

Asynchronous Neuromorphic Event-Driven Image Filtering

Sparse coding of spatio-temporal signals lowers computational cost and raises the efficiency of visual processing.

By SIO-HOI IENG, CHRISTOPH POSCH, *Senior Member IEEE*, AND RYAD BENOSMAN

ABSTRACT | This paper introduces a new methodology to process asynchronously sampled image data captured by a new generation of biomimetic vision sensors. Unlike conventional cameras, these neuromorphic sensors acquire data not at fixed points in time for the entire array (frame-based) but sparse in space and time, i.e., pixel-individually and precisely timed only if new information is available (event-based). In this paper, we introduce a filtering methodology for asynchronously acquired gray-level data from an event-driven time-encoding imager. The paper first studies the properties of level-crossing sampling parameters in order to define threshold level properties and associated bandwidth needs. In a second stage, we introduce asynchronous linear and nonlinear filtering techniques. Examples are shown and examined on real data. Finally, the paper introduces a methodology to compare frame-based versus event-based computational costs. Implementations and experiments show that event-based gray-level filtering produces equivalent filtering accuracy as compared to frame-based ones. The main result of this work shows that, based on the number of operations to be carried out, beyond 3 frames per second (fps), event-based processing outperforms frame-based processing in terms of computational cost.

KEYWORDS | Asynchronous filtering; computer vision; event-based imaging; filtering algorithms; image filtering; image processing; level-crossing sampling; neuromorphic vision

I. INTRODUCTION

Conventional imaging devices are built to sample scenes at a fixed frequency involving all pixels of the sensing device. This process naturally leads to high computational cost and

inefficient use of resources. If scene dynamics are known in advance and the system obeys the Shannon–Nyquist sampling theorem, often only very few pixels change between two consecutive frames, leading to the acquisition of large amounts of redundant data. This temporal inefficiency is most striking when the system is dealing with temporarily static or slow changing scenes. Given the frame structure of the acquired image data, all existing image processing techniques operate on entire images, thus often handling information already processed in previous frames.

An alternative to acquisition using a fixed frequency is to sample a time-varying signal not on the time axis but the amplitude axis, leading to nonuniform sampling rates that match the dynamics of the input signal (Fig. 1). This sampling approach is often referred to as asynchronous delta modulation [2] or continuous-time level-crossing sampling [3].

Recently, this sampling paradigm has advanced from the recording of 1-D signals to the real-time acquisition of 2-D image data. The asynchronous time-based image sensor (ATIS) described in [1] contains an array of autonomously operating pixels that combine an asynchronous level-crossing detector and an exposure measurement circuit. Each exposure measurement by an individual pixel is triggered by a level-crossing event. Hence, each pixel independently samples its illuminance, through an integrative measurement, upon detection of a change of a certain magnitude in this same illuminance, so establishing its instantaneous gray level after it has changed. The result of the exposure measurement (i.e., the new gray level) is asynchronously transmitted off the sensor together with the pixel's x, y -address. As a result, image information is not acquired frame-wise but conditionally only from parts in the scene where there is new information. Or in other words, only information that is relevant—because unknown—is acquired and transmitted and needs to be processed.

Manuscript received May 20, 2014; revised July 25, 2014; accepted July 29, 2014. Date of publication September 11, 2014; date of current version September 16, 2014. The authors are with the Institut de la Vision, University Pierre and Marie Curie, Paris 75012, France (e-mail: sio-hoi.ieng@upmc.fr).

Digital Object Identifier: 10.1109/JPROC.2014.2347355

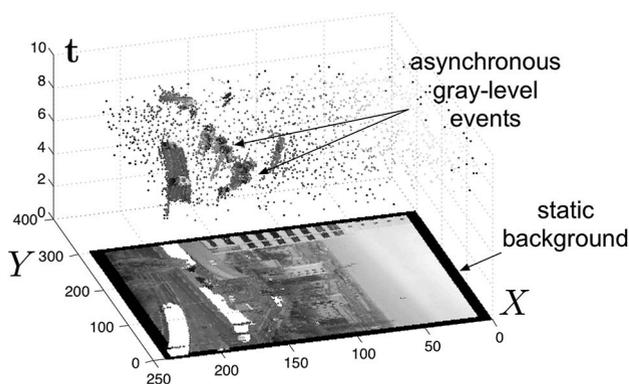


Fig. 1. Spatio-temporal space of imaging events: static objects and scene background are acquired first. Then, dynamic objects trigger pixel-individual, asynchronous gray-level events after each change. Frames are absent from this acquisition process.

Fig. 1 shows the general principle of asynchronous imaging spaces. As discussed above, frames are absent from this acquisition process. However, they can be reconstructed, when needed, at frequencies limited only by the temporal resolution of the pixel circuits. Static objects and background information, if required, can be recorded as a snapshot at the start of an acquisition. From then on, only pixel positions that have undergone a change and have consequently been remeasured are updated [1]. Moving objects in the visual scene describe a spatio-temporal surface at very high temporal resolution.

This novel paradigm of visual data acquisition calls for a new methodology in order to efficiently process the sparse, event-based image information without sacrificing its beneficial characteristics.

This paper presents a methodology for filtering asynchronously acquired gray-level data in a way to preserve as much as possible the temporal accuracy provided by the event-driven acquisition. The aim is to process each incoming event, rather than processing an entire frame. Although it would be possible to recreate frames from the event-driven sampling data and to apply classic filtering techniques, this approach, as we will show, is not useful as it removes all benefits of the event-driven mechanism, i.e., low data volume at high temporal resolution. In this work, linear and nonlinear filtering techniques are introduced and several standard filters are implemented and evaluated in order to assess their ability to process nonuniformly sampled data. Comparisons with conventional image-based processing are provided to emphasize the difference, especially in terms of computational costs.

II. PREVIOUS WORK

A. Asynchronous Image Sensors

Silicon retinas have been pioneered by the work of Mahowald [4]. Several artificial retinas based on the

address-event representation (AER) [5] have been built such as the ones presented in [6]–[15].

The latest generation silicon retina sensor ATIS used in this work is a time-domain encoding image sensors with 304×240 pixel resolution [1]. The sensor contains an array of fully autonomous pixels that combine an illuminance change detector circuit and a conditional exposure measurement block. The change detector individually and asynchronously initiates the measurement of an exposure/grayscale value only if—and immediately after—a brightness change of a certain magnitude has been detected in the field of view of the respective pixel. The exposure measurement circuit in each pixel individually encodes the absolute instantaneous pixel illuminance into the timing of asynchronous event pulses, more precisely into interevent intervals. Since the ATIS is not clocked like conventional cameras, the timing of events can be conveyed with a very accurate temporal resolution at the order of microseconds. The time-domain encoding of the intensity information automatically optimizes the exposure time separately for each pixel instead of imposing a fixed integration time for the entire array, resulting in exceptionally high dynamic range and improved signal-to-noise ratio. The pixel-individual change-detector-driven operation yields almost ideal temporal redundancy suppression, resulting in maximally sparse encoding of the image data.

B. Asynchronous Filtering

Signal filtering is generally applied before initiating more advanced processing techniques. In image processing, it is useful for feature detection and extraction (corners, edges, etc.), for preprocessing operations (signal smoothing, contrast enhancing, etc.), or for prediction purposes. Asynchronous, continuous-time signal coding and processing based on nonuniform sampling has been widely studied (e.g., [3] and [16]). One of the earliest applications of the level-crossing sampling in image processing can be found in [17] where the technique has been applied to compress binary images. However, filtering operations have been explored mainly for 1-D signals. So far, results have been obtained for synthesis of finite impulse response (FIR) and infinite impulse response (IIR) filters. In [18], the filter impulse response is sampled via interpolation at the sampling times of the asynchronous signal to filter. The resulting system has a more complex structure than conventional filtering schemes; however, this is compensated by the sparsity of the signal. In [19], the state-space presentation is used to avoid the need to resample the filter impulse response. In [20], a monodimensional convolution product algorithm is presented. The idea is to interpolate the two signals to allow identical sampling times for both of them. As a result conventional convolutions can be used. The main advantage is the possibility to convolute the input signal with any filter defined by sampled impulse response. One rare achievement on

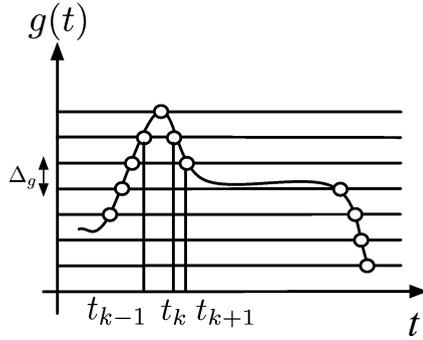


Fig. 2. Level crossing sampling.

event-based visual signal filtering has been implemented on a very large scale integration (VLSI) architecture [21]. Local convolutions are applied to active pixels based on the assumption that the 2-D filter is separable into two monodimensional convolutions. The pixel intensities are computed by event integration over time before being processed by the filter. However, this filter architecture is only suitable for linear filtering.

