HATS: Histograms of Averaged Time Surfaces for Robust Event-based Object Classification

Amos Sironi[1]*, Manuele Brambilla[1], Nicolas Bourdis[1], Xavier Lagorce[1], Ryad Benosman[1,2,3]
[1]PROPHESEE, Paris, France [2]Institut de la Vision, UPMC, Paris, France
[3]University of Pittsburgh Medical Center / Carnegie Mellon University
{asironi, mbrambilla, nbourdis, xlagorce}@prophesee.ai ryad.benosman@upmc.fr

Abstract

Event-based cameras have recently drawn the attention of the Computer Vision community thanks to their advantages in terms of high temporal resolution, low power consumption and high dynamic range, compared to traditional frame-based cameras. These properties make event-based cameras an ideal choice for autonomous vehicles, robot navigation or UAV vision, among others. However, the accuracy of event-based object classification algorithms, which is of crucial importance for any reliable system working in real-world conditions, is still far behind their frame-based counterparts. Two main reasons for this performance gap are: 1. The lack of effective low-level representations and architectures for event-based object classification and 2. The absence of large real-world event-based datasets. In this paper we address both problems. First, we introduce a novel event-based feature representation together with a new machine learning architecture. Compared to previous approaches, we use local memory units to efficiently leverage past temporal information and build a robust event-based representation. Second, we release the first large real-world event-based dataset for object classification. We compare our method to the state-of-the-art with extensive experiments, showing better classification performance and real-time computation.

1. Introduction

This paper focuses on the problem of object classification using the output of a neuromorphic asynchronous event-based camera [15, 14, 53]. Event-based cameras offer a novel path to Computer Vision by introducing a fundamentally new representation of visual scenes, with a drive towards real-time and low-power algorithms.

Contrary to standard frame-based cameras, which rely on a pre-defined acquisition rate, in event-based cameras, individual pixels asynchronously emit *events* when they observe a sufficient change of the local illuminance intensity (Figure 1). This new principle leads to significant reduction of memory usage and of power consumption and the information contained in standard videos of hundreds megabytes can be naturally compressed in an event stream of few hundreds kilobytes [36, 52, 63]. Additionally, the time

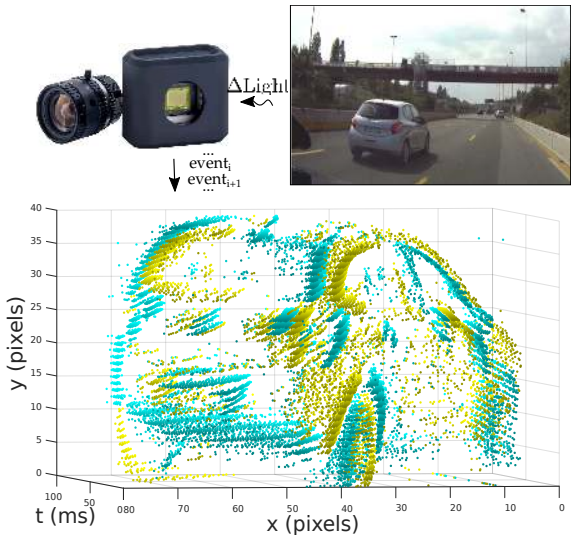

Figure 1: Pixels of an event-based camera asynchronously generate events as soon as a contrast change is detected in their field of view. As a consequence, the output of an event-based camera can be extremely sparse and with time resolution of order of microseconds. Because of the asynchronous nature of the data and the high resolution of the temporal component of the events, compared to the spatial one, standard Computer Vision methods can not be directly applied. **Top**: An event-based camera (left) recording a natural scene (right). **Bottom**: Visualization of the events stream generated by a moving object. ON and OFF events (Sec. 2) are represented by yellow and cyan dots respectively. This figure, as most of the figures in this paper, is best seen in color.

*This work was supported in part by the EU H2020 ULPEC project (grant agreement number 732642)

resolution of event-based cameras is orders of magnitude higher than frame-based cameras, reaching up to hundreds of microseconds. Finally, thanks to their logarithmic sensitivity to illumination changes, event-based cameras also have a much larger dynamic range, exceeding 120dB [52]. These characteristics make event-based cameras particularly interesting for applications with strong constraints on latency (*e.g.* autonomous navigation), power consumption (*e.g.* UAV vision and IoT), or bandwidth (*e.g.* tracking and surveillance).

However, due to the novelty of the field, the performance of event-based systems in real-world conditions is still inferior to their frame-based counterparts [28, 66]. We argue that two main limiting factors of event-based algorithms are: 1. the limited amount of work on low-level feature representations and architectures for event-based object classification; 2. the lack of large event-based datasets acquired in real-world conditions. In this work, we make important steps towards the solution of both problems.

We introduce a new event-based scalable machine learning architecture, relying on a low-level operator called Local Memory Time Surface. A time surface is a spatio-temporal representation of activities around an event relying on the arrival time of events from neighboring pixels [30]. However, the direct use of this information is sensitive to noise and non-idealities of the sensors. By contrast, we emphasize the importance of using the information carried by past events to obtain a robust representation. Moreover, we show how to efficiently store and access this past information by defining a new architecture based on local memory units, where neighboring pixels share the same memory block. In this way, the Local Memory Time Surfaces can be efficiently combined into a higher-order representation, which we call Histograms of Averaged Time Surfaces.

This results in an event-based architecture which is significantly faster and more accurate than existing ones [30, 33, 46]. Driven by brain-like asynchronous event based computations, this new architecture offers the perspective of a new class of machine learning algorithms that focus the computational effort only on active parts of the network.

Finally, motivated by the importance of large-scale datasets for the recent progress of Computer Vision systems [16, 28, 37], we also present a new real-world event-based dataset dedicated to car classification. This dataset is composed of about 24k samples acquired from a car driving in urban and motorway environments. These samples were annotated using a semi-automatic protocol, which we describe below. To the best of our knowledge this is the largest labeled event-based dataset acquired in real-world conditions.

We evaluate our method on our new event-based dataset and on four other challenging ones. We show that our method reaches higher classification rates and faster computation times than existing event-based algorithms.

