POLITECNICO DI TORINO
Repository ISTITUZIONALE

MESA: A Dynamical Attention-based Pre-processing Pipeline for High-throughput Event-based Computer Vision Tasks

Original
MESA: A Dynamical Attention-based Pre-processing Pipeline for High-throughput Event-based Computer Vision Tasks / Bich, P., Prono, L., Boretti, C., Pareschi, F., Rovatti, R., Setti, G.. - In: IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS. II, EXPRESS BRIEFS. - ISSN 1549-7747. - STAMPA. - 72:12(2025), pp. 2057-2061. [10.1109/TCSII.2025.3621345]

Availability:
This version is available at: 11583/3003607 since: 2025-12-02T21:36:47Z

Publisher:
IEEE

Published
DOI:10.1109/TCSII.2025.3621345

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

18 November 2026

This article has been accepted for publication in IEEE Transactions on Circuits and Systems II: Express Briefs. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCSII.2025.3621345

MESA: A Dynamical Attention-based Pre-processing Pipeline for High-throughput Event-based Computer Vision Tasks

Philippe Bich, *Student Member, IEEE,* Luciano Prono, *Member, IEEE,* Chiara Boretti, *Student Member, IEEE,* Fabio Pareschi, *Senior Member, IEEE,* Riccardo Rovatti, *Fellow, IEEE,* Gianluca Setti, *Fellow, IEEE*

***Abstract*—Dynamic Vision Sensors (DVS) offer a unique advantage in capturing changes in luminance asynchronously, providing high temporal resolution and efficiency, making them particularly suitable for applications like egocentric vision and autonomous driving. However, adapting the sparse and asynchronous nature of DVS data for traditional non-recurrent deep learning models, such as convolutional neural networks (CNNs) and transformer-based architectures, poses challenges. In fact, classical methods, such as time surfaces and voxel grids, convert event-based data into a form suitable for frame-based Deep Neural Networks (DNNs). While effective, these methods often sacrifice the fine-grained temporal details intrinsic to DVS data, especially when requiring high throughput predictions. This can diminish the advantages of DVS in capturing fast-moving or transient phenomena. We aim to contribute addressing this issue and propose a dynamic pre-processing pipeline called *Memory of Events through Spatial Attention* (MESA), that enhances the currently used event-based data representations. This is obtained by storing events in a memory tensor with pixel-wise adaptive forgetting factors generated in real time through a spatial-attention module. Tested on multiple computer vision tasks, this method enhances the performance of state-of-the-art non-recurrent DNNs with minimal computational cost. In particular, by using MESA, the accuracy on CIFAR10-DVS with MobileViT-v2s improves by more than 15% and with DETR-ResNet50, the mAP on the PEDRo object detection dataset is three times higher than the baseline achieved with time surfaces alone. Furthermore, when estimating pupil position on the 3ET+ dataset using MobileNet-v3s, MESA reduces the Euclidean distance error by 36% compared to using time surfaces alone.**

I. Introduction

In recent years, Deep Neural Networks (DNNs) have emerged as the de facto standard for tackling an extensive set of computer vision tasks. In particular, the literature enumerates a wide variety of non-recurrent DNN models, ranging from the broadly acclaimed convolutional models [1], [2] to the most recent and highly scalable transformer-based models [3], [4]. These models are employed for tasks from image classification to object detection, often in dynamic, quickly changing scenarios that require a high temporal resolution analysis of data in video streams.

In this area, event-based data encoding is a promising approach for speed-critical computer vision tasks, using information captured by Dynamic Vision Sensors (DVS) [5], also known as event-based cameras. These asynchronous devices respond only to the luminescence changes in the visual scene, generating a stream of temporally and spatially sparse events with a very high temporal resolution, in the order of 1 µs. This kind of representation adapts well to time-varying models such as the recurrent neural networks [6]. However, these architectures require a computationally intensive training process [7], which has led the current best-performing models to favor non-recurrent structures.

Because of this, a well-known approach is to use meaningful representations from streams of events that are suitable to standard non-recurrent computer vision models. Popular strategies involve time surfaces [8] and voxel grids [9]. While time surfaces flatten the events within a certain time range to a single image-like matrix, voxel grids are 3-dimensional structures where events are aggregated into small spatial-temporal bins; both approaches capture changes over time and space in a dense format suitable for processing by standard computer vision models. Note that, for each instance, these representations group all the events over a temporal window of predefined size.