III. LEVEL-CROSSING SAMPLING

An event in the context of this work is defined as the pair “gray-level value and its timestamp” at a given pixel coordinate. It is a sample of the luminance signal captured at the level of a single pixel along with its “exact” time of acquisition. The notion of event is important in emphasizing the scene-driven sampling process. Let us assume a signal expressed as a real function g , of time t . An event is then a pair $(t_k, g(t_k))$, and each sample is acquired according to the generic rule

$$\forall k \in \mathbb{N}, \quad g(t_{k+1}) = f(g(t_k)) \quad (1)$$

where f is an adequately chosen constraint on the signal amplitude g . A frequently used f is the affine function

$$f(g(t_k)) = g(t_k) \pm \Delta_g. \quad (2)$$

This means that each sample $g(t_k)$ is determined by adding or subtracting a constant positive term Δ_g to the previous one $g(t_{k-1})$. Fig. 2 shows an example waveform and resulting sampling points in time where f is an affine function.

Considering the function implemented in several neuromorphic sensors, f is defined as

$$f(g(t_k)) = e^{ag(t_k)} \quad (3)$$

where $a \in \mathbb{R}$ and $e^a = \exp(a)$. a is positive (respectively negative) if g is increasing (respectively decreasing). This equality is interesting as it says that each sample is just a multiple of the previous one, or equivalently, the k th sample is captured each time g increases or decreases by a factor of e^a .

We focus on (3) as it allows to compress large signal variations into a smaller domain and to encode a larger dynamic range. The sampling accuracy increases as e^a is approaching 1 in (3). Since e^a cannot be infinitely close to 1, this sampling operation is behaving as a low-pass temporal filter. Its cutoff frequency is related to the signal amplitude since the sampling accuracy decreases with increasing signal amplitude and to the threshold value a that sets the sampling resolution in the amplitude dimension. Without loss of generality, we can consider one single pixel i that measures the light intensity g_i . The k th sampled value $g_i(t_k)$ satisfies

$$g_i(t_k) = e^a g_i(t_{k-1}). \quad (4)$$

This is equivalent to detecting a constant change in log intensity as used in both the dynamic vision sensor (DVS) [13] and the ATIS [1] sensors. The sequence of sampling times t_k , with t_0 being the time of the first sample, depends on the threshold a and the signal g_i . The sampling of $\log(g_i)$ is slightly different from a standard level-crossing sampling as we do not predefine the set of levels to cross. However, we show the ordinate axes using a semi log scale in Fig. 2 to ease comprehension. The instantaneous sampling rate is defined by the time interval between two consecutive samples, hence the highest frequency component one can recover from the sampled signal limited by the smallest sampling frequency

$$F_e = \min_k \left\{ \frac{1}{\Delta t_k} \right\} \quad (5)$$

with $\Delta t_k = t_k - t_{k-1}$.

This is equivalent to look for the maximum time intervals $t_k - t_{k-1}$. If we note g_i^{-1} the inverse function of g_i , then we have to look for the maximum of the sequence

$$\Delta t_k = t_k - t_{k-1} = g_i^{-1}(e^a g_i(t_{k-1})) - t_{k-1}. \quad (6)$$

If $D = (g_i^{-1} \circ e^a g_i) - Id$, with Id being the identity function, then finding the maximum of $\{\Delta t_k\}$ is equivalent to maximizing D .

g is, in general, not invertible as it is, in general, not monotonic over its definition domain. However, if we observe g over a small neighborhood, it is, in practice, possible to define a local inverse function of g .

Let us examine the case of a monospectral signal $g_i(t) = A \cos(\omega_0 t) + B$ to derive a closed-form relation linking the signal parameter and a to F_e . Given the i th pixel, the event-based sampling operation produces samples satisfying

$$\forall k \in \mathbb{N}, \quad \left| \log \left(\frac{g_i(t_k)}{g_i(t_{k-1})} \right) \right| = a \quad (7)$$

with a now assumed being positive. We thus have either $g_i(t_k) = e^a g_i(t_{k-1})$ or $g_i(t_k) = e^{-a} g_i(t_{k-1})$ if $\{g_i(t_k)\}$ is either increasing or decreasing. There is no definite cutoff frequency as we are used to when dealing with constant rate sampled signals. For such asynchronous sampled signals, if F_e is the smallest sampling rate, then we define the filter cutoff frequency $F_c = F_e/2$. This choice is legitimated by the fact that frequency components are the most suppressed for parts of the signal sampled at the lowest rate. This is equivalent to look for the largest time interval between two consecutive samples: $\max_k \{\Delta t_k\}$.

A. Sampling Cutoff Frequency

According to (6), we examine the case of a mono-spectral signal $g_i(t) = A \cos(\omega_0 t) + B$, with $T = 2\pi/\omega_0$.

We assume the constants $0 \leq A \leq B$ to ease the understanding of the calculation. For other values of A and B , the principle is the same, but the analysis needs to be split into subintervals, defined by the zero crossings of the function itself and the zero crossings of its first derivative.

We introduce the two following functions: d^- and d^+ . They are the inverses of the intensity function g_i . g_i depends on t , therefore to determine the time t of every incoming event, we need to invert g_i . The inverse of g_i , d^- , and d^+ are expressed as recursive functions to follow the acquisition principle of the ATIS that uses relative contrast changes. Because the cosine function has different inverse functions depending on its monotonicity, we set d^- as the inverse of g_i over the intervals $[0, T/2[$ while d^+ is the one over $]T/2, T]$

$$d^- : \left[0, \frac{T}{2}\right[\rightarrow \left[0, \frac{T}{2}\right[$$

$$t \mapsto d^-(t) = \frac{1}{\omega_0} \cos^{-1} \left(\frac{e^{-a}(A \cos(\omega_0 t) + B) - B}{A} \right) \quad (8)$$

and

$$d^+ : \left]\frac{T}{2}, T\right] \rightarrow \left]\frac{T}{2}, T\right]$$

$$t \mapsto d^+(t) = T - \frac{1}{\omega_0} \cos^{-1} \left(\frac{e^a(A \cos(\omega_0 t) + B) - B}{A} \right). \quad (9)$$

Both functions are real valued if the argument of \cos^{-1} is within the interval $[-1, 1]$, i.e.,

$$-1 \leq \frac{e^{-a}(A \cos(\omega_0 t) + B) - B}{A} \leq 1. \quad (10)$$

Since \cos^{-1} is a decreasing function of t for $t \in [0, T/2]$, the upper bound of the interval of definition of d^- is

reached when the argument is equal to -1 (i.e., $d^-(t) = T/2$). This value is equal to

$$t^- = \frac{1}{\omega_0} \cos^{-1} \left(\frac{e^a(B - A) - B}{A} \right). \quad (11)$$

The same explanation is valid for d^+ with the upper bound being reached when the argument is equal to 1 (i.e., $d^+(t) = T$)

$$t^+ = \frac{1}{\omega_0} \cos^{-1} \left(\frac{e^{-a}(B + A) - B}{A} \right). \quad (12)$$

With these functions, it is possible to calculate iteratively the timestamps of the sampling satisfying (7)

$$\forall k \in \mathbb{N}, \quad t_k = \begin{cases} d^-(t_{k-1}), & \text{if } t_{k-1} \in [0, t^-] \\ d^+(t_{k-1}), & \text{if } t_{k-1} \in]\frac{T}{2}, t^+]. \end{cases} \quad (13)$$

The way t_k is defined, there is no sample in the intervals $]t^-, T/2[$ and $]t^+, T]$ since d^- and d^+ are not defined over these two intervals.

For a correct sampling of g , an intuitive choice is to have a sufficiently small. The bandwidth of the sampling operation is related to the value of a . Thus, we are looking for a direct relation between the cutoff frequency and the value of a .

Let us define the sequence $\{\Delta t_k\}_{k \in \mathbb{N}} = \{t_k - t_{k-1}\}_{k \in \mathbb{N}}$, and the function D

$$D : \mathbb{R} \rightarrow [0, t^- \cup]T/2, t^+]$$

$$t \mapsto D(t) \quad (14)$$

with

$$D(t) = \begin{cases} d^-(t) - t, & \text{if } t \in [0, t^-[\\ d^+(t) - t, & \text{if } t \in]T/2, t^+]. \end{cases}$$

Consequently, $D(t_k) = \Delta t_k$ (see Fig. 3). D is maximum when g_i is close to its maximal amplitude, i.e., when t is close to 0 or T , respectively. Hence, we have the relation

$$F_c = \frac{F_e}{2} = \frac{1}{2(t_1 - t_0)} = \frac{1}{2D(0)}$$

if we assume the first sample is taken at $t_0 = 0$. This value can be related directly to a

$$\frac{1}{2F_c} = D(0) = \frac{1}{\omega_0} \cos^{-1} \left(\frac{e^{-a}(A + B) - B}{A} \right). \quad (15)$$

This is defined in \mathbb{R} if the argument of the \cos^{-1} is within the interval $[-1, 1]$. Thus, a should satisfy the inequalities

$$0 \leq a \leq \log \left(\frac{B + A}{B - A} \right). \quad (16)$$

$(B + A)/(B - A)$ is the ratio of the maximum value of the signal to its minimum value, and a needs to satisfy (16).

