,\mu W$. Embedded event-camera systems where the sensor is directly interfaced to a processor have shown system-level power consumption (i.e., sensing plus processing) of $100\,mW$ or less [17], [52], [53], [54].

High Dynamic Range (HDR). The very high dynamic range of event cameras ($>120\,dB$) notably exceeds the $60\,dB$ of high-quality, frame-based cameras, making them able to acquire information from moonlight to daylight. It is due to the facts that the photoreceptors of the pixels operate in logarithmic scale and each pixel works independently, not waiting for a global shutter. Like biological retinas, DVS pixels can adapt to very dark as well as very bright stimuli.

2.3 Challenges Due To The Novel Sensing Paradigm

Event cameras represent a paradigm shift in acquisition of visual information. Hence, they pose the challenge of designing novel methods (algorithms and hardware) to process the acquired data and extract information from it in order to unlock the advantages of the camera. Specifically:

1) *Coping with different space-time output:* The output of event cameras is fundamentally different from that of standard cameras: events are asynchronous and spatially sparse, whereas images are synchronous and dense. Hence, frame-based vision algorithms designed for image sequences are not directly applicable to event data.

2) *Coping with different photometric sensing:* In contrast to the grayscale information that standard cameras provide, each event contains binary (increase/decrease) brightness change information. Brightness changes depend not only on the scene brightness, but also on the current and past motion between the scene and the camera.

3) *Coping with noise and dynamic effects:* All vision sensors are noisy because of the inherent shot noise in photons and from transistor circuit noise, and they also have non-idealities. This situation is especially true for event cameras, where the process of quantizing temporal contrast is complex and has not been completely characterized.

Therefore, new methods need to rethink the space-time, photometric and stochastic nature of event data. This poses the following questions: What is the best way to extract information from the events relevant for a given task? and How can noise and non-ideal effects be modeled to better extract meaningful information from the events?

2.4 Event Generation Model

An event camera [2] has independent pixels that respond to changes in their log photocurrent $L \doteq \log(I)$ ("brightness"). Specifically, in a noise-free scenario, an event $e_k \doteq (\mathbf{x}_k, t_k, p_k)$ is triggered at pixel $\mathbf{x}_k \doteq (x_k, y_k)^\top$ and at time t_k as soon as the brightness increment since the last event at the pixel, i.e.

$$\Delta L(\mathbf{x}_k, t_k) \doteq L(\mathbf{x}_k, t_k) - L(\mathbf{x}_k, t_k - \Delta t_k), \tag{1}$$

reaches a temporal contrast threshold $\pm C$ (Fig. 1b), i.e.,

$$\Delta L(\mathbf{x}_k, t_k) = p_k\, C, \tag{2}$$

where $C > 0$, Δt_k is the time elapsed since the last event at the same pixel, and the polarity $p_k \in \{+1, -1\}$ is the sign of the brightness change [2].

The contrast sensitivity C is determined by the pixel bias currents [55], [56], which set the speed and threshold voltages of the change detector in Fig. 1 and are generated by an on-chip digitally-programmed bias generator. The sensitivity C can be estimated knowing these currents [55]. In practice, positive ("ON") and negative ("OFF") events may be triggered according to different thresholds, C^+, C^-. Typical DVS's [2], [5] can set thresholds between $10\,\%$–$50\,\%$ illumination change. The lower limit on C is determined by noise and pixel-to-pixel mismatch (variability); setting C too low results in a storm of noise events, starting from pixels with low values of C. Experimental DVS's with higher photoreceptor gain are capable of lower thresholds, e.g., $1\,\%$ [57], [58], [59]; however these values are only obtained under very bright illumination and ideal conditions. Fundamentally, the pixel must react to a small change in the photocurrent in spite of the shot noise present in this current. This shot noise limitation sets the relation between threshold and speed of the DVS under a particular illumination and desired detection reliability condition [59], [60].

Events and the Temporal Derivative of Brightness: Eq. (2) states that event camera pixels set a threshold on magnitude of the brightness change since the last event happened. For a small Δt_k, such an increment (2) can be approximated using Taylor's expansion by $\Delta L(\mathbf{x}_k, t_k) \approx \frac{\partial L}{\partial t}(\mathbf{x}_k, t_k)\Delta t_k$, which allows us to interpret the events as providing information about the temporal derivative:

$$\frac{\partial L}{\partial t}(\mathbf{x}_k, t_k) \approx \frac{p_k\, C}{\Delta t_k}. \tag{3}$$

This is an indirect way of measuring brightness, since with standard cameras we are used to measuring absolute brightness. Note that DVS events are triggered by a change in brightness magnitude (2), not by the brightness derivative (3) exceeding a threshold. The above interpretation may be taken into account to design physically-grounded event-based algorithms, such as [7], [23], [24], [28], [61], [62], [63], [64], as opposed to algorithms that simply process events as a collection of points with vague photometric meaning.

Events are Caused by Moving Edges: Assuming constant illumination, linearizing (2) and using the brightness constancy assumption one can show that events are caused by moving edges. For small Δt, the intensity increment (2) can be approximated by[8]:

$$\Delta L \approx -\nabla L \cdot \mathbf{v} \Delta t, \tag{4}$$

that is, it is caused by an brightness gradient $\nabla L(\mathbf{x}_k, t_k) = (\partial_x L, \partial_y L)^\top$ moving with velocity $\mathbf{v}(\mathbf{x}_k, t_k)$ on the image plane, over a displacement $\Delta \mathbf{x} \doteq \mathbf{v}\Delta t$. As the dot product (4) conveys: (*i*) if the motion is parallel to the edge, no

8. Eq. (4) can be shown [65] by substituting the brightness constancy assumption (i.e., optical flow constraint) $\frac{\partial L}{\partial t}(\mathbf{x}(t), t) + \nabla L(\mathbf{x}(t), t) \cdot \dot{\mathbf{x}}(t) = 0$, with image-point velocity $\mathbf{v} \equiv \dot{\mathbf{x}}$, in Taylor's approximation $\Delta L(\mathbf{x}, t) \doteq L(\mathbf{x}, t) - L(\mathbf{x}, t - \Delta t) \approx \frac{\partial L}{\partial t}(\mathbf{x}, t)\Delta t$.

event is generated since $\mathbf{v} \cdot \nabla L = 0$; (*ii*) if the motion is perpendicular to the edge ($\mathbf{v} \parallel \nabla L$) events are generated at the highest rate (i.e., minimal time is required to achieve a brightness change of size $|C|$).

Probabilistic Event Generation Models: Equation (2) is an idealized model for the generation of events. A more realistic model takes into account sensor noise and transistor mismatch, yielding a mixture of frozen and temporally varying stochastic triggering conditions represented by a probability function, which is itself a complex function of local illumination level and sensor operating parameters. The measurement of such probability density was shown in [2] (for the DVS128), suggesting a normal distribution centered at the contrast threshold C. The 1-σ width of the distribution is typically 2-4% temporal contrast. This event generation model can be included in emulators [72] and simulators [73] of event cameras, and in event processing algorithms [24], [65]. Other probabilistic event generation models have been proposed, such as: the likelihood of event generation being proportional to the magnitude of the image gradient [74] (for scenes where large intensity gradients are the source of most event data), or the likelihood being modeled by a mixture distribution to be robust to sensor noise [7]. Future even more realistic models may include the refractory period (i.e., the duration in time that the pixel ignores log brightness changes after it has generated an event; the larger the refractory period the fewer events are produced by fast moving objects), and bus congestion [75].

The above event generation models are simple, developed to some extent based on sensor noise characterization. Just like standard image sensors, DVS's also have fixed pattern noise (**FPN**[9]), but in DVS it manifests as pixel-to-pixel variation in the event threshold. Standard DVS's can achieve minimum $C \approx \pm 15\,\%$, with a standard deviation of about $2.5\,\%$–$4\,\%$ contrast between pixels [2], [76], and there have been attempts to measure pixelwise thresholds by comparing brightness changes due to DVS events and due to differences of consecutive DAVIS APS frames [77]. However, understanding of *temporal* DVS pixel and readout noise is preliminary [2], [58], [75], [78], and noise filtering methods have been developed mainly based on computational efficiency, assuming that events from real objects should be more correlated spatially and temporally than noise events [38], [79], [80], [81], [82]. We are far from having a model that can predict event camera noise statistics under arbitrary illumination and biasing conditions. Solving this challenge would lead to better estimation methods.

2.5 Event Camera Availability

Table 1 summarizes the most popular or recent cameras. The numbers therein are approximate since they were not measured using a common testbed. Event camera characteristics are considerably different from other CMOS image sensor (**CIS**) technology, and so there is a need for an agreement on standard specifications to be better used by researchers. As Table 1 shows, since the first practical event camera [2] there has been a trend mainly to increase spatial resolution, increase readout speed, and add features, such as: gray level output (in ATIS and DAVIS), integration with an Inertial Measurement Unit (**IMU**) [83] and multi-camera timestamp synchronization [84]. IMUs act as a vestibular sense that may improve camera pose estimation, as in visual-inertial odometry. Only recently has the focus turned more towards the difficult task of reducing pixel size for economical mass production of sensors with large pixel arrays.

Pixel Size: The most widely used event cameras have quite large pixels: $40\,\mu\mathrm{m}$ (DVS128), $30\,\mu\mathrm{m}$ (ATIS), $18.5\,\mu\mathrm{m}$ (DAVIS240, DAVIS346) (Table 1). The smallest published DVS pixel [67] is $4.86\,\mu\mathrm{m}$; while conventional global shutter industrial APS are typically in the range of $2\,\mu\mathrm{m}$–$4\,\mu\mathrm{m}$. Low spatial resolution is certainly a limitation for application, although many of the seminal publications are based on the 128×128 pixel DVS128 [2]. The DVS with largest published array size has only about $1\,\mathrm{Mpixel}$ spatial resolution (1280×960 pixels [39]). Event camera pixel size has shrunk pretty closely following feature size scaling, which is remarkable considering that a DVS pixel is a mixed-signal circuit, which generally do not scale following technology. However, achieving even smaller pixels is difficult and may require abandoning the strictly asynchronous circuit design philosophy that the cameras started with. Camera cost is constrained by die size (since silicon costs about \$5-\$10/cm^2 in mass production), and optics (designing new mass production miniaturized optics to fit a different sensor format can cost tens of millions of dollars).

Fill Factor: A major obstacle for early event camera mass production prospects was the limited fill factor of the pixels (i.e., the ratio of a pixel's light sensitive area to its total area). Because the pixel circuit is complex, a smaller pixel area can be used for the photodiode that collects light. For example, a pixel with $20\,\%$ fill factor throws away 4 out of 5 photons. Obviously this is not acceptable for optimum performance; nonetheless, even the earliest event cameras could sense high contrast features under moonlight illumination [2]. Early CIS sensors dealt with this problem by including microlenses that focused the light onto the pixel photodiode. What is probably better, however, is to use back-side illumination technology (**BSI**). BSI flips the chip so that it is illuminated from the back, so that in principle the entire pixel area can collect photons. Nearly all smartphone cameras are now back illuminated, but the additional cost and availability of BSI fabrication has meant that only recently BSI event cameras were first demonstrated [85]. BSI also brings problems: light can create additional 'parasitic' photocurrents that lead to spurious 'leak' events [55].

Cost: Currently, a practical obstacle to adoption of event camera technology is the high cost of several thousand dollars per camera, similar to the situation with early time of flight, structured lighting and thermal cameras. The high costs are due to non-recurring engineering costs for the silicon design and fabrication (even when much of it is provided by research funding) and the limited samples available from prototype runs. It is anticipated that this price will drop precipitously once this technology enters mass production, as shown by the "Samsung SmartThings Vision" consumer-grade home monitoring device: it contains an event camera [5] and sells for 130 dollars.

9. https://en.wikipedia.org/wiki/Fixed-pattern_noise

Table 1
Comparison of commercial or prototype event cameras. Values are approximate since there is no standard measurement testbed.

	Supplier	iniVation			Prophesee				Samsung			CelePixel		Insightness
	Camera model	DVS128	DAVIS240	DAVIS346	ATIS	Gen3 CD	Gen3 ATIS	Gen 4 CD	DVS-Gen2	DVS-Gen3	DVS-Gen4	CeleX-IV	CeleX-V	Rino 3
Sensor specifications	Year, Reference	2008 [2]	2014 [4]	2017	2011 [3]	2017 [66]	2017 [66]	2020 [67]	2017 [5]	2018 [68]	2020 [39]	2017 [69]	2019 [70]	2018 [71]
	Resolution (pixels)	128 × 128	240 × 180	346 × 260	304 × 240	640 × 480	480 × 360	1280 × 720	640 × 480	640 × 480	1280 × 960	768 × 640	1280 × 800	320 × 262
	Latency (µs)	12μs @ 1klux	12μs @ 1klux	20	3	40 - 200	40 - 200	20 - 150	65 - 410	50	150	10	8	125μs @ 10lux
	Dynamic range (dB)	120	120	120	143	> 120	> 120	> 124	90	90	100	90	120	> 100
	Min. contrast sensitivity (%)	17	11	14.3 - 22.5	13	12	12	11	9	15	20	30	10	15
	Power consumption (mW)	23	5 - 14	10 - 170	50 - 175	36 - 95	25 - 87	32 - 73	27 - 50	40	130	-	400	20-70
	Chip size (mm2)	6.3 × 6	5 × 5	8 × 6	9.9 × 8.2	9.6 × 7.2	9.6 × 7.2	6.22 × 3.5	8 × 5.8	8 × 5.8	8.4 × 7.6	15.5 × 15.8	14.3 × 11.6	5.3 × 5.3
	Pixel size (µm2)	40 × 40	18.5 × 18.5	18.5 × 18.5	30 × 30	15 × 15	20 × 20	4.86 × 4.86	9 × 9	9 × 9	4.95 × 4.95	18 × 18	9.8 × 9.8	13 × 13
	Fill factor (%)	8.1	22	22	20	25	20	> 77	11	12	22	8.5	8	22
	Supply voltage (V)	3.3	1.8 & 3.3	1.8 & 3.3	1.8 & 3.3	1.8	1.8	1.1 & 2.5	1.2 & 2.8	1.2 & 2.8		1.8 & 3.3	1.2 & 2.5	1.8 & 3.3
	Stationary noise (ev/pix/s) at 25C	0.05	0.1	0.1	-	0.1	0.1	0.1	0.03	0.03		0.15	0.2	0.1
	CMOS technology (nm)	350	180	180	180	180	180	90	90	90	65/28	180	65	180
		2P4M	1P6M MIM	1P6M MIM	1P6M	1P6M CIS	1P6M CIS	BI CIS	1P5M BSI			1P6M CIS	CIS	1P6M CIS
	Grayscale output	no	yes	yes	yes	no	yes	no	no	no	no	yes	yes	yes
	Grayscale dynamic range (dB)	NA	55	56.7	130	NA	> 100	NA	NA	NA	NA	90	120	50
	Max. frame rate (fps)	NA	35	40	NA	NA	NA	NA	NA	NA	NA	50	100	30
Camera	Max. Bandwidth (Meps)	1	12	12	-	66	66	1066	300	600	1200	200	140	20
	Interface	USB 2	USB 2	USB 3		USB 3	USB 3	USB 3	USB 2	USB 3	USB 3			USB 2
	IMU output	no	1 kHz	1 kHz	no	1 kHz	1 kHz	no	no	1 kHz	no	no	no	1 kHz

Advanced Event Cameras

There are active developments of more advanced event cameras that are only available through scientific collaborations with the developers. Next, we discuss issues related to advanced camera developments and the types of new cameras that are being developed.

Color: Most diurnal animals have some form of color vision, and most conventional cameras offer color sensitivity. Early attempts at color sensitive event cameras [86], [87], [88] tried to use the "vertacolor" principle of splitting colors according to the amount of penetration of the different light wavelengths into silicon, pioneered by Foveon [89], [90]. However, it resulted in poor color separation performance. So far, there are few publications of practical color event cameras, with either integrated color filter arrays (**CFA**) [91], [92], [93] or color-splitter prisms [94]; splitters have a much higher cost than CFA.

Higher Contrast Sensitivity: Efforts have been made to improve the temporal contrast sensitivity of event cameras, leading to experimental sensors with higher sensitivity [57], [58], [59] (down to laboratory condition $\sim 1\,\%$). These sensors are based on variations of the idea of a thermal bolometer [95], i.e., increasing the gain before the change detector (Fig. 1) to reduce the input-referred FPN. However this intermediate preamplifier requires active gain control to avoid clipping. Increasing the contrast sensitivity is possible, at the expense of decreasing the dynamic range (e.g., [5]).

3 Event Processing

One of the key questions of the paradigm shift posed by event cameras is how to extract meaningful information from the event data to fulfill a given task. This is a very broad question, since the answer is application dependent, and it drives the algorithmic design of the task solver.

Event cameras acquire information in an asynchronous and sparse way, with high temporal resolution and low latency. Hence, the temporal aspect, specially latency, plays an essential role in the way events are processed. Depending on how many events are processed simultaneously, two categories of algorithms can be distinguished: (*i*) methods that operate on an *event-by-event basis*, where the state of the system (the estimated unknowns) can change upon the arrival of a single event, thus achieving minimum latency, and (*ii*) methods that operate on *groups or packets of events*, which introduce some latency. Discounting latency considerations, methods based on groups (i.e., temporal windows) of events can still provide a state update upon the arrival of each event if the window slides by one event. Hence, the distinction between both categories is more subtle: an event alone does not provide enough information for estimation, and so additional information, in the form of past events or extra knowledge, is needed. We review this categorization.

Orthogonally, depending on how events are processed, we can distinguish between model-based approaches and model-free (i.e., data-driven, machine learning) approaches. Assuming events are processed in an optimization framework, another classification concerns the type of objective or loss function used: geometric- vs. temporal- vs. photometric-based (e.g., a function of the event polarity or the event activity). Each category presents methods with advantages and disadvantages and current research focuses on exploring the possibilities that each method can offer.

3.1 Event Representations

Events are processed and often transformed into alternative representations (Fig. 3) that facilitate the extraction of meaningful information ("features") to solve a given task. Here we review popular representations of event data. Several of them arise from the need to aggregate the little information conveyed by individual events in the absence of additional knowledge. Some representations are simple, hand-crafted data transformations whereas others are more elaborate.

Individual events $e_k \doteq (\mathbf{x}_k, t_k, p_k)$ are used by event-by-event processing methods, such as probabilistic filters and Spiking Neural Networks (**SNNs**) (Section 3.3). The filter or SNN has additional information, built up from past events or given by additional knowledge, that is fused with the incoming event asynchronously to produce an output. Examples include: [7], [24], [61], [96], [97].

Event packet: Events $\mathcal{E} \doteq \{e_k\}_{k=1}^{N_e}$ in a spatio-temporal neighborhood are processed together to produce an output. Precise timestamp and polarity information is retained by this representation. Choosing the appropriate packet size N_e is critical to satisfy the assumptions of the algorithm (e.g., constant motion speed during the span of the packet), which varies with the task. Examples are [18], [19], [98], [99].

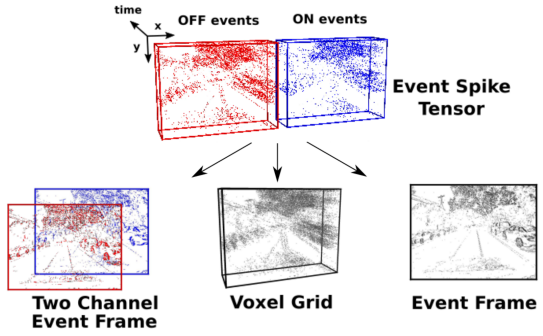


Figure 3. Different ways to convert events into more familiar representations, suitable for processing with modern learning architectures [111].

Event frame/image or 2D histogram: The events in a spatio-temporal neighborhood are converted in a simple way (e.g., pixel-wise by counting events, accumulating polarity, etc.) into an image (2D grid) that can be fed to image-based computer vision algorithms. Some algorithms may work in spite of the different statistics of event frames and natural images. However, this practice is not ideal in the event-based paradigm because it quantizes event timestamps, discards sparsity, and the resulting images are highly sensitive to the number of events used. Nevertheless the high impact of event frames in the literature [23], [26], [63], [100], [101], [102] is clear because (*i*) they are a simple way to convert an unfamiliar event stream into a familiar 2D representation that contains spatial information about scene edges, which are the most informative regions in natural images, (*ii*) they inform not only about the presence of events but also about their absence (which is informative), (*iii*) they have an intuitive interpretation (e.g., an edge map, a brightness increment image) and (*iv*) they are the data structure compatible with conventional computer vision.

