VK-SITS: Variable Kernel Speed Invariant Time Surface for Event-Based Recognition

Laure Acin[1][a], Pierre Jacob[2][b], Camille Simon-Chane[1][c] and Aymeric Histace[1][d]

[1]*ETIS UMR 8051, CY Cergy Paris University, ENSEA, CNRS, F-95000, Cergy, France*

[2]*Univ. Bordeaux, CNRS, Bordeaux INP, LaBRI, UMR 5800, F-33400 Talence, France*

Keywords: Event-based Camera, Event-based Vision, Asynchronous Camera, Machine Learning, Time-surface, Recognition.

Abstract: Event-based cameras are a recent non-conventional sensor which offer a new movement perception with low latency, high power efficiency, high dynamic range and high-temporal resolution. However, event data is asynchronous and sparse thus standard machine learning and deep learning tools are not optimal for this data format. A first step of event-based processing often consists in generating image-like representations from events, such as time-surfaces. Such event representations are proposed with specific applications. These event representations and learning algorithms are most often evaluated together. Furthermore, these methods are often evaluated in a non-rigorous way (i.e. by performing the validation on the testing set). We propose a generic event representation for multiple applications: a trainable extension of Speed Invariant Time Surface, coined VK-SITS. This speed and spatial-invariant framework is computationally fast and GPU-friendly. A second contribution is a new benchmark based on 10-Fold cross-validation to better evaluate event-based representation of DVS128 Gesture and N-Caltech101 recognition datasets. Our VK-SITS event-based representation improves recognition performance of state-of-art methods.

1 INTRODUCTION

Event-based cameras are a recent technology composed of autonomous pixels which acquire information only when they detect a brightness change in their individual field of view (Lichtsteiner et al., 2008; Posch et al., 2011). These cameras only record scene dynamics and there is no information redundancy. Other advantages of such cameras are their high dynamic range (over 120 dB), high temporal resolution ($\sim$ µs), low latency and low power consumption. For all these reasons, event-based cameras are an attractive sensor for movement recognition with efficent data processing.

Compared to standard cameras where data is dense (all pixel information is sent for each frame) and synchronous (all frames are acquired at a fixed frequency), data from event camera is sparse and asynchronous: only pixels which sense a change in brightness provide data and this data is sent as soon as a change is detected (see Figure 1). Standard image processing tools and methods from machine and deep learning are not adapted for this sparse and asynchronous data. The event-based processing community has been developing new tailored algorithms for this data (Gallego et al., 2020). A common strategy to use existing computer vision tools is to recreate image-like representations from events.

Event representation methods are usually evaluated for a given object recognition task. However, standard benchmarks are highly biased and overtuned. In most cases, the validation is performed on the testing set which makes model selection unreliable and reported results biased. Also, both training and testing sets are randomized a single time. The reported results highly depend on the random split, and might not correctly represent the benefits of the proposed method. This type of evaluation greatly limits repeatability and reproducibility, which limits the use of these methods in real-case scenarios. Finally, time-surfaces are often developed for a given deep-learning method and application; they are evaluated with the deep learning method used.

[a] https://orcid.org/0000-0001-9140-5406
[b] https://orcid.org/0000-0001-6427-5853
[c] https://orcid.org/0000-0002-4833-6190
[d] https://orcid.org/0000-0002-3029-4412

Acin, L., Jacob, P., Simon-Chane, C. and Histace, A.
VK-SITS: Variable Kernel Speed Invariant Time Surface for Event-Based Recognition.
DOI: 10.5220/0011779400003417
In *Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2023) - Volume 5: VISAPP*, pages 754-761
ISBN: 978-989-758-634-7; ISSN: 2184-4321

