A homeostatic gain control mechanism to improve event-driven object recognition

Antoine Grimaldi
Institut de Neurosciences de la Timone (UMR 7289), Aix Marseille Univ, CNRS,
Marseille, France
antoine.grimaldi@univ-amu.fr

Victor Boutin
Institut de Neurosciences de la Timone (UMR 7289), Aix Marseille Univ, CNRS,
Marseille, France
victor.boutin@univ-amu.fr

Laurent Perrinet
Institut de Neurosciences de la Timone (UMR 7289), Aix Marseille Univ, CNRS,
Marseille, France
laurent.perrinet@univ-amu.fr

Sio-Hoi Ieng
Sorbonne Université, INSERM, CNRS, Institut de la Vision
Paris, France
sio-hoi.ieng@upmc.fr

Ryad Benosman
Sorbonne Université, INSERM, CNRS, Institut de la Vision
Paris, France
ryad.benosman@upmc.fr

***Abstract*—We propose a neuromimetic architecture able to perform pattern recognition. To achieve this, we extended the existing event-based algorithm from [1] which introduced novel spatio-temporal features: time surfaces. Built from asynchronous events acquired by a neuromorphic camera, these time surfaces allow to code the local dynamics of a visual scene and create an efficient hierarchical event-based pattern recognition architecture. Inspired by biological findings and the efficient coding hypothesis, our main contribution is to integrate homeostatic regulation into the Hebbian learning rule. Indeed, in order to be optimally informative, average neural activity within a layer should be equally balanced across neurons. We used that principle to regularize neurons within the same layer by setting a gain depending on their past activity and such that they emit spikes with balanced firing rates. The efficiency of this technique was first demonstrated through a robust improvement in spatio-temporal patterns which were learnt during the training phase. In order to compare with state-of-the-art methods, we replicated past results on the same dataset as [1] and extended results in this study to the widely used N-MNIST dataset [2]. We expect to extend this fully event-driven approach to more naturalistic tasks, notably for ultra-fast object categorization.**

***Index Terms*—event-based vision, homeostasis, pattern recognition, efficient coding**

I. Introduction

Bio-inspired engineering aims at taking advantage of our understanding of the complex and impressively efficient mechanisms found in Nature. Event-based cameras perfectly illustrate this process. Also called silicon retinas, these sensors are inspired by biological retinas and allow capturing luminous information asynchronously. Unlike its classical frame-based counterpart, an event-based camera responds to the scene's dynamics in a pixel-wise fashion: when a light change is detected, an event is emitted. The event is labeled with an ON or OFF polarity whether it corresponds, respectively, to an increase or decrease in brightness (see Figure 1). Event-based cameras offer various advantages and notably: a high temporal resolution, energy efficiency, reduction of redundancy, and a high dynamic range. Numerous interesting applications and use cases of event-based cameras are nowadays flourishing in the scientific community (for a review, see [3]). This new technology, along with its Address Event Representation (AER) specification [4], can bring a paradigm shift in the way dynamical visual information is processed.

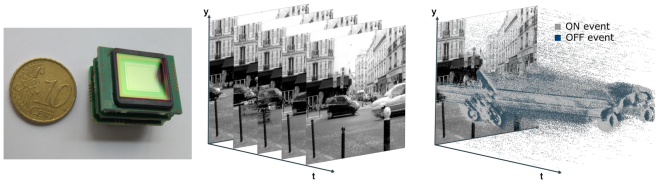

Fig. 1: A miniature event-based ATIS sensor (left) which, compared to classical frame-based representations (middle), outputs an event-based representation of the scene (right).