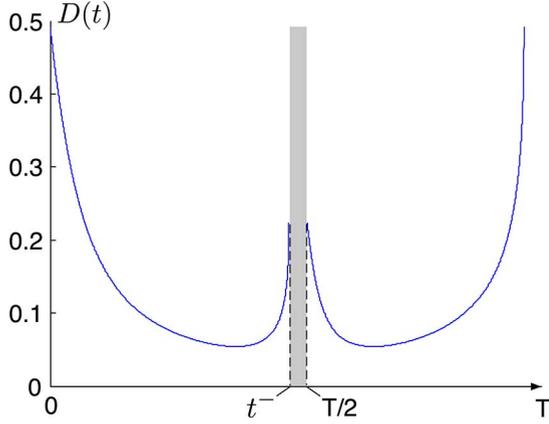


Fig. 3. Function of the time interval between two consecutive samples with $a = 4.85 \times 10^{-2}$, $A = 2$, and $B = 3$. The shaded band outlines the interval for which D is not defined on \mathbb{R} .

This result is also intuitive as it means that the ratio of any two consecutive samples cannot be larger than $(B + A)/(B - A)$. The smallest sampling frequency is reached when a is maximum, i.e., $a = \log((B + A)/(B - A))$ and we have $F_c = 2/D(0) = 2/T$.

Without loss of generality, we can drop index i as all considerations equally apply to any single pixel in the array. In what follows, g_k will refer to $g_i(t_k)$, where k expresses the k th sample of g_i .

The requirements given in (16) alone are not sufficient to guarantee a Nyquist compliant sampling of the monospectral signal. If $e^{\mp a}$ is not small enough, there may be a risk that the sample $g_k = e^{\mp a}g_{k-1}$ might be outside the interval $[B - A, B + A]$; consequently, we need an additional constraint on a .

This constraint can be formulated as: $\forall k \in \mathbb{N}$, $g_k = e^{\mp a}g_{k-1} = e^{\mp ak}g_0$. Then, g_k exists only if

$$B - A \leq e^{\mp ak}g_0 \leq B + A. \quad (17)$$

If we are considering the decreasing part of the cosine, this constraint is

$$\frac{1}{k} \log\left(\frac{g_0}{B - A}\right) \geq a \geq \frac{1}{k} \log\left(\frac{g_0}{B + A}\right). \quad (18)$$

The lower bound of a in (18) is maximum when $g_0 = B + A$, and this is also the value of a for which we achieve cutoff frequency F_c . Consequently, a has to be

$$0 \leq a \leq \frac{1}{k} \log\left(\frac{B + A}{B - A}\right). \quad (19)$$

As expected, the more samples are needed, the smaller a has to be.

Fig. 4 shows an example of the level sampling applied to a cosine with the following parameters: $A = 2$, $B = 3$, and $k = 17$, where k is the number of samples, then

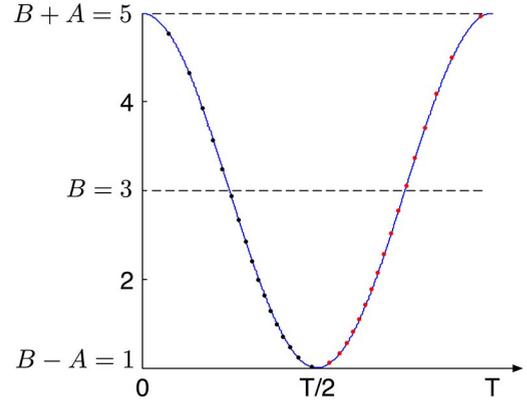


Fig. 4. Monospectral signal of parameters $A = 2$, $B = 3$, and $k = 17$ where k is the number of samples and consequently with $a \approx 2.35 \times 10^{-3}$.

$a \approx 2.35 \times 10^{-3}$. We can observe the increasing density of samples as the signal value decreases. The value of a , chosen according to (19), produces k samples for each half-period of the cosine.

Natural signals are usually much more complex than a monospectral signal, however we can see from this analysis that level crossing sampling effects low-pass filtering of the signal. a is set before the signal acquisition, but it must be chosen according to the signal's parameters A, B and to the needed number of samples k .

The acquisition based on (3) is then extended to a 2-D imaging sensor of $L \times C$ pixels. For the forthcoming sections, we are using the following definitions.

- The signal g_i acquired independently by the pixel $(x_i, y_i)^T$ is the function

$$g_i : \mathbb{R}^+ \rightarrow \mathbb{R}^+ \\ t \mapsto g_i(t). \quad (20)$$

- We define I , a spatio-temporal intensity function

$$I : \mathbb{N}^2 \times \mathbb{R}^+ \rightarrow \mathbb{R}^+ \\ (x_i, y_i, t) \mapsto I(x_i, y_i, t) = g_i(t). \quad (21)$$

With this definition, I is also the set of functions $\{g_i\}$ for $i \in [1, L \times C]$. I is the asynchronous spatio-temporal signal which cannot be processed in a standard image processing way unless frames are built at some given frequency. Constructing frames is always possible using the ATIS sensor [1] at arbitrary points in time, however, as we will show, generating frames from the output of an asynchronous, data-driven sampling sensor is not the preferred path to follow as one loses all the advantages of this signal sampling strategy.

Past unfiltered events captured at $t' < t$

Filtered events at t

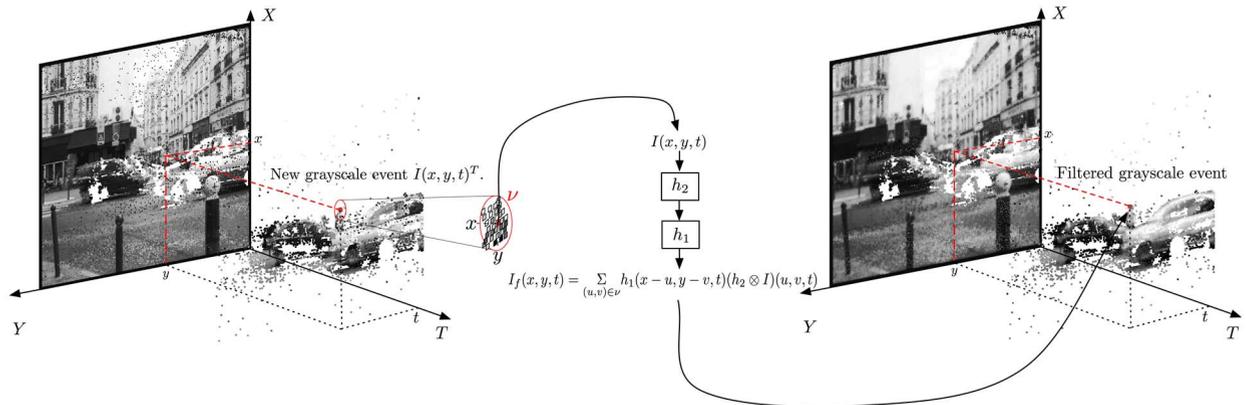


Fig. 5. Asynchronous spatio-temporal filtering mechanism: the vision sensor outputs a stream of events that are processed within a neighborhood ν by h_1 . The output of h_1 is a single event which amplitude is the filtered value. The filter h_2 is some temporal filter applied to I . In our case, it is reduced to the sampling operation performed by the sensor.