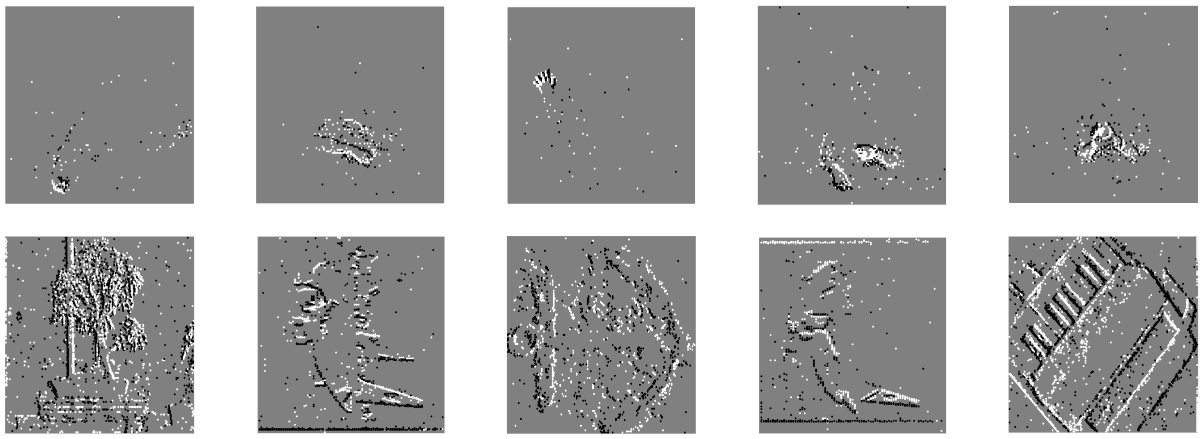
Figure 1: Samples from event-based datasets accumulated over 10 ms. Events are on a gray background and colored depending on their polarity: ON events are represented in white points and OFF events are represented in black. First row: DVS128 Gesture, from left to right air guitar, arm roll, right hand wave, air drum and hand clap classes. Second row: N-Caltech101, from left to right bonsai, seahorse, anchor, bird and truck classes.

Our proposition is twofold: First, we introduce Variable-Kernel SITS, coined VK-SITS, a representation inspired by SITS (Manderscheid et al., 2019) and EST (Gehrig et al., 2019). We use the principle of SITS to build a translation and speed-invariant time-surface while making the kernel learnable to improve its representation power. We extend this method to multiple kernel learning, unlike the models it is based on. Moreover, SITS has only been used for corner detection. We evaluate it on a recognition task, a new application for this method.

Second, we propose a new evaluation benchmark. Compared to the previous one, we fix the train-val-test splits, which ensures repeatability. Then, we evaluate the models based on a 10-Fold cross-validation. This ensures that model selection is correctly performed, real model performances are not biased, and that the results are reproducible. This also keeps the training time and complexity of the model evaluation traceable. The evaluation we propose is independent from the preprocessing and learning method used. These event representations are evaluated with the same classification network. This guarantees that we evaluate only the event representation and not the entire framework. We compare our representation to time-surface methods that are most often used in competitive applications: SITS (Manderscheid et al., 2019), TORE (Baldwin et al., 2022) and VoxelGrid (Zhu et al., 2019).

2 RELATED WORK

An increasingly deployed event representation is the "time-surface". A time-surface describes the spatial neighborhood of an event over an interval of time in two dimensions (Lagorce et al., 2017). Time-surfaces can thus be easily input to most of machine learning tools such as deep convolutional networks. However, the large majority of state-of-the-art time-surfaces, such as HOTS (Lagorce et al., 2017), HATS (Sironi et al., 2018), and SITS (Manderscheid et al., 2019), are not end-to-end trainable which limits the learning process. On the other hand, end-to-end trainable representations such as EST (Gehrig et al., 2019) are not speed-invariant, contrary to SITS. We know of no speed-invariant, fully trainable event representation, though such method would be a powerfull input for a classification network.

The rest of this section is divided into three parts: First we review the basics of event-based cameras and time-surface methods proposed in the literature following (Gehrig et al., 2019) framework. Then, we focus on SITS (Manderscheid et al., 2019) and EST (Gehrig et al., 2019) from which our method is inspired.

2.1 Time-Surface

Many event-based processing algorithms are based on a time-surface representation. This image-like representation is often the first step in recognition methods (Lagorce et al., 2017). It provides a 2D description of the past activity of an event neighborhood.

Let us start by introducing the event concept. When a brightness change, characterized by $\Delta L(\boldsymbol{x_i}, t_i) = p_i\, C$ with

$$\begin{cases} \Delta L(\boldsymbol{x_i}, t_i) = L(\boldsymbol{x_i}, t_i) - L(\boldsymbol{x_i}, t_i - \Delta t_i) \\ \hfill L = log(I) \end{cases} \tag{1}$$

happens at the pixel location $[x_i, y_i]^T$ and timestamp t_i, the camera records an event

$$\boldsymbol{e}_i = [x_i, y_i, t_i, p_i]^T. \tag{2}$$

The polarity of the event $p_i \in \{-1, 1\}$ corresponds to the sign of the brightness change. Event representations aim at encoding a sequence of N events $\mathcal{E} = \{\boldsymbol{e}_i\}_{i=1}^N$ with increasing temporality, into a meaningful form suitable for the subsequent task.