Many recent works aim to improve data representation to achieve performance advantages. For instance, [10], [11] select only sparse spatial portions of the input tensor – referred to as "activated patches" – to reduce the number of embeddings processed by a Vision Transformer (ViT), thereby effectively reducing the latency and power consumption. However, all the aforementioned representations rely on selecting a sufficiently long time window or waiting until enough information is available before processing, as each input segment alone may not contain adequate data for the DNN models to produce accurate outputs. This limitation becomes critical in tasks requiring high-throughput inference, where the use of very short time windows is mandatory. As an example, the use

P. Bich, L. Prono, C. Boretti and F. Pareschi are with the Department of Electronic and Telecommunication, Politecnico di Torino, 10129 Torino, Italy (e-mail: {philippe.bich, chiara.boretti, luciano.prono, fabio.pareschi}@polito.it.

R. Rovatti is with the Department of Electrical, Electronic, and Information Engineering, University of Bologna, 40136 Bologna, Italy (e-mail: riccardo.rovatti@unibo.it).

G. Setti is with King Abdullah University of Science and Technology (KAUST), Saudi Arabia (e-mail: gianluca.setti@kaust.edu.sa).

F. Pareschi and R. Rovatti are also with the Advanced Research Center on Electronic Systems (ARCES), University of Bologna, 40125 Bologna, Italy.

This study was carried out within the FAIR - Future Artificial Intelligence Research and received funding from the European Union Next-Generation EU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.3 – D.D. 1555 11/10/2022, PE00000013). This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Circuits and Systems II: Express Briefs. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCSII.2025.3621345

of activated patches as in [10], [11] would be particularly challenging with short temporal windows since all the patches would contain extremely sparse information.

To address this issue, a recent work [12] proposed a middle-ground solution, applying a preprocessing pipeline to an event-based eye-tracking task with stringent high-throughput requirements. With this approach, we accumulate time surfaces over time and store them in *memory channels*. At each time step, the memory channels are scaled by a constant forgetting factor $k \in (0, 1)$, which helps to discard outdated, no-longer-relevant information. While the use of memory and in general past information in neural estimators is not new [10], employing a lightweight preprocessing pipeline offers greater flexibility, enabling applicability to a wide range of estimators and ensuring robustness to their updates.

Although this solution proved to be effective, the use of a single, constant and manually tuned value of k is clearly sub-optimal. In this work, we improve the preprocessing pipeline and explore the possibility of extending the *memory of events* benefits beyond eye-tracking. We propose a flexible, dynamic pre-processing pipeline, called Memory of Events through Spatial Attention (MESA), to generate information-rich inputs from event-based data. Specifically: 1) we accumulate the event representation in a *memory tensor* and we apply a dynamic, pixel-by-pixel forgetting factor matrix $\boldsymbol{K}$, generated at each time-step with a spatial-attention module [13]; and 2) we test both time surface and voxel grid representations on multiple tasks which exploit state-of-the-art DNN models, ranging from regression in eye-tracking applications to classification and object detection.

When working with very short temporal windows, MESA greatly enhances the performance of state-of-the-art estimators in regression, classification and object detection tasks with minimal computational overhead. The rest of this paper is structured as follows. In Section II we present our methodology, in Section III we list the datasets and models used to test the proposed pipeline, and we provide information on the training process. In Section IV we present the numerical results of our experiments in terms of accuracy and computational overhead. Finally, the conclusion is drawn.

II. MESA METHODOLOGY

DVSs generate sparse and asynchronous streams of events that encode the pixel-local changes in the brightness of the scene. Each event is in the form $\boldsymbol{e} = (x, y, t, p)$, where x, y and t are the pixel coordinates and detection time of the luminescence alteration, and $p \in \{+1, -1\}$ is the polarity of the event (indicating whether brightness is increasing or decreasing). In our input pipeline, we employ two event representations, either time surfaces [8], or voxel grids [9], but the same pipeline can be used with many others available in the literature, such as TORE volumes [14]. In this work, the employed representations are generated by means of the *Tonic* framework [15].