IV. ASYNCHRONOUS FILTERING

The asynchronous filter applied to I is assumed to be a composition of a temporal function h_2 and a spatial function h_1 such that

$$I_f = (h_1 \circ h_2)(I). \quad (22)$$

h_2 is a monodimensional filter acting on each individual pixel. It is chosen to implement the level-crossing sampling as in Section III. Here one could also apply additional asynchronous filtering techniques such as the one presented in [22] which mainly consist of reconstruction or interpolation of the original signal.

h_1 implements a 2-D filtering operation on a local neighborhood ν_i , centered on a pixel i and activated at t_i . If the filtering is linear, it is a 2-D convolution product of $I(u, v, t)$, for $(u, v)^T \in \nu_i$ and $t \leq t_i$, with h_1 . If i is the latest updated pixel at time t_i , then all neighboring pixel timestamps in ν_i are going to be lower than t_i . The convolution with h_1 can be written as

$$(h_1 * I)(x_i, y_i, t_i) = \sum_{(u,v)^T \in \nu_i} h_1(x_i - u, y_i - v)I(u, v, t_i). \quad (23)$$

The major difference with a classical 2-D convolution appears when a new event is triggered at $t_i + dt$, for the same pixel i . Because of the recursive form, only one term has to be updated in the convolution

$$(h_1 * I)(x_i, y_i, t_i + dt) = (h_1 * I)(x_i, y_i, t_i) - h_1(0, 0)I(x_i, y_i, t_i) + h_1(0, 0)I(x_i, y_i, t_i + dt). \quad (24)$$

This equality shows that an asynchronously filtered signal is perfectly suited for an iterative algorithm. The

implementation is computationally efficient since we just need to perform one subtraction and one addition using the previously computed values to obtain the new one. This is more efficient than the frame-based convolution if only portions of the pixels in I have changed, which is true in most cases, the only requirement being to be able to store in a buffer previously filtered and unfiltered signals values. This approach is fundamentally different from existing event-based convolution techniques found in the literature because they operate on local spike integrations over a space-time neighborhood to estimate relative intensities of pixels [21], [23]. In our case, there is no need to integrate spikes since intensities are natively provided by the sensor.

The asynchronous filtering is an iterative operation that is suited for parallelized processing. Since a local spatial filtering is performed each time an event is received, an independent thread can be created to filter the signal at the active pixel's neighborhood. Fig. 5 illustrates the asynchronous and iterative filtering summarized by Algorithm 1.

Algorithm 1: Iterative asynchronous spatio-temporal filtering

- 1: **for** each $I(x_i, y_i, t_i)$ **do**
 - 2: define the spatial neighborhood ν_i of $(x_i, y_i)^T$
 - 3: compute the asynchronous temporal filtering $(h_2 \otimes I)(u, v, t_i)$ for $(u, v) \in \nu_i$ if h_2 is specified
 - 4: compute the spatial filtering $(h_1 \otimes (h_2 \otimes I))(u, v, t_i)$ over ν_i incrementally according (24).
 - 5: store the filtered value in I_f : $I_f(x_i, y_i, t_i) = (h_1 \otimes (h_2 \otimes I))(x_i, y_i, t_i)$.
 - 6: **end for**
-

The operator \otimes represents either a linear filtering operation or a nonlinear filtering operation. If a standard linear filtering is applied, \otimes is equivalent to the convolution product $*$.

Algorithm 1 shows a generic filtering operation. We will now move to a practical implementation of three asynchronous filters that have been implemented and tested on event-based signal output by the ATIS [1]: a Gaussian smoothing filter, a bilateral filter, and a Canny edge detector.

Usually, an unknown signal is processed on the fly, i.e., there is no prior knowledge on the signal before starting the sampling process. In this case, there is no explicit way to choose an optimal value for the threshold parameter a . For the following experiments, a was tuned to a value of about 10% relative change in pixel intensity, typically used with the ATIS sensor [1].

A. Gaussian Filter

Gaussian filters are easily parametrizable and separable. If the filter is assumed to be centered, the only parameter to set is the standard deviation. We are referring to the Gaussian function G as

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}. \quad (25)$$

B. Bilateral Filter

The bilateral filter has been introduced in [24] as a smoothing and edge-preserving filter. It is a nonlinear operator composed of two kernels that measure independently the spatial and photometric consistencies

$$I_f(x, y, t) = \sum_{(u,v) \in \mathcal{V}} I(u, v, t) g_1(\|(x, y) - (u, v)\|^2) \times g_2(\|I(x, y, t) - I(u, v, t)\|^2). \quad (26)$$

The kernels are similarity or weighting functions that decrease relative to the pixels' "difference." The g_1 and g_2 kernels are measuring, respectively, the spatial range and the luminance range between two pixels $(x, y)^T$ and $(u, v)^T$. Centered Gaussians are usually chosen for g_1 and g_2 , but if g_1 is set to 1, the bilateral filter is a simple linear low-pass filter. The bilateral filter is not linear, thus it cannot be implemented as a convolution operation. However, the general scheme described in Algorithm 1 is still valid. The only change is to replace (24) by (26) in Algorithm 1.

Algorithm 2: Gaussian gradient operator

Input: $I(x, y, t)$

Output: Stream of gradient events $\nabla(G * I)(x, y, t)$

1: **for** each event $I(x, y, t)$ **do**
 2: Apply Algorithm 1 with $h_1 = \partial G / \partial x = (-x / \sqrt{2\pi\sigma^3}) \exp(-x^2 / 2\sigma^2)$ to $I(x, y, t)$ to have $(\partial G * I / \partial x)(x, y, t) = I_x$.

3: Apply Algorithm 1 with $h_1 = \partial G / \partial y = (-y / \sqrt{2\pi\sigma^3}) \exp(-y^2 / 2\sigma^2)$ to $I(x, y, t)$ to have $(\partial G * I / \partial y)(x, y, t) = I_y$.
 4: **return** $\nabla G * I(x, y, t)$.
 5: **end for**

C. Edge Detector

Edges are interfaces separating two different homogeneous regions in the focal plane. They are defined as the locations where the signal rate of change is the steepest. To classify pixels as edge pixels, one needs to compute the spatial gradient. The higher the gradient's amplitude is, the more likely the pixel belongs to an edge. The gradient itself is simply computed by performing a convolution of the visual signal with a gradient operator such as the Sobel or Prewitt's filters or a Gaussian first derivative. The Gaussian first derivative is interesting since it smoothes the visual information and applies the derivative to the Gaussian function instead of the signal

$$\frac{\partial(G * I)}{\partial w} = \frac{\partial G}{\partial w} * I \text{ with } w = x \text{ or } y. \quad (27)$$

Event-based sensors are native contour detectors as pixel events occur most frequently around moving edges. The main processing needed here is to identify and suppress nonedge-related events and to localize edges as precisely as possible. The Canny edge detection method [27] is a good candidate to be implemented in an event-driven way as it produces thin and continuous edges if thresholds are correctly chosen. The Canny detector steps can then be summarized as:

- compute the spatial gradient;
- suppress the nonlocal maximum in the gradient's direction to ensure thin edges;
- apply a hysteresis thresholding to label pixels that do not have high gradient responses.

Three parameters have to be chosen to obtain a valuable edge detection: the Gaussian standard deviation and the upper and lower thresholds. The algorithm for each method is detailed by Algorithm 3.

Algorithm 3: Canny edge detector

Input: $\tau_1, \tau_2, I(x, y, t)$

Output: Stream of event marked as edge $I_e(x, y, t)$.

1: **for** each event $I(x, y, t)$ **do**
 2: Apply Algorithm 2 to get $\nabla(G * I)(x, y, t)$.
 3: Apply the nonlocal maxima suppression within ν .
 4: Apply a hysteresis thresholding on the gradient amplitude with τ_1 and τ_2 , respectively, being the high and low threshold values for pixel in ν .
 5: Mark $I(x, y, t)$ as edge if the maximum suppression operation did not suppress $I(x, y, t)$.
 6: **return** $I_e(x, y, t)$.
 7: **end for**

Frames generated from the unfiltered raw signal.



Frames generated from the low pass filtered signal.

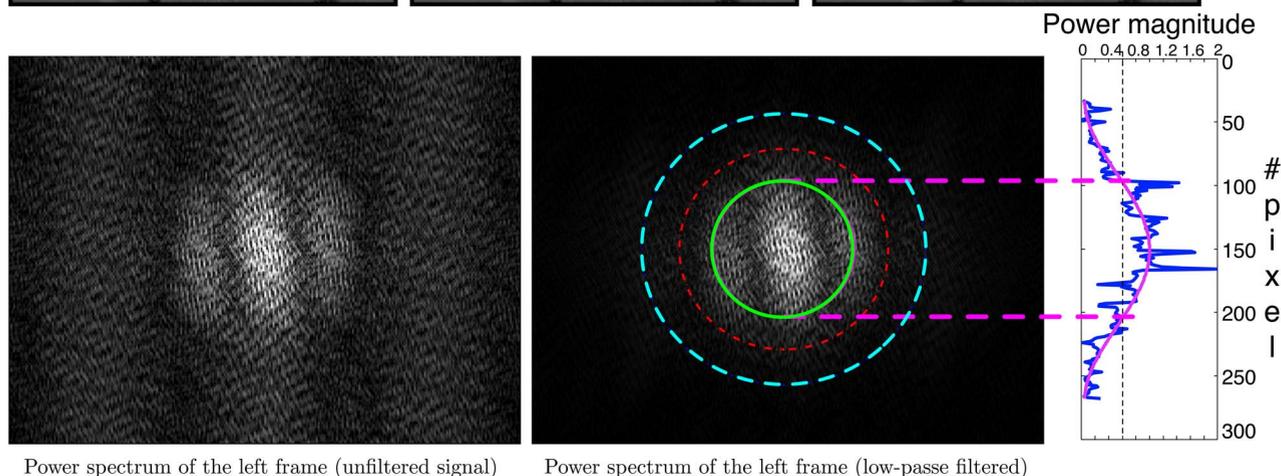


Fig. 6. Event-based spatio-temporal Gaussian low-pass filter. The top row of figures shows raw images of a traffic scene. The middle row contains the same data after asynchronous filtering. Frames are generated at an equivalent frame rate of 100 fps. The bottom row shows the power spectrums computed for frames generated from the spatio-temporal signal (unfiltered left and filtered right). Concentric circles delimit 50%, 25%, and 10% of the raw signal magnitude (frequencies are increasing from the center to the border). The purple curve shows the theoretic signal amplitude attenuation due to the low-pass filter.