Practically, events are points in a four dimensional points manifold spanned by their spatial coordinates x and y, their timestamp t and their polarity p. Mathematically, this is represented by an event-field with a measure f:

$$\boldsymbol{F}_{\pm}(x,y,t) = \sum_{\boldsymbol{e}_i} f_{\pm}(x,y,t)\ \delta(x - x_i, y - y_i, t - t_i) \tag{3}$$

where $\boldsymbol{F}_{+}$ (resp. $\boldsymbol{F}_{-}$) is computed with events with polarity $+1$ (resp. -1). Thanks to this formulation, event-field preserves the high temporal resolution of event-based cameras, but also enforces spatio-temporal locality. Thus, designing an event representation could be resumed by designing an efficient measure function f. In particular, event polarity assumes $f_{\pm}(x,y,t) = \pm 1$, and counting events (Maqueda et al., 2018) is retrieved by using $f_{\pm}(x,y,t) = 1$. Voxel grids (Zhu et al., 2019; Mueggler et al., 2018) and leaky surfaces (Cannici et al., 2019) are also two event representations that fall into this framework.

Time-surfaces generalize Equation 3 by considering a spatio-temporal convolution kernel, such that:

$$\begin{aligned} \boldsymbol{S}_{\pm}(x,y,t) &= (k \star \boldsymbol{F}_{\pm})(x,y,t) \\ &= \sum_{\boldsymbol{e}_i} f_{\pm}(x,y,t)\ k(x - x_i, y - y_i, t - t_i) \end{aligned} \tag{4}$$

where $\star$ is the convolution operation. In order to retrieve most known event representations, this continuous function is discretized, for most of them, into the spatio-temporal coordinates (x_l, y_m, t_n) where $x_l \in \{0, 1, \ldots, W-1\}, y_m \in \{0, 1, \ldots, H-1\}$ and $t_n \in \{t_0, t_0 + \Delta\, t, \ldots, t_0 + B\Delta\, t\}$ with (H, W) the image size, $\Delta\, t$ the bin size, and B the number of bins. In particular, HOTS (Lagorce et al., 2017) can be retrieved from Equation 4 by using $k(x,y,t) = \delta(x,y)\exp(-\frac{t}{\tau})$, HATS by adding spatio-temporal average (Sironi et al., 2018), and TORE (Baldwin et al., 2022) by applying kernel $k(x,y,t) = \delta(x,y)\log(1+t)$. DART (Ramesh et al., 2020) considers a log-polar reparameterization of the time-surface and uses a soft-assignment kernel to rings and wedges.

2.2 Speed Invariant Time Surface

Time-surfaces are dependent on the speed and direction of the movement. For this reason, the Speed Invariant Time Surface (SITS) has been formalized (Manderscheid et al., 2019). The aim is to have the same silhouette for events produced by the movement of an object, whatever the speed of the object. So, when an event arrives, it puts up a large value at this position in the time-surface and reduces neighborhood values, while being independent of the time value. In such manner, values of previous events are sequentially scaled down and similar values are reduced by a constant value, the time-surface created is independent with movement's speed. Mathematically, SITS is defined as follows:

$$\boldsymbol{S}_{\pm}(x,y,t) = \sum_{\boldsymbol{e}_i} f_{\pm}(x,y,t) k(x - x_i, y - y_i) \tag{5}$$

where $k(x,y)$ is defined as a fixed 3×3 convolution kernel in (Manderscheid et al., 2019)

$$\begin{pmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{pmatrix}. \tag{6}$$

Then, the time-surface is normalized between 0 and 9 to become invariant to speed and direction of the movement.

SITS is an improvement over most recent time-surface methods, as it is the only one to consider speed invariance. However, we observe that the kernel used by SITS is arbitrarily chosen. Moreover, in frame based algorithms it is common to learn convolution kernels as it is done in CNN.

2.3 Event Spike Tensor

(Gehrig et al., 2019) proposed Event Spike Tensor (EST) as the first end-to-end learnable event representation. Following Equation 4, the convolution kernel is learned in order to find a data-driven representation best suited for the task. EST replaces the handcrafted function with a multi-layer perceptron (MLP) composed of three layers. This MLP takes the relative spatio-temporal coordinates as input, then generates an activation map around it.

Thanks to this formulation, EST is both translation-invariant and end-to-end trainable. However, their proposed formulation is not speed-invariant by design, and the training process has to discover this property. In the following, we present the variable-kernel SITS (VK-SITS), which takes advantages of both representations.