Time surface (TS): A TS is a 2D map where each pixel stores a single time value (e.g., the timestamp of the last event at that pixel [79], [103]). Thus events are converted into an image whose "intensity" is a function of the motion history at that location, with brighter values corresponding to a more recent motion. TSs are called Motion History Images in classical computer vision [104]. They explicitly expose the rich temporal information of the events and can be updated asynchronously. Using an exponential kernel, TSs emphasize recent events over past events. To achieve invariance to motion speed, normalization is proposed [105], [106]. Compared to other grid-like representations of events, TSs highly compress information as they only keep one timestamp per pixel, thus their effectiveness degrades on textured scenes, in which pixels spike frequently. To make TSs less sensitive to noise, each pixel value may be computed by filtering the events in a space-time window [107]. More examples include [21], [108], [109], [110].

Voxel Grid: is a space-time (3D) histogram of events, where each voxel represents a particular pixel and time interval. This representation preserves better the temporal information of the events by avoiding to collapse them on a 2D grid (Fig. 3). If polarity is used the voxel grid is an intuitive discretization of a scalar field (polarity $p(x, y, t)$ or brightness variation $\partial L(x, y, t)/\partial t$) defined on the image plane, with absence of events marked by zero polarity. Each event's polarity may be accumulated on a voxel [112], [113] or spread among its closest voxels using a kernel [8], [114], [115]. Both schemes quantize event timestamps but the latter (interpolated voxel grid) provides sub-voxel accuracy.

3D point set: Events in a spatio-temporal neighborhood are treated as points in 3D space, $(x_k, y_k, t_k) \in \mathbb{R}^3$. Thus the temporal dimension becomes a geometric one. It is a sparse representation, and is used on point-based geometric processing methods, such as plane fitting [21] or PointNet [116].

Point sets on image plane: Events are treated as an evolving set of 2D points on the image plane. It is a popular representation among early shape tracking methods based on mean-shift or ICP [117], [118], [119], [120], [121], where events provide the only data needed to track edge patterns.

Motion-compensated event image [122], [123]: is a representation that depends not only on events but also on motion hypothesis. The idea of motion compensation is that, as an edge moves on the image plane, it triggers events on the pixels it traverses; the motion of the edge can be estimated by warping the events to a reference time and maximizing their alignment, producing a sharp image (i.e., histogram) of warped events (IWE) [123]. Hence, this representation (IWE) suggests a criterion to measure how well events fit a candidate motion: the sharper the edges produced by warping events, the better the fit [124]. Moreover, the resulting motion-compensated images have an intuitive meaning (i.e., the edge patterns causing the events) and provide a more familiar representation of visual information than the events. In a sense, motion compensation reveals a hidden ("motion-invariant") map of edges in the event stream. The images may be useful for further processing, such as feature tracking [63], [125]. There are motion-compensated versions of point sets [126], [127] and time surfaces [128], [129].

Reconstructed images: Brightness images obtained by image reconstruction (Section 4.6) can be interpreted as a more motion-invariant representation than event frames or TSs, and be used for inference [8] yielding first-rate results.

Event polarity may be considered in two ways: processing positive and negative events separately and merging results (e.g., using TSs [103]), or processing them together in a common representation (e.g., brightness increments images [63]), where polarity is often aggregated among neighboring events. Event polarity depends on motion direction, hence it is a nuisance for tasks that should be independent of motion, such as object recognition (to mitigate this, training data from multiple motion directions should be available). For motion estimation tasks, polarity may be useful, especially to detect abrupt changes of direction.

A general framework for converting event data into some of the above grid-based representations is presented in [111]. It also studies how the choice of representation passed to an artificial neural network (**ANN**) affects task performance and consequently proposes to automatically learn the representation that maximizes such performance.

3.2 Methods for Event Processing

Event processing systems consist of several stages: preprocessing (input adaptation), core processing (feature extraction and analysis) and post-processing (output creation). The event representations in Section 3.1 may occur at different stages: for example, in [122] an event packet is used at

pre-processing, and motion-compensated event images are the internal representation at the core processing stage. In other cases, the above representations may be used only at pre-processing: in [22] events are converted to event images and time surfaces that are then processed by an ANN.

The methods used to process events are influenced by the choice of representation and hardware platform available. These three factors influence each other. For example, it is natural to use dense representations and design algorithms accordingly that are executed on standard processors (e.g., CPUs or GPUs). At the same time, it is also natural to process events one-by-one on SNNs (Section 3.3) that are implemented on neuromorphic hardware (Section 5.1), in search for more efficient and low-latency solutions. Major exponents of event-by-event methods are filters (deterministic or probabilistic) and SNNs. For events processed in packets there are also many methods: hand-crafted feature extractors, deep neural networks (**DNNs**), etc. Next, we review some of the most common methods.

Event-by-event–based Methods: Deterministic filters, such as (space-time) convolutions and activity filters have been used for noise reduction, feature extraction [130], image reconstruction [61], [131] and brightness filtering [62], among other applications. Probabilistic filters (Bayesian methods), such as Kalman- and particle filters have been used for pose tracking in SLAM systems [7], [24], [25], [74], [96]. These methods rely on the availability of additional information (typically "appearance" information, e.g., grayscale images or a map of the scene), which may be provided by past events or by additional sensors. Then, each incoming event is compared against such information and the resulting mismatch provides innovation to update the filter state. Filters are a dominant class of methods for event-by-event processing because they naturally (*i*) handle asynchronous data, thus providing minimum processing latency, preserving the sensor's characteristics, and (*ii*) aggregate information from multiple small sources (e.g., events).

The other dominant class of methods takes the form of a multi-layer ANN (whether spiking or not) containing many parameters which must be computed from the event data. Networks trained with unsupervised learning typically act as feature extractors for a classifier (e.g., SVM), which still requires some labeled data for training [15], [103], [132]. If enough labeled data is available, supervised learning methods such as backpropagation can be used to train a network without the need for a separate classifier. Many approaches use packets of events during training (deep learning on frames), and later convert the trained network to an SNN that processes data event-by-event [133], [134], [135], [136], [137]. Event-by-event model-free methods have mostly been applied to classify objects [15], [103], [133], [134] or actions [16], [17], [138], and have targeted embedded applications [133], often using custom SNN hardware [15], [17] (Section 5.1). SNNs trained with deep learning typically provide higher accuracy than those relying on unsupervised learning for feature extraction, but there is growing interest in finding efficient ways to implement supervised learning directly in SNNs [138], [139] and in embedded devices [140].

Methods for Groups of Events: Because each event carries little information and is subject to noise, several events are often processed together to yield a sufficient signal-to-noise ratio for the problem considered. Methods for groups of events use the above representations (event packet, event frame, etc.) to gather the information contained in the events in order to estimate the problem unknowns, usually without requiring additional data. Hence, events are processed differently depending on their representation.

Many representations just perform data pre-processing to enable the re-utilization of image-based computer vision tools. In this respect, *event frames* are a practical representation that has been used by multiple methods on various tasks. In [100], [141] event frames allow to re-utilize traditional stereo methods, providing modest results. They also provide an adaptive frame rate signal that is profitable for camera pose estimation [26] (by image alignment) or optical flow computation [101] (by block matching). Event frames are also a simple yet effective input for image-based learning methods (DNNs, SVMs, Random Forests) [22], [102], [142], [143]. Few works design algorithms taking into account their photometric meaning (4). This was done in [23], showing that such a simple representation allows to jointly compute several visual quantities of interest (optical flow, brightness, etc.). Intensity increment images (4) are also used for feature tracking [63], image deblurring [28] or camera tracking [64].

Because *time surfaces* (TSs) are sensitive to scene edges and the direction of motion they have been utilized for many tasks involving motion analysis and shape recognition. For example, fitting local planes to the TS yields optical flow information [21], [144]. TSs are used as building blocks of hierarchical feature extractors, similar to neural networks, that aggregate information from successively larger space-time neighborhoods and is then passed to a classifier for recognition [103], [107]. TSs provide proxy intensity images for matching in stereo methods [110], [145], where the photometric matching criterion becomes temporal: matching pixels based on event concurrence and similarity of event timestamps across image planes. Recently, TSs have been probed as input to convolutional ANNs (**CNNs**) to compute optical flow [22], where the network acts both as feature extractor and velocity regressor. TSs are popular for corner detection using adaptations of image-based methods (Harris, FAST) [105], [108], [109] or new learning-based ones [106]. However, their performance degrades on highly textured scenes [109] due to the "motion overwriting" problem [104].

Methods working on *voxel grids* include variational optimization and ANNs (e.g., DNNs). They require more memory and often more computations than methods working on lower dimensional representations but are able to provide better results because temporal information is better preserved. In these methods voxel grids are used as an internal representation [112] (e.g., to compute optical flow) or as the multichannel input/output of a DNN [114], [115]. Thus, voxel grids are processed by means of convolutions [114], [115] or the operations derived from the optimality conditions of an objective function [112].

Once events have been converted to grid-like representations, countless tools from conventional vision can be applied to extract information: from feature extractors (e.g., CNNs) to similarity metrics (e.g., cross-correlation) that measure the goodness of fit or consistency between data and task-model hypothesis (the degree of event alignment, etc.).

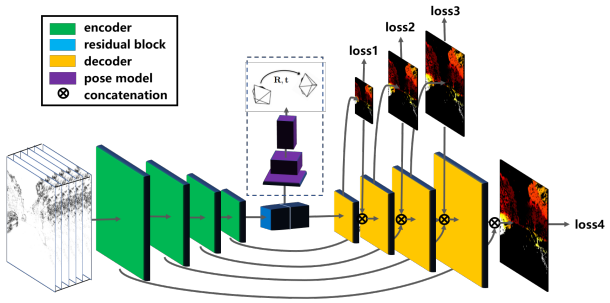


Figure 4. Events in a space-time volume are converted into an interpolated voxel grid (left) that is fed to a DNN to compute optical flow and ego-motion in an unsupervised manner [114]. Thus, modern tensor-based DNN architectures are re-utilized using novel loss functions (e.g., motion compensation) adapted to event data.

Such metrics are used as objective functions for classification (SVMs, CNNs), clustering, data association, motion estimation, etc. In the neuroscience literature there are efforts to design metrics that act directly on spikes (e.g., event stream), to avoid the issues that arise due to data conversion.

Deep learning methods for groups of events consist of a deep neural network (DNN). Sample applications include classification [146], [147], image reconstruction [8], [113], steering angle prediction [102], [148], and estimation of optical flow [22], [114], [149], depth [149] or ego-motion [114]. These methods differentiate themselves mainly in the representation of the input and in the loss functions optimized during training. Several representations have been used, such as event images [102], [143], TSs [22], [129], [149], voxel grids [114], [115] or point sets [116] (Section 3.1). While loss functions in classification tasks use manually annotated labels, networks for regression tasks from events may be supervised by a third party ground truth (e.g., a pose) [102], [143] or by an associated grayscale image [22] to measure photoconsistency, or be completely unsupervised (depending only on the training input events) [114], [149]. Loss functions for unsupervised learning from events are studied in [124]. In terms of architecture, most networks have an encoder-decoder structure, as in Fig. 4. Such a structure allows the use of convolutions only, thus minimizing the number of network weights. Moreover, a loss function can be applied at every spatial scale of the decoder.

Finally, *motion compensation* is a technique to estimate the parameters of the motion that best fits a group of events. It has a continuous-time warping model that allows to exploit the fine temporal resolution of events (Section 3.1), and hence departs from conventional image-based algorithms. Motion compensation can be used to estimate ego-motion [122], [123], optical flow [114], [123], [126], [150], depth [19], [123], [124], motion segmentation [128], [150], [151] or feature motion for VIO [125], [127]. The technique in [99] also has a continuous-time motion model, albeit not used for motion compensation but rather to fuse event data with IMU data. To find the parameters of the continuous-time motion models [99], [124], standard optimization methods, e.g., conjugate gradient or Gauss-Newton, may be applied.

The *number of events per group* (i.e., size of the spatio-temporal neighborhood) is an important hyper-parameter of many methods. While this number highly depends on the processing algorithm and the available resources, there are two main strategies [11], [113], [122]: constant number of events or constant observation time (i.e., constant frame rate). Utilizing a constant number of events fits more naturally with the camera's output and scene dynamics, whereas a constant frame rate selects a varying number of events: sometimes too few or too many (depending on the scene) for the subsequent module in the processing pipeline.

3.3 Biologically Inspired Visual Processing

Biological principles and computational primitives drive the design of event camera pixels and some of the event-processing algorithms (and hardware), such as Spiking Neural Networks (**SNNs**).

Visual pathways: The DVS [2] was inspired by the function of biological visual pathways, which have "transient" pathways dedicated to processing dynamic visual information in the so-called "where" pathway. Animals ranging from insects to humans all have these transient pathways. In humans, the transient pathway occupies about 30 % of the visual system. It starts with transient ganglion cells, which are mostly found in retina outside the fovea. It continues with magno layers of the thalamus and particular sublayers of area V1. It then continues to area MT and MST, which are part of the dorsal pathway where many motion selective cells are found [44]. The DVS corresponds to the part of the transient pathway(s) up to retinal ganglion cells. Similarly, the grayscale events of the ATIS correspond to the "sustained" or "what" pathway through the parvo layers of the brain [36], [42].

Event processing by SNNs: Artificial neurons, such as Leaky-Integrate and Fire or Adaptive Exponential, are computational primitives inspired in neurons found in the mammalian's visual cortex. They are the basic building blocks of artificial SNNs. A neuron receives input spikes ("events") from a small region of the visual space (a receptive field), which modify its internal state (membrane potential) and produce an output spike (action potential) when the state surpasses a threshold. Neurons are connected in a hierarchical way, forming an SNN. Spikes may be produced by pixels of the event camera or by neurons of the SNN. Information travels along the hierarchy, from the event camera pixels to the first layers of the SNN and then through to higher (deeper) layers. Most first layer receptive fields are based on Difference of Gaussians (selective to center-surround contrast), Gabor filters (selective to oriented edges), and their combinations. The receptive fields become increasingly more complex as information travels deeper into the network. In ANNs, the computation performed by inner layers is approximated as a convolution. One common approach in artificial SNNs is to assume that a neuron will not generate any output spikes if it has not received any input spikes from the preceding SNN layer. This assumption allows computation to be skipped for such neurons. The result of this visual processing is almost simultaneous with the stimulus presentation [152], which is very different from traditional CNNs, where convolution is computed simultaneously at all locations at fixed time intervals.

Tasks: Bio-inspired models have been adopted for several low-level visual tasks. For example, event-based *optical*

flow can be estimated by using spatio-temporally oriented filters [79], [130], [153] that mimic the working principle of receptive fields in the primary visual cortex [154], [155]. The same type of oriented filters have been used to implement a spike-based model of *selective attention* [156] based on the biological proposal from [157]. Bio-inspired models from binocular vision, such as recurrent lateral connectivity and excitatory-inhibitory neural connections [158], have been used to solve the event-based *stereo* correspondence problem [40], [159], [160], [161], [162] or to control binocular vergence on humanoid robots [163]. The visual cortex has also inspired the hierarchical feature extraction model proposed in [164], which has been implemented in SNNs and used for *object recognition*. The performance of such networks improves the better they extract information from the precise timing of the spikes [165]. Early networks were hand-crafted (e.g., Gabor filters) [52], but recent efforts let the network build receptive fields through brain-inspired learning, such as Spike-Timing Dependent Plasticity (**STDP**), yielding better recognition rates [132]. This research is complemented by approaches where more computationally inspired types of supervised learning, such as back-propagation, are used in deep networks to efficiently implement spiking deep convolutional networks [139], [166], [167], [168], [169]. The advantages of the above methods over their traditional vision counterparts are lower latency and higher efficiency.

To build small, efficient and reactive computational systems, *insect vision* is also a source of inspiration for event-based processing. To this end, systems for fast and efficient obstacle avoidance and target acquisition in small robots have been developed [170], [171], [172] based on models of neurons driven by DVS output that respond to looming objects and trigger escape reflexes.

4 Algorithms / Applications

In this section, we review several works on event-based vision, presented according to the task addressed. We start with low-level vision on the image plane, such as feature detection, tracking, and optical flow estimation. Then, we discuss tasks that pertain to the 3D structure of the scene, such as depth estimation, visual odometry (**VO**) and historically related subjects, e.g., intensity image reconstruction. Finally, we consider motion segmentation, recognition and coupling perception with control.

4.1 Feature Detection and Tracking

Feature detection and tracking on the image plane are fundamental building blocks of many vision tasks such as visual odometry, object segmentation and scene understanding. Event cameras make it possible to track asynchronously, adapted to the dynamics of the scene and with low latency, high dynamic range and low power (Section 2.2). Thus, they allow to track in the "blind" time between the frames of a standard camera. To do so, the methods developed need to deal with the unique space-time and photometric characteristics of the visual signal: events report only brightness changes, asynchronously (Section 2.3).

Challenges: Since events represent brightness changes, which depend on motion direction, one of the main challenges of feature detection and tracking with event cameras is overcoming the variation of scene appearance caused by such motion dependency (Fig. 5). Tracking requires the establishment of correspondences between events (or features built from the events) at different times (i.e., data association), which is difficult due to the varying appearance. The second main challenge consists of dealing with sensor noise and possible event clutter caused by the camera motion.

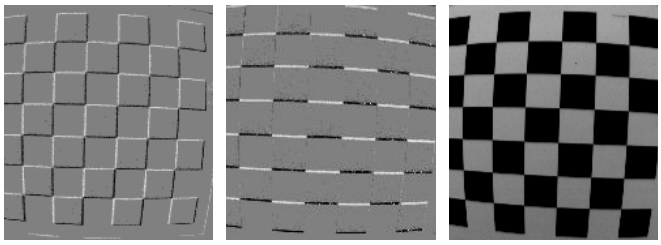


Figure 5. The challenge of data association. Panels (a) and (b) show events from a scene (c) under two different motion directions: (a) diagonal and (b) up-down. Intensity increment images (a) and (b) are obtained by accumulating event polarities over a short time interval: pixels that do not change intensity are represented in gray, whereas pixels that increased or decreased intensity are represented in bright and dark, respectively. Clearly, it is not easy to establish event correspondences between (a) and (b) due to the changing appearance of the edge patterns in (c) with respect to the motion. Image adapted from [63].

Literature Review: Early event-based feature methods were very simple and focused on demonstrating the low-latency and low-processing requirements of event-driven vision systems. Hence they assumed a stationary camera scenario and tracked moving objects as clustered *blob-like sources of events* [6], [12], [14], [117], [173], circles [174] or lines [53]. Only pixels that generated events needed to be processed. Simple Gaussian correlation filters sufficed to detect blobs of events, which could be modeled by Gaussian Mixtures [175]. For tracking, each incoming event was associated to the nearest existing blob/feature and used to asynchronously update its parameters (location, size, etc.). Circles [174] and lines [53] were treated as blobs in the Hough transform space. These methods were used in traffic monitoring and surveillance [14], [117], [175], high-speed robotic tracking [6], [12] and particle tracking in fluids [173] or microrobotics [174]. However, they worked only for a limited class of object shapes.

Tracking of more complex, high-contrast *user-defined shapes* has been demonstrated using event-by-event adaptations of the Iterative Closest Point (ICP) algorithm [118], gradient descent [119], Mean-shift and Monte-Carlo methods [176], or particle filtering [177]. The iterative methods in [118], [119] used a nearest-neighbor strategy to associate incoming events to the target shape and update its transformation parameters, showing very high-speed tracking (200 kHz equivalent frame rate). Other works [176] handled geometric transformations of the target shape (*aka* "kernel") by matching events against a pool of rotated and scaled versions of it. The predefined kernels tracked the object without overlapping themselves due to a built-in repulsion mechanism. Complex objects, such as faces or human bodies, have been tracked with part-based shape models [178], where objects are represented as a set of basic elements linked by springs [179]. The part trackers simply follow incoming blobs of events generated by ellipse-like shapes, and the

elastic energy of this virtual mechanical system provides a quality criterion for tracking. In most tracking methods events are treated as individual points (without polarity) and update the system's state asynchronously, with minimal latency. The performance of the methods strongly depends on the tuning of several model parameters, which is done experimentally according to the object to track [176], [178].

The previous methods require a priori knowledge or user input to determine the objects to track. This restriction is valid for scenarios like tracking cars on a highway or balls approaching a goal, where knowing the objects greatly simplifies the computations. But when the space of objects becomes larger, methods to determine more *realistic features* become necessary. The features proposed in [120], [126] consist of local edge patterns that are represented as point sets. Incoming events are registered to them by means of some form of ICP. Other methods [27], [125] proposed to re-utilize well-known feature detectors [180] and trackers [181] on patches of motion-compensated event images (Section 3.1), providing good results. All these methods allowed to track features for cameras moving in natural scenes, hence enabling ego-motion estimation in realistic scenarios [121], [125], [127]. Features built from motion-compensated events (in image form [125] or point-set form [126]) provide a useful representation of edge patterns. However, they depend on motion direction, and, therefore, trackers suffer from drift as event appearance changes over time [63]. To track with no drift, motion-invariant features are needed.