2. Event-based camera

Conventional cameras encode the observed scene by producing dense information at a fixed frame-rate. As explained in Sec. 1, this is an inefficient way to encode natural scenes. Following this observation, a variety of event-based cameras [36, 52, 63] have been designed over the past few years, with the goal to encode the observed scene adaptively, based on its content.

In this work, we consider the ATIS camera [52]. The ATIS camera contains an array of fully asynchronous pixels, each composed of an illuminance *relative change detector* and a *conditional exposure measurement block*. The relative change detector reacts to changes in the observed scene, producing information in the form of *asynchronous address events* [4], known henceforth as *events*. Whenever a pixel detects a change in illuminance intensity, it emits an event containing its x-y position in the pixel array, the microsecond timestamp of the observed change and its polarity: i.e. whether the illuminance intensity was increasing (ON events) or decreasing (OFF events). The conditional exposure measurement block measures the absolute luminous intensity observed by a pixel [49]. In the ATIS, the measurement itself is not triggered at fixed frame-rate, but only when a change in the observed scene is detected by the relative change detector.

In this work, the luminous intensity measures from the ATIS camera were used only to generate ground-truth annotations for the dataset presented in Sec. 5. By contrast, the object classification pipeline was designed to operate on change-events only, in order to support generic event-based cameras, whether or not they include the ATIS feature to generate grey levels. In this way, any event-based camera can be used to demonstrate the potential of our approach, while leaving the possibility for further improvement when gray level information is available [39].

3. Related work

In this section, we first briefly review frame-based object classification, then we describe previous work on event-based features and object classification. Finally, we discuss existing event-based datasets.

Frame-based Features and Object Classification There is a vast literature on spatial [40, 13, 67, 57] and spatio-temporal [31, 73, 60] feature descriptors for frame-based Computer Vision. Early approaches mainly focus on handcrafting feature representations for a given problem by using domain knowledge. Well-designed features combined with shallow classifiers have driven research in object recognition for many decades [72, 13, 18] and helped understanding and modeling important properties of the object classification problem, such as local geometric invari-

ants, color and light properties, etc. [74, 1].

In the last few years, the availability of large datasets [16, 37] and effective learning algorithms [32, 26, 68] shifted the research direction towards data driven learning of feature representations [2, 22]. Typically this is done by optimizing the weights of several layers of elementary feature extraction operations, such as spatial convolutions, pixel-wise transformations, pooling etc. This allowed an impressive improvement in the performance of image classification approaches and many others Computer Vision problems [28, 66, 75]. Deep Learning models, although less easily interpretable, also allowed understanding higher order geometrical properties of classical problems [8].

By contrast, the work on event-based Computer Vision is still in its early stages and it is unclear which feature representations and architectures are best suited for this problem. Finding adequate low-level feature operations is a fundamental topic both for understanding the properties of event-based problems and also for finding the best architectures and learning algorithms to solve them.

Event-based Features and Object Classification Simultaneous Localization and Mapping use-cases [25, 54] drove the majority of prior work on event-based features [11, 43] for stable detection and tracking. Corner detectors, have been defined in [11, 71, 43], while the works of [61, 7] focused on edge and line extraction.

Recently, [12] introduced a feature descriptor based on local distributions of optical flow and applied it to corner detection and gesture recognition. It is inspired by its frame-based counterpart [10], but in [12] the algorithm for computing the optical flow relies on the temporal information carried by the events. One limitation of [12] is that the quality of the descriptor strongly depends on the quality of the flow. As a consequence, it loses accuracy in presence of noise or poorly contrasted edges.

Event-based classification algorithms can be divided in two categories: unsupervised learning methods and supervised ones. Most unsupervised approaches train artificial neural networks by reproducing or imitating the learning rules observed in biological neural networks [21, 42, 38, 3, 65, 41]. Supervised methods [47, 29, 34, 51], similar to what is done in frame-based Computer Vision, try to optimize the weights of artificial networks by minimizing a smooth error function.

The most commonly used architectures for event-based cameras are Spiking Neural Networks (SNN) [5, 59, 24, 17, 9, 76, 46]. SNN are a promising research field; however, their performance is limited by the discrete nature of the events, which makes it difficult to properly train a SNN with gradient descent. To avoid this, some authors [50] use predefined Gabor filters as weights in the network. Others propose to first train a conventional Convolutional Neural Networks (CNN) and then to convert the weights to a SNN [9, 58]. In both cases, the obtained solutions are suboptimal and typically the performance is lower than conventional CNNs on frames. Other methods consider a smoothed version of the transfer function of a SNN and directly optimize it [33, 45, 69]. The convergence of the corresponding optimization problem is still very difficult to obtain and typically only few layers and small networks can be trained.

Recently, [30] proposed an interesting alternative to SNNs by introducing a hierarchical representation based on the definition of *Time Surface*. In [30], learning is unsupervised and performed by clustering time surfaces at each layer, while the last layer sends its output to a classifier. The main limitations of this method are its high latency, due to the increasing time window needed to compute the time surfaces and the high computational cost of the clustering algorithm.

We propose a much simpler yet effective feature representation. We generalize time surfaces by introducing a memory effect in the network by storing the information carried by past events. We then build our representation by applying a regularization scheme both in time and space to obtain a compact and fast representation. Although the architecture is scalable, we show that once a memory process is introduced, a single layer is sufficient to outperform a multilayer approach directly relying on time surfaces. This reduces computation, but more importantly, adds more generalization and robustness to the network.

Event-based Datasets An issue of previous work on event-based object classification is that the proposed solutions are tested either on very small datasets [64, 30], or on datasets generated by converting standard videos or images to an event-based representation [48, 23, 35]. In the first case, the small size of the test set prevents an accurate evaluation of the methods. In the second case, the dataset size is large enough to create a valid tool for testing new algorithms. However, since the datasets are generated from static images, the real dynamics of a scene and the temporal resolution of event-based cameras can not be fully employed and there is no guarantee that a method tested on this kind of artificial data will behave similarly in real-world conditions.

The authors of [44] released an event-based dataset adapted to test visual odometry algorithms. Unfortunately, this dataset does not contain labeled information for an object recognition task.

The need of large real-world datasets is a major slowing factor for event-based vision [70]. By releasing a new labeled real-world event-based dataset, and defining an efficient semi-automated protocol based on a single event-based camera, we intend to accelerate progress toward a robust and accurate event-based object classifier.