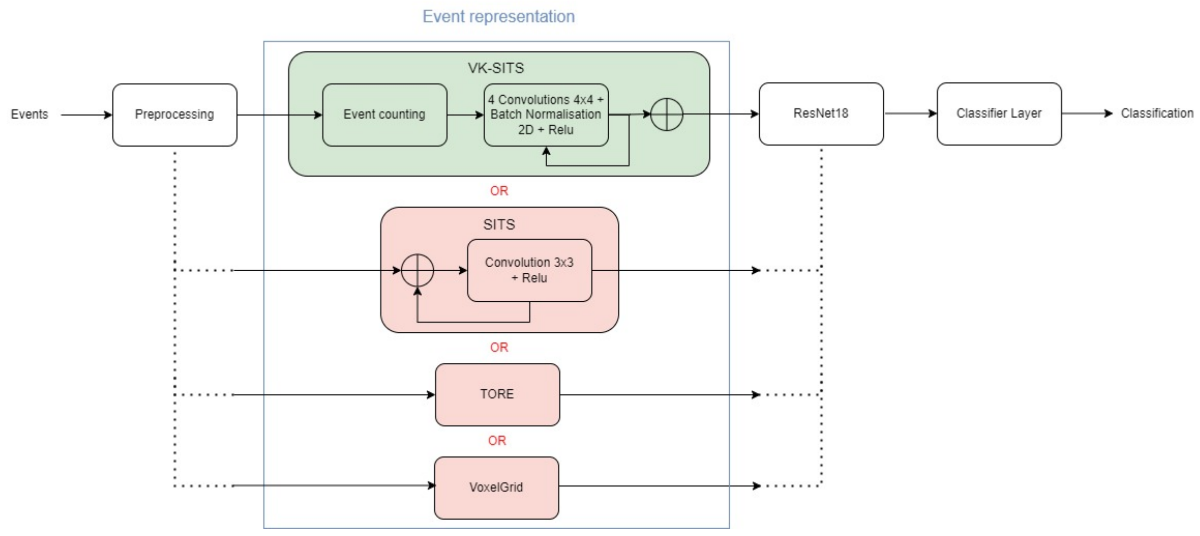

Figure 2: Overview of the paper. In green is our method, in pink are the methods with which our method is compared to. For a stream of events, we preprocess the data: first we sort the data chronologically, then we choose a window of 40 000 events randomly, we sub-sample the window by 10 and if necessary, we use padding if there isn't enough data. The second part is our method. We compare our method to TORE (Baldwin et al., 2022), SITS (Manderscheid et al., 2019) and VoxelGrid (Zhu et al., 2019) methods using the same preprocessing and neural network.

3 MATERIALS AND METHODS

In this section, we describe our method, the datasets used to evaluate our method and finally our experimental setup. The source code is publicly available for download.[1].

3.1 Variable Kernel SITS

While our method is based on the SITS kernel, we use several kernels to simultaneously focus on different features and we learn these kernels to better fit the data. Figure 2 shows an overview of our method, which we present in three steps: processing, event representation and classification.

3.1.1 Preprocessing

First, events are sorted in chronological order, in case the sequence does not respect the AER protocol (Delbrück et al., 2010). Then, we randomly choose a window of 40000 consecutive events in case sequences are not cut right at the start and end of the movement. We then sub-sample the sequence by a factor of 10 to obtain a sequence of 4000 events. Deep learning algorithms are robust to sub-sampling and we noticed that these short sequences are sufficient to perform the recognition task while decreasing the computation cost.

[1]source code: https://github.com/LaureAcin/Event-based-processing.git

3.1.2 Representation

To compute the representation, events are transformed in tensor as done in (Maqueda et al., 2018), then four kernels are applied on events: two filters per polarity.

Mathematically, we fix a tensor $\mathcal{E}_{\pm}$ which encodes the number of events per polarity in each pixel of the camera:

$$\mathcal{E}_{\pm}(x,y) \in \mathbb{N}^{H\times W} \tag{7}$$

with H and W respectively the height and the width of the event-based sensor. Then, we apply kernels $\mathcal{K}_k$ with $k \in \{0,1\}$ to create time-surfaces $\mathcal{T}_{\pm}(x,y,k)$ for each kernel $\mathcal{K}_k$ as:

$$\mathcal{T}_{\pm}(x,y,k) = \mathcal{K}_k \star \mathcal{E}_{\pm}(x,y) \tag{8}$$