The Canny edge detector is implemented based on two temporary matrices. The first matrix stores the last value of pixel spatial gradient. The second matrix stores the label of the pixel according to the upper and lower thresholds: 0 not an edge, 1 potential edge, and 2 reliable edge.

V. EXPERIMENTS AND RESULTS

The ATIS neuromorphic sensor [1] is used to capture indoor and outdoor dynamic scenes. All experiments are

carried out with a static camera, observing a natural scene. The outdoor sequence is a traffic scene captured with the camera placed on the sidewalk, facing the street. The indoor scene uses a checkerboard that is randomly moving in front of the camera. The indoor sequence is designed to test and measure the Canny edge detector, since we can extract a ground truth from the known geometry of the checkerboard. Each asynchronously filtered result is compared to the frame-based filtering results.

A. Gaussian Filter

The first set of experiments are tested using a Gaussian low-pass filter with the outdoors sequence. Each active pixel is filtered according to Algorithm 1. The filter is tested for several increasing values of σ (ranging from 0.5 to 2). The scene has static portions which are updated rarely, and also regions which are frequently updated because of the car traffic. Fig. 6 shows the result with a Gaussian low-pass filter with $\sigma = 1$ applied to the sequence which lasts 18 s, with a mean event rate of 2.8 kevents/s. The first row shows sample frames selected randomly from a set of frames created at a rate of 100 frames per second (fps). The first row represents the raw signal while the second shows the event-based smoothing filtering. The last row shows the 2-D Fourier transform power spectrums of both the raw image (left) and the filtered image (right) generated by the event-based filtering. The magnitude spectrum of the filtered signal shown on the right of the last row shows a clear attenuation of the high-frequency components (the frequency is increasing from the center toward the edges of the plot). Three circles are added to indicate zones at which the power is reduced to 50%, 25%, and 10%. One

can clearly see that the filtering is effective. In the present case, it is difficult to compare with frame-based filtering, because it is difficult to set a ground truth measurement of noise in the stream of events. Without controlling the amount of noise in the acquisition, it is impossible to assess which method performs better.

B. Bilateral Filter

The bilateral filter is applied to the same traffic sequence. The experimental procedure is similar to the Gaussian filtering. The kernel measuring the luminance similarity g_2 and the spatial similarity g_1 is set with $\sigma_1 = 0.75$ with the same standard deviation value as the Gaussian function in the linear smoothing ($\sigma_2 = 1$) to ensure comparison. We briefly show that the edge preserving filter is producing the expected results: smoothing the homogeneous parts of the signal and keeping as much as possible the sharp transitions at edges. Fig. 7 is showing different filtered signals as meshes to outline the edge preserving effect of the bilateral filter. The raw signal is shown in the top-left figure, the result of the asynchronous bilateral filter is shown in the top-right figure, the frame-based bilateral filter's result is shown in

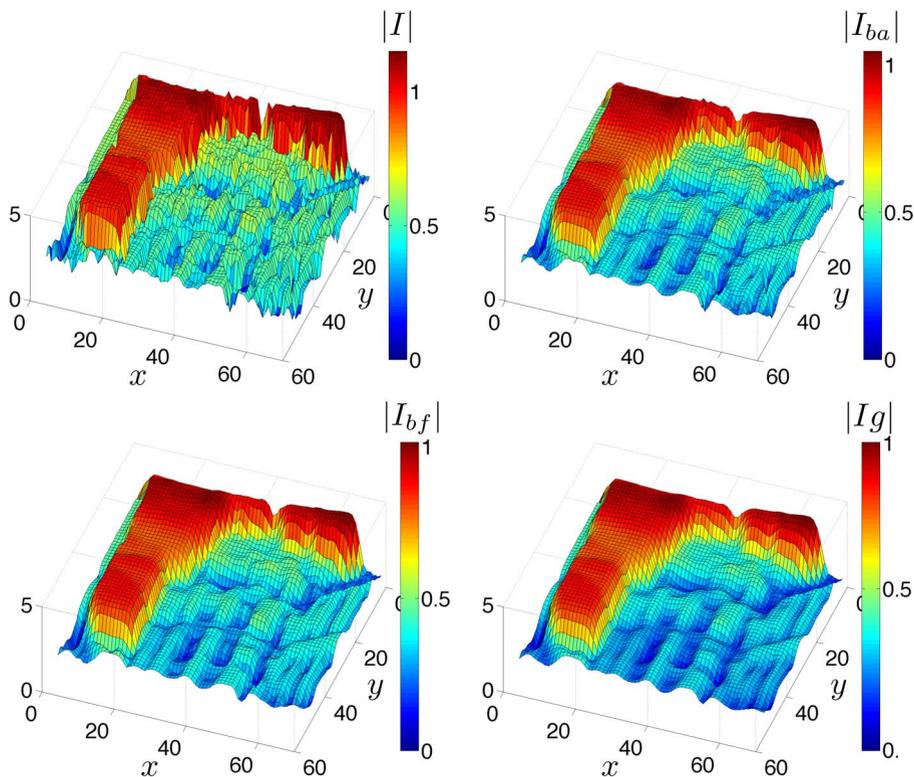


Fig. 7. Asynchronous bilateral filter compared to the frame-based bilateral and low-pass linear filters. The input signal for the bilateral filters is the same as the one used in Fig. 6. The amplitude of the raw signal ($|I|$) is shown in the top-left figure, the amplitude of the asynchronous bilateral filtered signal ($|I_{ba}|$) is shown in the top-right figure, the amplitude of the frame-based bilateral filtered signal ($|I_{bf}|$) is shown in the bottom-left figure, and the low-pass filtered signal ($|I_g|$) is shown in the bottom-right figure. The same Gaussian function is used for both low-pass and bilateral filters. Edge sharpness is more preserved as shown for both bilateral filters, while the homogeneous regions are smoothed as expected.

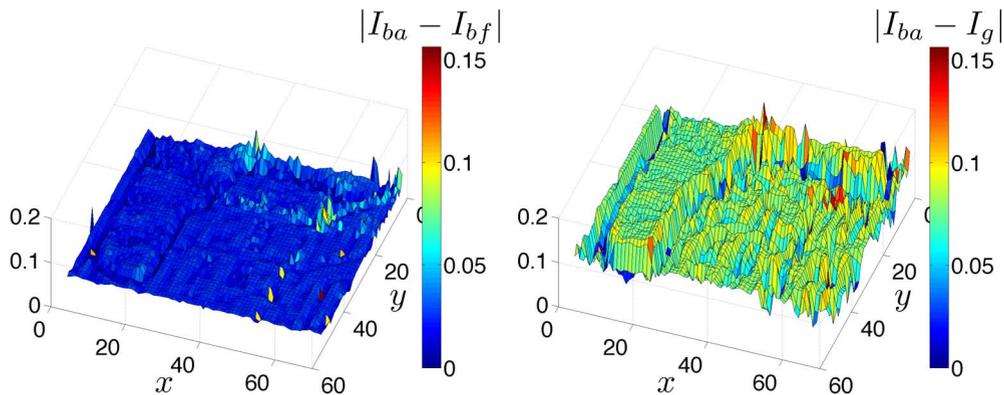


Fig. 8. Error surfaces of the bilateral filtered signals (normalized by the max of $|I_{ba}|$). The left figure shows the relative error between the event-based and frame-based filtered signals ($|I_{ba} - I_{bf}|$), with a mean value of 2.2%. The right figure shows the same relative error between the event-based and Gaussian filtered signals ($|I_{ba} - I_g|$), with a mean value of 7.1%.

the bottom-left figure, and, finally, for comparison purpose, the signal filtered with the g_2 kernel alone (i.e., a linear luminance low-pass filter) is shown in the bottom-right figure. In Fig. 8, we also show the relative errors (normalized by the max of $|I_{ba}|$) between the asynchronous and nonasynchronous bilateral filters' results and between the asynchronous bilateral and Gaussian filters' results. The mean errors are 2.2% and 7.1%, showing clearly comparable performances of both bilateral filters.