Combining Events and Frames: Data association (Fig. 5) simplifies if the absolute intensity of the pattern to be tracked (Fig. 5c, i.e., a motion-invariant representation or "map" of the feature) is available. This is the approach followed by works that leverage the strengths of a combined frame- and event-based sensor (à la DAVIS [4]). The algorithms in [63], [120], [121] automatically detect arbitrary edge patterns (features) on the frames and track them asynchronously with events. The feature location is given by the Harris corner detector [180] and the feature descriptor is given by the edge pattern around the corner: [120], [121] convert Canny edges to point sets used as templates for ICP tracking, thus they assume events are mostly triggered at strong edges. In contrast, the edge pattern in [63] is given by the frame intensities, and tracking consists of finding the motion parameters that minimize the photometric error between the events and their frame prediction using a generative model (4). A comparison of five feature trackers is provided in [63], showing that the generative model is most accurate, with sub-pixel performance, albeit it is computationally expensive. Finally, [63] also shows the interesting fact that an event-based sensor suffices: frames can be replaced by images reconstructed from events (Section 4.6) and still achieve similar detection and tracking results.

Corner Detection and Tracking: Since event cameras naturally respond to edges in the scene, they shorten the detection of lower-level primitives such as keypoints or "corners". Such primitives identify pixels of interest around which local features can be extracted without suffering from the aperture problem, and therefore provide reliable tracking information. The method in [182] computes corners as the intersection of two moving edges, which are obtained by fitting planes in the space-time stream of events. To deal with event noise, least-squares is supplemented by a sampling technique similar to RANSAC. This method of fitting planes locally to time surfaces has also been profitable to estimate optical flow [21] and "event lifetime" [144], which are obtained from the coefficients of the planes. Recently, extensions of popular frame-based keypoint detectors, such as Harris [180] and FAST [183], have been developed for event cameras [105], [108], [109], by operating on time surfaces (TSs) as if they were natural intensity images. In [108] the TS is binarized before applying the derivative filters of Harris' detector. To speed up detection, [109] replaces the derivative filters with pixelwise comparisons on two concentric circles of the TS around the current event. Moving corners produce local TSs with two clearly separated regions: recent vs. old events. Hence, corners are obtained by searching for arcs of contiguous pixels with higher TS values than the rest. The method in [105] improves the detector in [109] and proposes a strategy to track the corners. Assuming corners follow continuous trajectories on the image plane and the detected event corners are accurate, these are threaded by proximity along trajectories, following a tree-based hypothesis graph. The above TS-based hand-crafted corner detectors suffer from variations of the TS due to changes in motion direction. To overcome them, [106] proposes a data-driven method to learn the TS appearance of intensity-image corners. To this end, a grayscale input (from DAVIS or ATIS camera) provides the supervisory signal to label the corners. As a trade-off between accuracy and speed, a random forest classifier is used. Event corners find multiple applications, such as visual odometry or ego-motion segmentation [184]; yet there are only a few demonstrations.

Table 2
Classification of several optical flow methods according to whether they provide normal (N) or full flow (F), sparse (S) or dense (D) estimates, whether they are model-based or model-free (Artificial Neural Network - ANN), and neuro-biologically inspired or not.

Reference	N/F?	S/D?	Model?	Bio?
Delbruck [79], [186]	Normal	Sparse	Model	Yes
Benosman et al. [186], [187]	Full	Sparse	Model	No
Orchard et al. [153]	Full	Sparse	ANN	Yes
Benosman et al. [21], [186]	Normal	Sparse	Model	No
Barranco et al. [188]	Normal	Sparse	Model	No
Brosch et al. [130]	Normal	Sparse	Model	Yes
Bardow et al. [112]	Full	Dense	Model	No
Liu et al. [101]	Full	Sparse	Model	No
Gallego [123], Stoffregen [150]	Full	Sparse	Model	No
Haessig et al. [189]	Normal	Sparse	ANN	Yes
Zhu et al. [22], [114]	Full	Dense	ANN	No
Ye et al. [149]	Full	Dense	ANN	No
Paredes-Vallés [97]	Full	Sparse	ANN	Yes

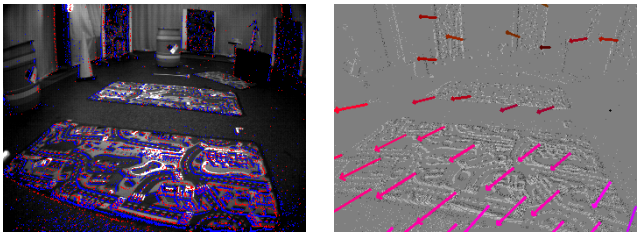
Opportunities: In spite of the abundance of detection and tracking methods, they are rarely evaluated on common datasets for performance comparison. Establishing benchmark datasets [185] and evaluation procedures will foster progress in this and other topics. Also, in most algorithms, parameters are defined experimentally according to the tracking target. It would be desirable to have adaptive parameter tuning to increase the range of operation of the trackers. Learning-based feature detection and tracking methods also offer considerable room for research.

4.2 Optical Flow Estimation

Optical flow estimation is the problem of computing the velocity of objects on the image plane without knowledge about the scene geometry or motion. The problem is ill-posed and thus requires regularization to become tractable. Event-based optical flow estimation is challenging because of the unfamiliar way in which events encode visual information (Section 2). In conventional cameras optical flow is obtained by analyzing two consecutive images. These provide temporal and spatial derivatives that are substituted in the brightness constancy law (page 4), which together with smoothness assumptions provide enough equations to solve for the flow at each image pixel. In contrast, events provide neither absolute brightness nor spatially continuous data. Each event does not carry enough information to determine flow, and so events need to be aggregated to produce an estimate, which leads to the unusual question of where in the x-y-t-space of the image plane spanned by the events is flow computed. Ideally one would like to know the flow field over the whole space, which deems computationally expensive. In practice, optical flow is computed only at specific points: at the event locations, or at images with artificially-chosen times. Nevertheless, computing flow from events is attractive because they represent edges, which are the parts of the scene where flow estimation is less ambiguous, and because their fine timing information allows measuring high speed flow [11]. Finally, another challenge is to design a flow estimation algorithm that is biologically plausible, i.e., compatible with what is known from neuroscience about early processing in the primate visual cortex, and that can be implemented efficiently in neuromorphic processors.

Literature Review: Table 2 lists some event-based optical flow methods, categorized according to different criteria. Early works [187] tried to adapt classical approaches in computer vision to event-based data (Fig. 6). These are based on the brightness constancy assumption [181], and discussion focused on whether events carried enough information to estimate flow with such approaches [130]. Events allow to estimate the temporal derivative of brightness (3), and so additional assumptions were needed to approximate the spatial derivative ∇L in order to apply such classical methods [181]. However, due to the potentially very small number of events generated at each pixel as an edge crosses over it, it is difficult to estimate derivatives $(\nabla L, \partial L/\partial t)$ reliably [130], which leads gradient-based methods like [187] to inconclusive flow estimates. Approaches that consider the local distribution of events in the x-y-t-space, as in [21], are more robust and therefore preferred.

The method in [21] reasons about the local distribution of events geometrically, in terms of time surfaces and planar approximations. As an edge moves it produces events that resemble points on a surface in space-time (the time surface, Section 3). The surface slopes in the x-t and y-t cross sections encode the edge motion, thus optical flow is estimated by fitting planes to the surface and reading the slopes from the plane coefficients. In spite of providing only normal flow (i.e., the component of the optical flow perpendicular to the edge), the method works even in the case of only a few generated events. Of course, the goodness of fit depends on the size of the spatio-temporal neighborhood (this remark generalizes to other methods). If the neighborhood is too small then the plane fit may become arbitrary. If the neighborhood is too large then the event stream may not be well approximated by a local plane.



(a) *Indoor flying* scene from [190] (b) Sparse optical flow

Figure 6. Optical flow estimation with an event camera. (a) Events (with polarity in red/blue) overlaid on a grasycale frame; (b) Sparse optical flow computed using method [181] on brightness increment images (as in Fig. 5). Flow is colored according to magnitude and direction.

A hierarchical architecture for optical flow estimation building on experimental findings of the primate visual system is proposed in [130]. It applies a set of spatio-temporal filters on the event stream to yield selectivity to different motion speeds and directions (à la Gabor filters) while maintaining the sparse representation of events. Such filters are formally equivalent to spatio-temporal correlation detectors. Other biologically-inspired methods [97], [153] can also be interpreted as filter banks sampling the event stream along different spatio-temporal orientations; [153] and [130] define hand-crafted filters, whereas [97] learns them from event data using a novel STDP rule. The SNN in [153] detects motion patterns by delaying events through synaptic connections and employing neurons as coincidence detectors. Its neurons are sensitive to 8 speeds and 8 directions (i.e., 64 velocities) over receptive fields of 5×5 pixels. These methods are implementable in neuromorphic hardware, offering low-power, efficient computations.

The method in [191] targets the issue of flow estimation at textured regions. It converts events into event frames and applies a Gabor filter bank. Then, assuming constancy of the (spatio-temporal) edges of the event frame, optical flow is given by the phase gradient [192] of the filter bank output. Due to the aperture problem, only the velocity in the direction of the spatial phase gradient can be computed.

Methods like [23], [112] estimate optical flow jointly with other quantities, notably image intensity, so that the quantities involved bring in well-known equations and boost each other towards convergence. Knowing image intensity, or equivalently $(\nabla L, \partial L/\partial t)$, is desirable since it can be used on the brightness constancy law to provide constraints on the optical flow. In this respect, [112] combines multiple equations ((2), brightness constancy, smoothness priors, etc.) as penalty terms into an objective function that is optimized via calculus of variations. The method finds the optical flow and image intensity on the image plane that minimizes the objective function, i.e., that best explains the distribution of events in the x-y-t-space (using a voxel grid). Thus, it outputs a dense flow (i.e., flow at every pixel). Flow vectors at pixels where no events were produced (i.e., regions of homogeneous brightness) are due to the smoothness priors, thus they are less reliable than those computed at pixels

where events were triggered (i.e., at edges).

The method in [101] estimates optical flow by computing event frames (Section 3) at an adaptive rate and applying video coding techniques (block matching). It can be interpreted as finding the optical flow vector that best matches the distributions of events within two cuboids (collapsed into event frames). Thus, the optical flow problem is posed as that of finding event correspondences, i.e., events triggered by the same scene point (at different times). The method defines two sets of events ("blocks") and a similarity metric to compare them. It is assumed that the appearance of event frames do not change significantly for short times and hence simple metrics, such as sum of absolute distances, suffice to compare them. The method can be implemented in FPGA, trading off efficiency for accuracy.

The framework in [123], [124], [150] computes optical flow by maximizing the sharpness of image patches obtained by warping cuboids of events, producing motion-compensated images (Section 3). It can be interpreted as applying an adaptive filter to the events, where the filter coefficients define the spatio-temporal direction that maximizes the filter's response. Motion compensation was also used to compute flow in [126], albeit using point sets.

Recently, deep learning methods have emerged [22], [114], [149]. These are based on the availability of large amounts of event data paired with an ANN. In [22], an encoder-decoder CNN is trained using a self-supervised scheme to estimate dense optical flow. The loss function measures the error between DAVIS grayscale images aligned using the flow produced by the network. The trained network is able to accurately predict optical flow from events only, passed as time surfaces and event frames. The work [149] presents the first monocular ANN architecture to estimate dense optical flow, depth and ego-motion from events only. The input to the ANN consists of events over multiple time slices, given as event frames and time surfaces with per-pixel average timestamps. This reduces event noise and preserves the structure of the event stream better than [22]. The network is trained unsupervised, measuring the photometric error between the events in neighboring time slices aligned using the estimated flow. Later, [22] was extended to unsupervised learning of flow and ego-motion in [114] using a motion-compensation loss function in terms of time surfaces.

Evaluation: Optical flow estimation is computationally expensive. Some methods [22], [112], [114], [149] require a GPU, while other approaches are more lightweight [101], albeit not as accurate. Few algorithms [21], [101], [130], [153] have been pushed to hardware logic circuits that offload CPU and minimize latency. The review [186] compared some early event-based optical flow methods [21], [79], [187], but only on flow fields generated by a rotating camera, i.e., lacking motion parallax and occlusion. For newer methods, there are multiple trade offs (accuracy vs. efficiency vs. latency) that have not been properly quantified yet.

Opportunities: Comprehensive datasets with accurate ground truth optical flow in multiple scenarios (varying texture, speed, parallax, occlusions, illumination, etc.) and a common evaluation methodology would be essential to assess progress and reproducibility in this paramount low-level vision task. Providing ground truth *event-based* optical flow in real scenes is challenging, especially for moving objects not conforming to the motion field induced by the camera's ego-motion. A thorough quantitative comparison of existing event-based optical flow methods would help identify key ideas to develop improved methods.

4.3 3D reconstruction. Monocular and Stereo

Depth estimation with event cameras is a broad field. It can be divided according to the considered scenario and camera setup or motion, which determine the problem assumptions.

Instantaneous Stereo: Most works on depth estimation with event cameras target the problem of "instantaneous" stereo, i.e., 3D reconstruction using events on a very short time (ideally on a per-event basis) from two or more synchronized cameras that are rigidly attached. Being synchronized, the events from different image planes share a common clock. These works follow the classical two-step stereo solution: first solve the event correspondence problem across image planes (i.e., epipolar matching) and then triangulate the location of the 3D point [193]. The main challenge is finding correspondences between events; it is the computationally intensive step. Events are matched (*i*) using traditional stereo metrics (e.g., normalized cross-correlation) on event frames [141], [194] or time surfaces [145] (Section 3), and/or (*ii*) by exploiting simultaneity and temporal correlations of the events across sensors [145], [195], [196]. These approaches are *local*, matching events by comparing their neighborhoods since events cannot be matched based on individual timestamps [166], [197]. Additional constraints, such as the epipolar constraint [198], ordering, uniqueness, edge orientation and polarity may be used to reduce matching ambiguities and false correspondences, thus improving depth estimation [18], [166], [199]. Event matching can also be done by comparing local context descriptors [200], [201] of the spatial distribution of events on both stereo image planes.

Global approaches produce better depth estimates (i.e., less sensitive to ambiguities) than local approaches by considering additional regularity constraints. In this category, we find extensions of Marr and Poggio's cooperative stereo algorithm [158] for the case of event cameras [40], [160], [161], [162], [202]. These approaches consist of a network of disparity sensitive neurons that receive events from both cameras and perform various operations (amplification, inhibition) that implement matching constraints (uniqueness, continuity) to extract disparities. They use not only the temporal similarity to match events but also their spatio-temporal neighborhoods, with iterative nonlinear operations that result in an overall globally-optimal solution. A discussion of cooperative stereo is provided in [42]. Also in this category are [203], [204], [205], which use Belief Propagation on a Markov Random Field or semiglobal matching [206] to improve stereo matching. These methods are primarily based on optimization, trying to define a well-behaved energy function whose minimizer is the correct correspondence map. The energy function incorporates regularity constraints, which enforce coupling of correspondences at neighboring points and therefore make the solution map less sensitive to ambiguities than local methods, at the expense of computational effort. A table

(a) (b)

Figure 7. Example of monocular depth estimation with a hand-held event camera. (a) Scene, (b) semi-dense depth map, pseudo-colored from red (close) to blue (far). Image courtesy of [19].

comparing different stereo methods is provided in [207]; however, it should be interpreted with caution since the methods were not benchmarked on the same dataset.

Recently, brute-force space-sweeping using dedicated hardware (a GPU) has been proposed [208]. The method is based on ideas similar to [19], [123]: the correct depth manifests as "in focus" voxels of displaced events in the Disparity Space Image [19], [209]. In contrast, other approaches pair event cameras with neuromorphic processors (Section 5.1) to produce fully event-based low-power ($100\,\mathrm{mW}$), high-speed stereo systems [161], [207]. There is an efficiency vs. accuracy trade-off that has not been quantified yet.

Most of the methods above are demonstrated in scenes with static cameras and few moving objects, so that correspondences are easy to find due to uncluttered event data. Event matching happens with low latency, at high rate ($\sim 1\,\mathrm{kHz}$) and consuming little power, which shows that event cameras are promising for high-speed 3D reconstructions of moving objects or in uncluttered scenes.

Multi-Perspective Panoramas: Some works [210], [211] also target the problem of instantaneous stereo (depth maps produced using events over very short time intervals), but using two non-simultaneous event cameras. These methods exploit a constrained hardware setup (two rotating event cameras with known motion) to either (*i*) recover intensity images on which conventional stereo is applied [210] or (*ii*) match events using temporal metrics [211].

Monocular Depth Estimation: Depth estimation with a single event camera has been shown in [19], [25], [123]. It is a significantly different problem from previous ones because temporal correlation between events across multiple image planes cannot be exploited. These methods recover a semi-dense 3D reconstruction of the scene (i.e., 3D edge map) by integrating information from the events of a moving camera over time, and therefore require knowledge of camera motion. Hence they do not pursue instantaneous depth estimation, but rather depth estimation for SLAM [212].

The method in [25] is part of a pipeline that uses three filters operating in parallel to jointly estimate the motion of the event camera, a 3D map of the scene, and the intensity image. Their depth estimation approach requires using an additional quantity—the intensity image—to solve for data association. In contrast, [19] (Fig. 7) proposes a space-sweep method that leverages the sparsity of the event stream to perform 3D reconstruction without having to establish event matches or recover the intensity images. It back-projects events into space, creating a ray density volume [213], and then finds scene structure as local maxima of ray density. It is computationally efficient and used for VO in [26].

Stereo Depth for SLAM: Recently, inspired by work in small-baseline multi-view stereo [214], a stereo depth estimation method for SLAM has been presented [110]. It obtains a semi-dense 3D reconstruction of the scene by optimizing the local spatio-temporal consistency of events across image planes using time surfaces. It does not follow the classical paradigm of event matching plus triangulation [145], but rather a forward-projection approach that enables depth estimation without establishing event correspondences explicitly. The method opens the door for bringing the advantages of event cameras to event-based stereo SLAM applications such as self-driving cars.

Depth Estimation using Structured Light: All the above 3D reconstruction methods are passive, i.e., do not interfere with the scene. In contrast, there are some works on event-based active 3D reconstruction, based on emitting light onto the scene and measuring reflection with event cameras [20], [215], [216]. For example, [215] combines a DVS with a pulsed line laser to allow fast terrain reconstruction, in the style of a 3D line scanner. Motion Contrast 3D scanning [20] is a structured light technique that simultaneously achieves high resolution, high speed and robust performance in challenging 3D scanning environments (e.g., strong illumination, or highly reflective and moving surfaces). Active systems with pulsed lasers exploit the high temporal resolution and redundancy suppression of event cameras, but they are application specific and may not be safe (depending on the power of the laser needed to scan far away objects).

Opportunities: Although there are many methods for event-based depth estimation, it is difficult to compare their performance since they are not evaluated on the same dataset. In this sense, it would be desirable to (*i*) provide a comprehensive dataset and testbed for event-based depth evaluation and (*ii*) benchmark many existing methods on the dataset, to be able to compare their performance.

4.4 Pose Estimation and SLAM

Addressing the Simultaneous Localization and Mapping (SLAM) problem with event cameras has been difficult because most methods and concepts developed for conventional cameras (feature detection, matching, iterative image alignment, etc.) are not applicable or were not available; events are fundamentally different from images. The challenge is therefore to design new SLAM techniques that are able to unlock the camera's advantages (Sections 2.3 and 2.2), showing their usefulness to tackle difficult scenarios for current frame-based cameras. Historically, the design goal of such techniques has focused on preserving the low-latency nature of the data, i.e., being able to produce a state estimate for every incoming event (Section 3). However, each event does not contain enough information to estimate the state from scratch (e.g., the six degrees of freedom (DOF) pose of a calibrated camera), and so the goal becomes that each event be able to asynchronously update the state of the system. Probabilistic (Bayesian) filters [217] are popular in event-based SLAM [7], [24], [74], [218] because they naturally fit with this description. Their main adaptation for event cameras consists of designing sensible likelihood functions based on the event generation process (Section 2.4).

Table 3
Event-based methods for pose tracking and/or mapping with an event camera. The type of motion is noted with labels "2D" (3-DOF motions, e.g., planar or rotational) and "3D" (free 6-DOF motion in 3D space). Columns indicate whether the method performs tracking ("Track") and depth estimation ("Depth") using only events ("Event"), the type of scene considered ("Scene"), and any additional requirements. Only [25], [26] address the most general scenario using only events.