We can consider an event $\boldsymbol{e}_i$ at the position (x_i, y_i) as a Dirac $\delta_{\pm}(x_i,y_i)$ and so the tensor $\mathcal{E}_{\pm}$ as a sum of Diracs:

$$\mathcal{E}_{\pm} = \sum_{\boldsymbol{e}_i}(\delta_{\pm}(x_i,y_i)) \tag{9}$$

So, we can define $\mathcal{T}_{\pm}(x,y,k)$ as:

$$\begin{aligned} T_{\pm}(x,y,k) &= \mathcal{K}_k \star \sum_{\boldsymbol{e}_i}(\delta_{\pm}(x_i,y_i)) \\ &= \sum_{\boldsymbol{e}_i} \mathcal{K}_k(x-x_i, y-y_i) \end{aligned} \tag{10}$$

Batch Normalisation and Relu functions are applied next to keep a normalisation process and we obtain time-surfaces of events.

3.1.3 Classification

We use a ResNet18 network (He et al., 2016) pre-trained on ImageNet (Russakovsky et al., 2015). We replace the first layer of ResNet18 by a layer adapted to the size of the input data, as needed. The last layer of ResNet18 is also deleted and replaced by a classification layer.

3.2 Evaluation

We evaluate our method using two datasets: a small real world dataset, DVS128 Gesture and a larger dataset, N-Caltech101.

DVS128 Gesture. is a real world dataset composed of 11 hand and arm gestures performed by 29 subjects in 3 different illumination conditions, resulting in a total of 1342 sequences (Amir et al., 2017). This dataset was created in laboratory with a fixed background and controlled illumination (natural, fluorescent and LED lights are used). The scenes are aquired with a DVS128, so the event-streams cover a range of 128×128 pixels.

We respect the original split between training and testing: 23 subjects are used for training and 6 subjects in the test set. To perform a 10-Fold cross validation we successively extract 4 subjects from the training set and use them for the validation. This extraction is performed using a sliding window of step 2 subjects. Finally, the training set of DVS128 Gesture represents about 65% of the data, the validation set is about 14% and the testing set represents 21%. We avoid subject bias by keeping all sequences of a given subject grouped in the same set (training, testing or validation).

N-Caltech101. is obtained by recording static images of the Caltech101 dataset on a computer display (Orchard et al., 2015) with a moving ATIS event-based camera mimicking sacades (Posch et al., 2011). The dataset contains 100 objects classes and one background class. Each category contains between 45 and 400 sequences, resulting in a total of 8709 sequences. Since the recordings are performed with an ATIS sensor, the resulting event-streams cover a range of 240×304 pixels, even though the input Caltech images are of varying sizes.

20 % of each class is reserved for testing. A 10-Fold cross validation is performed by splitting the remaining data in 10 parts per class, each part is successively used as a validation set while the remaining 9 form the training set. Each set thus respects the unbalanced statistics of the global database. The training set represents 72% of the data, the validation set 8% and testing set represents 20%.

Evaluation Pipeline. We compare VK-SITS to three other competitive event representation methods: SITS (Manderscheid et al., 2019), TORE (Baldwin et al., 2022) and VoxelGrid (Zhu et al., 2019). The evaluation pipeline, described in Figure 2, consists in successively training a ResNet18 network on both datasets, using the Adam and SGD optimizer for all four representation methods. The input data undergoes the same pre-processing. This way, the influence of the event representation on the recognition task can be isolated.

Global parameters are fine-tuned empirically. For VK-SITS the radius of the time-surface kernel is set to $r = 4$, the dilatation is set to $d = 1$ and we use 4 filters. Common parameters between SITS and VK-SITS are set to the same value. We use a memory of 100 for TORE and 100 bins for VoxelGrid. We use a learning rate of 10^{-4} for the Adam optimizer and 10^{-2} for SGD. In all experiments we train during 150 epochs. We save the model with the highest average top-1 accuracy and we save the number of epochs that were needed to train the best model, and we use it with the testing set.

4 RESULTS AND DISCUSSION

During training, we calculate the average time per step for training models, the average number of epochs needed to train the models and the average of classification top-1 accuracy from the 10-Fold cross-validation. Average, we calculate during testing the mean of classification top-1 accuracy, standard deviation, minimum and maximum top-1 accuracy obtained.

Recognition performance on the DVS128 Gesture dataset are given in Table 1. Our work achieves the lowest average learning time by step for both SGD (1.75s) and Adam (0.60s) as well as the highest training mean top-1 accuracy with the two optimizers too (89.29 % with SGD and 89.30 % with Adam). Slightly better results are obtained using Adam optimizer. The lowest average number of epochs needed to train models is obtained by SITS using SGD (90 epochs) and TORE using Adam (89 epochs).