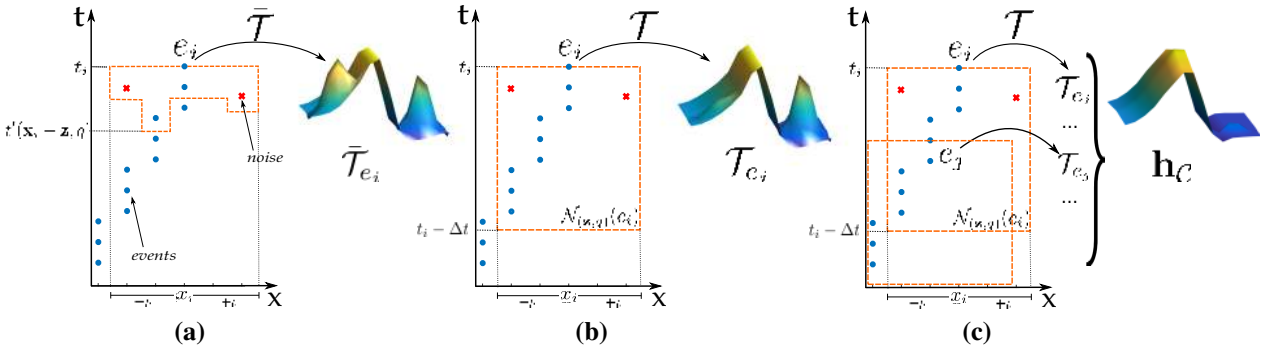

Figure 2: Time surface computation around an event e_i, in presence of noise. Noisy events are represented as red crosses, non-noisy events as blue dots. For clarity of visualization only the x-t component of the event stream and a single polarity are shown. **(a)** In [30] the time surface $\bar{\mathcal{T}}_{e_i}$ (Eq. (2)) is computed by considering only the times $t'(\mathbf{x}_i + \mathbf{z}, q)$ of the last events in a neighborhood of e_i (orange dashed line). As a consequence, noisy events can have a large weight in $\bar{\mathcal{T}}_{e_i}$. This is visible from the spurious peaks in the surface $\bar{\mathcal{T}}_{e_i}$. **(b)** By contrast, the definition of Local Memory Time Surface $\mathcal{T}_{e_i}$ of Eq. (3), considers the contribution of all past events in a spatio-temporal window $\mathcal{N}_{(\mathbf{z},q)}(e_i)$. In this way, the ratio of noisy events considered to compute $\mathcal{T}$ is smaller and the result better describes the real dynamics of the underling stream of events. **(c)** The time surface can be further regularized by spatially averaging the time surfaces for all the events in a neighborhood (Eq. (6)). Thanks to both the spatial and temporal regularization, the contribution of noise is almost completely suppressed.

4. Method

In this section, we formalize the event-based representation of visual scenes and describe our event-based architecture for object classification.

4.1. Time Surfaces

Given an event-based sensor with pixel grid size $M \times N$, a stream of events is given by a sequence

$$\mathcal{E} = \{e_i\}_{i=1}^{I}, \text{ with } e_i = (\mathbf{x}_i, t_i, p_i), \tag{1}$$

where $\mathbf{x}_i = (x_i, y_i) \in [1, \dots, M] \times [1, \dots, N]$ are the coordinates of the pixel generating the event, $t_i \geq 0$ the timestamp at which the event was generated, with $t_i \leq t_j$ for $i < j$, and $p_i \in \{-1, 1\}$ the *polarity* of the event, with $-1, 1$ meaning respectively OFF and ON events, and I is the number of events. From now on we will refer to individual events by e_i and to a sequence of events by $\{e_i\}$.

In [30], the concept of time surface is introduced to describe local spatio-temporal patterns around an event. A time surface can be formalized as a local spatial operator acting on an event e_i by $\bar{\mathcal{T}}_{e_i}(\cdot,\cdot) : [-\rho, \rho]^2 \times \{-1, 1\} \to \mathbb{R}$, where ρ is the radius of the spatial neighborhood used to compute the time surface.

For an event $e_i = (\mathbf{x}_i, t_i, p_i)$, and $(\mathbf{z}, q) \in [-\rho, \rho]^2 \times \{-1, 1\}$, $\bar{\mathcal{T}}_{e_i}$ is given by

$$\bar{\mathcal{T}}_{e_i}(\mathbf{z}, q) = \begin{cases} e^{-\frac{t_i - t'(\mathbf{x}_i + \mathbf{z}, q)}{\tau}} & \text{if } p_i = q \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Where $t'(\mathbf{x}_i + \mathbf{z}, q)$ is the time of the last event with polarity q received from pixel $\mathbf{x}_i + \mathbf{z}$ (Fig. 2(a)), and τ is a decay factor giving less weight to events further in the past. Intuitively, a time surface encodes the dynamic context in a neighborhood of an event, hence providing both temporal and spatial information. Therefore, this compact representation of the content of the scene can be useful to classify different patterns.

4.2. Local Memory Time Surfaces

To build the feature representation, we start by generalizing the time surface $\bar{\mathcal{T}}_{e_i}$ of Eq. (2). As shown in Fig. 2(a) using only the time $t'(\mathbf{x}_i + \mathbf{z}, q)$ of the last event received in the neighborhood of the time surface pixel $\mathbf{x}_i$, leads to a descriptor which is too sensitive to noise or small variations in the event stream.

To avoid this problem, we compute the time surface by considering the history of the events in a temporal window of size Δt. More precisely, we define a local memory time surface $\mathcal{T}_{e_i}$ as

$$\mathcal{T}_{e_i}(\mathbf{z}, q) = \begin{cases} \sum_{e_j \in \mathcal{N}_{(\mathbf{z},q)}(e_i)} e^{-\frac{t_i - t_j}{\tau}} & \text{if } p_i = q \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where

$$\mathcal{N}_{(\mathbf{z},q)}(e_i) = \{e_j : \mathbf{x}_j = \mathbf{x}_i + \mathbf{z}, t_j \in [t_i - \Delta t, t_i), p_j = q\}. \tag{4}$$

As shown in Fig. 2(b), this formulation more robustly describes the real dynamics of the scene while resisting noise and small variations of events. In the supplementary material we compare the results obtained by using Eq. (2) or Eq. (3) on an object classification task, showing the advantage of using the local memory formulation to achieve better accuracy.