Reference	Dim	Track	Depth	Scene	Event	Additional requirements
Cook [23]	2D	✓	✗	natural	✓	rotational motion only
Weikersdorfer [218]	2D	✓	✗	B&W	✓	scene parallel to motion
Kim [24]	2D	✓	✗	natural	✓	rotational motion only
Gallego [122]	2D	✓	✗	natural	✓	rotational motion only
Reinbacher [98]	2D	✓	✗	natural	✓	rotational motion only
Censi [74]	3D	✓	✗	B&W	✗	attached depth sensor
Weikersdorfer [219]	3D	✓	✓	natural	✗	attached RGB-D sensor
Mueggler [220]	3D	✓	✗	B&W	✓	3D map of lines
Gallego [7]	3D	✓	✗	natural	✗	3D map of the scene
Rebecq [19]	3D	✗	✓	natural	✓	pose information
Kueng [121]	3D	✓	✓	natural	✗	intensity images
Kim [25]	3D	✓	✓	natural	✓	image reconstruction
Rebecq [26]	3D	✓	✓	natural	✓	–

Since events are caused by the apparent motion of intensity edges, the majority of maps emerging from SLAM systems naturally consist only of scene edges, i.e., semi-dense maps (Fig. 8 and [19]). However, note that an event camera does not directly measure intensity gradients but only temporal changes (2), and so the presence, orientation and strength of edges (on the image plane and in 3D) must be estimated together with the camera's motion. The strength of the intensity gradient at a scene point is correlated with the firing rate of events corresponding to that point, and it enables reliable tracking [98]. Edge information for tracking may also be obtained from gradients of brightness maps [7], [24], [25] used in generative models (Section 2.4).

The event-based SLAM problem in its most general setting (6-DOF motion and natural 3D scenes) is a challenging problem that has been addressed step-by-step in scenarios with increasing complexity. Three complexity axes can be identified: dimensionality of the problem, type of motion and type of scene. The literature is dominated by methods that address the localization subproblem first (i.e., motion estimation) because it has fewer degrees of freedom to estimate. Regarding the type of motion, solutions for constrained motions, such as rotational or planar (both being 3-DOF), have been investigated before addressing the most complex case of a freely moving camera (6-DOF). Solutions for artificial scenes in terms of photometry (high contrast) and/or structure (line-based or 2D maps) have been proposed before focusing on the most difficult case: natural scenes (3D and with arbitrary photometric variations). Some proposed solutions require additional sensing (e.g., RGB-D) to reduce the complexity of the problem. This, however, introduces some of the bottlenecks present in frame-based systems (e.g., latency and motion blur). Table 3 classifies the related work using these complexity axes.

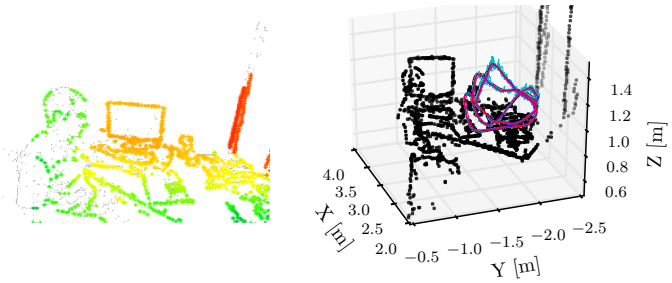
Camera Tracking Methods: Pose tracking with an event camera was first presented in [96]. It proposed a particle filter to track the motion of a ground robot that was viewing a flat scene, which was parallel to the plane of motion and consisted of artificial B&W line patterns. The main innovation was the design of the likelihood function to quantify the event generation probability given the robot's pose and scene map. The function was based on the reprojection error between the event's location and the nearest edge in the map. In [74], a standard grayscale camera was attached to a DVS to estimate, using a Bayesian filter, the small displacement between the current event and the previous frame of the standard camera. The system was developed for planar motion and B&W scenes. The likelihood function was proportional to the strength of the gradient of the map at the event's location (Section 2.4). In [221], pose tracking under a non-holonomic and planar motion was proposed, supporting loop closure and topologically-correct trajectories. It converted events into frames and used traditional method SeqSLAM.

Estimation of the 3D orientation of an event camera has been addressed in [24], [98], [122]. The rotational motion of the camera was tracked by [24] using a particle filter, whose likelihood function was approximately Gaussian centered at a contrast sensitivity $C \approx 0.2$ (Section 2.4). The work in [98] estimated the camera's rotation by minimization of a photometric error at the event locations; it used a map of event probabilities that represented the strength of the scene edges [218]. Finally, the motion-compensation optimization framework (Section 3) was introduced in [122] to estimate the angular velocity of the camera rather than its absolute rotation. The above systems are restricted to rotational motions, and, thus, do not account for translation or depth. Nevertheless they inspire ideas to solve more complex problems, such as [25], [123], [125].

Camera tracking in 6-DOF is the most challenging one. An event-based algorithm to track the pose of a DVS during very high-speed motion was presented in [220]. This was a hand-crafted method developed for artificial, B&W line-based maps, rather than a probabilistic filter. It assumed that events were generated only by one of the map lines (the one closest to the event), which were tracked and intersected to provide points for PnP methods (camera resectioning). A continuous-time formulation and extension of such method was given in [222]. It computed the camera trajectory, rather than individual poses, by non-linear optimization of the event-to-line reprojection error. By contrast, [7], [64] showed 6-DOF high-speed tracking capabilities in natural scenes. They used a generative model: [7] proposed a probabilistic filter with a robust likelihood function comprising mixture densities (Section 2.4), whereas [64] pursued non-linear optimization of the photometric error between brightness increment images (Section 3.1) and their prediction given the scene map. The latter gave slightly better results.

Tracking and Mapping: Let us focus on methods that address the tracking-and-mapping problem. Cook et al. [23] proposed a generic message-passing algorithm within an interacting network to jointly estimate ego-motion, image intensity and optical flow from events. However, the system was restricted to rotational motion. Joint estimation is appealing because it allows to employ as many equations as possible relating the variables (e.g., (4) and rotational prior) in the hope of finding a better solution to the problem.

An event-based 2D SLAM system was presented in [218] by extension of [96], and thus it was restricted to planar motion and high-contrast scenes. The method used a particle filter for tracking, with the event likelihood function inversely related to the the reprojection error of the event with respect to the map. The map of scene edges was



(a) Events & projected map. (b) Camera trajectory & 3D map.

Figure 8. Event-based SLAM. (a) Reconstructed scene from [223], with the reprojected semi-dense map colored according to depth and overlaid on the events (in gray), showing the good alignment between the map and the events. (b) Estimated camera trajectory (several methods) and semi-dense 3D map (i.e., point cloud). Image courtesy of [99].

concurrently built; it consisted of an occupancy map [217], with each pixel representing the probability that the pixel triggered events. The method was extended to 3D in [219], but it relied on an external RGB-D sensor attached to the event camera for depth estimation. The depth sensor introduced bottlenecks, which deprived the system of the low latency and high-speed advantages of event cameras.

The filter-based approach in [24] showed how to simultaneously track the 3D orientation of a rotating event camera and create high-resolution panoramas of natural scenes. It operated probabilistic filters in parallel for both subtasks. A panoramic gradient was built using per-pixel Kalman filters, each one estimating the orientation and strength of the scene edge at its location. This gradient map was then upgraded to an absolute intensity one with super-resolution and HDR properties by Poisson integration. SLAM during rotational motion was also presented in [98], where camera tracking was performed by minimization of a photometric error at the event locations given a probabilistic edge map. The map was simultaneously built, and each map point represented the probability of events being generated at that location [218]. Hence it was a panoramic occupancy map measuring the strength of the scene edges.

Recently, solutions to the full problem of event-based 3D SLAM for 6-DOF motions and natural scenes, not relying on additional sensing, have been proposed [25], [26] (Table 3). The approach in [25] extends [24] and consists of three interleaved probabilistic filters to perform pose tracking as well as depth and intensity estimation. However, it suffers from limited robustness (especially during initialization) due to the assumption of uncorrelated depth, intensity gradient, and camera motion. Furthermore, it is computationally intensive, requiring a GPU for real-time operation. In contrast, the semi-dense approach in [26] shows that intensity reconstruction is not needed for depth estimation or pose tracking. The approach has a geometric foundation: it performs space sweeping for 3D reconstruction [19] and edge-map alignment (non-linear optimization with few events per frame) for pose tracking. The resulting SLAM system runs in real-time on a CPU.

Trading off latency for efficiency, probabilistic filters [24], [25], [218] can operate on small groups of events. Other approaches are natively designed for groups, based for example on non-linear optimization [26], [122], [123], and run in real time on the CPU. Processing multiple events simultaneously is also beneficial to reduce noise.

Opportunities: The above-mentioned SLAM methods lack loop-closure capabilities to reduce drift. Currently, the scales of the scenes on which event-based SLAM has been demonstrated are considerably smaller than those of frame-based SLAM. However, trying to match both scales may not be a sensible goal since event cameras may not be used to tackle the same problems as standard cameras; both sensors are complementary, as argued in [7], [27], [63], [74]. Stereo event-based SLAM is another unexplored topic, as well as designing more accurate, efficient and robust methods than the existing monocular ones. Robustness of SLAM systems can be improved by sensor fusion with IMUs [27], [212].

4.5 Visual-Inertial Odometry (VIO)

The robustness of event-based visual odometry and SLAM systems can be improved by combining an event camera with an inertial measurement unit (IMU) rigidly attached. VIO is qualitatively different from VO: a VO system may lose track (i.e., fail to produce an output pose). In contrast, a VIO system does not "fail" (there is always an output pose), it drifts. For this and other reasons, some event cameras have an integrated IMU (see Table 1).

A key issue in VIO is how to temporally fuse data from the synchronous, high-rate IMU (e.g., 1 kHz) and the asynchronous event camera. There are three options in the literature: (i) use an asynchronous probabilistic filter [127], (ii) use pre-integration theory [224] to convert IMU data into lower-rate pieces of information at desired times where events are collected [27], [125], [127] or (iii) consider a continuous-time framework so that both sensor measurements are referred to a common temporal axis that is also used to model the solution (i.e., the camera poses) [99].

Feature-based VIO: The majority of existing event-based VIO systems are "feature-based", consisting of two stages: first, features tracks are extracted from the events (Section 4.1), and then these point trajectories on the image plane are fused with IMU data using modern VIO algorithms, such as [224], [225], [226]. That is, a front-end converts the photometric information conveyed by the events into geometric information that is then processed by highly optimized geometric VIO pipelines. For example, [127] tracked features using [126], and combined them with IMU data by means of a Kalman filter [225]. Recently, [125] proposed to synthesize motion-compensated event images from spatio-temporal windows of events (Section 3) and then detect-and-track features using classical methods [181], [183]. Feature tracks were fused with inertial data by means of keyframe-based nonlinear optimization [226] to recover the camera trajectory and a sparse map of 3D landmarks. Later, the nonlinear optimization in [125] was extended to also fuse intensity frames [27], and was demonstrated on a resource-constrained platform (quadrotor), enabling it to fly in low light and HDR scenarios by exploiting the advantages of event cameras. The above methods are benchmarked on the 6-DOF motion dataset [223], and each method outperforms its predecessor.

Reprojection-error–based VIO: The work in [99] presents a different approach, fusing events and inertial data using a continuous-time framework [227]. As opposed to

the above-mentioned feature-based methods, it optimizes a combined objective with inertial- and event-reprojection error terms over a segment of the camera trajectory, in the style of visual-inertial bundle adjustment.

Opportunities: The above works adapt state-of-the-art VIO methods for conventional cameras by first converting events into geometric information. However, it should be possible to avoid this conversion step and directly recover the camera motion and scene structure from the events, as suggested by [123]; for example, by optimizing a function with photometric (i.e., event firing rate [26]) and inertial error terms, akin to VI-DSO [228] for standard cameras.

Stereo event-based VIO is an unexplored topic, and it would be interesting to see how ideas from event-based depth estimation can be combined with SLAM and VIO.

Also to be explored are learning-based approaches to tackle all of the above problems. Currently, literature is dominated by model-based methods, but, as it happened in frame-based vision, we anticipate that learning-based methods will also play a major role in event-based processing. Some works in this direction are [102], [103], [107], [114].

4.6 Image Reconstruction

Events represent brightness changes, and so, in ideal conditions (noise-free scenario, perfect sensor response, etc.) integration of the events yields "absolute" brightness. This is intuitive, since events are just a non-redundant (i.e., "compressed") per-pixel way of encoding the visual content in the scene. Event integration or, more generically, image reconstruction (Fig. 9) can be interpreted as "decompressing" the visual data encoded in the event stream. Due to the very high temporal resolution of the events, brightness images can be reconstructed at very high frame rate (e.g., 2 kHz–5 kHz [8], [229]) or even continuously in time [61].

As the literature reveals, the insight about image reconstruction from events is that it requires regularization. Event cameras have independent pixels that report brightness changes, and, consequently, per-pixel integration of such changes during a time interval only produces brightness increment images. To recover the absolute brightness at the end of the interval, an offset image (i.e., the brightness image at the start of the interval) would need to be added to the increment [77], [223]. Surprisingly, some works have used spatial and/or temporal smoothing [61], [131], [229], [230] to reconstruct brightness starting from a zero initial condition, i.e., without knowledge of the offset image. Other forms of regularization, using learned features from natural scenes [8], [113], [115], [229] are also effective.

Literature Review: Image reconstruction from events was first established in [23] under rotational camera motion and static scene assumptions. These assumptions together with the brightness constancy (4) were used in a message-passing algorithm between pixels in a network of visual maps to jointly estimate several quantities, such as scene brightness. Also under the above motion and scene assumptions, [24] showed how to reconstruct high-resolution panoramas from the events, and they popularized the idea of event-based HDR image reconstruction. Each pixel of the panoramic image used a Kalman filter to estimate the brightness gradient (based on (4)), which was then integrated using Poisson reconstruction to yield absolute brightness. The method in [231] exploited the constrained motion of a platform rotating around a single axis to reconstruct images that were then used for stereo depth estimation.

Figure 9. Image reconstruction example. Camera pointing at the Sun, in front of a traffic sign. Left: view from a standard camera, showing severe under-exposure on the foreground. Middle: frame from the DAVIS [4], showing severe under- and over-exposed areas. Right: HDR image reconstructed from the events. Image courtesy of [26].

Motion restrictions were then replaced by regularizing assumptions to enable image reconstruction for generic motions and scenes [112]. In this work, image brightness and optical flow were simultaneously estimated using a variational framework that contained several penalty terms (on data fitting (1) and smoothness of the solution) to best explain a space-time volume of events discretized as a voxel grid. This method was the first to show reconstructed video from events in dynamic scenes. Later [131], [229], [230] showed that image reconstruction was possible even without having to estimate motion. This was done using a variational image denoising approach based on time surfaces [131], [230] or using sparse signal processing with a patch-based learned dictionary that mapped events to image gradients, which were then Poisson-integrated [229]. Concurrently, the VO methods in [25], [26] extended the image reconstruction technique in [24] to 6-DOF camera motions by using the computed scene depth and poses: [25] used a robust variational regularizer to reduce noise and improve contrast of the reconstructed image, whereas [26] showed image reconstruction as an ancillary result, since it was not needed to achieve VO. Recently, [61] proposed a temporal smoothing filter for image reconstruction and for continuously fusing events and frames. The filter acted independently on every pixel, thus showing that no spatial regularization on the image plane was needed to recover brightness, although it naturally reduced noise and artefacts at the expense of sacrificing some real detail. More recently, [8], [115] has presented a deep learning approach that achieves considerable gains over previous methods and mitigates visual artefacts. Reflecting back on earlier works, the motion restrictions or hand-crafted regularizers that enabled image reconstruction have been replaced by perceptual, data-driven priors from natural scenes that consequently produced more natural-looking images. Note that image reconstruction methods used in VO or SLAM [23], [24], [25] assume static scenes, whereas methods with weak or no motion assumptions [8], [61], [112], [115], [131], [229], [230] are naturally used to reconstruct videos of arbitrary (e.g., dynamic) scenes.

Besides image reconstruction from events, another category of methods tackles the problem of fusing events

and frames (e.g., from the DAVIS [4]), thus enhancing the brightness information from the frames with high temporal resolution and HDR properties of events [28], [61], [77]. These methods also do not rely on motion knowledge and are ultimately based on (2). The method in [77] performs direct event integration between frames, pixel-wise. However, the fused brightness becomes quickly corrupted by event noise (due to non-ideal effects, sensitivity mismatch, missing events, etc.), and so fusion is reset with every incoming frame. To mitigate noise, events and frames are fused in [61] using a per-pixel, temporal complementary filter that is high-pass in the events and low-pass in the frames. It is an efficient solution that takes into account the complementary sensing modality of events and frames: frames carry slow-varying brightness information (i.e., low temporal frequency), whereas events carry "change" information (i.e., high frequency). The fusion method in [28] exploits the high temporal resolution of the events to additionally remove motion blur from the frames, producing high frame-rate, sharp video from a single blurry frame and events. It is based on a double integral model (one integral to recover brightness and another one to remove blur) within an optimization framework. A limitation of the above methods is that they still suffer from artifacts due to event noise. These might be mitigated if combined with learning-based approaches [8].

Image quality: The quality of the reconstructed image is directly affected by noise in the contrast threshold (Section 2.4), which changes per pixel (due to manufacturing mismatch) and also due to dynamical effects (incident light, time, etc.) [77]. Image quality has been quantified in several works [8], [61], [113], [115], [230] and is also affected by the spatial resolution of the sensor.

Applications: Image reconstruction implies that, in principle, it is possible to convert the events into brightness images and then apply mature computer vision algorithms [8], [93], [115]. This can have a high impact on both, event- and frame-based communities. The resulting images capture high-speed motions and HDR scenes, which may be beneficial in some applications, but it comes at the expense of computational cost, latency and power consumption.

Despite image reconstruction having been useful to support tasks such as recognition [229], SLAM [25] or optical flow estimation [112], there are also works in the literature, such as [107], [114], [123], [149], showing that it is not needed to fulfill such tasks. One of the most valuable aspects of image reconstruction is that it provides scene representations (e.g., appearance maps [7], [24]) that are more *invariant* to motion than events and also facilitate establishing event correspondences, which is one of the biggest challenges of some event data processing tasks, such as feature tracking [63].

4.7 Motion Segmentation

Segmentation of moving objects viewed by a stationary event camera is simple because events are solely imputable to the motion of the objects (assuming constant illumination) [117], [119], [176]. The challenges arise in the scenario of a moving camera because events are triggered everywhere on the image plane [13], [128], [151] (Fig. 10), produced by moving objects and the static scene (whose apparent motion is induced by the camera's ego-motion) and the goal is to infer this causal classification for each event. However, each event carries very little information, and therefore it is challenging to perform the mentioned per-event classification.

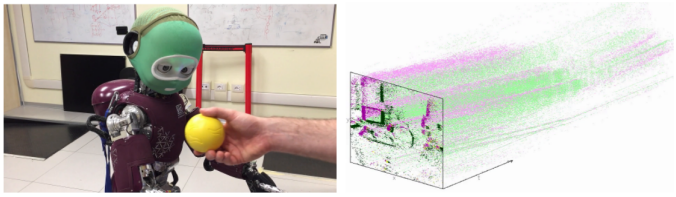


Figure 10. The iCub humanoid robot from IIT has two event cameras in the eyes. Here, it segments and tracks a ball under event clutter produced by the motion of the head. Right: space-time visualization of the events on the image frame, colored according to polarity (positive in green, negative in red). Image courtesy of [177].

Overcoming these challenges has been done by tackling segmentation scenarios of increasing complexity, obtained by reducing the amount of additional information given to solve the problem. Such additional information adopts the form of known object shape or known motion, i.e., the algorithm knows "what object to look for" or "what type of motion it expects" and objects are segmented by detecting (in-)consistency with respect to the expectation. The less additional information is provided, the more unsupervised the problem becomes (e.g., clustering). In such a case, segmentation is enabled by the key insight that moving objects produce distinctive traces of events on the image plane and it is possible to infer the trajectories of the objects that generate those traces, yielding the segmented objects [151]. Like clustering, this is a joint optimization problem in the motion parameters of the objects (i.e., the "clusters") and the event-object associations (i.e., the segmentation).

Literature Review: Considering known object shape, [13] presents a method to detect and track a circle in the presence of event clutter caused by the moving camera. It is based on the Hough transform using optical flow information extracted from temporal windows of events. The method was extended in [177] using a particle filter to improve tracking robustness: the duration of the observation window was dynamically selected to accommodate for sudden motion changes due to accelerations of the object. More generic object shapes were detected and tracked by [184] using event corners (Section 4.1) as geometric primitives. In this method, additional knowledge of the robot joints controlling the camera motion was required.