Interesting solutions were found to solve classical computer vision problems like optical flow [5–7], 3D reconstruction [8–10] or the Simultaneous Localization and Mapping (SLAM) problem [11, 12]. In this work, we focus on event-driven pattern recognition and in particular digit recognition as defined by the event-based, augmented version of MNIST called N-MNIST [2]. Some existing approaches train Artificial Neural Networks (ANN) and then convert them into Spiking Neural Networks (SNN) [13–15] with overall good classification results. Alternatively, some competitive event-driven algorithms are developed using backpropagation adapted for SNN [16–18]. More recently, it was proposed to use saccades to boost object recognition [19]. In [1], object recognition is achieved through a feedforward hierarchical architecture using time surfaces: a spatio-temporal object making use of the local dynamics of a scene. Then, these are assembled in a Hierarchy of Time-Surfaces (HOTS). Using a form of Hebbian learning, the network is able to learn in an unsupervised way

progressively more complex spatio-temporal features which appear on the event stream. It was shown to make accurate predictions on a letter and digit dataset [20], on a flipped card dataset [21] and on a dataset of scenes with faces.

We identified some limitations in the HOTS algorithm [1]. For instance, an unequal activation of the different features during learning can lead to a poor variety of time surfaces and consequently to a loss in efficiency. While it is controllable in simplified datasets, we observed a performance drop when replicating HOTS in order to learn to classify digits from the N-MNIST dataset. In [22], robustness was achieved by averaging time surfaces gathered in a temporal window Δt. Here, we propose the use of a simple and bio-plausible homeostatic gain control mechanism. Unsupervised learning of the features is qualitatively improved by balancing activations of the different neurons within the same layer. The robustness of the classification is tested over different amounts of spatial and temporal noise added to the input event stream. A full implementation of the algorithm is available at https://github.com/SpikeAI/HOTS and allows to reproduce all results presented in this paper, and we will give links to reproducible notebooks within this text. As a summary, we extended an existing event-driven object categorization algorithm by including a simple bio-plausible regulation rule in its training. We show qualitative and quantitative improvements on the unsupervised learning of time surfaces and on the resulting classification performances.

II. Methods

A. Datasets

To load the events, we use the community-built *tonic* python package. It currently offers the possibility to load five different event-based datasets and is based on the PyTorch language [23]. This allows to load event streams in a standard fashion and to optionally apply data augmentation methods to the event streams. Once loaded, an event-based camera recording is a $N_{ev} \times 4$ matrix where N_{ev} represents the number of events and the 4 columns are respectively for the x and y positions on the pixel grid, the time and polarity values. Timestamps are in microseconds and polarities are 0 and 1 respectively for OFF and ON events.

1) Poker-DVS: Poker-DVS [24] is one of the first publicly available DVS recordings from a real-world scene and was used to test performances of [1]. It consists of 131 occurrences of the four different card symbols as extracted from 3 separate DVS recordings while browsing very quickly poker cards. The sensor size is 31×31 pixels and recordings last between 10 to 30 ms. In *tonic*, there is an available training set of 48 samples and a testing set of 20 samples.

2) N-MNIST: The widely used N-MNIST dataset [2] was recorded by moving an event-based camera in front of a screen on which digitalized MNIST digits [25] were projected. MNIST digits are originally 28×28 pixels and they were resized to project to 28×28 pixels of the *Asynchronous Time Based Image Sensor* (ATIS) camera [26]. For N-MNIST, *tonic* registered maximum values of x and y are equal to 34 due to the saccades of the camera, such that sensor size is 34×34 pixels.

3) DVS_barrel: The DVS_barrel dataset [20] is a collection of digits and letters captured by a DVS. Movement (and thus events) is created by a rotating barrel with printed characters. The dataset is composed of 76 samples, 36 samples, one per class, for the training set, and 40 samples for the testing set. Each sample of DVSbarrel is a 32×32 pixels event-based recording. Unlike past datasets, DVS_barrel is not included in the *tonic* package.

B. HOTS algorithm

The HOTS network is fully described in [1] and we give here its guiding principles. The network can be assimilated to a Spiking Convolutional Neural Network (SCNN).