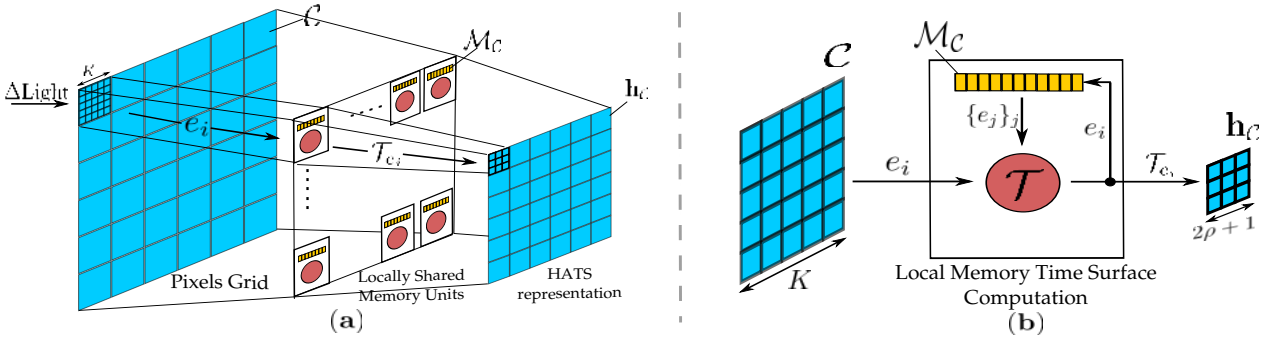

Figure 3: Overview of the proposed architecture. **(a)** The pixel grid is divided into cells $\mathcal{C}$ of size $K \times K$. When a change of light is detected by a pixel, an event e_i is generated. Then, the time surface $\mathcal{T}_{e_i}$ is computed and used to update the histogram $\mathbf{h}_{\mathcal{C}}$. The *HATS* representation is obtained by the concatenation of the histograms $\mathbf{h}_{\mathcal{C}}$. **(b)** Detail of the Local Memory Time Surface computation using local memory units. For each input event, the time surface of Eq. (3) is computed by using the past events e_j's stored in the cell's local memory unit $\mathcal{M}_C$ (Sec. 4.4). After computation, $\mathcal{T}_{e_i}$ is used to update the histogram $\mathbf{h}_{\mathcal{C}}$ of the corresponding cell, while event e_i is added to the memory unit. For simplicity, the polarity of the event and the normalization of the histograms are not considered in the scheme.

The name *Local Memory Time Surfaces* comes from the fact that past events $\{e_j\}$ in $\mathcal{N}_{(\mathbf{z},q)}(e_i)$ need to be stored in memory units in order to prevent the algorithm from 'forgetting' past information. In Sec. 4.4, we will describe how memory units can be shared efficiently by neighboring pixels. In this way, we can compute a robust feature representation without significant increase in memory requirements.

4.3. Histograms of Averaged Time Surfaces

The local memory time surfaces of Eq. (3) is the elementary spatio-temporal operator we use in our approach. In this section, we describe how this new type of time surface can be used to define a compact representation of an event stream useful for object classification.

Inspired by [13] in frame-based vision, we group adjacent pixels in cells $\{\mathcal{C}_l\}_{l=1}^{L}$ of size $K \times K$. Then, for each cell $\mathcal{C}$, we sum the components of the time surfaces computed on events from $\mathcal{C}$ into histograms. More precisely, for a cell $\mathcal{C}$ we have:

$$\bar{\mathbf{h}}_{\mathcal{C}}(\mathbf{z}, p) = \sum_{e_i \in \mathcal{C}} \mathcal{T}_{e_i}(\mathbf{z}, p), \tag{5}$$

where, with an abuse of notation, we write $e_i \in \mathcal{C}$ if and only if pixel coordinates (x_i, y_i) of the event belong to $\mathcal{C}$.

A characteristic of event-based sensors is that the amount of events generated by a moving object is proportional to its contrast: higher contrast objects generate more events than low contrast objects. To make the cell descriptor more invariant to contrast, we therefore normalize $\bar{\mathbf{h}}$ by the number of events $|\mathcal{C}|$ contained in the spatio-temporal window used to compute it. This results in the averaged histogram:

$$\mathbf{h}_{\mathcal{C}}(\mathbf{z}, p) = \frac{1}{|\mathcal{C}|}\bar{\mathbf{h}}_{\mathcal{C}}(\mathbf{z}, p) = \frac{1}{|\mathcal{C}|}\sum_{e_i \in \mathcal{C}} \mathcal{T}_{e_i}(\mathbf{z}, p). \tag{6}$$

Algorithm 1 *HATS* with shared memory units

Input: Events $\mathcal{E} = \{e_i\}_{i=1}^{I}$ Parameters: $\rho, \Delta t, \tau, K$
Output: *HATS* representation $\mathbf{H}(\{e_i\})$
Initialize: $\mathbf{h}_{\mathcal{C}_l} = \mathbf{0},\ |\mathcal{C}_l| = 0,\ \mathcal{M}_{\mathcal{C}_l} = \emptyset$, for all l
for $i = 1, \ldots, I$ **do**
    $\mathcal{C}_l \leftarrow \text{getCell}(x_i, y_i)$
    $\mathcal{T}_{e_i} \leftarrow \text{computeTimeSurface}(e_i, \mathcal{M}_{\mathcal{C}_l})$
    $\mathbf{h}_{\mathcal{C}_l} \leftarrow \mathbf{h}_{\mathcal{C}_l} + \mathcal{T}_{e_i}$
    $\mathcal{M}_{\mathcal{C}_l} \leftarrow \mathcal{M}_{\mathcal{C}_l} \cup e_i$
    $|\mathcal{C}_l| \leftarrow |\mathcal{C}_l| + 1$
return $\mathbf{H} = [\mathbf{h}_{\mathcal{C}_1}/|\mathcal{C}_1|, \ldots, \mathbf{h}_{\mathcal{C}_L}/|\mathcal{C}_L|]^\intercal$

An example of a cell histogram $\mathbf{h}_{\mathcal{C}}(\mathbf{z}, p)$ is shown in Fig. 2(c). Given a stream of events, our final descriptor, which we call *HATS* for Histograms of Averaged Time Surfaces, is given by concatenating every $\mathbf{h}_{\mathcal{C}}$, for all positions $\mathbf{z}$, polarities and cells $1, \ldots, L$:

$$\mathbf{H}(\{e_i\}) = [\mathbf{h}_{\mathcal{C}_1}, \ldots, \mathbf{h}_{\mathcal{C}_L}]^\intercal. \tag{7}$$

Fig. 3(a) shows an overview of our method.