Segmentation has been addressed by [128], [150], [151] under mild assumptions leveraging the idea of motion-compensated event images [122] (Section 3). Essentially this technique associates events that produce sharp edges when warped according to a motion hypothesis. The simplest hypothesis is a linear motion model (i.e., constant optical flow), yet it is sufficiently expressive: for short times, scenes may be described as collections of objects producing events that fit different linear motion models. Such a scene description is what the cited segmentation algorithms seek for. Specifically, the method in [150] first fits a linear motion-compensation model to the dominant events, then removes these and fits another linear model to the remaining events, greedily. Thus, it clusters events

according to optical flow, yielding motion-compensated images with sharp object contours. Similarly, [128] detects moving objects in clutter by fitting a motion-compensation model to the dominant events (i.e., the background) and detecting inconsistencies with respect to it (i.e., the objects). They test the method in challenging scenarios inaccessible to standard cameras (HDR, high-speed) and release a dataset. The work in [151] proposes an iterative clustering algorithm that jointly estimates the event-object associations (i.e., segmentation) and the motion parameters of the objects (i.e., clusters) that produce sharpest motion-compensated event images. It allows for general parametric motion models [123] to describe each object and produces better results than greedy methods [128], [150]. In [129] a learning-based approach for segmentation using motion-compensation is proposed: ANNs are used to estimate depth, ego-motion, segmentation masks of independently moving objects and object 3D velocities. An event-based dataset is provided for supervised learning, which includes accurate pixel-wise motion masks of 3D-scanned objects that are reliable even in poor lighting conditions and during fast motion.

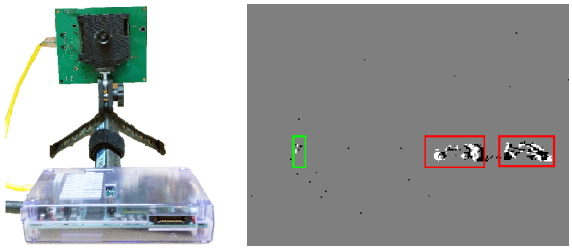
Segmentation is a paramount topic in frame-based vision, yet it is rather unexplored in event-based vision. As more complex scenes are addressed and more advanced event-based vision techniques are developed, more works targeting this challenging problem are expected to appear.

4.8 Recognition

Algorithms: Recognition algorithms for event cameras have grown in complexity, from template matching of simple shapes to classifying arbitrary edge patterns using either traditional machine learning on hand-crafted features or modern deep learning methods. This evolution aims at endowing recognition systems with more expressibility (i.e., approximation capacity) and robustness to data distortions.

Early research with event-based sensors began with tracking a moving object using a static sensor. An event-driven update of the position of a model of the object shape was used to detect and track objects with a known simple shape, such as a blob [6], circle [52], [232] or line [53]. Simple shapes can also be detected by matching against a predefined template, which removes the need to describe the geometry of the object. This *template matching* approach was implemented using convolutions in early hardware [52].

For more complex objects, templates can be used to match low level features instead of the entire object, after which a *classifier* can be used to make a decision based on the distribution of features observed [103]. Nearest Neighbor classifiers are typically used, with distances calculated in feature space. Accuracy can be improved by increasing feature invariance, which can be achieved using a hierarchical model where feature complexity increases in each layer. With a good choice of features, only the final classifier needs to be retrained when switching tasks. This leads to the problem of selecting which features to use. Hand-crafted orientation features were used in early works, but far better results are obtained by learning the features from the data itself. In the simplest case, each template can be obtained from an individual sample, but such templates are sensitive to noise in the sample data [15]. One may follow a generative approach, learning features that enable to accurately reconstruct the input, as was done in [134] with a Deep Belief Network (DBN). More recent work obtains features by unsupervised learning, clustering the event data and using the center of each cluster as a feature [103]. During inference, each event is associated to its closest feature, and a classifier operates on the distributions of features observed. With the rise of *deep learning* in frame-based computer vision, many have sought to leverage deep learning tools for event-based recognition, using back-propagation to learn features. This approach has the advantage of not requiring a separate classifier at the output, but the disadvantage of requiring far more labeled data for training.



(a) (b)

Figure 11. IBM's TrueNorth neurosynaptic system performing recognition of moving objects. (a) A DAVIS240C sensor with FPGA attached performing tracking and sending tracked regions to IBM's TrueNorth NS1e evaluation platform for classification. (b) Tracking and classification results on a street scene. Red boxes indicate cars and the green box indicates a pedestrian.

Most learning-based approaches convert events/spikes into (dense) tensors, a convenient representation for image-based hierarchical models, e.g., ANNs (Fig 11). There are different ways the value of each tensor element can be computed (Section 3.1). Simple methods use the time surfaces, or event histogram frames. A more robust method uses time surfaces with exponential decay [103] or with average timestamps [107]. Image reconstruction methods (Section 4.6) may also be used. Some recognition approaches rely on converting spikes to frames during inference [146], [229], while others convert the trained ANN to an SNN which can operate directly on the event data [133]. Similar ideas can be applied for tasks other than recognition [22], [102]. As neuromorphic hardware advances (Section 5.1), there is increasing interest in learning directly in SNNs [139] or even directly in the neuromorphic hardware itself [140].

Tasks: Early tasks focused on detecting the presence of a simple shape (such as a circle) from a static sensor [6], [52], [232], but soon progressed to the classification of more complex shapes, such as card pips [133], block letters [15] and faces [103], [229]. A popular task throughout has been the classification of hand-written digits. Inspired by the role it has played in frame-based computer vision, a few event-based MNIST datasets have been generated from the original MNIST dataset [57], [233]. These datasets remain a good test for algorithm development, with many algorithms now achieving over $98\,\%$ accuracy on the task [107], [138], [139], [234], [235], [236], but few would propose digit recognition as a strength of event-based vision. More difficult tasks involve either more difficult objects, such as the Caltech-101 and Caltech-256 datasets (both of which are still considered easy by computer vision) or more difficult scenarios, such as

recognition from on-board a moving vehicle [107]. Very few works tackle these tasks so far, and those that do typically fall back on generating event frames and processing them using a traditional deep learning framework.

A key challenge for recognition is that event cameras respond to relative motion in the scene (Section 2.3), and thus require either the object or the camera to be moving. It is therefore unlikely that event cameras will be a strong choice for recognizing static or slow moving objects, although little has been done to combine the advantages of frame- and event-based cameras for these applications. The event-based appearance of an object is highly dependent on the above-mentioned relative motion (Fig. 5), thus tight control of the camera motion could be used to aid recognition [233].

Since the camera responds to dynamic signals, obvious applications include recognizing objects by the way they move [237], or recognizing dynamic scenes such as gestures or actions [16], [17]. These tasks are typically more challenging than static object recognition because they include a time dimension, but this is exactly where event cameras excel.

Opportunities: Event cameras exhibit many alluring properties, but event-based recognition has a long way to go if it is to compete with modern frame-based approaches. While it is important to compare event- and frame-based methods, one must remember that each sensor has its own strengths. The ideal acquisition scenario for a frame-based sensor consists of both the sensor and object being static, which is the worst possible scenario for event cameras. For event-based recognition to find widespread adoption, it will need to find applications which play to its strengths. Such applications are unlikely to be similar to well established computer vision recognition tasks which play to the frame-based sensor's strengths. Instead, such applications are likely to involve resource constrained recognition of dynamic sequences, or recognition from on-board a moving platform. Finding and demonstrating the use of event-based sensors in such applications remains an open challenge.

Although event-based datasets have improved in quality in recent years, there is still room for improvement. Much more data is being collected, but annotation remains challenging. There is not yet an agreed upon or standard tool or format for annotations. Many event-based datasets are derived from frame-based vision. While these datasets have played an important role in the field, they inherently play to the strengths of frame-based vision and are thus unlikely to give rise to new event-based sensor applications. Data collection and annotation is a tiresome and thankless task, but developing an easy to use pipeline for collecting and annotating event-based data would be a significant contribution to the field, especially if the tools can mature to the stage where the task can be outsourced to laymen.

4.9 Neuromorphic Control

In living creatures, most information processing happens through spike-based representation: spikes encode the sensory data; spikes perform the computation; and spikes transmit actuator "commands". Therefore, biology shows that the event-based paradigm is, in principle, applicable not just to perception and inference, but also to control.

Neuromorphic-vision-driven Control Architecture: In this type of architecture (Fig. 12), there is a neuromorphic sensor, an event-based estimator, and a traditional controller. The estimator computes a state, and the controller computes the control based on the provided state. The controller is not aware of the asynchronicity of the architecture.

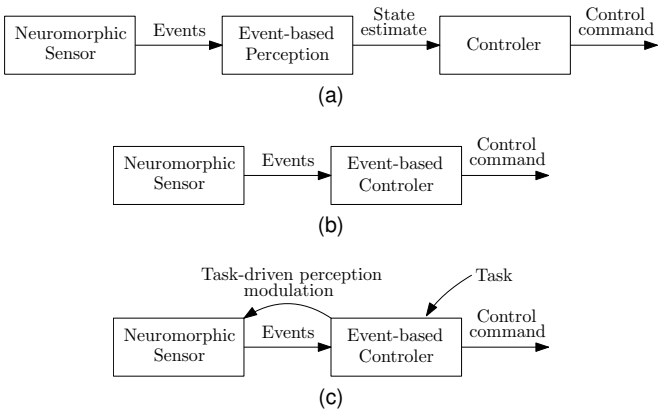


Figure 12. Control architectures based on neuromorphic events. In a neuromorphic-vision-driven control architecture (a), a neuromorphic sensor produces events, an event-based perception system produces state estimates, and a traditional controller is called asynchronously to compute the control signal. In a native neuromorphic-based architecture (b), the events generate directly changes in control. Finally, (c) shows an architecture in which the task informs the events that are generated.

Neuromorphic-vision-driven control architectures have been demonstrated since the early days of neuromorphic cameras, and they have proved the two advantages of low latency and computational efficiency. The earliest demonstrators were the spike-based convolutional target tracking demo in the CAVIAR project [52] and the "robot goalie" described in [6], [12]. Another early example was the pencil-balancing robot [53]. In that demonstrator two DVS's observed a pencil as inverted pendulum placed on a small movable cart. The pencil's state in 3D was estimated in below 1 ms latency. A simple hand tuned PID controller kept the pencil balanced upright. It was also demonstrated on an embedded system, thereby establishing the ability to run on severely constrained computing resources.

Event-based Control Theory: Event-based techniques can be motivated from the perspective of control and decision theory. Using a biological metaphor, event-based control can be understood as a form of what economics calls *rational inattention* [238]: more information allows for better decisions, but if there are costs associated to obtaining or processing the information, it is rational to take decisions with only partial information available.

In event-based control, the control signal is changed asynchronously [239]. There are several variations of the concept depending on how the "control events" are generated. One important distinction is between *event-triggered control* and *self-triggered control* [240]. In *event-based control* the events are generated "exogenously" based on certain condition; for example, a "recompute control" request might be triggered when the trajectory's tracking error exceeds a threshold. In *self-triggered control*, the controller decides by itself when is the next time it should be called based on the situation. For example, a controller might decide to "sleep" for longer if the state is near the target, or to recompute the control signal sooner if it is required.

The advantages of event-based control are usually jus-

tified considering a trade-off between computation / communication cost and control performance. The basic consideration is that, while the best control performance is obtained by recomputing the control infinitely often (for an infinite cost), there are strongly diminishing returns. A solid principle of control theory is that the control frequency depends on the time constant of the plant and the sensor: it does not make sense to change the control much quicker than the new incoming information or the speed of the actuators. This motivates choosing control frequencies that are comparable with the plant dynamics and adapt to the situation. For example, one can show that an event-triggered controller achieves the same performance with a fraction of the computation; or, conversely, a better performance with the same amount of computation. In some cases (scalar linear Gaussian) these trade-offs can be obtained in closed form [241], [242]. (Analogously, certain trade-offs can be obtained in closed form for perception [243].)

Unfortunately, the large literature in event-based control is of restricted utility for the embodied neuromorphic setting. Beyond the superficial similarity of dealing with "events" the settings are quite different. For example, in network-based control, one deals with typically low-dimensional states and occasional events—the focus is on making the most of each single event. By contrast, for an autonomous vehicle equipped with event cameras, the problem is typically how to find useful signals in potentially millions of events per second. Particularizing the event-based control theory to the neuromorphic case is a relatively young avenue of research [244], [245], [246], [247]. The challenges lie in handling the non-linearities typical of the vision modality, which prevents clean closed-form results.

Open questions in Neuromorphic Control: Finally, we describe some of open problems in this topic.

Task-driven sensing: In animals, perception has value because it is followed by action, and the information collected is *actionable information* that helps with the task. A significant advance would be the ability for a controller to modulate the sensing process based on the task and the context. In current hardware there is limited software-modulated control for the sensing processing, though it is possible to modulate some of the hardware biases. Integration with region-of-interest mechanisms, heterogeneous camera bias settings, etc. would provide additional flexibility and more computationally efficient control.

Thinking fast and slow: Existing research has focused on obtaining low-latency control, but there has been little work on how to integrate this sensorimotor level into the rest of an agent's cognitive architecture. Using again a bio-inspired metaphor, and following Kahneman [248], the fast/instinctive/"emotional" system must be integrated with the slower/deliberative system.

5 Event-based Systems and Applications

5.1 Neuromorphic Computing

Neuromorphic engineering tries to capture some of the unparalleled computational power and efficiency of the brain by mimicking its structure and function. Typically this results in a massively parallel hardware accelerator for SNNs (Section 3.3), which is how we will define a neuromorphic processor. Since the neuron spikes within such a processor are inherently asynchronous, a neuromorphic processor is the best computational partner for an event camera. Neuromorphic processors act on the events injected by the event camera directly, without conversion, and offer better data-processing locality (spatially and temporally) than standard architectures such as CPUs, yielding low power and low latency computer vision systems.

Table 4
Comparison between selected neuromorphic processors, ordered by neuron model type.

Processor	SpiNNaker	TrueNorth	Loihi	DYNAP	Braindrop
Reference	[249]	[250]	[251]	[252]	[253]
Manufacturer	U. Manchester	IBM	Intel	aiCTX	Stanford U.
Year	2011	2014	2018	2017	2018
Neuron model	Software	Digital	Digital	Analog	Analog
On-chip learning	Yes	No	Yes	No	No
CMOS technol.	130 nm	28 nm	14 nm	180 nm	28 nm
Neurons/chip	4 k*	1024 k	131 k	1 k	4 k
Neurons/core	255*	256	1024	256	4096
Cores/chip	16*	4096	128	4	1
Synapses/chip	16 M	268 M	130 M	128 k	16 M
Boards	4- or 48-chip	1- or 16-chip	4- or 8-chip,	1-chip	1-chip
Software stack	sPyNNaker PACMAN	CPE/Eedn NSCP	Nengo Nx SDK	cAER libcAER	Nengo

Neuromorphic processors may be categorized by their neuron model implementations (Table 4), which are broadly divided between analog neurons (Neurogrid, BrainScaleS, ROLLS, DYNAP-se), digital neurons (TrueNorth, Loihi, ODIN) and software neurons (SpiNNaker). Some architectures also support on-chip learning (Loihi, ODIN, DYNAP-le). When evaluating a neuromorphic processor for an event-based vision system, the following criteria should be considered in addition to the processor's functionality and performance: (*i*) the software development ecosystem: a minimal toolchain includes an API to compose and train a network, a compiler to prepare the network for the hardware, and a runtime library to deploy the network in hardware, (*ii*) event-based vision systems typically require that a processor be available as a standalone system suitable for mobile applications, and not just hosted in a remote server, (*iii*) the availability of neuromorphic processors.

Several developments are necessary to enable a more widespread use of these processors, such as: (*i*) developing a more user-friendly ecosystem (an easier way to program the desired method for deployment in hardware), (*ii*) enabling more processing capabilities of the hardware platform, (*iii*) increasing the availability of devices beyond early access programs targeted at selected partners.

The following processors (Table 4) have the most mature developer workflows, combined with the widest availability of standalone systems. More details are given in [254], [255].

SpiNNaker (Spiking Neural Network Architecture) uses general-purpose ARM cores to simulate biologically realistic models of the human brain [256]. Unlike the other processors that choose specific simplified neuron models to embed in custom transistor circuits, SpiNNaker implements neurons as software running on the cores, sacrificing hardware acceleration to maximize model flexibility. The SpiNNaker has been coupled with event cameras for stereo depth estimation [161], [257], optic flow computation [257], [258], and for object tracking [259] and recognition [260].

TrueNorth uses digital neurons to perform real-time

(a) (b)

Figure 13. (a) Embedded DVS128 on Pushbot as standalone closed-loop perception-computation-action system, used in navigation and obstacle-avoidance tasks [265]. (b) Drone with downward-looking event camera, used for autonomous flight [27].

inference. Each chip simulates 1 M (million) neurons and 256 M synapses, distributed among 4096 neurosynaptic cores. There is no on-chip learning, so networks are trained offline using a GPU or other processor [261]. TrueNorth has been paired with event cameras for gesture recognition [17], stereo reconstruction [207] and optical flow estimation [189].

Loihi uses digital neurons to perform real-time inference and online learning. Each chip simulates up to 131 thousand spiking neurons and 130 M synapses. A learning engine in each neuromorphic core updates each synapse using rules that includes STDP and reinforcement learning [251]. Non-spiking networks can be trained in TensorFlow and approximated by spiking networks for Loihi using the Nengo Deep Learning toolkit from Applied Brain Research [262]. There are plans to build a larger system called Pohoiki Springs with 768 Loihi chips (100 M neurons and 100 billion synapses), which is still small compared to the human brain's 800 trillion synapses.

DYNAP: The Dynamic Neuromorphic Asynchronous Processor (DYNAP) from aiCTX has two variants, one optimized for scalable inference (Dynap-se), and another for online learning (Dynap-le).

Braindrop prototypes a single core of the 1 M-neuron Brainstorm system [253]. It is programmed using Nengo [263] and implements the Neural Engineering Framework [264]. Braindrop is Stanford University's follow-on to the Neurogrid processor.

5.2 Applications in Real-Time On-Board Robotics

As event-based vision sensors often produce significantly less data per time interval compared to traditional cameras, multiple applications can be envisioned where extracting relevant vision information can happen in real-time within a simple computing system directly connected to the sensor, avoiding USB connection. Fig 13 shows an example of such, where a dual-core ARM micro controller running at 200 MHz with 136 kB on-board SRAM fetches and processes events in real-time. The combined embedded system of sensor and micro controller here operate a simple wheeled robot in tasks such as line following, active and passive object tracking, distance estimation, and simple mapping [265].

A different example of near-sensor processing ("edge computing") is the Speck SoC[10], which combines a DVS and the Dynap-se neuromorphic CNN processor. Its peak power consumption is less than 1 mW and latency is less than 30 ms. Application domains are low-power, continuous object detection, surveillance, and automotive systems.

Event cameras have also been used on-board quadrotors with limited computational resources, both for autonomous landing [266] or flight [27] (Fig. 13b), in challenging scenes.

6 Resources

The List of Event-based Vision Resources [9] is a collaborative effort to collect relevant resources in the field: links to information (papers, videos, organizations, companies and workshops) as well as drivers, code, datasets, simulators and other essential tools in event-based vision.

6.1 Software

To date, there is no open-source standard library integrated to OpenCV that provides algorithms for event-based vision. This would be a very desirable resource to accelerate the adoption of event cameras. There are, however, many quite highly developed open-source software resources:

- *jAER* [267][11] is a Java-based environment for event sensors and processing like noise reduction, feature extraction, optical flow, de-rotation using IMU, CNN and RNN inference, etc. Several non-mobile robots [6], [12], [53], [268] and even one mobile DVS [146] robot have been built in jAER, although Java is not ideal for mobile robots. It provides a desktop GUI based interface for easily recording and playing data that also exposes the complex internal configurations of these devices. It mainly supports the sensors developed at the Inst. of Neuroinformatics (UZH-ETH Zurich) that are distributed by iniVation.
- *libcaer*[12] is a minimal C library to access, configure and get data from iniVation and aiCTX neuromorphic sensors and processors. It supports the DVS and DAVIS cameras, and the Dynap-SE neuromorphic processor.
- The ROS DVS package[13] developed in the Robotics and Perception Group of UZH-ETH Zurich is based on libcaer. It provides C++ drivers for the DVS and DAVIS. It is popular in robotics since it integrates with the Robot Operating System (ROS) [269] and therefore provides high-level tools for easily recording and playing data, connecting to other sensors and actuators, etc. Popular datasets [190], [223] are provided in this format. The package also provides a calibration tool for both intrinsic and stereo calibration.
- The event-driven YARP Project[14] [270] comprises libraries to handle neuromorphic sensors, such as the DVS, installed on the iCub humanoid robot, along with algorithms to process event data. It is based on the Yet Another Robot Platform (YARP) middleware.
- pyAER[15] is a python wrapper around libcaer developed at the Inst. of Neuroinformatics (UZH-ETH) that will probably become popular for rapid experimentation.