The input of the network is a stream of ON or OFF events. An event, or spike, δ_i is defined by its timing t_i, and its address $a_i = (x_i, y_i, \mathbf{p}_i)$, where x_i and y_i are the coordinates on the pixel grid and $\mathbf{p}_i$ is 0 or 1 respectively for an OFF or ON event. To make use of the temporal dynamics of this flux of events, [1] introduces time surfaces. Time surfaces are local matrices of analog values obtained by applying an exponential decay to the event stream. A time surface is defined locally in time relatively to the constant τ of the exponential decay and in space by a spatial window. In order to use the information given by the two polarities of the event stream, contributions of ON and OFF events will be stored in two distinct matrices. A time surface is thus represented by a 3D matrix of dimensions $(2R+1) \times (2R+1) \times 2$ where R is the parameter that defines the size of the spatial window and 2 is the number of polarities. Specifically, for an event δ_i emitted at time t_i in the network at the address a_i, analog values assigned to the time surface defined around this event are:

$$S_i(a_{ts_i}) = e^{-\frac{t_i - \bar{t}(a_{ts_i})}{\tau}}$$

where τ is a time constant and $\bar{t}(a_{ts_i})$ is the time of the last event recorded at address a_{ts_i} which is defined as follow:

$$a_{ts} = (x_{ts_i}, y_{ts_i}, \mathbf{p}) \left| \begin{array}{l} x_{ts_i} \in [x_i - R, x_i + R] \\ y_{ts_i} \in [y_i - R, y_i + R] \\ \mathbf{p} \in [0, 1] \end{array} \right. .$$

This 3D matrix is then sent to the first layer of the network and compared to weight matrices $W_{\mathbf{n}}$ associated to each of the N neurons of the layer. This match is computed as the normalized scalar product (or correlation coefficient):

$$\beta_{\mathbf{n}} = \frac{\langle W_{\mathbf{n}}, S_i \rangle}{\|W_{\mathbf{n}}\| \cdot \|S_i\|}$$

As a consequence, each $W_{\mathbf{n}}$ acts as a convolution kernel, but computations need only to be performed locally around address a_i of event δ_i to produce a vector of the possible matches $\beta_{\mathbf{n}}$. Then, the neuron which has its weights matrix the most similar to the input time surface S_i produces an event. Its index is simply $\mathbf{n}^* = \arg\max_{\mathbf{n}} \beta_{\mathbf{n}}$. The output of the first layer is also an event stream that keeps the temporal, t_i,

and spatial, x_i and y_i, information of the incoming events but modifies its polarity to match that of the best matching neuron: $\mathbf{n}^*$. As a consequence, neurons within a layer $\mathbf{L}$ are competing to produce spikes and each incoming event produces a post-synaptic event. Besides, once a postsynaptic event is chosen, a Hebbian-like mechanism is used to take the selected weight matrix closer to the observed time surface. This mechanism is similar in principle to that used by the k-means algorithm of other unsupervised learning schemes [27]. More generally, each layer takes input events from its previous layer and feeds events to the next by reproducing the steps described above. The number of neurons, the time constant and the size of time surfaces will increase when passing from one layer to the next. As a consequence, the network will learn more and more complex spatio-temporal features hierarchically. The output of the last layer is finally fed to a classifier layer which will recognize the object in the input. In the methodology defined in [1], classification is performed using a "bag-of-words representation" by accumulating output spikes of the last layer of the network in a histogram representation (see below).

Up to the classification layer, the algorithm that is presented here is fully event-driven, i.e. computations are performed only when an incoming event is observed. When observing the definition of the time surfaces and the structure of the network, one can see the equivalence to a SNN of leaky integrate-and-fire (LIF) models in analogy with a SCNN.