Similarly to standard Computer Vision methods, we can further group adjacent cells into blocks and perform a block-normalization scheme to obtain more invariance to velocity and contrast [13]. In Sec. 6, we show how this simple representation obtains higher accuracy for event-based object classification compared to previous approaches.

4.4. Architecture with Locally Shared Memory Units

Irregular access in event-based cameras is a well known limiting factor for designing efficient event-based algorithms. One of the main problems is that the use of standard

hardware accelerations, such as GPU, is not trivial due to the sparse and asynchronous nature of the events. For example, accessing spatial neighbors on contiguous memory blocks can impose significant overheads when processing event-based data.

The architecture computing the *HATS* representation allows to overcome this memory access issue (Fig. 3). From Eq. (5) we notice that for every incoming event e_i, we need to iterate over all events in a past spatio-temporal neighborhood. Since, for small values of ρ, most of the past events would not be in the neighborhood of e_i, looping through the entire temporally ordered event stream would be prohibitively expensive and inefficient. To avoid this, we notice that, for $\rho \approx K$, the events falling in the same cell $\mathcal{C}$, will share most of the neighbors $\mathcal{N}_{(\mathbf{z},q)}$ used to compute Eq. (3). Following this observation, for every cell, we define a shared memory unit $\mathcal{M}_C$, where past events relevant for $\mathcal{C}$ are stored. In this way, when a new event arrives in $\mathcal{C}$, we update Eq. (5) by only looping through $\mathcal{M}_C$, which contains only the relevant past events to compute the Local Memory Time Surface of Eq. (3) (Fig. 3(b)).

Algorithm 1 describes the computation of *HATS* with memory units. Although this was not the scope of this paper, we notice that Algorithm 1 can be easily parallelized and implemented in dedicated neuromorphic chips [62].

5. Datasets

We validated our approach on five different datasets: four datasets generated by converting standard frame-based datasets to events (namely, the N-MNIST [48], N-Caltech101 [48], MNIST-DVS [63] and CIFAR10-DVS [35] datasets) and a novel dataset, recorded from real-world scenes and introduced for the first time in this paper, which we call N-CARS. We made the N-CARS dataset publicly available for download at `http://www.prophesee.ai/dataset-n-cars/`.

5.1. Datasets Converted from Frames

N-MNIST, N-Caltech101, MNIST-DVS and CIFAR10-DVS are four publicly available datasets created by converting the popular frame-based MNIST [32], Caltech101 [20] and CIFAR10 [27] to an event-based representation.

N-MNIST and N-Caltech101 were obtained by displaying each sample image on an LCD monitor, while an ATIS sensor (Section 2) was moving in front of it [48]. Similarly, the MNIST-DVS and CIFAR10-DVS datasets were created by displaying a moving image on a monitor and recorded with a fixed DVS sensor [63].

In both cases, the result is a conversion of the images of the original datasets into a stream of events suited for evaluating event-based object classification. Fig. 4(a,b) shows some representative examples of the datasets generated from frames, for the N-MNIST and N-Caltech101.

5.2. Dataset Acquired Directly as Events: N-CARS

The datasets described in the previous section are good datasets for a first evaluation of event-based classifiers. However, since they were generated by displaying images on a monitor, they are not very representative of data from real-world situations. The main shortcoming results from the limited and predefined motion of the objects.

To overcome these limitations, we created a new dataset by directly recording objects in urban environments with an event-based sensor. The dataset was obtained with the following semi-automatic protocol. First, we captured approximately 80 minutes of video using an ATIS camera (Section 2) mounted behind the windshield of a car. The driving was conducted in a natural way, without particular regards for video quality or content. In a second stage, we converted gray-scale measurements from the ATIS sensor to conventional gray-scale images. We then processed them with a state-of-the-art object detector [55, 56], to automatically extract bounding boxes around cars and background samples. Finally, the data was manually cleaned to ensure that the samples were correctly labeled.

Since the gray-scale measurements have the same time resolution of the change detection events, the gray-level images can be easily synchronized with the change detection events. Thus, the positions and timestamps of the bounding boxes can be directly used to extract the corresponding event-based samples from the full event stream. Thanks to our semi-automated protocol, we generated a two-class dataset composed of 12,336 car samples and 11,693 non-cars samples (background). The dataset was split in 7940 car and 7482 background training samples, and 4396 car and 4211 background testing samples. Each example lasts 100 milliseconds. More details on the dataset can be found in the supplementary material.

We called this new dataset N-CARS. As shown in Fig. 4(c) the N-CARS is a challenging dataset, containing cars at different poses, speeds and occlusions, as well as a large variety of background scenarios.

6. Experiments

6.1. Event-based Object Classification

Once the features have been extracted from the events sequences of the database, the problem reduces to a conventional classification problem. To highlight the contribution of our feature representation to classification accuracy, we used a simple linear SVM classifier in all our experiments. A more complex classifier, such as non-linear SVM or Convolutional Neural Networks, could be used to further improve the results.