In the 10-Fold testing, TORE obtains the best results during testing, in terms of average, minimum and maximum top-1 accuracy and standard deviation with both optimizers. For this dataset, our work does not obtain the best results but is in the same range as the other methods. Figure 3 shows that the differences

Table 1: Comparison between our method and SITS (Manderscheid et al., 2019), TORE (Baldwin et al., 2022) and VoxelGrid (Zhu et al., 2019) methods on DVS128 Gesture and N-Caltech101 according to mean of classification top-1 accuracy (%), time performance by step (in s) and number of epochs needed to learn models. Variance, minimum and maximum of accuracy are reported for testing phase using a 10-Fold cross-validation. Best values per column and optimizer highlighted in in bold.

		Training				Testing			
						Top-1 accuracy			
Dataset	Method	Optim.	Avg. time per step (s)	Avg. # of epochs	Avg. top-1 acc.	Avg.	σ	Min	Max
DVS128 Gesture	**VK-SITS**	SGD	**1.75**	137	**89.29**	87.43	2.22	84.03	89.93
	SITS	SGD	3.59	**90**	88.81	87.67	2.44	83.33	89.93
	TORE	SGD	2.93	102	88.64	**88.16**	**1.59**	**85.76**	**91.32**
	VoxelGrid	SGD	2.09	126	81.56	80.38	4.39	71.88	85.42
	VK-SITS	Adam	**0.60**	104	**89.30**	87.92	1.40	85.76	89.58
	SITS	Adam	2.57	116	89.15	88.30	1.59	85.42	90.63
	TORE	Adam	3.03	**89**	88.81	**88.37**	**1.31**	**86.81**	**91.32**
	VoxelGrid	Adam	1.10	128	84.63	83.16	2.37	78.47	86.46
N-Caltech101	**VK-SITS**	SGD	**0.31**	75	**73.75**	**73.52**	**0.52**	**72.82**	**74.21**
	SITS	SGD	2.31	83	73.12	72.66	0.98	70.92	73.96
	TORE	SGD	2.08	**69**	73.21	72.27	0.86	70.80	73.33
	VoxelGrid	SGD	0.57	100	72.74	72.29	1.09	70.20	73.58
	VK-SITS	Adam	**0.25**	80	71.50	71.08	**0.73**	69.86	71.88
	SITS	Adam	1.79	49	71.61	71.33	0.94	70.17	72.60
	TORE	Adam	1.93	58	72.16	72.21	0.90	**70.89**	73.73
	VoxelGrid	Adam	3.11	**39**	**72.84**	**72.69**	1.14	70.52	**73.98**

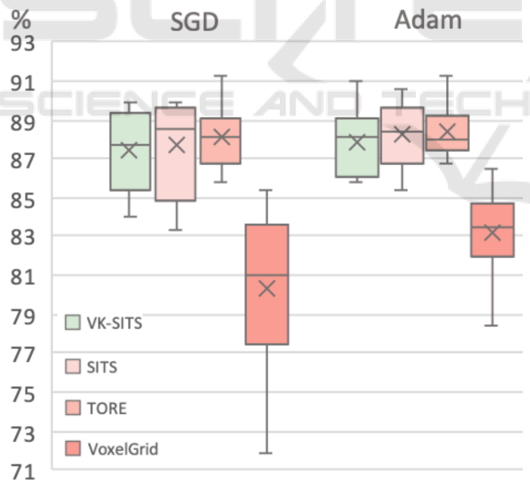

Figure 3: 10-Fold classification accuracy for DVS128 Gesture database per event representation: VK-SITS (this work), SITS (Manderscheid et al., 2019), TORE (Baldwin et al., 2022) and VoxelGrid (Zhu et al., 2019).

in accuracy between VK-SITS, SITS and TORE are not significant. Only the VoxelGrid event representation method provides meaningfully lower recognition results.

Table 1 shows the recognition performance on the N-Caltech101 dataset. VK-SITS reaches higher results than other methods for the average time per step with SGD (0.31 s) and Adam (0.25 s). The smallest numbers of epochs needed to train the models is achieved with TORE using SGD (69 epochs) and VoxelGrid using Adam (39 epochs). We report a fast overfitting when we use Adam optimizer for all methods tested. This can be explained by the heterogeneity of the dataset. Best overall top-1 accuracy is reached with VK-SITS using SGD (highest training and testing average top-1 accuracy, highest min and max testing accuracy, best standard deviation).