C. Homeostasis

In this work, we follow the same method defined in [1] but complement it by using for each layer a homeostatic gain $\gamma_{\mathbf{n}}$ controlling for the activation of neurons during the learning. Such mechanisms of regulation are generally observed in living systems and are well justified in terms of efficient coding [28]. We use the simple heuristic derived in [29] to redefine the postsynaptic activation as $\beta_{\mathbf{n}} = \gamma_{\mathbf{n}} \cdot \frac{\langle W_{\mathbf{n}}, S_i \rangle}{\|W_{\mathbf{n}}\| \cdot \|S_i\|}$ with $\gamma_{\mathbf{n}} = e^{\lambda \cdot (p_{\mathbf{n}} \cdot N_{\mathbf{L}})}$, where λ is a regularization parameter, $p_{\mathbf{n}}$ is the relative activation frequency of neuron $\mathbf{n}$ and $N_{\mathbf{L}}$ the total number of features of the layer. The homeostatic gain rule was found empirically such that $\gamma_{\mathbf{n}} > 1$ if neuron $\mathbf{n}$ is less activated than the average $\frac{1}{N_{\mathbf{L}}}$ on the layer and $\gamma_{\mathbf{n}} < 1$ in the opposite case. The activation frequency of the neuron is measured by simple counting of the history of its activations. This regulation rule allows to train the different neurons in a balanced fashion and avoid the response of some of them for too specific features. When the activity of neurons within a layer is balanced, we reach the efficient coding hypothesis, making our algorithm more efficient while being bio-plausible and robust. Note that once this equilibrium state is reached, $\gamma_{\mathbf{n}} = 1$ and the coding is similar to that of HOTS.

In practice, we observed that adding homeostasis to balance the activation of neurons allows avoiding heuristics to reach convergence. In [1], weight matrices or synaptic weights associated with each neuron are initialized with the first incoming time surfaces. This method makes the learning of weight matrices very sensitive to initialization. In addition, the hierarchy is learnt sequentially, one layer after the other. Here, they are initialized at random and allow spikes to cross each layer when they enter the network even if a layer is not fully trained. This method for the unsupervised learning phase makes our method more similar to the conditions faced by living systems.

D. Classifier

In the original HOTS algorithm [1], classification is performed by comparing the activation histograms across the neurons of the last layer of the network to that observed on average for each given class. We noticed a drop of efficiency for this method when datasets have a large number of training samples and decided to use a simple k-Nearest Neighbors (k-NN) as done in [30]. In practice, we first trained the hierarchical network using unsupervised learning, then we recorded activation histograms for each training sample, and finally performed a weighted k-NN with euclidean distance on the activation histogram of the tested sample.

To test for the robustness of the proposed algorithm, we used the *tonic* package to transform and augment the dataset. In particular, it allows the addition of spatial or temporal jitter to the input stream. As relevant information is supposed to be represented by the timing and position of input spikes, we can assume that classification performance should get worse as the jitter increases. Therefore, we will use this module to test differentially the robustness of the algorithms by progressively adding some noise to the input signal. To do so, we use a subset composed of 1000 randomly selected samples from the N-MNIST testing set. We keep the exact same subset and apply different amounts of spatial and temporal jitter independently.

III. Results

A. Replication of HOTS

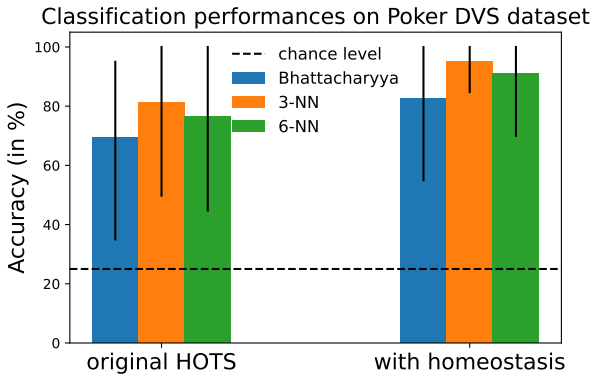

Fig. 2: Accuracy scores with and without homeostasis averaged over 100 trials with different clusterings for each trial (chance level is at 25%). We use the Bhattacharyya distance to compare with the averaged histogram per class, and k-NN are performed with one histogram per training sample. Error bars represent the 5% and 95% quantiles.