Another quantitative measurement is also produced to show the amount of blur around the edges by fitting Gaussians a set of randomly selected edges. The idea is that the fitting Gaussians' variance increases with the image blur. To estimate the variances, we applied the procedure presented in [25]. The estimated blur is compared with the one produced by applying g_2 kernel alone and the one produced by the frame-based bilateral filter. If we define σ_{ba} , σ_{bf} , and σ_g as the mean standard

deviations for the asynchronous bilateral filtered signal, the frame-based bilateral filtered one, and the low-pass filtered one, we get

$$\frac{\sigma_{ba}}{\sigma_g} = 0.92 \quad \text{and} \quad \frac{\sigma_{bf}}{\sigma_g} = 0.91. \quad (28)$$

Statistically, both event-based and frame-based bilateral filters keep sharper edges than the asynchronous Gaussian low-pass filter. This shows that both frame-based and event-based bilateral filters achieve similar performances.

C. Canny Edge Detection

This experiment tests the asynchronous edge detection using a gradient filter to label events as either edges or not. The Canny edge detector described in Algorithm 3 has

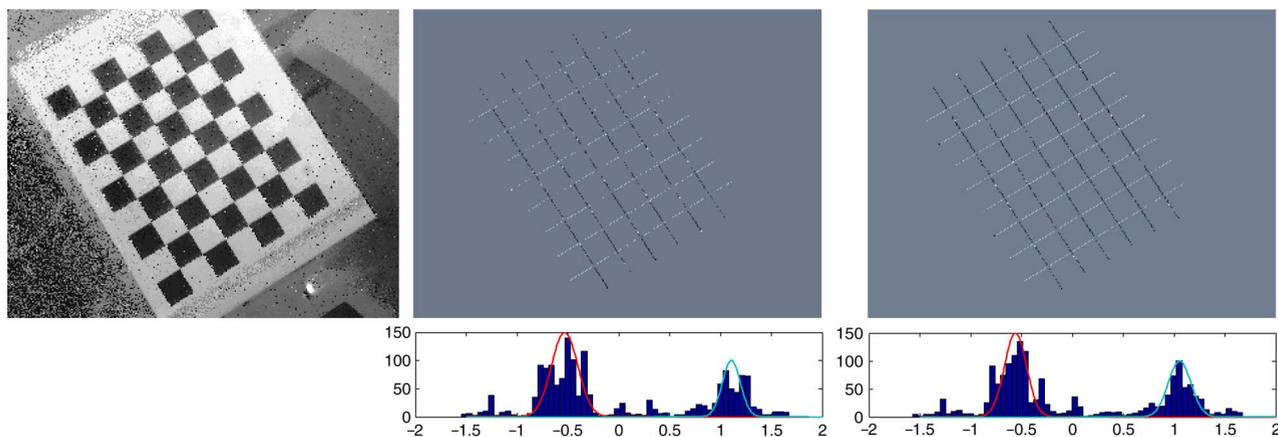


Fig. 9. Gradient direction distributions at edges detected by the asynchronous Canny detection (center) and the frame-based Canny edge detector (right). The ground truth is extracted from the frame generated at time t . The distribution of both results is similar.

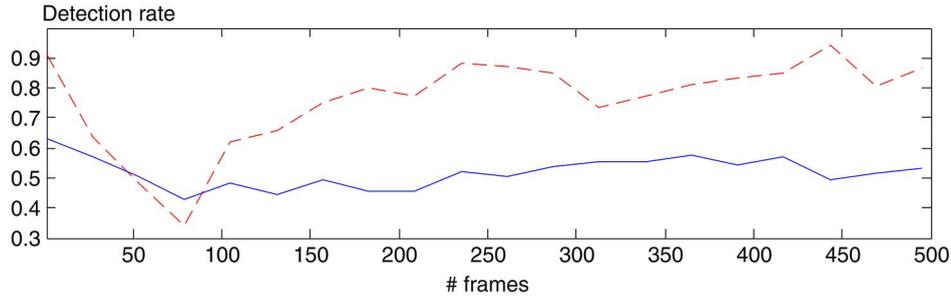


Fig. 10. Asynchronous Canny edge detection result (plain curve) compared to the frame-based Canny algorithm (dashed curve). The same filtering parameters are used for both cases: Gaussians set with the same variance, identical upper and lower thresholds for the hysteresis thresholding. The detection rate is the ratio of the number of pixels detected as edge to the number of ground truth edges.

been implemented and applied to a scene showing a moving checkerboard. The checkerboard is generating straight edges on the sensor's focal plane. The Canny edge detection performances are assessed by examining the detected edges' locations and orientations. We have generated frames from the event stream at a frame rate of 60 fps. A ground truth is built to evaluate the detector performances. We have estimated the homographies mapping the real checkerboard to each generated frame with the classic computer vision technique called the direct linear transformation [26]. The homographies allow to map, on each frame, the checkerboard's edges. A detected edge is considered as correct if it coincides exactly with an edge given by the ground truth. Fig. 9 shows the detected edges whose directions are coded by a gray-level scale. The asynchronous edge detection is shown in the middle, while the frame-based Canny edge detection is shown on the right. In both cases, the detector parameters (Gaussian standard deviation, threshold values) are set to the same values ($\sigma = 1.5$, high and low thresholds, respectively, set to 0.15 multiplied by the mean value of ν_i defined in (23) and 0.03 multiplied by the mean value of ν_i). The distributions of the gradient directions are plotted below the edge images. We can observe similar

performance for the orientations estimation: the mean values and the standard deviation for each set of directions are as follows:

- the first set of directions for the asynchronous detector and the frame-based Canny's detector: -0.512 ± 0.131 rad, -0.529 ± 0.110 rad;
- the second set of direction for the asynchronous detector and the frame-based Canny's detector: 1.054 ± 0.089 rad, 1.025 ± 0.117 rad.

As for the detection rates shown by Fig. 10, we can see that the frame-based Canny detector shows a better performance. The detection rate for the sequence is around 76% with the frame-based Canny, while the asynchronous Canny's detection rate is around 51%. However, one interesting effect of the asynchronous filter can be observed at the beginning of the sequence: while the frame-based Canny detection rate drops because of a sudden change in the acceleration (however, it recovers its performances a few iterations after), the asynchronous detection rate remains slightly more stable. The asynchronous formulation of the edge detector seems to be less sensitive to steep acceleration changes since events are generated at an almost continuous rate that better matches the scene's dynamic.

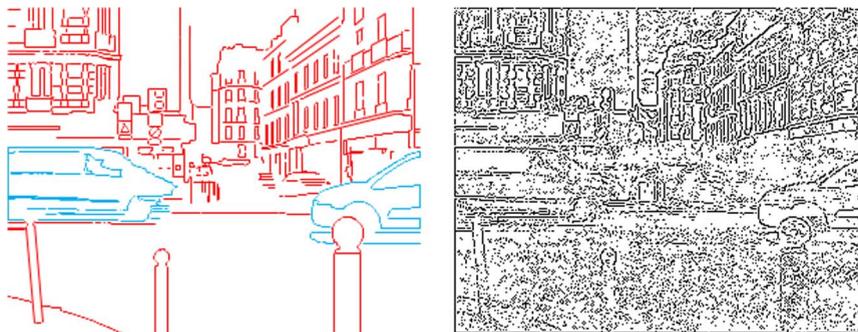


Fig. 11. Canny edge detector implemented asynchronously and applied to the traffic scene. The image on the left shows the manually marked pixels used as ground truth for evaluating the edge detector results shown in the right. Blue pixels outline moving objects while red pixels show the nonmoving ones. Both sets of pixels are used indifferently to evaluate the detection rate.

Table 1 Detection Rate of the Asynchronous Canny Detection According to the Gaussian Filter Variance

σ	Asynch. Canny detector (%)	Frame Canny detector (%)
0.50	0.91	0.95
0.75	0.90	0.94
1	0.87	0.91
2	0.78	0.78

A second edge detection has been carried out on another sequence, showing a real traffic scene (see Fig. 11). For that experiment, the ground truth is a set of randomly selected frames in which we have manually extracted all edges on 16 randomly chosen images, i.e., without the use of any edge detection algorithm one can find in the literature. In this case, a detected edge is accounted for as correct if its distance to an edge pixel of the ground truth is at most 1 pixel. The mean detection rate for several values of the Gaussian variance is displayed in Table 1. The frame-based detector is only slightly better than the asynchronous detector. Finally, the increase of the detection accuracy as σ decreases is complying with the localization term Λ , introduced in Canny’s original work [27], if the Gaussian derivative filter is used. This is an interesting observation since we are checking for how close the detected pixels are to the manually extracted edges. The localization term is actually maximized if the pixels detected as edges are as close as possible to the center of true edges. The localization is defined as

$$\Lambda \propto \frac{|h_1'(0)|}{\sqrt{\int_{\mathbb{R}} h_1^2(x) dx}} = \sqrt{\frac{4}{3\sigma\sqrt{\pi}}} \quad (29)$$

with \propto standing for “proportional to.”