These event structures are sampled within time intervals of size Δ_t over the entire sensor area. A new representation is generated each discrete time step $n = 1, 2, 3, \ldots$

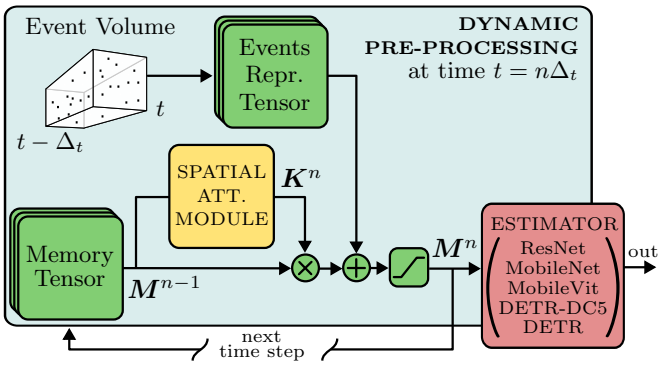


Fig. 1. Pre-processing pipeline. A tensor-like representation of an event stream – in the time range $(t, t + \Delta_t)$ – is accumulated (with 0 to 1 clipping) over time and stored in a memory tensor $\boldsymbol{M}^n$. At each time step n, the values corresponding to each pixel of the memory tensor $\boldsymbol{M}^{n-1}$ are dampened with varying strength by a forgetting matrix $\boldsymbol{K}^n$, dynamically generated by the spatial attention module.

corresponding to times $t = n\Delta_t$ and encoded in a tensor $\boldsymbol{R}^n \in \mathbb{R}^{H \times W \times C}$, where H and W are the number of pixels in the DVS sensor along height and width, respectively, while C is the number of channels, whose definition varies depending on the representation. We define as $\mathcal{E}_p^n(x, y)$ the set of the events with polarity p located at pixel (x, y) in the time range $t \in ((n-1)\Delta_t, n\Delta_t]$. With $\mathcal{E}^n(x, y)$, we consider both positive and negative events in the corresponding set.

Time surfaces (ts) are image-like matrices where each pixel is defined as

$$\boldsymbol{R}^n_{\text{ts},p}(x, y) = \exp\left[-\frac{1}{\tau}\left(n\Delta_t - \max_{\boldsymbol{e} \in \mathcal{E}_p^n(x,y)} t_{\boldsymbol{e}}\right)\right] \tag{1}$$

here $t_{\boldsymbol{e}}$ is the time of event $\boldsymbol{e}$ and τ is a time constant. Using time surfaces, the resulting event representation $\boldsymbol{R}^n_{\text{ts}}$ is a tensor with $C = 2$ channels, namely $\boldsymbol{R}^n_{\text{ts},+1}$ encoding positive events and $\boldsymbol{R}^n_{\text{ts},-1}$ encoding negative events.

Conversely, voxel grids (vg) are defined as 3-dimensional structures. The time dimension is discretized in B bins of size $\delta_t = \Delta_t/(B-1)$, so the voxel grid is a tensor with $C = B$ channels. Each pixel at position (x, y) of channel z is then defined as

$$\boldsymbol{R}^n_{\text{vg}}(x, y, z) = \sum_{\boldsymbol{e} \in \mathcal{E}^n(x,y)} p_{\boldsymbol{e}} \max\left(0, 1 - |z\delta_t - t^*_{\boldsymbol{e}}|\right) \tag{2}$$

where $t^*_{\boldsymbol{e}} = \left[t_{\boldsymbol{e}} - (n-1)\Delta_t\right]/\delta_t$ is the time of event $\boldsymbol{e}$ shifted to the event representation range and normalized to the bin size δ_t and $p_{\boldsymbol{e}}$ is its polarity. The max operator in (2) is equivalent to the bilinear sampling kernel defined in [16], which means that the closer an event is to the center of a temporal bin, the more it influences the corresponding value, and a single event can influence two adjacent bins at the same time.

For each time step n, either the time surface or the voxel grid representations are accumulated over time into a *memory tensor* $\boldsymbol{M}^n \in \mathbb{R}^{H \times W \times C}$ as

$$\boldsymbol{M}^n = \left[\boldsymbol{K}^n \odot \boldsymbol{M}^{n-1} + \boldsymbol{R}^n\right]_0^1 \tag{3}$$

where $\boldsymbol{K}^n \in (0, 1)^{H \times W}$ is the *forgetting matrix* with the same spatial height and width of $\boldsymbol{M}^n$, $\odot$ is an element-wise

© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Circuits and Systems II: Express Briefs. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCSII.2025.3621345