The VoxelGrid representation with Adam optimizer also performs well (highest training and testing average top-1 accuracy and highest maximum accuracy), though less consistently over all 10 folds. This is evident in the high standard deviation of the accuracy of 1.14, compared to 0.73 for VK-SITS. Figure 4 visually confirms that VK-SITS with the SGD optimizer achieves the best classification results for the N-Caltech 101 database. VK-SITS is more stable and repeatable than other methods tested.

Results are mixed for DVS-Gesture, a simple dataset on which most methods obtain good results with no significantly better method. However, the more challenging N-Caltech101 dataset shows that VK-SITS permits the best recognition compared to SITS, TORE and VoxelGrid. Using several learnable kernels creates a representation which better fits the data and can focus on different features. We retain the speed-invariant advantage of SITS. VK-SITS also consistently permits a faster learning.

Finally, DVS128 Gesture sequences are in a range from 35 267 to 1 594 557 events and N-Caltech101 sequences are in a range from 6718 to 399 321 events. Considering these large ranges, preprocessing phase make difference regarding others methods, with less events than others methods, we obtain results close to other papers.

In the original two datasets, only the split between training and testing set is done. Most other works use the same set for validation and testing. The results are thus biased and higher than what can be achieved on unseen data. In this configuration, TORE completed by a GoogLeNet network obtains 96,2% on DVS128 Gesture and 79.8% on N-Caltech101 (Baldwin et al., 2022). Similarly, VoxelGrid reaches 75.4% on N-Caltech101 (Gehrig et al., 2019). The proper training, validation and testing split performed in this work allows for the first time to realistically and fairly compare four event representation methods for a deep-learning recognition task.

SITS was developed for corner detection, this work is the first evaluation of SITS for a deep-learning recognition task. The original SITS implementation uses a random forest for corner detection whereas we use a ResNet18 for classification. Finally, SITS is a fully asynchronous event-per-event method which can process up to 1.6 Mev/s on a single CPU, to provide the event representation. Our method processes sequences of 40 000 events downsampled to 4000 events. Though the recognition is quite quick (150 µs to compute the representation and perform the classification) there is an inherent latency given the time to accumulate the necessary events. This latency is of the order of 50 ms for DVS128 Gesture and 10 ms for N-Caltech101.

5 CONCLUSION

The paper introduces VK-SITS, an event-based representation based on the SITS time-surface and on the Event Spike Tensor. VK-SITS is speed-invariant, translation invariant, and characterized by four end-to-end trainable kernels. This allows VK-SITS to learn faster than the other representations evaluated. VK-SITS is compared to three state-of-the-art event representations using a unified preprocessing on a recognition task using ResNet18 on two commonly used event-based datasets, DVS128 Gesture and N-Caltech101. The comparison is performed with a 10-Fold cross-validation with a proper training, valida-

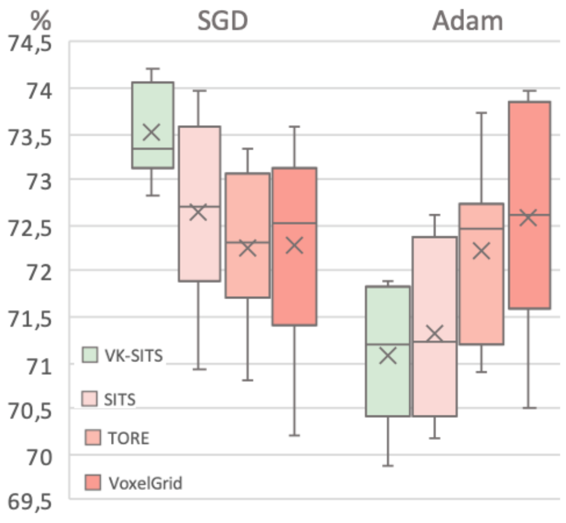

Figure 4: 10-Fold classification accuracy for N-Caltech101 database per event representation: VK-SITS (this work), SITS (Manderscheid et al., 2019), TORE (Baldwin et al., 2022) and VoxelGrid (Zhu et al., 2019).

tion and testing split. When trained with the SGD optimizer, VK-SITS provides more robust learning results.

This paper provides a methodological contribution by proposing a pipeline to compare event representation methods independently from the deep-learning networks used. We also implement and evaluate for the first time SITS for a deep-learning based recognition task.