The performances of the filters using both event- and image-based filtering provide comparable results, with little advantage for the frame-based implementation. However, these performances are established using only spatial criteria that tend to be in favor of the frame-based acquisition. The ground truth we have built assesses only the spatial performances of the filters and not necessarily the temporal ones. This has been done because it is difficult to create asynchronous data from frames, and the amount of computations that would be needed to match a frame-based temporal precision of 1 μ s on the sequence and on large portions of time goes beyond our computation capacities. It is also important to emphasize that neuromorphic sensors are noisy. In the carried out experiments, noise was not removed and has been accounted for information. This increases the amount of performance errors, especially when edges’ locations are at stake.

In Section VI, we will show that the overall comparison of computational efficiency is in favor of event-based filtering. For comparable performances, event-based filter-

ing will be shown to be at least 5–10 orders of magnitude more efficient than frame-based filtering.

VI. RESOURCE CONSUMPTION: EVENT-BASED VERSUS FRAME-BASED FILTERING

This section provides a quantitative analysis of the computation cost when gray-level frames are generated from the asynchronous output according to two different strategies: at a fixed event rate or at a fixed frequency.

Frames generated at a constant event rate are scene dependent. If pixels update quickly but only locally, the problem of processing the same information several times will arise similarly to the case where frames are generated at fixed frequency. At some point, frame-based filtering computational cost should be equivalent to that of the event-based filtering. We are aiming at identifying the event rate and the frame rate at which both filtering techniques consume the same resources.

To establish the amount of operations performed by both frame- and event-based filtering techniques, we set the following definitions.

- Let us consider the spatial convolution given by (23) over a neighborhood ν_i of dimensions $S \times S$. For one pixel, this operation is made of S^2 products and $S^2 - 1$ additions. For one given pixel, this set of operations defines a calculation unit that is referred to as u . This is exactly the computation performed on one pixel for both frame-based and event-based filterings.
- N is the total number of captured events.
- K is the number of frames generated from the N events.
- n is a fixed number of events per frame.

The calculation unit defined for the convolution is just an example; it can be extended to a more complex set of operations. The only required assumption is that a single pixel be processed identically by both synchronous or asynchronous algorithms. For that reason, we do not consider the iterative form of the convolution given by (24), which requires less computation resources.

A. Frames Generated at Constant Events Rates

We now examine the case of frames generated at a constant event rate equal to n . Because N , K , and n are integers, then

$$\exists q! \in \mathbb{N} | N = Kn + q, \quad \text{with } q < n. \quad (30)$$

This means that K frames are generated for the N acquired events, and q is then the remainder of the Euclidian division of N by n .

The number of calculation units consumed by K frames of $L \times C$ pixels filtered with the frame-based filtering technique is

$$K.LC = \frac{N - q}{n} LC. \quad (31)$$

We can bound the cost function by two functions that do not depend on q

$$\forall n \in \mathbb{N}, \quad \left(\frac{N}{n} - 1\right)LC < \frac{N - q}{n} LC < \frac{N}{n} LC. \quad (32)$$

We define $g_l(n) = ((N/n) - 1)LC$ and $g_h(n) = (N/n)LC$, the lower and the upper bounds of the number of computation unit necessary to process all frames generated for each set of n events. Let us find the inequalities on n such that $g_l(n) \leq N$ and then n such that $g_h(n) \leq N$. This will allow us to give a bounding interval to localize n such that $((N - q)/n)LC \leq N$, i.e., n leading to a less costly frame-based filtering than the event-based one. We have

$$g_l(n) \leq N \Leftrightarrow n \geq LC \frac{N}{LC + N} \quad (33)$$

and

$$g_h(n) \leq N \Leftrightarrow n \geq LC. \quad (34)$$

We observe that $\lim_{N \rightarrow \infty} LC(N/(N + LC)) = LC$. This implies the rate n to be greater or equal to LC to achieve a less consuming processing for the frame-based filter. The upper bound function g_h has the same asymptotic behavior; it can be concluded that as long as the event number per frame n does not exceed the number of pixels of the camera, then the event-based technique is less computationally expensive. The hypothesis of $N \rightarrow +\infty$ is also easily satisfied if compared to LC because, in practice, $N \gg LC$.

Fig. 12 shows the computation cost of the frame-based filtering when frames are generated according to n . A family of functions are drawn according to the total number of events N . The black curve at the bottom shows the lower bound function g_l when it meets the value N : it provides an estimation of the rate where the frame-based filtering consumes as many resources as the event-based filtering.

B. Frames Generated at Constant Frame Rates

If frames are generated at a constant frame rate F , for a total duration of T , then the number of computation units is equal to $T \times F \times LC$. The event-based processing still consumes N computation units. Both techniques consume the same resources if

$$N = T \times F \times LC \Rightarrow F = \frac{N}{T \times LC}. \quad (35)$$

Statistics established over 50 different sequences (both indoor and outdoor) show that the ratio N/T is closely

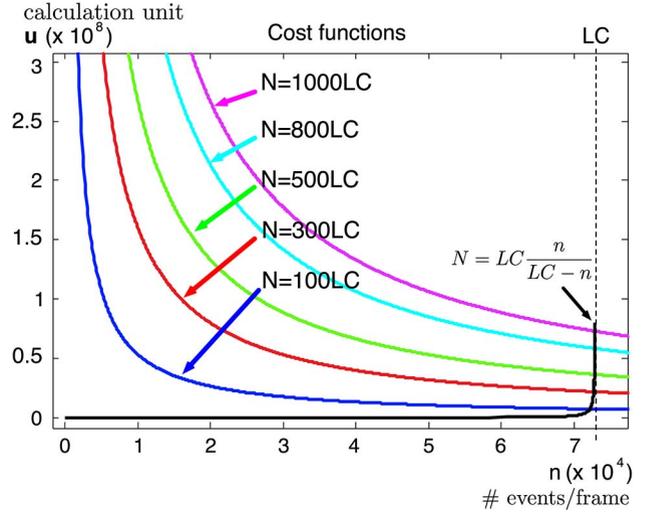


Fig. 12. Cost functions of frame-based filtering compared to event-based filtering. The intersections of the black curve with the cost functions shows the rate at which both filtering techniques consume the same computation resources. Frame-based filtering is more computationally expensive than event-based filtering as long as the event rate is smaller than the number of pixels of the sensor, i.e., $n < LC$.

around 2×10^5 for the ATIS. This leads to a frame rate $F \simeq 3$, below which the event-based filtering consumes more resources than the frame-based filtering. This statistical result shows that event-based filtering is usually, if not always, more computationally efficient than frame-based filtering, since 15 fps is already considered to be low frequency in image processing.

VII. DISCUSSION AND CONCLUSION

In this paper, we introduced a generic methodology to filter event-based spatio-temporal visual signals. The approach is generic and can be extended to any nonlinear filter, and to more complex operations such as the Canny edge detector. In this work, we also provided an insight into level-crossing sampling, by defining its properties and the setting boundaries of its parameters. We provided the link between bandwidth and level-crossing threshold. The paper presented an explicit comparison methodology to estimate the computational cost of event-based versus frame-based algorithms.

Results show that, as long as the number of events used to create a frame is smaller than the total number of pixels of the sensor, the event-based processing is more efficient than the frame-based one, assuming that the same operation is performed on both data types and a single pixel is processed in the same manner in both frame-based and event-based algorithms.

In Section V, asynchronous event-based filters were evaluated and compared with standard frame-based filters.

If performance is quantified from a spatial perspective, the frame-based algorithms perform slightly better than the event-based algorithms.

The work in this paper outlined two main advantages in using the asynchronous filter: the low computational cost and the high temporal resolution dynamic. The asynchronous filter preserves the temporal resolution and the dynamics of the signal and it lowers the computational costs, the tradeoff being that in some cases, the results may be less accurate than in the frame-based filtering. This is due to the fact that event-based filtering processes locally around each incoming event without taking into account the general statistics from the whole image that, in some cases, might be needed. Another reason for the lower performance of the asynchronous filter (e.g., in the case of the complex nonlinear Canny's edge detector) is likely because the implementation of the frame-based algorithms is usually very optimized for a 2-D signal. On the contrary, the asynchronous Canny detector presented in this paper is a straight translation of the detector algorithm without specific optimization. This leaves plenty of room for improvement of dedicated hardware and software.