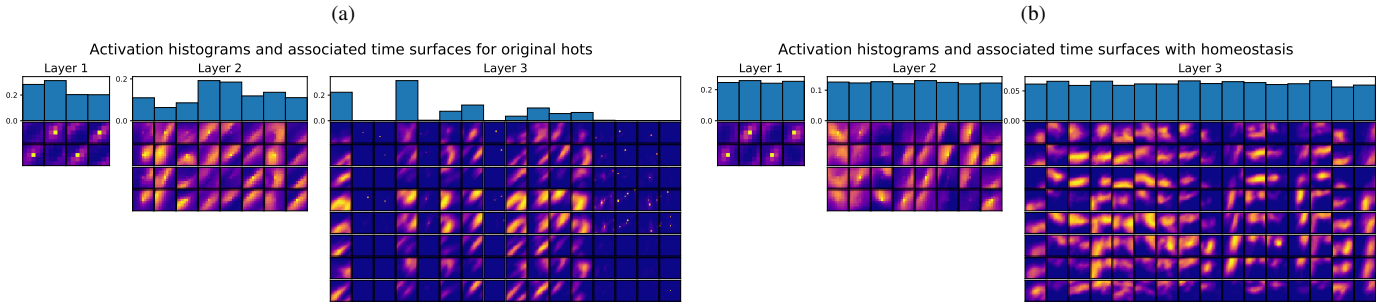

Fig. 3: Activation histograms and time surfaces obtained in the unsupervised learning algorithm for the original HOTS network (a) (replicated from [1]) and for the bio-plausible version with homeostasis (b). Associated time surfaces are plotted below histogram bins. The different lines are the different polarities of the features (ON and OFF for the first layer), that is, the output neurons of the previous layer for the next one. Note the unbalanced histograms in the left figure. Some time surfaces are so rarely activated that they remain close to the first inputs they were initialized with. These unevenly matured clusters lead to an inefficient coding within the network.

We had access to two datasets, DVS_barrel and Poker-DVS used in [1] and followed the original methodology to replicate the results.

For DVS_barrel, we were able to obtain comparable results with a small drop of performances: we reached an accuracy of 98% instead of 100% announced in [1]. The classification was performed as in [1] with histogram comparison using the Bhattacharyya distance. Results for classification in this dataset are shared in an online notebook: RESULTS_01_SimpleAlphabet. We obtain similar accuracy values close to the maximum, such that the impact of the homeostatic gain is not significant.

For Poker-DVS, we were not able to replicate past results with the original parameters and had to perform parameter tuning to obtain the best performance of the original method. In order to reach 95% recognition accuracy close to the results obtained in [1], we changed the time constant for $\tau = 70$ µs and used a k-NN with k=3. With these parameters, accuracy improved and reached 100% when including homeostasis. In general, poorer performances were obtained when using one histogram per class with Bhattacharyya distance instead of one histogram per sample with k-NN. With this small dataset, we could easily test the impact of clustering over classification. We repeated the unsupervised learning of the features and then the training and testing of the network to check for classification variability with different clustering results. Figure 2 illustrates this variability. Note the larger error bars when using the original method compared to the higher average accuracy and the more stable performance when adding the homeostasis rule.

One can infer that clustering is highly dependent on its initialization and then, given the fact that it is initialized with the first time surfaces as input, the original method will have greater variability in clustering. This directly impacts accuracy as is shown in the comparison with the method using random initialization. Transferring the past methodology to a more bio-plausible one makes it more robust in terms of classification performances. Parameters tuning and results are shared in RESULTS_02_PokerDVS.

B. Testing the method on a widely used dataset: N-MNIST

Once the original method compared to ours with the datasets used in [1], we will now extend the study to a commonly used dataset: N-MNIST. The rest of the study will give results using a k-NN classifier (with $k \in \{1, 12\}$) as it is applied on the space of activation histograms [30].