The parameters for all methods were optimized by splitting the training set and using 20% of the data for validation. Once the best settings were found, the classifier was

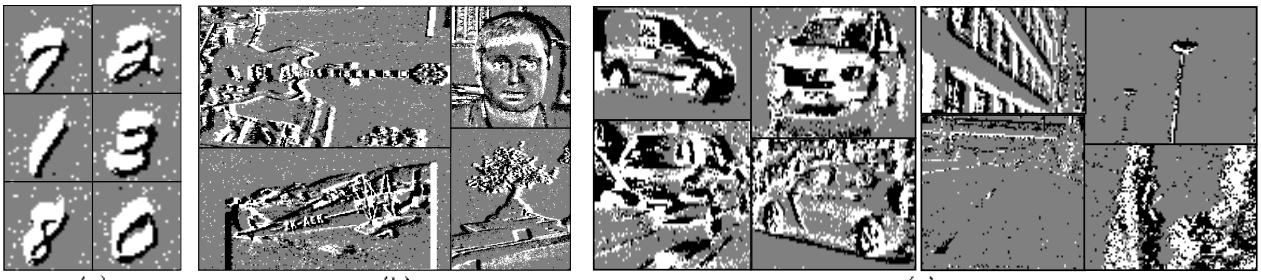

Figure 4: Sample snapshots from the datasets used for the experiments of Sec. 6. The snapshots are obtained by cumulating 100ms of events. Black pixels represents OFF events, white pixels ON events. **(a)** N-MNIST Dataset. **(b)** N-Caltech101 dataset. **(c)** N-CARS Dataset. Left: positive samples; Right: negative samples. Notice that the N-MNIST and N-Caltech101 datasets have been generated by moving an event-based camera in front of a LCD screen displaying static images. By contrast, our dataset has been acquired in real-world conditions, therefore it fully exploits the temporal resolution of the camera by capturing the real dynamics of the objects.

retrained on the whole training set.

We noticed little influence of the ρ and τ parameters to accuracy, while small K's improved performance for low resolution inputs. When the input duration is larger than the value of Δt used to compute the time surfaces (Eq. 4), we compute the features every Δt and then stack them together.

The baselines methods we consider are *HOTS* [30], *H-First* [50] and Spiking Neural Networks (*SNN*) [33, 46]. For *H-First* we used the code provided by the authors online. For *SNN* we report the results previously published, when available, while for *HOTS* we used our implementation of the method described in [30]. As with *HATS* features, we used a linear SVM on the features extracted with *HOTS*. Notice that this is in favour of *HOTS*, since linear SVM is a more powerful classifier than the one used by the authors [30].

Given that no code is available for SNN, we also compared our results with those of a 2-layer SNN architecture we implemented using predefined Gabor filters [6]. We then again train a linear SVM on the output of the network. We call this approach *Gabor-SNN*. This allowed us to obtain the results for SNN when not readily available in the literature.

Results on the Datasets Converted from Frames The results for the N-MNIST, N-Caltech101, MNIST-DVS and CIFAR10-DVS datasets are given in Tab. 1. As it is usually done, we report the results in terms of classification accuracy. The complete set of parameters used for the methods are reported in the supplementary material.

Our method has the highest classification rate ever reported for an event-based classification method. The performance improvement is higher for the more challenging N-Caltech101 and CIFAR10-DVS datasets. *HOTS* and a predefined *Gabor-SNN* have similar performance, while the *H-First* learning mechanism is too simple to reach good performance.

Results on the N-CARS Datasets For the N-CARS dataset, the *HATS* parameters used are $K = 10$, $\rho = 3$ and $\tau = 10^9 \mu s$. In this case, block normalization was not applied because it did not improve results. Since the N-CARS dataset contains only two classes, cars and non-cars, we can consider it as a binary classification problem. Therefore, we also analyze the performance of the methods using ROC curves analysis [19]. The Area Under the Curve (AUC) and the accuracy (Acc.) for our method and the baselines are shown in Tab. 2, while the ROC curves are presented in the supplementary material.

From the results, we see that our method outperforms the baselines by a large margin. The variability contained in a real-world dataset, such as the N-CARS one, is too large for both the *H-First* and *HOTS* learning algorithms to converge to a good feature representation. A predefined *Gabor-SNN* architecture has better accuracy than *H-First* and *HOTS*, but still 11% lower than our method. The spatio-temporal regularization implemented in our method is more robust to the noise and variability contained in the dataset.

6.2. Latency and Computational Time

Latency is a crucial characteristic for many applications requiring fast reaction time. In this section, we compare *HATS* , *HOTS* and *Gabor-SNN* in terms of their computational time and latency on the N-CARS dataset. All methods are implemented in C++ and run on a laptop equipped with an Intel i7 CPU (64bits, 2.7GHz) and 16GB of RAM.

Tab. 3 compares the average computational times to process a sample. Average computational time per sample was computed by dividing the total time spent to compute the features on the full training set by the number of training samples. As we can see, our method is more than 20x faster than *HOTS* and almost 40x times faster than a 2-layer SNN. In particular our method is 13 times faster than real time. We also report the average number of events processed per second in Kilo-events per second (Kev/s).

Table 1: Comparison of classification accuracy on datasets converted from frames. Our method has the highest classification rate ever reported for an event-based classification method.

	N-MNIST	N-Caltech101	MNIST-DVS	CIFAR10-DVS
H-First [50]	0.712	0.054	0.595	0.077
HOTS [30]	0.808	0.210	0.803	0.271
Gabor-SNN	0.837	0.196	0.824	0.245
HATS (this work)	**0.991**	**0.642**	**0.984**	**0.524**
Phased LSTM [46]	0.973	-	-	-
Deep SNN [33]	0.987	-	-	-

Table 2: Comparison of classification results on the N-CARS dataset. The table reports the global classification accuracy (Acc.) and the AUC score (the higher the better). Our method outperforms the baselines by a large margin.

N-CARS	Acc.	AUC
H-First [50]	0.561	0.408
HOTS [30]	0.624	0.568
Gabor-SNN	0.789	0.735
HATS (this work)	**0.902**	**0.945**

Latency represents the time period used to accumulate evidence in order to reach a decision on the object class. In our case, this time period is given by the time window used to compute the features, as longer time windows results in higher latency. Notice that with this definition, the latency is independent from both the computational time and the classification accuracy.

There is a trade-off between latency and classification accuracy: on one side longer time periods yield more information at the cost of higher latency, on the other side they lead to risk of mixing dynamics from separate objects or even different dynamics from the same object. We study this trade-off by plotting the accuracy as a function of the latency for the different methods (Fig. 5). The results were averaged over 5 repetitions. By using only 10ms of events, *HATS* has higher performance than the baselines applied to the full 100ms events stream. The performance of *HATS* does not completely saturate, probably due to the presence of cars with really small apparent motion in the dataset.

We also notice that the performance of *Gabor-SNN* is unstable, especially for low latency. This is due to the spiking architecture of *Gabor-SNN* for which small variations in the input of a layer can cause large differences at its output.