From a hardware point of view, several dedicated and massively parallel computing platforms will be soon available. Breakthroughs in event-based data processing are expected. Because each pixel of an event-based sensor operates independently, it is possible to speculate that processing each pixel's output by an independent thread each time it notifies the availability of new data will produce even more outstanding computational efficiency. To cite some examples of massively parallel computing platforms, we have the SpiNNaker [28] multicore system or the NEUROGRID [29] silicon neuron network implemented on-chip. They also promise to deliver results with much lower energy consumption than conventional computation platforms.

If the spatial accuracy is the primary criterion of performance, frame-based techniques implemented on dedicated hardware seem to be a better choice than asynchronous techniques. But if a real-time processing is an issue, it is unlikely for a frame-based algorithm to be able to match a frame rate of 10^6 fps (the ATIS temporal accuracy is $1 \mu\text{s}$). For such a case, frames have to be dropped. Consequently, highly fast motion scenes can be processed accurately only by asynchronous algorithms. ■

REFERENCES

- [1] 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.
- [2] T. Hawkes and P. Simonpieri, "Signal coding using asynchronous delta modulation," *IEEE Trans. Commun.*, vol. COM-22, no. 5, pp. 729–731, May 1974.
- [3] C. Vezyrtzis and Y. Tsividis, "Processing of signals using level-crossing sampling," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2009, no. 1, pp. 2293–2296.
- [4] Mahowald, *An Analog VLSI System for Stereoscopic Vision*. Norwell, MA, USA: Kluwer, 1994.
- [5] J. Lazzaro and J. Wawrzyniec, "A multi-sender asynchronous extension to the AER protocol," in *Proc. Conf. Adv. Res. VLSI*, 1995, pp. 158–169.
- [6] A. Andreou and K. Boahen, "A 48 000 pixels, 590 000 transistors silicon retina in current-mode subthreshold CMOS," in *Proc. Symp. Circuits Syst.*, 1994, vol. 1, pp. 97–102.
- [7] E. Culurciello, R. Etienne-Cummings, and K. Boahen, "A biomorphic digital image sensor," *IEEE J. Solid-State Circuits*, vol. 38, no. 2, pp. 281–294, Feb. 2003.
- [8] E. Culurciello and R. Etienne-Cummings, "Second generation of high dynamic range, arbitrated digital imager," in *Proc. Int. Symp. Circuits Syst.*, 2004, vol. 4, pp. IV-828–IV-831.
- [9] P. Lichtsteiner, T. Delbruck, and J. Kramer, "Improved ON/OFF temporally differentiating address-event imager," in *Proc. 11th IEEE Int. Conf. Electron. Circuits Syst.*, 2004, pp. 211–214.
- [10] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128×128 120 dB $15 \mu\text{s}$ latency asynchronous temporal contrast vision sensor," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, Feb. 2008.
- [11] K. A. Zaghoul and K. Boahen, "Optic nerve signals in a neuromorphic chip II: Testing and results," *Trans. Biomed. Eng.*, vol. 51, no. 4, pp. 667–675, 2004.
- [12] P.-F. Ruedi et al., "A 128×128 pixel 120 dB dynamic range vision sensor chip for image contrast and orientation extraction," in *Dig. Tech. Papers IEEE Int. Solid-State Circuits Conf.*, 2003, pp. 226–490.
- [13] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128×128 120 dB 30 mw asynchronous vision sensor that responds to relative intensity change," in *Dig. Tech. Papers IEEE Int. Solid-State Circuits Conf.*, 2006, pp. 2060–2069.
- [14] U. Mallik, M. Clapp, E. Choi, G. Cauwenberghs, and R. Etienne-Cummings, "Temporal change threshold detection imager," in *Dig. Tech. Papers IEEE Int. Solid-State Circuits Con.*, 2005, pp. 362–603.
- [15] T. Delbruck and P. Lichtsteiner, "Fast sensory motor control based on event-based hybrid neuromorphic-procedural system," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2007, pp. 845–849.
- [16] Y. Tsividis, "Event-driven data acquisition and digital signal processing—a tutorial," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 8, pp. 577–581, Aug. 2010.
- [17] J. Mark and T. Todd, "A nonuniform sampling approach to data compression," *IEEE Trans. Commun.*, vol. COM-29, pp. 24–32, 1981.
- [18] B. Bidégaray-Fesquet and L. Fesquet, "A fully nonuniform approach to FIR filtering," *Proc. Int. Conf. Sampling Theory Appl.*, 2009, pp. 1–4.
- [19] L. Fesquet and B. Bidégaray-Fesquet, "IIR digital filtering of non-uniformly sampled signals via state representation," *Signal Process.*, vol. 90, no. 10, pp. 2811–2821, Oct. 2010.
- [20] F. Aeschlimann, "Traitement du Signal chantilloné non uniformément: Algorithme et Architecture," Ph.D. dissertation, Dept. Tech. Inf. Microelectron. Integrat. Syst. Architect., Inst. Nat. Polytechnique de Grenoble, Grenoble, France, 2006.
- [21] T. Serrano-Gotarredona, A. G. Andreou, and B. Linares-Barranco, "AER image filtering architecture for vision-processing systems," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 46, no. 9, pp. 1064–1071, Sep. 1999.
- [22] A. Lazar and L. Toth, "Perfect recovery and sensitivity analysis of time encoded bandlimited signals," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 10, pp. 2060–2073, Oct. 2004.
- [23] L. Camunas-Mesa et al., "An event-driven multi-kernel convolution processor module for event-driven vision sensors," *IEEE J. Solid-State Circuits*, vol. 47, no. 2, pp. 504–517, Feb. 2012.
- [24] C. Tomasi and R. R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. Int. Conf. Comput. Vis.*, 1998, pp. 839–846.
- [25] G. Cao, Y. Zhao, and R. Ni, "Edge-based blur metric for tamper detection," *J. Inf. Hiding Multimedia Signal Process.*, vol. 1, pp. 20–27, 2010.
- [26] Y. Abdel-Aziz and H. Karara, "Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry," in *Proc. Symp. Close-Range Photogramm.*, 1971, pp. 1–18.
- [27] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Jun. 1986.
- [28] S. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The SpiNNaker project," *Proc. IEEE*, vol. 102, no. 5, pp. 652–665, May 2014.
- [29] B. V. Benjamin et al., "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proc. IEEE*, vol. 102, no. 5, pp. 699–716, May 2014.

ABOUT THE AUTHORS

Sio-Hoi Leng received the Ph.D. degree in computer vision from the University Pierre and Marie Curie, Paris, France, in 2005.

He is now an Associate Professor at the University Pierre and Marie Curie and a member of the Vision Institute, Paris, France. He worked on the geometric modeling of noncentral catadioptric vision sensors and their link to the caustic surface. His current research interests include computer vision, with special reference to the understanding of general vision sensors, cameras networks, neuro-morphic event-driven vision and event-based signal processing.



Christoph Posch (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in electrical engineering and experimental physics from Vienna University of Technology, Vienna, Austria, in 1995 and 1999, respectively.

From 1996 to 1999, he worked on analog CMOS and BiCMOS IC design for particle detector readout and control at CERN, the European Laboratory for Particle Physics, Geneva, Switzerland. From 1999 onwards he was with Boston University, Boston, MA, USA, engaging in applied research and mixed-signal integrated circuit design for high-energy physics instrumentation. In 2004, he joined the newly founded Neuroinformatics and Smart Sensors Group at the AIT Austrian Institute of Technology (formerly Austrian Research Centers ARC), Vienna, Austria, where he was promoted to Principal Scientist in 2007. Since 2012, he has been co-directing the Neuromorphic Vision and Natural Computation group at the



Institut de la Vision in Paris, France, and has been appointed Research Professor at Université Pierre et Marie Curie, Paris VI, France. His research interests include neuromorphic analog VLSI, CMOS image and vision sensors, biology-inspired signal processing, and biomedical devices and systems. He is co-founder and scientific advisor of two high-tech start-up companies, has authored more than 90 scientific publications and holds several patents in the area of artificial vision and image sensing.

Dr. Posch has been recipient and co-recipient of eight IEEE awards, including the Jan van Vessel Award at the IEEE International Solid-State Circuits Conference (ISSCC) in 2006, the Best Paper Award at ICECS 2007, and Best Live Demonstration Awards at ISCAS 2010 and BioCAS 2011. He is a member of the Biomedical and Life Science Circuits and Systems, Sensory Systems, and Neural Systems and Applications Technical Committees of the IEEE Circuits and Systems Society.

Ryad Benosman is a full Professor at the Université Pierre et Marie Curie, Paris, France. He leads the Neuromorphic Vision and Natural Computation group. He focuses mainly on neuromorphic engineering, visual computation, and sensing. He is a pioneer of omnidirectional vision systems, complex perception systems, variant scale sensors, and noncentral sensors. His current research interests include the understanding of the computation operated by the visual system with the goal to establish the link between computational and biological vision. He is also invested in applying neuromorphic computation to retina prosthetics and is a cofounder of the startup Pixium Vision.