For this dataset, we first show results of the clustering phase, i.e. the time surfaces learnt by the different neurons in each layer of the network. In figure 3 (a), we can observe the time surfaces, represented by weight matrices, as they are learnt by the network without homeostasis. Different blocs correspond to different layers, and within a layer, a column corresponds to a single neuron. In a column, lines correspond to different polarities for each associated time surface in the previous layer: In the hierarchy of the network, polarity number $\mathbf{p}$ in layer $\mathbf{L}$ is linked to polarity number $\mathbf{p}'$ of layer $\mathbf{L}-1$. On top of the plot for each layer, we show activation histograms for the different neurons. Notice the unbalanced activations and learning of the neurons for the original method. In figure 3 (a), neurons that are never or rarely activated have kernels that remain close to the time surface they were initialized with, meaning that they did not learn from enough inputs. Thus, they will respond only to very specific time surfaces, probably the ones they were initialized with, like grand-mother cells. When trying initialization with different, randomly selected, digits, great variability was observed in learnt time surfaces. The network clustering is not stable and leads to significant differences in encoding for the same dataset. This can result in a great variability of performances as observed in figure 2. If initialized at random and using the same methodology, only a few kernels have structured patterns after clustering and most kernels remain at their initial, random state.

Homeostasis prevents the network from following this behavior and balances the learning between all neurons within the same layer (see figure 3 (b)). It leads to more generic

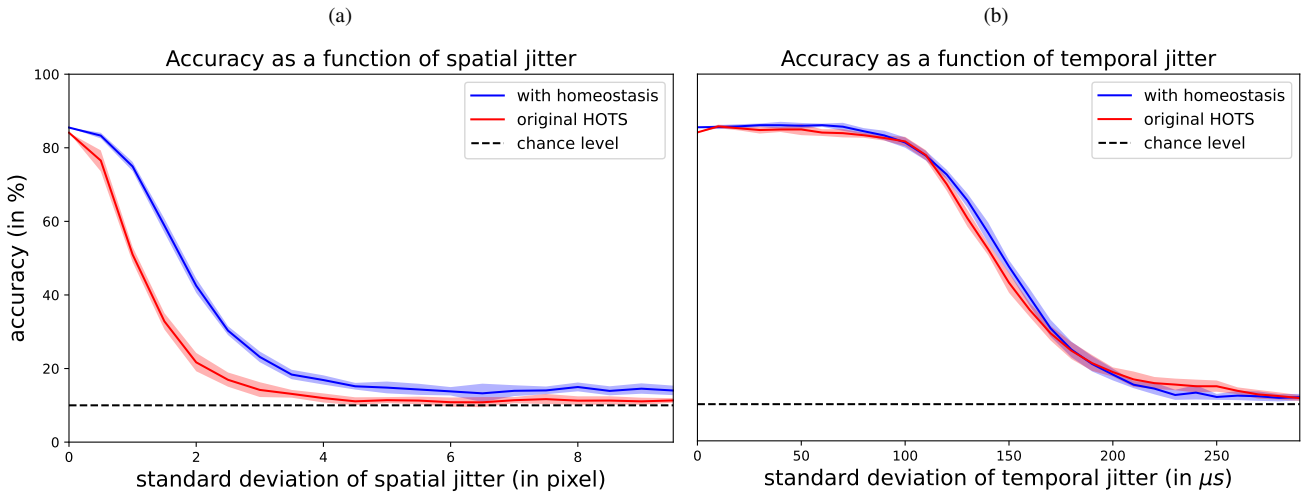

Fig. 4: Classification performances on N-MNIST using histogram distances and as a function of (a) spatial or (b) temporal jitter. Blue curves show results of the HOTS method with homeostasis and these are compared to the performance of the original HOTS algorithm (Red curves). Transparent outlines represent the 5% and 95% quantiles.

kernels which are equally activated, in line with the efficient coding hypothesis. Similar results can be observed with previously used datasets, as shown in notebooks RESULTS_01_SimpleAlphabet and RESULTS_02_PokerDVS. Note that our study still gives results far behind the state-of-the-art and does not improve significantly the accuracy of the original algorithm. However, this work achieves a slightly better performance while being more bio-plausibility. Moreover, it is simpler and does not need to include heuristic rules, suggesting better robustness.