10. https://www.speck.ai/

11. https://jaerproject.org
12. https://github.com/inilabs/libcaer
13. https://github.com/uzh-rpg/rpg_dvs_ros
14. https://github.com/robotology/event-driven
15. https://github.com/duguyue100/pyaer

Other open-source software utilities and processing algorithms (in Python, CUDA, Matlab, etc.) are spread throughout the web, on the pages of the research groups working on event-based vision [9]. Proprietary software includes the development kits (SDKs) developed by companies.

6.2 Datasets and Simulators

Datasets and simulators are fundamental tools to facilitate adoption of event-driven technology and advance its research. They allow to reduce costs (currently, event cameras are considerably more expensive than standard cameras) and to monitor progress with quantitative benchmarks (as in traditional computer vision: the case of datasets such as Middlebury, MPI Sintel, KITTI, EuRoC, etc.).

The number of event-based vision datasets and simulators is growing. Several of them are listed in [9], sorted by task. Broadly, they can be categorized as those that target motion estimation (regression) tasks and those that target recognition (classification) tasks. In the first group, there are datasets for optical flow, SLAM, object tracking, segmentation, etc. The second group comprises datasets for object and action recognition.

Datasets for optical flow include [22], [186], [271]. Since ground-truth optical flow is difficult to acquire, [186] considers only flow during purely rotational motion recorded with an IMU, and so, the dataset lacks flow due to translational (parallax) motion. The datasets in [22], [271] provide optical flow as the motion field induced on the image plane by the camera motion and the depth of the scene (measured with a range sensor, such as an RGB-D camera, a stereo pair or a LiDAR). Naturally, ground truth optical flow is subject to noise and inaccuracies in alignment and calibration of the different sensors involved.

Datasets for pose estimation and SLAM include [219][16], [190], [223], [271], [272]. The most popular one is described in [223], which has been used to benchmark visual odometry and visual-inertial odometry methods [27], [98], [99], [122], [123], [125], [127]. This dataset is also popular to evaluate corner detectors [105], [109] and feature trackers [63], [121].

Datasets for recognition are currently of limited size compared to traditional computer vision ones. They consist of cards of a deck (4 classes), faces (7 classes), handwritten digits (36 classes), gestures (rocks, papers, scissors) in dynamic scenes, cars, etc. Neuromorphic versions of popular frame-based computer vision datasets, such as MNIST and Caltech101, have been obtained by using saccade-like motions [233], [273]. Newer datasets [17], [107], [274], [275] are acquired in real scenarios (not generated from frame-based data). These datasets have been used in [15], [103], [107], [133], [146], [147], among others, to benchmark event-based recognition algorithms.

The DVS *emulators* in [72], [276] and the *simulator* in [223] are based on the working principle of an ideal DVS pixel (2). Given a virtual 3D scene and the trajectory of a moving DAVIS within it, the simulator generates the corresponding stream of events, intensity frames and depth maps. The simulator has been extended in [73]: it uses an adaptive rendering scheme, is more photo-realistic, includes an event noise model and returns ground truth optical flow.

16. http://ebvds.neurocomputing.systems

A comprehensive characterization of the noise and dynamic effects of existing event cameras has not been carried out yet, and so, the noise models used are, currently, only a coarse approximation. In the future, it would be desirable to develop more realistic sensor models so that prototyping on simulated data transferred more easily to real data.

6.3 Workshops

There are two yearly Summer schools fostering research, among other topics, on event-based vision: the Telluride Neuromorphic Cognition Engineering Workshop (27th edition in 2020, in USA) and the Capo Caccia Workshop (12th edition in 2020, in Europe). Recently, workshops have been organized alongside major robotics conferences (IROS'15[17], ICRA'17[18] or IROS'18[19]). Live demos of event-based systems have been shown at top-tier conferences, such as ISSCC'06, NIPS'09, CVPR'18, ECCV'18, ICRA'17, IROS'18, CVPR'19[20], multiple ISCAS, etc. As the event-based vision community grows, more workshops and live demos are expected to happen in traditional computer vision venues.

7 Discussion

Event-based vision is a topic that spans many fields, such as computer vision, robotics and neuromorphic engineering. Each community focuses on exploiting different advantages of the event-based paradigm. Some focus on the low power consumption for "always on" or embedded applications on resource-constrained platforms; others favor low latency to enable highly reactive systems, and others prefer the availability of information to better perceive the environment (high temporal resolution and HDR), with fewer constraints on computational resources.

Event-based vision is an emerging technology in the era of mature frame-based camera hardware and software. Comparisons are, in some terms, unfair since they are not carried out under the same maturity level. Nevertheless event cameras show potential, able to overcome some of the limitations of frame-based cameras, reaching new scenarios previously inaccessible. There is considerable room for improvement (research and development), as pointed out in numerous opportunities throughout the paper.

There is no agreement on what is the best method to process events, notably because it depends on the application. There are different trade-offs involved, such as latency vs. power consumption and accuracy, or sensitivity vs. bandwidth and processing capacity. For example, reducing the contrast threshold and/or increasing the resolution produces more events, which will be processed by an algorithm and platform with finite capacity. A challenging research area is to quantify such trade-offs and to develop techniques to dynamically adjust the sensor and/or algorithm parameters for optimal performance.

Another big challenge is to develop bio-inspired systems that are natively event-based end-to-end (from perception to control and actuation) that are also more efficient and

17. http://innovative-sensing.mit.edu/
18. http://rpg.ifi.uzh.ch/ICRA17_event_vision_workshop.html
19. http://www.jmartel.net/irosws-home
20. http://rpg.ifi.uzh.ch/CVPR19_event_vision_workshop.html

long-term solutions than synchronous, frame-based systems. Event cameras pose the challenge of rethinking perception, control and actuation, and, in particular, the current main stream of deep learning methods in computer vision: adapting them or transferring ideas to process events while being as top-performing. Active vision (pairing perception and control) is specially relevant on event cameras because the events distinctly depends on motion, which may be due to the actuation of a robot.

Event cameras can be seen as an entry point for more efficient, near-sensor processing, such that only high-level, non-redundant information is transmitted, thus reducing bandwidth, latency and power consumption. This could be done by pairing an event camera with hardware on the same sensor device (Speck in Section 5.2), or by alternative bio-inspired imaging sensors, such as cellular processor arrays [277] which every pixel has a processor that allows to perform several types of computations with the brightness of the pixel and its neighbors.

8 Conclusion

Event cameras are revolutionary sensors that offer many advantages over traditional, frame-based cameras, such as low latency, low power, high speed and high dynamic range. Hence, they have a large potential for computer vision and robotic applications in challenging scenarios currently inaccessible to traditional cameras. We have provided an overview of the field of event-based vision, covering perception, computing and control, with a focus on the working principle of event cameras and the algorithms developed to unlock their outstanding properties in selected applications, from low-level vision to high-level vision. Neuromorphic perception and control are emerging topics; and so, there are plenty of opportunities, as we have pointed out throughout the text. Many challenges remain ahead, and we hope that this paper provides an introductory exposition of the topic, as a step in humanity's longstanding quest to build intelligent machines endowed with a more efficient, bio-inspired way of perceiving and interacting with the world.

References

[1] M. Mahowald and C. Mead, "The silicon retina," *Scientific American*, vol. 264, no. 5, pp. 76–83, May 1991.

[2] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128×128 120 dB 15 μs latency asynchronous temporal contrast vision sensor," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[3] C. Posch, D. Matolin, and R. Wohlgenannt, "A QVGA 143 dB dynamic range frame-free PWM image sensor with lossless pixel-level video compression and time-domain CDS," *IEEE J. Solid-State Circuits*, vol. 46, no. 1, pp. 259–275, Jan. 2011.

[4] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240x180 130dB 3μs latency global shutter spatiotemporal vision sensor," *IEEE J. Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, 2014.

[5] B. Son, Y. Suh, S. Kim, H. Jung, J.-S. Kim, C. Shin, K. Park, K. Lee, J. Park, J. Woo, Y. Roh, H. Lee, Y. Wang, I. Ovsiannikov, and H. Ryu, "A 640x480 dynamic vision sensor with a 9μm pixel and 300Meps address-event representation," in *IEEE Intl. Solid-State Circuits Conf. (ISSCC)*, 2017.

[6] T. Delbruck and P. Lichtsteiner, "Fast sensory motor control based on event-based hybrid neuromorphic-procedural system," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2007, pp. 845–848.

[7] G. Gallego, J. E. A. Lund, E. Mueggler, H. Rebecq, T. Delbruck, and D. Scaramuzza, "Event-based, 6-DOF camera tracking from photometric depth maps," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 10, pp. 2402–2412, Oct. 2018.

[8] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza, "High speed and high dynamic range video with an event camera," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

[9] https://github.com/uzh-rpg/event-based_vision_resources.

[10] T. Delbruck, "Neuromorphic vision sensing and processing," in *Eur. Solid-State Device Research Conf. (ESSDERC)*, 2016, pp. 7–14.

[11] S.-C. Liu, B. Rueckauer, E. Ceolini, A. Huber, and T. Delbruck, "Event-Driven Sensing for Efficient Perception: Vision and audition algorithms," *IEEE Signal Process. Mag.*, vol. 36, no. 6, pp. 29–37, Nov 2019.

[12] T. Delbruck and M. Lang, "Robotic goalie with 3ms reaction time at 4% CPU load using event-based dynamic vision sensor," *Front. Neurosci.*, vol. 7, p. 223, 2013.

[13] A. Glover and C. Bartolozzi, "Event-driven ball detection and gaze fixation in clutter," in *IEEE Int. Conf. Intell. Robot. Syst. (IROS)*, 2016.

[14] M. Litzenberger, A. N. Belbachir, N. Donath, G. Gritsch, H. Garn, B. Kohn, C. Posch, and S. Schraml, "Estimation of vehicle speed based on asynchronous data from a silicon retina optical sensor," in *IEEE Intell. Transp. Sys. Conf.*, 2006, pp. 653–658.

[15] G. Orchard, C. Meyer, R. Etienne-Cummings, C. Posch, N. Thakor, and R. Benosman, "HFirst: A temporal approach to object recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 10, pp. 2028–2040, 2015.

[16] J. H. Lee, T. Delbruck, M. Pfeiffer, P. K. Park, C.-W. Shin, H. Ryu, and B. C. Kang, "Real-time gesture interface based on event-driven processing from stereo silicon retinas," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 12, pp. 2250–2263, 2014.

[17] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. D. Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza, J. Kusnitz, M. Debole, S. Esser, T. Delbruck, M. Flickner, and D. Modha, "A low power, fully event-based gesture recognition system," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017.

[18] P. Rogister, R. Benosman, S.-H. Ieng, P. Lichtsteiner, and T. Delbruck, "Asynchronous event-based binocular stereo matching," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 2, pp. 347–353, 2012.

[19] H. Rebecq, G. Gallego, E. Mueggler, and D. Scaramuzza, "EMVS: Event-based multi-view stereo—3D reconstruction with an event camera in real-time," *Int. J. Comput. Vis.*, vol. 126, no. 12, pp. 1394–1414, 2018.

[20] N. Matsuda, O. Cossairt, and M. Gupta, "MC3D: Motion contrast 3D scanning," in *IEEE Int. Conf. Comput. Photography (ICCP)*, 2015.

[21] R. Benosman, C. Clercq, X. Lagorce, S.-H. Ieng, and C. Bartolozzi, "Event-based visual flow," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 2, pp. 407–417, 2014.

[22] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "EV-FlowNet: Self-supervised optical flow estimation for event-based cameras," in *Robotics: Science and Systems (RSS)*, 2018.

[23] M. Cook, L. Gugelmann, F. Jug, C. Krautz, and A. Steger, "Interacting maps for fast visual interpretation," in *Int. Joint Conf. Neural Netw. (IJCNN)*, 2011, pp. 770–776.

[24] H. Kim, A. Handa, R. Benosman, S.-H. Ieng, and A. J. Davison, "Simultaneous mosaicing and tracking with an event camera," in *British Mach. Vis. Conf. (BMVC)*, 2014.

[25] H. Kim, S. Leutenegger, and A. J. Davison, "Real-time 3D reconstruction and 6-DoF tracking with an event camera," in *Eur. Conf. Comput. Vis. (ECCV)*, 2016.

[26] H. Rebecq, T. Horstschäfer, G. Gallego, and D. Scaramuzza, "EVO: A geometric approach to event-based 6-DOF parallel tracking and mapping in real-time," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 593–600, 2017.

[27] A. Rosinol Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate SLAM? combining events, images, and IMU for robust visual SLAM in HDR and high speed scenarios," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 994–1001, Apr. 2018.

[28] 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 *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.

[29] G. Cohen, S. Afshar, A. van Schaik, A. Wabnitz, T. Bessell, M. Rutten, and B. Morreale, "Event-based sensing for space situational awareness," in *Proc. Advanced Maui Optical and Space Surveillance Technol. Conf. (AMOS)*, 2017.

[30] T.-J. Chin, S. Bagchi, A. Eriksson, and A. van Schaik, "Star tracking using an event camera," in *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, 2019.

[31] P. Lichtsteiner and T. Delbruck, "64x64 event-driven logarithmic temporal derivative silicon retina," in *IEEE Workshop on Charge-Coupled Devices and Advanced Image Sensors*, 2005, pp. 157–160.

[32] ——, "A 64x64 AER logarithmic temporal derivative silicon retina," in *Research in Microelectr. & Electron., PhD*, vol. 2, 2005.

[33] P. Lichtsteiner, "An AER temporal contrast vision sensor," Ph. D. Thesis, ETH Zurich, Dept. of Physics (D-PHYS), Zurich, Switzerland, 2006.

[34] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128x128 120dB 30mW asynchronous vision sensor that responds to relative intensity change," in *IEEE Intl. Solid-State Circuits Conf. (ISSCC)*, 2006, pp. 2060–2069.

[35] D. Neil, "Deep neural networks and hardware systems for event-driven data," Ph.D. dissertation, ETH-Zurich, Zurich, Switzerland, 2017.

[36] C. Posch, T. Serrano-Gotarredona, B. Linares-Barranco, and T. Delbruck, "Retinomorphic event-based vision sensors: Bioinspired cameras with spiking output," *Proc. IEEE*, vol. 102, no. 10, pp. 1470–1484, Oct. 2014.

[37] K. A. Boahen, "A burst-mode word-serial address-event link-I: Transmitter design," *IEEE Trans. Circuits Syst. I*, vol. 51, no. 7, pp. 1269–1280, Jul. 2004.

[38] S.-C. Liu, T. Delbruck, G. Indiveri, A. Whatley, and R. Douglas, *Event-Based Neuromorphic Systems*. John Wiley & Sons, 2015.

[39] Y. Suh, S. Choi, M. Ito, J. Kim, Y. Lee, J. Seo, H. Jung, D.-H. Yeo, S. Namgung, J. Bong, J. seok Kim, P. K. J. Park, J. Kim, H. Ryu, and Y. Park, "A 1280x960 Dynamic Vision Sensor with a 4.95-μm pixel pitch and motion artifact minimization," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2020.

[40] M. Mahowald, "VLSI analogs of neuronal visual processing: A synthesis of form and function," Ph.D. dissertation, California Institute of Technology, Pasadena, California, May 1992.

[41] D. il Dan Cho and T. jae Lee, "A review of bioinspired vision sensors and their applications," *Sens. Mater.*, vol. 27, no. 6, pp. 447–463, 2015.

[42] L. Steffen, D. Reichard, J. Weinland, J. Kaiser, A. Rönnau, and R. Dillmann, "Neuromorphic stereo vision: A survey of bio-inspired sensors and algorithms," *Front. Neurorobot.*, vol. 13, p. 28, 2019.

[43] T. Delbruck and C. A. Mead, "Time-derivative adaptive silicon photoreceptor array," in *Proc. SPIE, Infrared sensors: Detectors, Electron., and Signal Process.*, vol. 1541, 1991, pp. 92–99.

[44] S.-C. Liu and T. Delbruck, "Neuromorphic sensory systems," *Current Opinion in Neurobiology*, vol. 20, no. 3, pp. 288–295, 2010.

[45] T. Delbruck, B. Linares-Barranco, E. Culurciello, and C. Posch, "Activity-driven, event-based vision sensors," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2010, pp. 2426–2429.

[46] T. Delbruck, "Fun with asynchronous vision sensors and processing," in *Eur. Conf. Comput. Vis. Workshops (ECCVW)*, 2012, pp. 506–515.

[47] C. Posch, D. Matolin, and R. Wohlgenannt, "A QVGA 143dB dynamic range asynchronous address-event PWM dynamic image sensor with lossless pixel-level video compression," in *IEEE Intl. Solid-State Circuits Conf. (ISSCC)*, 2010, pp. 400–401.

[48] E. Culurciello, R. Etienne-Cummings, and K. A. Boahen, "A biomorphic digital image sensor," *IEEE J. Solid-State Circuits*, vol. 38, no. 2, pp. 281–294, Feb. 2003.

[49] G. Orchard, D. Matolin, X. Lagorce, R. Benosman, and C. Posch, "Accelerated frame-free time-encoded multi-step imaging," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2014, pp. 2644–2647.

[50] R. Berner, C. Brandli, M. Yang, S.-C. Liu, and T. Delbruck, "A 240x180 10mw 12μs latency sparse-output vision sensor for mobile applications," in *Proc. Symp. VLSI*, 2013, pp. C186–C187.

[51] E. Fossum, "CMOS image sensors: electronic camera-on-a-chip," *IEEE Trans. Electron Devices*, vol. 44, no. 10, pp. 1689–1698, 1997.

[52] R. Serrano-Gotarredona, M. Oster, P. Lichtsteiner, A. Linares-Barranco, R. Paz-Vicente, F. Gomez-Rodriguez, L. Camunas-Mesa, R. Berner, M. Rivas-Perez, T. Delbruck, S.-C. Liu, R. Douglas, P. Hafliger, G. Jimenez-Moreno, A. C. Ballcels, T. Serrano-Gotarredona, A. J. Acosta-Jimenez, and B. Linares-Barranco, "CAVIAR: A 45k neuron, 5M synapse, 12G connects/s AER hardware sensory-processing-learning-actuating system for high-speed visual object recognition and tracking," *IEEE Trans. Neural Netw.*, vol. 20, no. 9, pp. 1417–1438, 2009.

[53] J. Conradt, M. Cook, R. Berner, P. Lichtsteiner, R. J. Douglas, and T. Delbruck, "A pencil balancing robot using a pair of AER dynamic vision sensors," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2009, pp. 781–784.

[54] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017, pp. 3530–3538.

[55] Y. Nozaki and T. Delbruck, "Temperature and parasitic photocurrent effects in dynamic vision sensors," *IEEE Trans. Electron Devices*, vol. 64, no. 8, pp. 3239–3245, Aug. 2017.

[56] ——, "Authors Reply to Comment on Temperature and Parasitic Photocurrent Effects in Dynamic Vision Sensors," *IEEE Trans. Electron Devices*, vol. 65, no. 7, pp. 3083–3083, Jul. 2018.

[57] T. Serrano-Gotarredona and B. Linares-Barranco, "A 128 $\times$128 1.5contrast sensitivity 0.9dynamic vision sensor using transimpedance preamplifiers," *IEEE J. Solid-State Circuits*, vol. 48, no. 3, pp. 827–838, Mar. 2013.

[58] M. Yang, S.-C. Liu, and T. Delbruck, "A dynamic vision sensor with 1% temporal contrast sensitivity and in-pixel asynchronous delta modulator for event encoding," *IEEE J. Solid-State Circuits*, vol. 50, no. 9, pp. 2149–2160, 2015.

[59] D. P. Moeys, F. Corradi, C. Li, S. A. Bamford, L. Longinotti, F. F. Voigt, S. Berry, G. Taverni, F. Helmchen, and T. Delbruck, "A sensitive dynamic and active pixel vision sensor for color or neural imaging applications," *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, no. 1, pp. 123–136, Feb. 2018.

[60] A. Rose, *Vision: Human and Electronic*. Plenum Press, New York, 1973.

[61] C. Scheerlinck, N. Barnes, and R. Mahony, "Continuous-time intensity estimation using event cameras," in *Asian Conf. Comput. Vis. (ACCV)*, 2018.

[62] ——, "Asynchronous spatial image convolutions for event cameras," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 816–822, 2019.

[63] D. Gehrig, H. Rebecq, G. Gallego, and D. Scaramuzza, "EKLT: Asynchronous photometric feature tracking using events and frames," *Int. J. Comput. Vis.*, 2019.

[64] S. Bryner, G. Gallego, H. Rebecq, and D. Scaramuzza, "Event-based, direct camera tracking from a photometric 3D map using nonlinear optimization," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2019.