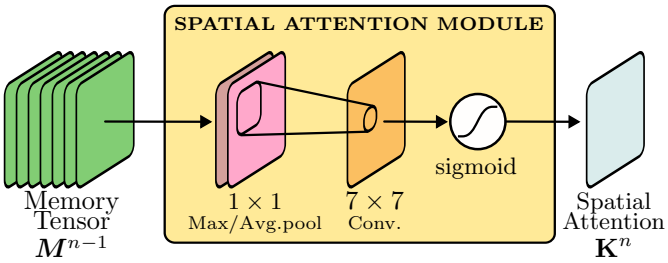


Fig. 2. Spatial attention module structure: a 2-channel tensor is generated by taking the average and the maximum of the memory tensor channels; then a 7×7 convolution filter is applied and a sigmoid function is applied to get the forgetting matrix $\boldsymbol{K}^n$ with values between 0 and 1.

multiplication operator that scales all the values corresponding to a spatial position in $\boldsymbol{M}^{n-1}$ by the corresponding value in matrix $\boldsymbol{K}^n$ and operator $[\cdot]_0^1$ saturates the argument between 0 and 1. Processing starts at time step $n = 1$, with $\boldsymbol{M}^0$ defined as an all-zero tensor. Fig. 1 shows in detail the proposed pre-processing pipeline.

The forgetting matrix $\boldsymbol{K}^n$ is composed of values in the range $(0, 1)$ and it is generated dynamically each time step by a *spatial attention* module [13]. The module selects what parts of the memory tensor are to be "forgotten" quickly (scaling values close to 0) and what parts are to be "remembered" (scaling values close to 1). This is done by condensing all the channels of the memory tensor into 2 channels (for each pixel, we encode the channel-wise maximum and average value) and by applying a 7×7 convolutional filter followed by a sigmoid activation function (we apply zero-padding to keep the spatial dimensions unchanged). This kind of approach allow us to dynamically keep only the important pixels stored in the memory tensor to maximize the performance on the task. Fig. 2 shows the spatial attention module structure.

This input pipeline is compatible with any non-recurrent computer vision model, allowing image-driven estimators to be employed off-the-shelf with minimal modifications and some fine-tuning. Compared to recurrent DNNs approaches, this moves the time dependency of the model entirely to the input pipeline, greatly simplifying the optimization process, i.e., it is not necessary to unroll the model during training with a backpropagation-through-time approach. Furthermore, this makes the strategy robust to changes or updates in the backbone estimator, always allowing easy integration of the latest state-of-the-art models. Finally, we can select Δ_t as small as possible, increasing the throughput of the algorithm without sacrificing the performance. In contrast, simply using time surfaces or voxel grids would be unsuitable for high-throughput applications, since a small Δ_t would lead to inputs with limited information content.

III. Case Studies and Training

A. Datasets

To validate the proposed pre-processing pipeline, we train and test it on multiple datasets related to different computer vision tasks:

- **3ET+** [17] (Regression): an event-based eye-tracking dataset. It comprises 52 videos entirely captured by means of an event-based camera from 13 different subjects performing different eyes activities.
- **N-MNIST** [18] (Classification): an event-based version of the MNIST dataset, consisting of event-based data representing hand-written digits. It contains 70 000 recordings with each image captured as an event stream.
- **CIFAR-10 DVS** [19] (Classification): a neuromorphic adaptation of the CIFAR-10 dataset, containing 10 000 event streams generated from static images across 10 object classes using a Dynamic Vision Sensor.
- **N-CALTECH101** [18] (Classification): the event-based counterpart of the original CALTECH101 dataset [20]. It retains its structure with 101 object categories and provides, on average, 50 event-based recordings per class.
- **PEDRo** [21] (Object Detection): a pedestrian detection and tracking dataset recorded with an event-based camera. It includes 43 259 bounding boxes included in 119 recordings, with scenarios involving pedestrians in dynamic environments.

B. Employed architectures

To solve the aforementioned tasks, we fine-tune and employ multiple state-of-the-art estimators in conjunction with the proposed pre-processing pipeline:

- **ResNet-18** [1] (Regression, Classification): a classical yet effective convolutional neural network belonging to the Residual Network (ResNet) family, which introduces identity-based skip connections to enable the training of deeper architectures.
- **MobileNet-v3s** [2] (Regression, Classification): a lightweight convolutional neural network architecture specifically optimized for efficiency and performance on mobile and embedded devices.
- **MobileViT-v2s** [4] (Regression, Classification): is the transformer-based version of MobileNet, designed for efficient inference on mobile devices by combining convolutional layers with lightweight transformer blocks for improved performance.
- **DETR** [3] (Object Detection): the state-of-the-art architecture for object detection that redefines the traditional pipeline by eliminating the need for region proposals, anchor boxes, and non-maximum suppression. DETR formulates object detection as a direct set prediction problem using a transformer-based encoder-decoder architecture[1].
- **DETR-DC5** [3] (Object Detection): a variant of the original DETR architecture that enhances detection performance. This improvement is achieved by incorporating dilated convolutions into the backbone network, which effectively increases the receptive field without reducing spatial resolution[2].