Figure 4 represents classification performances as a function of both spatial (a) and temporal (b) jitter. Jitter is applied only on the testing set to add noise to the signal used for classification. With no jitter, the overall advantage is for the method with homeostasis, though the difference is only slightly significant. As expected, the higher the jitter, the stronger its negative impact on classification. The drop in accuracy as a function of jitter approximately follows a sigmoïd function reaching chance level (i.e. 10 % accuracy), such that one can define a critical standard deviation of jitter in pixel or in μs where accuracy drops to half its maximal value compared to chance level. This half-saturation level reveals a signature value for the relevant information contained in the signal.

For figure 4 (b), mean accuracy for the original method and ours reaches its half-saturation level respectively for a temporal jitter with a standard deviation equal to 145 µs and 150 µs. These values are very similar showing that HOTS is robust to temporal jitter. This robustness can originate from the use of time surfaces for signal encoding. Indeed, we compare the signal as input to smooth time surfaces with a scalar product on the whole spatial window gathering a subset of recent events. This technique makes the encoding of input events more robust to local temporal variations.

In figure 4 (a) we focus on spatial jitter and one can observe a shift to the right for the blue curve. For the original HOTS method and ours respectively, half-saturation is reached for a spatial jitter with a standard deviation of 1 pixel and 2 pixels. It means that homeostasis gives more resilience to the network. This resilience can come from the better clustering phase of the method with homeostasis, allowing more robust encoding.

IV. Discussion

To summarize, we have proposed to include a bio-inspired homeostatic mechanism within the existing HOTS algorithm. By testing the efficiency of the original and modified algorithms, we have demonstrated that this additional mechanism is beneficial to the robustness of the network's learning, as shown in the learnt time surfaces but also quantitatively as it impacts positively the classification performance.

This biologically plausible homeostatic regulation rule allows reaching higher and more stable performances on all the datasets studied in this work. In particular, we were able with this study to observe resilience to different types of noise in the N-MNIST classification task. Our implementation of the [1] algorithm and of the extensions proposed in this study is fully available at https://github.com/SpikeAI/HOTS.

This work opens the way to further scientific perspectives in the domain of event-driven algorithms, and in particular in the use of time surfaces as a proxy for information processing. Future work should explore the efficiency of these kernels in capturing the relevant signal and in improving future algorithms.

Acknowledgments

Antoine Grimaldi and Laurent Perrinet received funding from the European Union ERA-NET CHIST-ERA 2018 research and innovation program under grant agreement No ANR-19-CHR3-0008-03.

References

[1] X. Lagorce et al. "HOTS: A Hierarchy of Event-Based Time-Surfaces for Pattern Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.7 (2017), pp. 1346–1359.

[2] G. Orchard et al. "Converting static image datasets to spiking neuromorphic datasets using saccades". In: *Frontiers in neuroscience* 9 (2015), p. 437.

[3] G. Gallego et al. "Event-based vision: A survey". In: *arXiv preprint arXiv:1904.08405* (2019).

[4] K. A. Boahen. "Point-to-point connectivity between neuromorphic chips using address events". In: *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* 47.5 (2000), pp. 416–434.

[5] R. Benosman et al. "Event-based visual flow". In: *IEEE transactions on neural networks and learning systems* 25.2 (2013), pp. 407–417.

[6] S. Tschechne et al. "Bio-inspired optic flow from event-based neuromorphic sensor input". In: *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*. Springer. 2014, pp. 171–182.

[7] P. Bardow et al. "Simultaneous optical flow and intensity estimation from an event camera". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 884–892.