7. Conclusion and Future Work

In this work, we presented a new feature representation for event-based object recognition by introducing the notion of Histograms of Averaged Time Surfaces. It validates the idea that information is contained in the relative time between events, provided a regularization scheme is introduced to limit the effect of noise. The proposed architecture makes efficient use of past information by using local memory units shared by neighboring pixels, outperforming existing spike based methods in both accuracy and efficiency.

In the future, we plan to extend our method by using a feature representation also for the memory units, instead of using raw events. This could be done for example by training a network to learn linear weights to apply to the incoming time surfaces.

Table 3: Average computational times per sample (the lower the better) and average number of events processed per second, in Kilo-events per second Kev/s (the higher the better), on the N-CARS dataset. Since each sample is 100ms long, our method is more than 13 times faster than real time, while *HOTS* and *Gabor-SNN* are respectively 1,5 and 2,8 times slower than real time.

N-CARS	Average Comp. Time per Sample (ms)	Kev/s
HOTS [30]	157.57	25.68
Gabor-SNN	285.95	14.15
HATS (this work)	**7.28**	**555.74**

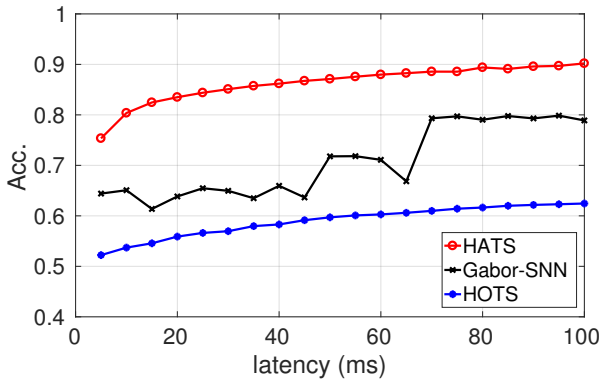

Figure 5: Accuracy as a function of latency on the N-CARS dataset. Our method is consistently more accurate than the baselines and already reaches better performance by using only events contained in the first 10ms of the samples.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *TPAMI*, 2011. 3

[2] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *TPAMI*, 2013. 3

[3] O. Bichler, D. Querlioz, S. J. Thorpe, J.-P. Bourgoin, and C. Gamrat. Extraction of temporally correlated features from dynamic vision sensors with spike-timing-dependent plasticity. *Neural Networks*, 2012. 3

[4] K. A. Boahen. Point-to-point connectivity between neuromorphic chips using address-events. *IEEE Trans. Circuits Syst. II*, 2000. 2

[5] S. M. Bohte, J. N. Kok, and H. La Poutre. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 2002. 3

[6] A. C. Bovik, M. Clark, and W. S. Geisler. Multichannel texture analysis using localized spatial filters. *TPAMI*, 1990. 7

[7] C. Brändli, J. Strubel, S. Keller, D. Scaramuzza, and T. Delbruck. Elisedan event-based line segment detector. In *Event-based Control, Communication, and Signal Processing (EBCCSP), International Conference on*, 2016. 3

[8] J. Bruna and S. Mallat. Invariant scattering convolution networks. *TPAMI*, 2013. 3

[9] Y. Cao, Y. Chen, and D. Khosla. Spiking deep convolutional neural networks for energy-efficient object recognition. *IJCV*, 2015. 3

[10] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal. Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In *CVPR*, 2009. 3

[11] X. Clady, S.-H. Ieng, and R. Benosman. Asynchronous event-based corner detection and matching. *Neural Networks*, 2015. 3

[12] X. Clady, J.-M. Maro, S. Barré, and R. B. Benosman. A motion-based feature for event-based pattern recognition. *Frontiers in neuroscience*, 2017. 3

[13] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *CVPR*, 2005. 2, 5

[14] T. Delbrück, B. Linares-Barranco, E. Culurciello, and C. Posch. Activity-driven, event-based vision sensors. In *Proc. IEEE International Symposium on Circuits and Systems*, 2010. 1

[15] T. Delbrück and C. Mead. An electronic photoreceptor sensitive to small changes in intensity. In *NIPS*, 1989. 1

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009. 2, 3

[17] 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 *International Joint Conference on Neural Networks*, 2015. 3

[18] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *TPAMI*, 2012. 2

[19] T. Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 2006. 7

[20] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *TPAMI*, 2006. 6

[21] R. Gütig and H. Sompolinsky. The tempotron: a neuron that learns spike timing–based decisions. *Nature neuroscience*, 2006. 3

[22] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 2006. 3

[23] Y. Hu, H. Liu, M. Pfeiffer, and T. Delbruck. Dvs benchmark datasets for object tracking, action recognition, and object recognition. *Frontiers in neuroscience*, 2016. 3

[24] N. Kasabov, K. Dhoble, N. Nuntalid, and G. Indiveri. Dynamic evolving spiking neural networks for on-line spatio- and spectro-temporal pattern recognition. *Neural Networks*, 2013. 3

[25] H. Kim, S. Leutenegger, and A. J. Davison. Real-time 3d reconstruction and 6-dof tracking with an event camera. In *ECCV*, 2016. 3

[26] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[27] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009. 6

[28] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS*, 2012. 2, 3

[29] X. Lagorce, S.-H. Ieng, X. Clady, M. Pfeiffer, and R. B. Benosman. Spatiotemporal features for asynchronous event-based data. *Frontiers in neuroscience*, 2015. 3

[30] X. Lagorce, G. Orchard, F. Galluppi, B. E. Shi, and R. B. Benosman. Hots: a hierarchy of event-based time-surfaces for pattern recognition. *TPAMI*, 2017. 2, 3, 4, 7, 8

[31] I. Laptev. On space-time interest points. *IJCV*, 2005. 2

[32] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In *NIPS*, 1989. 3, 6

[33] J. H. Lee, T. Delbruck, and M. Pfeiffer. Training deep spiking neural networks using backpropagation. *Frontiers in neuroscience*, 2016. 2, 3, 7, 8

[34] H. Li, G. Li, and L. Shi. Classification of spatiotemporal events based on random forest. In *Advances in Brain Inspired Cognitive Systems: International Conference*, 2016. 3

[35] H. Li, H. Liu, X. Ji, G. Li, and L. Shi. Cifar10-dvs: An event-stream dataset for object classification. *Frontiers in neuroscience*, 11:309, 2017. 3, 6