[65] G. Gallego, C. Forster, E. Mueggler, and D. Scaramuzza, "Event-based camera pose tracking using a generative event model," 2015, arXiv:1510.01972.

[66] Prophesee Evaluation Kits, https://www.prophesee.ai/event-based-evk/, 2019.

[67] T. Finateu, A. Niwa, D. Matolin, K. Tsuchimoto, A. Mascheroni, E. Reynaud, P. Mostafalu, F. Brady, L. Chotard, F. LeGoff, H. Takahashi, H. Wakabayashi, Y. Oike, and C. Posch, "A 1280x720 back-illuminated stacked temporal contrast event-based vision sensors with 4.86μm pixels, 1.066geps readout, programmable event-rate controller and compressive data-formatting pipeline," in *IEEE Intl. Solid-State Circuits Conf. (ISSCC)*, 2020.

[68] H. E. Ryu, "Industrial DVS design; key features and applications," http://rpg.ifi.uzh.ch/docs/CVPR19workshop/CVPRW19_Eric_Ryu_Samsung.pdf.

[69] M. Guo, J. Huang, and S. Chen, "Live demonstration: A 768 x 640 pixels 200meps dynamic vision sensor," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2017.

[70] S. Chen and G. Menghan, "Live demonstration: CeleX-V: a 1M pixel multi-mode event-based sensor," in *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, 2019.

[71] Insightness Event-based Sensor Modules, http://www.insightness.com/technology/.

[72] M. L. Katz, K. Nikolic, and T. Delbruck, "Live demonstration: Behavioural emulation of event-based vision sensors," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2012, pp. 736–740.

[73] H. Rebecq, D. Gehrig, and D. Scaramuzza, "ESIM: an open event camera simulator," in *Conf. on Robotics Learning (CoRL)*, 2018.

[74] A. Censi and D. Scaramuzza, "Low-latency event-based visual odometry," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2014.

[75] M. Yang, S.-C. Liu, and T. Delbruck, "Analysis of encoding degradation in spiking sensors due to spike delay variation," *IEEE Trans. Circuits Syst. I*, vol. 64, no. 1, pp. 145–155, Jan. 2017.

[76] C. Posch and D. Matolin, "Sensitivity and uniformity of a 0.18μm CMOS temporal contrast pixel array," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2011, pp. 1572–1575.

[77] C. Brandli, L. Muller, and T. Delbruck, "Real-time, high-speed video decompression using a frame- and event-based DAVIS sensor," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2014.

[78] M. Yang, S.-C. Liu, and T. Delbruck, "Comparison of spike encoding schemes in asynchronous vision sensors: Modeling and design," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2014, pp. 2632–2635.

[79] T. Delbruck, "Frame-free dynamic digital vision," in *Proc. Int. Symp. Secure-Life Electron.*, 2008, pp. 21–26.

[80] A. Khodamoradi and R. Kastner, "O(N)-space spatiotemporal filter for reducing noise in neuromorphic vision sensors," *IEEE Trans. Emerg. Topics Comput.*, vol. PP, no. 99, pp. 1–1, 2018.

[81] D. Czech and G. Orchard, "Evaluating noise filtering for event-based asynchronous change detection image sensors," in *IEEE Int. Conf. Biomed. Robot. and Biomechatron. (BioRob)*, 2016, pp. 19–24.

[82] V. Padala, A. Basu, and G. Orchard, "A noise filtering algorithm for event-based asynchronous change detection image sensors on TrueNorth and its implementation on TrueNorth," *Front. Neurosci.*, vol. 12, 2018.

[83] T. Delbruck, V. Villanueva, and L. Longinotti, "Integration of dynamic vision sensor with inertial measurement unit for electronically stabilized event-based vision," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2014, pp. 2636–2639.

[84] R. Berner, "Highspeed USB2.0 AER interfaces," Master's thesis, ETH Zurich, Dept. of Electrical and Information Eng. (D-ITET), Zurich, Switzerland, 2006.

[85] G. Taverni, D. P. Moeys, C. Li, C. Cavaco, V. Motsnyi, D. S. S. Bello, and T. Delbruck, "Front and back illuminated Dynamic and Active Pixel Vision Sensors comparison," *IEEE Trans. Circuits Syst. II*, vol. 65, no. 5, pp. 677–681, 2018.

[86] D. B. Fasnacht and T. Delbruck, "Dichromatic spectral measurement circuit in vanilla CMOS," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2007, pp. 3091–3094.

[87] R. Berner, P. Lichtsteiner, and T. Delbruck, "Self-timed vertacolor dichromatic vision sensor for low power pattern detection," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2008, pp. 1032–1035.

[88] L. Farian, J. A. Leñero-Bardallo, and P. Häfliger, "A bio-inspired AER temporal tri-color differentiator pixel array," *IEEE Trans. Biomed. Circuits Syst.*, vol. 9, no. 5, pp. 686–698, Oct. 2015.

[89] R. B. Merrill, "Color separation in an active pixel cell imaging array using a triple-well structure," US Patent US5 965 875A, 1999.

[90] R. F. Lyon and P. M. Hubel, "Eyeing the camera: Into the next century," in *Proc. IS&T/SID 10th Color Imaging Conf.*, 2002, pp. 349–355.

[91] C. Li, C. Brandli, R. Berner, H. Liu, M. Yang, S.-C. Liu, and T. Delbruck, "Design of an RGBW color VGA rolling and global shutter dynamic and active-pixel vision sensor," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2015.

[92] D. P. Moeys, C. Li, J. N. P. Martel, S. Bamford, L. Longinotti, V. Motsnyi, D. S. S. Bello, and T. Delbruck, "Color temporal contrast sensitivity in dynamic vision sensors," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2017, pp. 1–4.

[93] C. Scheerlinck, H. Rebecq, T. Stoffregen, N. Barnes, R. Mahony, and D. Scaramuzza, "CED: color event camera dataset," in *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, 2019.

[94] A. Marcireau, S.-H. Ieng, C. Simon-Chane, and R. B. Benosman, "Event-based color segmentation with a high dynamic range sensor," *Front. Neurosci.*, vol. 12, 2018.

[95] C. Posch, D. Matolin, R. Wohlgenannt, T. Maier, and M. Litzenberger, "A microbolometer asynchronous Dynamic Vision Sensor for LWIR," *IEEE Sensors J.*, vol. 9, no. 6, pp. 654–664, Jun. 2009.

[96] D. Weikersdorfer and J. Conradt, "Event-based particle filtering for robot self-localization," in *IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, 2012.

[97] F. Paredes-Valles, K. Y. W. Scheper, and G. C. H. E. de Croon, "Unsupervised learning of a hierarchical spiking neural network for optical flow estimation: From events to global motion perception," *IEEE Trans. Pattern Anal. Mach. Intell.*, Mar. 2019.

[98] C. Reinbacher, G. Munda, and T. Pock, "Real-time panoramic tracking for event cameras," in *IEEE Int. Conf. Comput. Photography (ICCP)*, 2017, pp. 1–9.

[99] E. Mueggler, G. Gallego, H. Rebecq, and D. Scaramuzza, "Continuous-time visual-inertial odometry for event cameras," *IEEE Trans. Robot.*, vol. 34, no. 6, pp. 1425–1440, Dec. 2018.

[100] J. Kogler, C. Sulzbachner, and W. Kubinger, "Bio-inspired stereo vision system with silicon retina imagers," in *Int. Conf. Comput. Vis. Syst. (ICVS)*, 2009, pp. 174–183.

[101] M. Liu and T. Delbruck, "Adaptive time-slice block-matching optical flow algorithm for dynamic vision sensors," in *British Mach. Vis. Conf. (BMVC)*, 2018.

[102] A. I. Maqueda, A. Loquercio, G. Gallego, N. García, and D. Scaramuzza, "Event-based vision meets deep learning on steering prediction for self-driving cars," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018.

[103] X. Lagorce, G. Orchard, F. Gallupi, B. E. Shi, and R. Benosman, "HOTS: A hierarchy of event-based time-surfaces for pattern recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1346–1359, 2017.

[104] M. A. R. Ahad, J. K. Tan, H. Kim, and S. Ishikawa, "Motion history image: its variants and applications," *Mach. Vis. and Applications*, vol. 23, no. 2, pp. 255–281, Mar. 2012.

[105] I. Alzugaray and M. Chli, "Asynchronous corner detection and tracking for event cameras in real time," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 3177–3184, Oct. 2018.

[106] J. Manderscheid, A. Sironi, N. Bourdis, D. Migliore, and V. Lepetit, "Speed invariant time surface for learning to detect corner points with event-based cameras," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.

[107] A. Sironi, M. Brambilla, N. Bourdis, X. Lagorce, and R. Benosman, "HATS: Histograms of averaged time surfaces for robust event-based object classification," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018.

[108] V. Vasco, A. Glover, and C. Bartolozzi, "Fast event-based Harris corner detection exploiting the advantages of event-driven cameras," in *IEEE Int. Conf. Intell. Robot. Syst. (IROS)*, 2016.

[109] E. Mueggler, C. Bartolozzi, and D. Scaramuzza, "Fast event-based corner detection," in *British Mach. Vis. Conf. (BMVC)*, 2017.

[110] Y. Zhou, G. Gallego, H. Rebecq, L. Kneip, H. Li, and D. Scaramuzza, "Semi-dense 3D reconstruction with a stereo event camera," in *Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 242–258.

[111] D. Gehrig, A. Loquercio, K. G. Derpanis, and D. Scaramuzza, "End-to-end learning of representations for asynchronous event-based data," in *Int. Conf. Comput. Vis. (ICCV)*, 2019.

[112] P. Bardow, A. J. Davison, and S. Leutenegger, "Simultaneous optical flow and intensity estimation from an event camera," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2016.

[113] S. Mostafavi I., L. Wang, Y.-S. Ho, and K.-J. Y. Yoon, "Event-based high dynamic range image and very high frame rate video generation using conditional generative adversarial networks," *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.

[114] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "Unsupervised event-based learning of optical flow, depth, and egomotion," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.

[115] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza, "Events-to-video: Bringing modern computer vision to event cameras," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.

[116] Y. Sekikawa, K. Hara, and H. Saito, "EventNet: Asynchronous recursive event processing," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.

[117] M. Litzenberger, C. Posch, D. Bauer, A. N. Belbachir, P. Schön, B. Kohn, and H. Garn, "Embedded vision system for real-time object tracking using an asynchronous transient vision sensor," in *Digital Signal Processing Workshop*, 2006, pp. 173–178.

[118] Z. Ni, A. Bolopion, J. Agnus, R. Benosman, and S. Régnier, "Asynchronous event-based visual shape tracking for stable haptic feedback in microrobotics," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1081–1089, 2012.

[119] Z. Ni, S.-H. Ieng, C. Posch, S. Régnier, and R. Benosman, "Visual tracking using neuromorphic asynchronous event-based cameras," *Neural Computation*, vol. 27, no. 4, pp. 925–953, 2015.

[120] D. Tedaldi, G. Gallego, E. Mueggler, and D. Scaramuzza, "Feature detection and tracking with the dynamic and active-pixel vision sensor (DAVIS)," in *Int. Conf. Event-Based Control, Comm. Signal Proc. (EBCCSP)*, 2016.

[121] B. Kueng, E. Mueggler, G. Gallego, and D. Scaramuzza, "Low-latency visual odometry using event-based feature tracks," in *IEEE Int. Conf. Intell. Robot. Syst. (IROS)*, 2016.

[122] G. Gallego and D. Scaramuzza, "Accurate angular velocity estimation with an event camera," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 632–639, 2017.

[123] G. Gallego, H. Rebecq, and D. Scaramuzza, "A unifying contrast maximization framework for event cameras, with applications to motion, depth, and optical flow estimation," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018.

[124] G. Gallego, M. Gehrig, and D. Scaramuzza, "Focus is all you need: Loss functions for event-based vision," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.

[125] H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization," in *British Mach. Vis. Conf. (BMVC)*, 2017.

[126] A. Z. Zhu, N. Atanasov, and K. Daniilidis, "Event-based feature tracking with probabilistic data association," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2017.

[127] ——, "Event-based visual inertial odometry," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017.

[128] A. Mitrokhin, C. Fermuller, C. Parameshwara, and Y. Aloimonos, "Event-based moving object detection and tracking," in *IEEE Int. Conf. Intell. Robot. Syst. (IROS)*, 2018.

[129] A. Mitrokhin, C. Ye, C. Fermuller, Y. Aloimonos, and T. Delbruck, "EV-IMO: Motion segmentation dataset and learning pipeline for event cameras," in *IEEE Int. Conf. Intell. Robot. Syst. (IROS)*, 2019.

[130] T. Brosch, S. Tschechne, and H. Neumann, "On event-based optical flow detection," *Front. Neurosci.*, vol. 9, Apr. 2015.

[131] C. Reinbacher, G. Graber, and T. Pock, "Real-time intensity-image reconstruction for event cameras using manifold regularisation," in *British Mach. Vis. Conf. (BMVC)*, 2016.

[132] H. Akolkar, S. Panzeri, and C. Bartolozzi, "Spike time based unsupervised learning of receptive fields for event-driven vision," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2015.

[133] J. A. Perez-Carrasco, B. Zhao, C. Serrano, B. Acha, T. Serrano-Gotarredona, S. Chen, and B. Linares-Barranco, "Mapping from frame-driven to frame-free event-driven vision systems by low-rate rate coding and coincidence processing–application to feed-forward ConvNets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 11, pp. 2706–2719, Nov. 2013.

[134] P. O'Connor, D. Neil, S.-C. Liu, T. Delbruck, and M. Pfeiffer, "Real-time classification and sensor fusion with a spiking deep belief network," *Front. Neurosci.*, vol. 7, p. 178, 2013.

[135] P. U. Diehl, D. Neil, J. Binas, M. Cook, S.-C. Liu, and M. Pfeiffer, "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 4, 2015, pp. 2933–2940.

[136] S. K. Esser, P. A. Merolla, J. V. Arthur, A. S. Cassidy, R. Appuswamy, A. Andreopoulos, D. J. Berg, J. L. McKinstry, T. Melano, D. R. Barch, C. di Nolfo, P. Datta, A. Amir, B. Taba, M. D. Flickner, and D. S. Modha, "Convolutional networks for fast, energy-efficient neuromorphic computing," *Proc. National Academy of Sciences*, vol. 113, no. 41, pp. 11 441–11 446, 2016.

[137] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Front. Neurosci.*, vol. 11, p. 682, 2017.

[138] S. B. Shrestha and G. Orchard, "SLAYER: Spike layer error reassignment in time," in *Conf. Neural Inf. Process. Syst. (NIPS)*, 2018.

[139] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training deep spiking neural networks using backpropagation," *Front. Neurosci.*, vol. 10, p. 508, 2016.

[140] E. Neftci, "Data and power efficient intelligence with neuromorphic learning machines," *iScience*, 2018.

[141] J. Kogler, C. Sulzbachner, M. Humenberger, and F. Eibensteiner, "Address-event based stereo vision with bio-inspired silicon retina imagers," in *Advances in Theory and Applications of Stereo Vision*. InTech, 2011, pp. 165–188.

[142] H. Li, G. Li, and L. Shi, "Classification of spatiotemporal events based on random forest," in *Advances in Brain Inspired Cognitive Systems (BICS)*, 2016.

[143] A. Nguyen, T. Do, D. G. Caldwell, and N. G. Tsagarakis, "Real-time 6DOF pose relocalization for event cameras with stacked spatial LSTM networks," in *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, 2019.

[144] E. Mueggler, C. Forster, N. Baumli, G. Gallego, and D. Scaramuzza, "Lifetime estimation of events from dynamic vision sensors," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2015.

[145] S.-H. Ieng, J. Carneiro, M. Osswald, and R. Benosman, "Neuromorphic event-based generalized time-based stereovision," *Front. Neurosci.*, vol. 12, p. 442, 2018.

[146] D. P. Moeys, F. Corradi, E. Kerr, P. Vance, G. Das, D. Neil, D. Kerr, and T. Delbruck, "Steering a predator robot using a mixed frame/event-driven convolutional neural network," in *Int. Conf. Event-Based Control, Comm. Signal Proc. (EBCCSP)*, 2016.

[147] I.-A. Lungu, F. Corradi, and T. Delbruck, "Live demonstration: Convolutional neural network driven by dynamic vision sensor playing RoShamBo," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2017.

[148] J. Binas, D. Neil, S.-C. Liu, and T. Delbruck, "DDD17: End-to-end DAVIS driving dataset," in *ICML Workshop on Machine Learning for Autonomous Vehicles*, 2017.

[149] C. Ye, A. Mitrokhin, C. Parameshwara, C. Fermüller, J. A. Yorke, and Y. Aloimonos, "Unsupervised learning of dense optical flow and depth from sparse event data," *arXiv e-prints*, 2018, 1809.08625.

[150] T. Stoffregen and L. Kleeman, "Simultaneous optical flow and segmentation (SOFAS) using Dynamic Vision Sensor," in *Australasian Conf. Robot. Autom. (ACRA)*, 2017.

[151] T. Stoffregen, G. Gallego, T. Drummond, L. Kleeman, and D. Scaramuzza, "Event-based motion segmentation by motion compensation," in *Int. Conf. Comput. Vis. (ICCV)*, 2019.

[152] L. A. Camunas-Mesa, T. Serrano-Gotarredona, and B. Linares-Barranco, "Event-driven sensing and processing for high-speed robotic vision," in *IEEE Biomed. Circuits Syst. Conf. (BioCAS)*, 2014.

[153] G. Orchard, R. Benosman, R. Etienne-Cummings, and N. V. Thakor, "A spiking neural network architecture for visual motion estimation," in *IEEE Biomed. Circuits Syst. Conf. (BioCAS)*, 2013.

[154] E. Chicca, P. Lichtsteiner, T. Delbruck, G. Indiveri, and R. Douglas, "Modeling orientation selectivity using a neuromorphic multi-chip system," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2006.

[155] R. L. D. Valois, N. P. Cottaris, L. E. Mahon, S. D. Elfar, and J. Wilson, "Spatial and temporal receptive fields of geniculate and cortical cells and directional selectivity," *Vision Research*, vol. 40, no. 27, pp. 3685–3702, 2000.

[156] F. Rea, G. Metta, and C. Bartolozzi, "Event-driven visual attention for the humanoid robot iCub," *Front. Neurosci.*, vol. 7, 2013.

[157] L. Itti and C. Koch, "Computational modelling of visual attention," *Nature Reviews Neuroscience*, vol. 2, no. 3, pp. 194–203, 2001.

[158] D. Marr and T. Poggio, "Cooperative computation of stereo disparity," *Science*, vol. 194, no. 4262, pp. 283–287, 1976.

[159] M. Mahowald, *The Silicon Retina*. Boston, MA: Springer US, 1994, pp. 4–65.

[160] M. Osswald, S.-H. Ieng, R. Benosman, and G. Indiveri, "A spiking neural network model of 3D perception for event-based neuromorphic stereo vision systems," *Sci Rep*, vol. 7, no. 1, Jan. 2017.

[161] G. Dikov, M. Firouzi, F. Röhrbein, J. Conradt, and C. Richter, "Spiking cooperative stereo-matching at 2ms latency with neuromorphic hardware," in *Conf. Biomimetic and Biohybrid Systems*, 2017, pp. 119–137.

[162] E. Piatkowska, A. N. Belbachir, and M. Gelautz, "Asynchronous stereo vision for event-driven dynamic stereo sensor using an adaptive cooperative approach," in *Int. Conf. Comput. Vis. Workshops (ICCVW)*, 2013.

[163] V. Vasco, A. Glover, Y. Tirupachuri, F. Solari, M. Chessa, and C. Bartolozzi, "Vergence control with a neuromorphic iCub," in *IEEE-RAS Int. Conf. Humanoid Robots (Humanoids)*, 2016.

[164] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, no. 11, pp. 1019–1025, Nov. 1999.

[165] H. Akolkar, C. Meyer, X. Clady, O. Marre, C. Bartolozzi, S. Panzeri, and R. Benosman, "What can neuromorphic event-driven precise timing add to spike-based pattern recognition?" *Neural Computation*, vol. 27, no. 3, pp. 561–593, Mar. 2015.

[166] L. A. Camunas-Mesa, T. Serrano-Gotarredona, S. H. Ieng, R. B. Benosman, and B. Linares-Barranco, "On the use of orientation filters for 3D reconstruction in event-driven stereo vision," *Front. Neurosci.*, vol. 8, p. 48, 2014.

[167] M. B. Milde, D. Neil, A. Aimar, T. Delbrück, and G. Indiveri, "ADaPTION: Toolbox and benchmark for training convolutional neural networks with reduced numerical precision weights and activation," *arXiv e-prints*, Nov. 2017.