[8] J. Hidalgo-Carrió et al. "Learning Monocular Dense Depth from Events". In: *arXiv preprint arXiv:2010.08350* (2020).

[9] M. Osswald et al. "A spiking neural network model of 3D perception for event-based neuromorphic stereo vision systems". In: *Scientific reports* 7.1 (2017), pp. 1–12.

[10] A. Z. Zhu et al. "Realtime time synchronized event-based stereo". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 433–447.

[11] G. Gallego et al. "Event-based, 6-DOF camera tracking from photometric depth maps". In: *IEEE transactions on pattern analysis and machine intelligence* 40.10 (2017), pp. 2402–2412.

[12] H. Kim et al. "Real-time 3D reconstruction and 6-DoF tracking with an event camera". In: *European Conference on Computer Vision*. Springer. 2016, pp. 349–364.

[13] D. Neil et al. "Effective sensor fusion with event-based sensors and deep network architectures". In: *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE. 2016, pp. 2282–2285.

[14] A. Patino-Saucedo et al. "Event-driven implementation of deep spiking convolutional neural networks for supervised classification using the SpiNNaker neuromorphic platform". In: *Neural Networks* 121 (2020), pp. 319–328.

[15] C. Frenkel et al. "A 28-nm Convolutional Neuromorphic Processor Enabling Online Learning with Spike-Based Retinas". In: *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE. 2020, pp. 1–5.

[16] S. B. Shrestha et al. "Slayer: Spike layer error reassignment in time". In: *arXiv preprint arXiv:1810.08646* (2018).

[17] J. H. Lee et al. "Training deep spiking neural networks using backpropagation". In: *Frontiers in neuroscience* 10 (2016), p. 508.

[18] Y. Wu et al. "Spatio-temporal backpropagation for training high-performance spiking neural networks". In: *Frontiers in neuroscience* 12 (2018), p. 331.

[19] A. Yousefzadeh et al. "Active perception with dynamic vision sensors. Minimum saccades with optimum recognition". In: *IEEE transactions on biomedical circuits and systems* 12.4 (2018), pp. 927–939.

[20] G. Orchard et al. "HFirst: a temporal approach to object recognition". In: *IEEE transactions on pattern analysis and machine intelligence* 37.10 (2015), pp. 2028–2040.

[21] J. A. Pérez-Carrasco et al. "Mapping from frame-driven to frame-free event-driven vision systems by low-rate rate coding and coincidence processing–application to feedforward ConvNets". In: *IEEE transactions on pattern analysis and machine intelligence* 35.11 (2013), pp. 2706–2719.

[22] A. Sironi et al. "HATS: Histograms of averaged time surfaces for robust event-based object classification". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1731–1740.

[23] A. Paszke et al. "Pytorch: An imperative style, high-performance deep learning library". In: *arXiv preprint arXiv:1912.01703* (2019).

[24] T. Serrano-Gotarredona et al. "Poker-DVS and MNIST-DVS. Their history, how they were made, and other details". In: *Frontiers in neuroscience* 9 (2015), p. 481.

[25] Y. LeCun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[26] C. Posch et al. "A QVGA 143dB dynamic range asynchronous address-event PWM dynamic image sensor with lossless pixel-level video compression". In: *2010 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE. 2010, pp. 400–401.

[27] L. U. Perrinet et al. "Emergence of filters from natural scenes in a sparse spike coding scheme". In: *Neurocomputing* 58–60.C (2003), pp. 821–6.

[28] L. U. Perrinet. "Role of homeostasis in learning sparse representations". In: *Neural Computation* 22.7 (July 17, 2010), pp. 1812–36.

[29] L. U. Perrinet. "An adaptive homeostatic algorithm for the unsupervised learning of visual features". In: *Vision* 3.3 (2019), p. 47.

[30] J.-M. Maro et al. "Event-based gesture recognition with dynamic background suppression using smartphone computational capabilities". In: *Frontiers in neuroscience* 14 (2020), p. 275.