[36] P. Lichtsteiner, C. Posch, and T. Delbruck. A 128x128 120db 15us latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid State Circuits*, 2008. 1, 2

[37] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014. 2, 3

[38] B. Linares-Barranco, T. Serrano-Gotarredona, L. A. Camuñas-Mesa, J. A. Perez-Carrasco, C. Zamarreño-Ramos, and T. Masquelier. On spike-timing-dependent-plasticity, memristive devices, and building a self-learning visual cortex. *Frontiers in neuroscience*, 2011. 3

[39] H. Liu, D. P. Moeys, G. Das, D. Neil, S.-C. Liu, and T. Delbrück. Combined frame-and event-based detection and tracking. In *Circuits and Systems, 2016 IEEE International Symposium on*, 2016. 2

[40] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999. 2

[41] D. Martí, M. Rigotti, M. Seok, and S. Fusi. Energy-efficient

neuromorphic classifiers. *Neural computation*, 2016. 3

[42] T. Masquelier and S. J. Thorpe. Unsupervised learning of visual features through spike timing dependent plasticity. *PLoS computational biology*, 2007. 3

[43] E. Mueggler, C. Bartolozzi, and D. Scaramuzza. Fast event-based corner detection. In *BMVC*, 2017. 3

[44] 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. *The International Journal of Robotics Research*, 2017. 3

[45] D. Neil, M. Pfeiffer, and S.-C. Liu. Learning to be efficient: Algorithms for training low-latency, low-compute deep spiking neural networks. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*. ACM, 2016. 3

[46] D. Neil, M. Pfeiffer, and S.-C. Liu. Phased lstm: Accelerating recurrent network training for long or event-based sequences. In *NIPS*, 2016. 2, 3, 7, 8

[47] 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. *Frontiers in neuroscience*, 2013. 3

[48] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in Neuroscience*, 2015. 3, 6

[49] G. Orchard, D. Matolin, X. Lagorce, R. Benosman, and C. Posch. Accelerated frame-free time-encoded multi-step imaging. In *Circuits and Systems, 2014 IEEE International Symposium on*, 2014. 2

[50] G. Orchard, C. Meyer, R. Etienne-Cummings, C. Posch, N. Thakor, and R. Benosman. Hfirst: A temporal approach to object recognition. *TPAMI*, 2015. 3, 7, 8

[51] X. Peng, B. Zhao, R. Yan, H. Tang, and Z. Yi. Bag of events: An efficient probability-based feature extraction method for aer image sensors. *IEEE transactions on neural networks and learning systems*, 2017. 3

[52] 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. *Solid-State Circuits, IEEE Journal of*, 2011. 1, 2

[53] C. Posch, T. Serrano-Gotarredona, B. Linares-Barranco, and T. Delbruck. Retinomorphic event-based vision sensors: Bioinspired cameras with spiking output. *Proceedings of the IEEE*, 2014. 1

[54] H. Rebecq, T. Horstschaefer, G. Gallego, and D. Scaramuzza. Evo: A geometric approach to event-based 6-dof parallel tracking and mapping in real time. *IEEE Robotics and Automation Letters*, 2017. 3

[55] J. Redmon and A. Farhadi. YOLO9000: better, faster, stronger. *CoRR*, 2016. 6

[56] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 6

[57] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *ICCV*, 2011. 2

[58] B. Rueckauer, I.-A. Lungu, Y. Hu, and M. Pfeiffer. Theory and tools for the conversion of analog to spiking convolutional neural networks. *arXiv preprint arXiv:1612.04052*, 2016. 3

[59] A. Russell, G. Orchard, Y. Dong, Ş. Mihalas, E. Niebur, J. Tapson, and R. Etienne-Cummings. Optimization methods for spiking neurons and networks. *IEEE transactions on neural networks*, 2010. 3

[60] M. Scherer, M. Walter, and T. Schreck. Histograms of oriented gradients for 3d object retrieval. In *WSCG*, 2010. 2

[61] S. Seifozzakerini, W.-Y. Yau, B. Zhao, and K. Mao. Event-based hough transform in a spiking neural network for multiple line detection and tracking using a dynamic vision sensor. In *BMVC*, 2016. 3

[62] R. Serrano-Gotarredona, T. Serrano-Gotarredona, A. Acosta-Jimenez, and B. Linares-Barranco. A neuromorphic cortical-layer microchip for spike-based event processing vision systems. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2006. 6

[63] T. Serrano-Gotarredona and B. Linares-Barranco. A 128 x 128 1.5% contrast sensitivity 0.9% fpn 3 μs latency 4 mw asynchronous frame-free dynamic vision sensor using transimpedance preamplifiers. *Solid-State Circuits, IEEE Journal of*, 2013. 1, 2, 6

[64] T. Serrano-Gotarredona and B. Linares-Barranco. Poker-dvs and mnist-dvs. their history, how they were made, and other details. *Frontiers in neuroscience*, 2015. 3

[65] S. Sheik, M. Pfeiffer, F. Stefanini, and G. Indiveri. Spatio-temporal spike pattern classification in neuromorphic systems. In *Biomimetic and Biohybrid Systems*. 2013. 3

[66] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *TPAMI*, 2017. 2, 3

[67] J. Sivic and A. Zisserman. Efficient visual search of videos cast as text retrieval. *TPAMI*, 2009. 2

[68] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 2014. 3

[69] 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. *Frontiers in neuroscience*, 2017. 3

[70] C. Tan, S. Lallee, and G. Orchard. Benchmarking neuromorphic vision: lessons learnt from computer vision. *Frontiers in neuroscience*, 2015. 3

[71] V. Vasco, A. Glover, and C. Bartolozzi. Fast event-based harris corner detection exploiting the advantages of event-driven cameras. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, 2016. 3

[72] P. Viola and M. J. Jones. Robust real-time face detection. *IJCV*, 2004. 2

[73] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *BMVC*, 2009. 2

[74] A. Witkin. Scale-space filtering: A new approach to multi-scale description. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on.*, 1984. 3

[75] S. Xie and Z. Tu. Holistically-nested edge detection. *IJCV*, 2017. 3

[76] B. Zhao, R. Ding, S. Chen, B. Linares-Barranco, and H. Tang. Feedforward categorization on aer motion events using cortex-like features in a spiking neural network. *IEEE transactions on neural networks and learning systems*, 2015. 3