[168] E. Stromatias, M. Soto, T. Serrano-Gotarredona, and B. Linares-Barranco, "An event-driven classifier for spiking neural networks fed with synthetic or dynamic vision sensor data," *Front. Neurosci.*, vol. 11, Jun. 2017.

[169] E. Neftci, C. Augustine, S. Paul, and G. Detorakis, "Event-driven random back-propagation: Enabling neuromorphic deep learning machines," *Front. Neurosci.*, vol. 11, p. 324, 2017.

[170] M. B. Milde, O. J. N. Bertrand, H. Ramachandran, M. Egelhaaf, and E. Chicca, "Spiking elementary motion detector in neuromorphic systems," *Neural Computation*, vol. 30, no. 9, pp. 2384–2417, Sep. 2018.

[171] H. Blum, A. Dietmller, M. Milde, J. Conradt, G. Indiveri, and Y. Sandamirskaya, "A neuromorphic controller for a robotic vehicle equipped with a dynamic vision sensor," in *Robotics: Science and Systems (RSS)*, 2017.

[172] L. Salt, D. Howard, G. Indiveri, and Y. Sandamirskaya, "Differential evolution and bayesian optimisation for hyper-parameter selection in mixed-signal neuromorphic circuits applied to UAV obstacle avoidance," *arXiv e-prints*, 2017, 1704.04853.

[173] D. Drazen, P. Lichtsteiner, P. Häfliger, T. Delbrück, and A. Jensen, "Toward real-time particle tracking using an event-based dynamic vision sensor," *Experim. Fluids*, vol. 51, no. 5, pp. 1465–1469, 2011.

[174] Z. Ni, C. Pacoret, R. Benosman, S.-H. Ieng, and S. Régnier, "Asynchronous event-based high speed vision for microparticle tracking," *J. Microscopy*, vol. 245, no. 3, pp. 236–244, 2012.

[175] E. Piatkowska, A. N. Belbachir, S. Schraml, and M. Gelautz, "Spatiotemporal multiple persons tracking using dynamic vision sensor," in *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, 2012.

[176] X. Lagorce, C. Meyer, S.-H. Ieng, D. Filliat, and R. Benosman, "Asynchronous event-based multikernel algorithm for high-speed visual features tracking," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 8, pp. 1710–1720, 2015.

[177] A. Glover and C. Bartolozzi, "Robust visual tracking with a freely-moving event camera," in *IEEE Int. Conf. Intell. Robot. Syst. (IROS)*, 2017.

[178] D. R. Valeiras, X. Lagorce, X. Clady, C. Bartolozzi, S.-H. Ieng, and R. Benosman, "An asynchronous neuromorphic event-driven visual part-based shape tracking," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 12, pp. 3045–3059, Dec. 2015.

[179] M. A. Fischler and R. A. Elschlager, "The representation and matching of pictorial structures," *IEEE Trans. Comput.*, vol. C-22, no. 1, pp. 67–92, Jan. 1973.

[180] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. Fourth Alvey Vision Conf.*, vol. 15, 1988, pp. 147–151.

[181] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Int. Joint Conf. Artificial Intell. (IJCAI)*, 1981.

[182] X. Clady, S.-H. Ieng, and R. Benosman, "Asynchronous event-based corner detection and matching," *Neural Netw.*, vol. 66, pp. 91–106, 2015.

[183] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Eur. Conf. Comput. Vis. (ECCV)*, 2006.

[184] V. Vasco, A. Glover, E. Mueggler, D. Scaramuzza, L. Natale, and C. Bartolozzi, "Independent motion detection with event-driven cameras," in *IEEE Int. Conf. Adv. Robot. (ICAR)*, 2017.

[185] Y. Hu, H. Liu, M. Pfeiffer, and T. Delbruck, "DVS benchmark datasets for object tracking, action recognition, and object recognition," *Front. Neurosci.*, vol. 10, p. 405, 2016.

[186] B. Rueckauer and T. Delbruck, "Evaluation of event-based algorithms for optical flow with ground-truth from inertial measurement sensor," *Front. Neurosci.*, vol. 10, no. 176, 2016.

[187] R. Benosman, S.-H. Ieng, C. Clercq, C. Bartolozzi, and M. Srinivasan, "Asynchronous frameless event-based optical flow," *Neural Netw.*, vol. 27, pp. 32–37, 2012.

[188] F. Barranco, C. Fermuller, and Y. Aloimonos, "Contour motion estimation for asynchronous event-driven cameras," *Proc. IEEE*, vol. 102, no. 10, pp. 1537–1556, Oct. 2014.

[189] G. Haessig, A. Cassidy, R. Alvarez-Icaza, R. Benosman, and G. Orchard, "Spiking optical flow for event-based sensors using IBM's truenorth neurosynaptic system," *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, no. 4, pp. 860–870, Aug. 2018.

[190] A. Z. Zhu, D. Thakur, T. Ozaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The multivehicle stereo event camera dataset: An event camera dataset for 3D perception," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2032–2039, Jul. 2018.

[191] F. Barranco, C. Fermuller, and Y. Aloimonos, "Bio-inspired motion estimation with event-driven sensors," in *Int. Work-Conf. Artificial Neural Netw. (IWANN), Advances in Comput. Intell.*, 2015, pp. 309–321.

[192] D. J. Fleet and A. D. Jepson, "Computation of component image velocity from local phase information," *Int. J. Comput. Vis.*, vol. 5, no. 1, pp. 77–104, 1990.

[193] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003, 2nd Edition.

[194] S. Schraml, A. N. Belbachir, N. Milosevic, and P. Schön, "Dynamic stereo vision system for real-time tracking," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2010, pp. 1409–1412.

[195] J. Kogler, M. Humenberger, and C. Sulzbachner, "Event-based stereo matching approaches for frameless address event stereo data," in *Int. Symp. Adv. Vis. Comput. (ISVC)*, 2011, pp. 674–685.

[196] J. Lee, T. Delbruck, P. K. J. Park, M. Pfeiffer, C.-W. Shin, H. Ryu, and B. C. Kang, "Live demonstration: Gesture-based remote control using stereo pair of dynamic vision sensors," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2012.

[197] E. Piatkowska, A. N. Belbachir, and M. Gelautz, "Cooperative and asynchronous stereo vision for dynamic vision sensors," *Meas. Sci. Technol.*, vol. 25, no. 5, p. 055108, Apr. 2014.

[198] R. Benosman, S.-H. Ieng, P. Rogister, and C. Posch, "Asynchronous event-based Hebbian epipolar geometry," *IEEE Trans. Neural Netw.*, vol. 22, no. 11, pp. 1723–1734, 2011.

[199] J. Carneiro, S.-H. Ieng, C. Posch, and R. Benosman, "Event-based 3D reconstruction from neuromorphic retinas," *Neural Netw.*, vol. 45, pp. 27–38, 2013.

[200] D. Zou, P. Guo, Q. Wang, X. Wang, G. Shao, F. Shi, J. Li, and P.-K. J. Park, "Context-aware event-driven stereo matching," in *IEEE Int. Conf. Image Process. (ICIP)*, 2016, pp. 1076–1080.

[201] D. Zou, F. Shi, W. Liu, J. Li, Q. Wang, P.-K. J. Park, C.-W. Shi, Y. J. Roh, and H. E. Ryu, "Robust dense depth map estimation from sparse DVS stereos," in *British Mach. Vis. Conf. (BMVC)*, 2017.

[202] M. Firouzi and J. Conradt, "Asynchronous event-based cooperative stereo matching using neuromorphic silicon retinas," *Neural Proc. Lett.*, vol. 43, no. 2, pp. 311–326, 2016.

[203] J. Kogler, F. Eibensteiner, M. Humenberger, C. Sulzbachner, M. Gelautz, and J. Scharinger, "Enhancement of sparse silicon retina-based stereo matching using belief propagation and two-stage postfiltering," *J. Electron. Imag.*, vol. 23, no. 4, pp. 1–15, 2014.

[204] Z. Xie, S. Chen, and G. Orchard, "Event-based stereo depth estimation using belief propagation," *Front. Neurosci.*, vol. 11, 2017.

[205] Z. Xie, J. Zhang, and P. Wang, "Event-based stereo matching using semiglobal matching," *Int. J. Advanced Robotic Systems*, vol. 15, no. 1, 2018.

[206] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 328–341, Feb. 2008.

[207] A. Andreopoulos, H. J. Kashyap, T. K. Nayak, A. Amir, and M. D. Flickner, "A low power, high throughput, fully event-based stereo system," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018.

[208] A. Z. Zhu, Y. Chen, and K. Daniilidis, "Realtime time synchronized event-based stereo," in *Eur. Conf. Comput. Vis. (ECCV)*, 2018.

[209] R. Szeliski, *Computer Vision: Algorithms and Applications*, ser. Texts in Computer Science. Springer, 2010.

[210] S. Schraml, A. N. Belbachir, and H. Bischof, "An event-driven stereo system for real-time 3-D 360 panoramic vision," *IEEE Trans. Ind. Electron.*, vol. 63, no. 1, pp. 418–428, Jan. 2016.

[211] ——, "Event-driven stereo matching for real-time 3D panoramic vision," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2015, pp. 466–474.

[212] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. D. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, 2016.

[213] R. T. Collins, "A space-sweep approach to true multi-image matching," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 1996.

[214] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *Int. Conf. Comput. Vis. (ICCV)*, 2011, pp. 2320–2327.

[215] C. Brandli, T. Mantel, M. Hutter, M. Höpflinger, R. Berner, R. Siegwart, and T. Delbruck, "Adaptive pulsed laser line extraction for terrain reconstruction using a dynamic vision sensor," *Front. Neurosci.*, vol. 7, p. 275, 2014.

[216] J. N. P. Martel, J. Müller, J. Conradt, and Y. Sandamirskaya, "An active approach to solving the stereo matching problem using

event-based sensors," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2018.

[217] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, Cambridge, MA, 2005.

[218] D. Weikersdorfer, R. Hoffmann, and J. Conradt, "Simultaneous localization and mapping for event-based vision systems," in *Int. Conf. Comput. Vis. Syst. (ICVS)*, 2013.

[219] D. Weikersdorfer, D. B. Adrian, D. Cremers, and J. Conradt, "Event-based 3D SLAM with a depth-augmented dynamic vision sensor," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2014.

[220] E. Mueggler, B. Huber, and D. Scaramuzza, "Event-based, 6-DOF pose tracking for high-speed maneuvers," in *IEEE Int. Conf. Intell. Robot. Syst. (IROS)*, 2014, pp. 2761–2768.

[221] M. Milford, H. Kim, S. Leutenegger, and A. Davison, "Towards visual SLAM with event-based cameras," in *The Problem of Mobile Sensors Workshop in conjunction with RSS*, 2015.

[222] E. Mueggler, G. Gallego, and D. Scaramuzza, "Continuous-time trajectory estimation for event-based vision sensors," in *Robotics: Science and Systems (RSS)*, 2015.

[223] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM," *Int. J. Robot. Research*, vol. 36, no. 2, pp. 142–149, 2017.

[224] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual-inertial odometry," *IEEE Trans. Robot.*, vol. 33, no. 1, pp. 1–21, 2017.

[225] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, Apr. 2007, pp. 3565–3572.

[226] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial SLAM using nonlinear optimization," *Int. J. Robot. Research*, 2015.

[227] A. Patron-Perez, S. Lovegrove, and G. Sibley, "A spline-based trajectory representation for sensor fusion and rolling shutter cameras," *Int. J. Comput. Vis.*, vol. 113, no. 3, pp. 208–219, 2015.

[228] L. von Stumberg, V. Usenko, and D. Cremers, "Direct sparse visual-inertial odometry using dynamic marginalization," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2018.

[229] S. Barua, Y. Miyatani, and A. Veeraraghavan, "Direct face detection and video reconstruction from event cameras," in *IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, 2016, pp. 1–9.

[230] G. Munda, C. Reinbacher, and T. Pock, "Real-time intensity-image reconstruction for event cameras using manifold regularisation," *Int. J. Comput. Vis.*, vol. 126, no. 12, pp. 1381–1393, 2018.

[231] A. N. Belbachir, S. Schraml, M. Mayerhofer, and M. Hofstaetter, "A novel HDR depth camera for real-time 3D 360-degree panoramic vision," in *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, 2014.

[232] G. Wiesmann, S. Schraml, M. Litzenberger, A. N. Belbachir, M. Hofstatter, and C. Bartolozzi, "Event-driven embodied system for feature extraction and object recognition in robotic applications," in *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, 2012.

[233] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Front. Neurosci.*, vol. 9, p. 437, 2015.

[234] D. Neil and S.-C. Liu, "Effective sensor fusion with event-based sensors and deep network architectures," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2016, pp. 2282–2285.

[235] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," *Front. Neurosci.*, vol. 12, 2018.

[236] A. Yousefzadeh, G. Orchard, T. Serrano-Gotarredona, and B. Linares-Barranco, "Active perception with dynamic vision sensors. minimum saccades with optimum recognition," *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, no. 4, pp. 927–939, Aug. 2018.

[237] X. Clady, J.-M. Maro, S. Barré, and R. B. Benosman, "A motion-based feature for event-based pattern recognition," *Front. Neurosci.*, vol. 10, Jan. 2017.

[238] C. A. Sims, "Implications of rational inattention," *J. Monetary Economics*, vol. 50, pp. 665–690, 2003.

[239] M. Miskowicz, *Event-Based Control and Signal Processing*, ser. Embedded Systems. CRC Press, 2018.

[240] P. Tabuada, "An introduction to event-triggered and self-triggered control," in *IEEE Conf. Decision Control (CDC)*, 2012.

[241] K. J. Aström, *Event Based Control*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 127–147.

[242] B. Wang and M. Fu, "Comparison of periodic and event-based sampling for linear state estimation," *IFAC Proc. Volumes (IFAC-PapersOnline)*, vol. 19, pp. 5508–5513, 2014.

[243] A. Censi, E. Mueller, E. Frazzoli, and S. Soatto, "A power-performance approach to comparing sensor families, with application to comparing neuromorphic to traditional vision sensors," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2015.

[244] E. Mueller, A. Censi, and E. Frazzoli, "Low-latency heading feedback control with neuromorphic vision sensors using efficient approximated incremental inference," in *IEEE Conf. Decision Control (CDC)*, 2015.

[245] P. Singh, S. Z. Yong, J. Gregoire, A. Censi, and E. Frazzoli, "Stabilization of linear continuous-time systems using neuromorphic vision sensors," in *IEEE Conf. Decision Control (CDC)*, 2016.

[246] A. Censi, "Efficient neuromorphic optomotor heading regulation," in *IEEE Am. Control Conf. (ACC)*, 2015.

[247] E. Mueller, A. Censi, and E. Frazzoli, "Efficient high speed signal estimation with neuromorphic vision sensors," in *Int. Conf. Event-Based Control, Comm. Signal Proc. (EBCCSP)*, 2015.

[248] D. Kahneman, *Thinking, fast and slow*. Farrar, Straus & Giroux, 2011.

[249] S. B. Furber, D. R. Lester, L. A. Plana, J. D. Garside, E. Painkras, S. Temple, and A. D. Brown, "Overview of the SpiNNaker system architecture," *IEEE Trans. Comput.*, vol. 62, no. 12, pp. 2454–2467, 2013.

[250] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam, B. Taba, M. Beakes, B. Brezzo, J. B. Kuang, R. Manohar, W. P. Risk, B. Jackson, and D. S. Modha, "TrueNorth: Design and tool flow of a 65 mW 1 million neuron programmable neurosynaptic chip," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 10, pp. 1537–1557, 2015.

[251] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[252] S. Moradi, N. Qiao, F. Stefanini, and G. Indiveri, "A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic asynchronous processors (DYNAPs)," *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, no. 1, pp. 106–122, 2018.

[253] A. S. Neckar, "Braindrop: a mixed signal neuromorphic architecture with a dynamical systems-based programming model," Ph.D. dissertation, Stanford University, Stanford, CA, Jun. 2018.

[254] L. A. Camunas-Mesa, B. Linares-Barranco, and T. Serrano-Gotarredona, "Neuromorphic spiking neural networks and their memristor-CMOS hardware implementations," *Materials*, vol. 12, no. 17, p. 2745, Aug. 2019.

[255] B. Rajendran, A. Sebastian, M. Schmuker, N. Srinivasa, and E. Eleftheriou, "Low-power neuromorphic hardware for signal processing applications: A review of architectural and system-level design approaches," *IEEE Signal Process. Mag.*, vol. 36, no. 6, pp. 97–110, Nov 2019.

[256] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The SpiNNaker Project," *Proc. IEEE*, vol. 102, no. 5, pp. 652–665, 2014.

[257] G. Haessig, F. Galluppi, X. Lagorce, and R. Benosman, "Neuromorphic networks on the SpiNNaker platform," in *IEEE Int. Conf. Artificial Intell. Circuits Syst. (AICAS)*, 2019.

[258] C. Richter, F. Röhrbein, and J. Conradt, "Bio inspired optic flow detection using neuromorphic hardware," in *Bernstein Conf.*, 2014.

[259] A. Glover, A. B. Stokes, S. Furber, and C. Bartolozzi, "ATIS + SpiNNaker: a fully event-based visual tracking demonstration," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. Workshops (IROSW)*, 2018.

[260] T. Serrano-Gotarredona, B. Linares-Barranco, F. Galluppi, L. Plana, and S. Furber, "ConvNets experiments on SpiNNaker," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2015.

[261] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[262] P. Blouw, X. Choo, E. Hunsberger, and C. Eliasmith, "Benchmarking Keyword Spotting Efficiency on Neuromorphic Hardware," *arXiv e-prints*, p. arXiv:1812.01739, Dec. 2018.

[263] T. Bekolay, J. Bergstra, E. Hunsberger, T. DeWolf, T. Stewart, D. Rasmussen, X. Choo, A. Voelker, and C. Eliasmith, "Nengo:

a Python tool for building large-scale functional brain models," *Front. Neuroinf.*, vol. 7, p. 48, 2014.
[264] C. Eliasmith, T. C. Stewart, X. Choo, T. Bekolay, T. DeWolf, Y. Tang, and D. Rasmussen, "A large-scale model of the functioning brain," *Science*, vol. 338, no. 6111, pp. 1202–1205, 2012.
[265] N. Waniek, J. Biedermann, and J. Conradt, "Cooperative SLAM on small mobile robots," in *IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, 2015.
[266] B. J. P. Hordijk, K. Y. Scheper, and G. C. D. Croon, "Vertical landing for micro air vehicles using event-based optical flow," *J. Field Robot.*, vol. 35, no. 1, pp. 69–90, Jan. 2017.
[267] jAER Project, https://github.com/SensorsINI/jaer, 2007.
[268] T. Delbruck, M. Pfeiffer, R. Juston, G. Orchard, E. Müggler, A. Linares-Barranco, and M. W. Tilden, "Human vs. computer slot car racing using an event and frame-based DAVIS vision sensor," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2015.
[269] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *ICRA Workshop Open Source Softw.*, 2009.
[270] A. Glover, V. Vasco, M. Iacono, and C. Bartolozzi, "The event-driven software library for YARP-with algorithms and iCub applications," *Front. Robotics and AI*, vol. 4, p. 73, 2018.
[271] F. Barranco, C. Fermuller, Y. Aloimonos, and T. Delbruck, "A dataset for visual navigation with neuromorphic methods," *Front. Neurosci.*, vol. 10, p. 49, 2016.
[272] J. Delmerico, T. Cieslewski, H. Rebecq, M. Faessler, and D. Scaramuzza, "Are we ready for autonomous drone racing? the UZH-FPV drone racing dataset," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2019.
[273] C. Tan, S. Lallee, and G. Orchard, "Benchmarking neuromorphic vision: lessons learnt from computer vision," *Front. Neurosci.*, vol. 9, p. 374, 2015.
[274] S. Miao, G. Chen, X. Ning, Y. Zi, K. Ren, Z. Bing, and A. Knoll, "Neuromorphic vision datasets for pedestrian detection, action recognition, and fall detection," *Front. Neurorobot.*, 2019.
[275] E. Calabrese, G. Taverni, C. Awai Easthope, S. Skriabine, F. Corradi, L. Longinotti, K. Eng, and T. Delbruck, "DHP19: Dynamic vision sensor 3D human pose dataset," in *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, 2019.
[276] G. P. Garcia, P. Camilleri, Q. Liu, and S. B. Furber, "pyDVS: An extensible, real-time dynamic vision sensor emulator using off-the-shelf hardware," in *IEEE Symp. Series Comput. Intell. (SSCI)*, 2016.
[277] S. J. Carey, A. Lopich, D. R. Barr, B. Wang, and P. Dudek, "A 100,000 fps vision sensor with embedded 535 GOPS/W 256x256 SIMD processor array," in *VLSI Circuits Symp.*, 2013.