To further evaluate the potential of VK-SITS as a generic event representation, it should be tested on other tasks and databases. SL-Animals for example, represents 19 different animals in American sign language and would provide a new application (Vasudevan et al., 2021). The evaluation methodology presented in this work is a rigorous comparison of an off-line situation. One of the strengths of event-based cameras is their ability to quickly record high speed movements. As such, it is important to also evaluate these recognition tasks in an on-line setting. The implementation of real-time event-based algorithms is non trivial, especially with the advent of increasingly large sensors. Future work should concentrate on comparing the on-line performances of event-based representations and recognition algorithms.

REFERENCES

Amir, A., Taba, B., Berg, D., Melano, T., Mckinstry, J., Nolfo, C. D., Nayak, T., Andreopoulos, A., Garreau, G., Mendoza, M., Kusnitz, J., Debole, M., Esser, S., Delbruck, T., Flickner, M., and Modha, D. (2017). A

Low Power, Fully Event-Based Gesture Recognition System. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 7388–7397.

Baldwin, R. W., Liu, R., Almatrafi, M., Asari, V., and Hirakawa, K. (2022). Time-Ordered Recent Event (TORE) Volumes for Event Cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1.

Cannici, M., Ciccone, M., Romanoni, A., and Matteucci, M. (2019). Asynchronous Convolutional Networks for Object Detection in Neuromorphic Cameras. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 1656–1665.

Delbrück, T., Linares-Barranco, B., Culurciello, E., and Posch, C. (2010). Activity-driven, event-based vision sensors. In *IEEE International Symposium on Circuits and Systems*, pages 2426–2429.

Gallego, G., Delbruck, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A., Conradt, J., Daniilidis, K., and Scaramuzza, D. (2020). Event-based Vision: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1):154–180.

Gehrig, D., Loquercio, A., Derpanis, K. G., and Scaramuzza, D. (2019). End-to-End Learning of Representations for Asynchronous Event-Based Data. In *IEEE International Conference on Computer Vision*, pages 5632–5642.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

Lagorce, X., Orchard, G., Galluppi, F., Shi, B. E., and Benosman, R. B. (2017). HOTS: A Hierarchy of Event-Based Time-Surfaces for Pattern Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(7):1346–1359.

Lichtsteiner, P., Posch, C., and Delbruck, T. (2008). A 128$\times$ 128 120 dB 15 μs Latency Asynchronous Temporal Contrast Vision Sensor. *IEEE Journal of Solid-State Circuits*, 43(2):566–576.

Manderscheid, J., Sironi, A., Bourdis, N., Migliore, D., and Lepetit, V. (2019). Speed Invariant Time Surface for Learning to Detect Corner Points with Event-Based Cameras. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10237–10246.

Maqueda, A. I., Loquercio, A., Gallego, G., Garcia, N., and Scaramuzza, D. (2018). Event-based Vision meets Deep Learning on Steering Prediction for Self-driving Cars. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5419–5427.

Mueggler, E., Gallego, G., Rebecq, H., and Scaramuzza, D. (2018). Continuous-Time Visual-Inertial Odometry for Event Cameras. *IEEE Transactions on Robotics*, 34.

Orchard, G., Jayawant, A., Cohen, G. K., and Thakor, N. (2015). Converting Static Image Datasets to Spiking Neuromorphic Datasets Using Saccades. *Frontiers in Neuroscience*, 9.

Posch, C., Matolin, D., and Wohlgenannt, R. (2011). A QVGA 143 dB Dynamic Range Frame-Free PWM Image Sensor with Lossless Pixel-Level Video Compression and Time-Domain CDS. *IEEE Journal of Solid-State Circuits*, 46.

Ramesh, B., Yang, H., Orchard, G., Thi, N. A. L., Zhang, S., and Xiang, C. (2020). DART: Distribution Aware Retinal Transform for Event-based Cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(11):2767–2780.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115.

Sironi, A., Brambilla, M., Bourdis, N., Lagorce, X., and Benosman, R. (2018). HATS: Histograms of Averaged Time Surfaces for Robust Event-based Object Classification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1731–1740.

Vasudevan, A., Negri, P., Ielsi, C. D., Linares-Barranco, B., and Serrano-Gotarredona, T. (2021). SL-Animals-DVS: Event-Driven Sign Language Animals Dataset. *Pattern Analysis and Applications*, 25(3):505–520.

Zhu, A. Z., Yuan, L., Chaney, K., and Daniilidis, K. (2019). Unsupervised Event-based Learning of Optical Flow, Depth, and Egomotion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 989–997.