[1]We fine-tuned the pre-trained models `facebook/detr-resnet-50` and `facebook/detr-resnet-101` available on HuggingFace.

[2]We fine-tuned from the pre-trained model available on HuggingFace `facebook/detr-resnet-50-dc5`.

© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Circuits and Systems II: Express Briefs. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCSII.2025.3621345

TABLE I
COMBINED RESULTS FOR DIFFERENT MODELS ON DIFFERENT TASKS. THE BEST MODEL FOR EACH CONFIGURATION IS HIGHLIGHTED IN **BOLD**. WE REPORT IN LIGHT GRAY THE RESULTS OBTAINED WITH THE MEMORY-BASED METHODS WE PROPOSE.

Dataset	DNN	Time surfaces			Voxel grids		
		without memory	*static memory*	*with MESA*	*without memory*	*static memory*	*with MESA*
Task: Regression		**Metric: Euclidean distance** (lower is better)					
3ET	ResNet-18	10.30 ± 0.85	**9.01 ± 0.14**	**8.90 ± 0.40**	10.02 ± 0.47	9.29 ± 0.22	**8.80 ± 0.19**
	MobileNet-v3s	11.26 ± 0.41	10.27 ± 0.49	**7.23 ± 0.17**	11.07 ± 0.52	9.06 ± 0.15	**7.15 ± 0.37**
	MobileViT-v2s	10.01 ± 0.42	9.06 ± 0.51	**7.08 ± 0.29**	9.57 ± 0.23	8.80 ± 0.25	**8.23 ± 0.12**
Task: Classification		**Metric: Accuracy (%)** (higher is better)					
N-MNIST	ResNet-18	82.45 ± 0.95	87.16 ± 0.93	**90.30 ± 0.93**	80.76 ± 0.17	89.60 ± 0.65	**92.97 ± 0.42**
	MobileNet-v3s	73.20 ± 2.01	76.63 ± 2.06	**79.16 ± 0.77**	72.96 ± 0.91	81.38 ± 1.17	**84.52 ± 0.25**
	MobileViT-v2s	82.66 ± 1.04	88.15 ± 0.46	**91.82 ± 0.48**	82.18 ± 0.12	88.93 ± 0.84	**92.67 ± 0.24**
CIFAR-10 DVS	ResNet-18	49.71 ± 0.25	52.70 ± 1.23	**58.91 ± 0.63**	44.90 ± 0.48	49.63 ± 0.81	**53.19 ± 0.08**
	MobileNet-v3s	47.15 ± 0.11	**52.61 ± 0.90**	**53.82 ± 0.48**	42.73 ± 0.18	44.85 ± 0.27	**50.02 ± 0.18**
	MobileViT-v2s	52.86 ± 0.31	58.16 ± 1.28	**65.24 ± 0.80**	47.72 ± 0.83	52.51 ± 0.63	**58.85 ± 0.71**
N-CALTECH101	ResNet-18	64.56 ± 0.52	68.65 ± 0.79	**70.52 ± 0.24**	63.46 ± 0.25	70.26 ± 0.76	**74.36 ± 0.12**
	MobileNet-v3s	58.62 ± 0.19	65.84 ± 0.91	**68.02 ± 0.97**	59.52 ± 0.27	**70.38 ± 1.84**	**72.38 ± 0.23**
	MobileViT-v2s	65.96 ± 0.20	72.93 ± 0.96	**74.83 ± 0.43**	65.39 ± 0.25	75.05 ± 0.66	**79.05 ± 0.23**
Task: Object detection		**Metric: mAP**$_{.5:.95}$ (higher is better)					
PEDRO	DETR-Resnet50	0.1195 ± 0.0008	0.3285 ± 0.0055	**0.3818 ± 0.0072**	0.2533 ± 0.0125	0.3923 ± 0.0183	**0.4421 ± 0.0243**
	DETR-Resnet101	0.2475 ± 0.0121	0.3965 ± 0.0234	**0.5557 ± 0.0113**	0.3843 ± 0.0211	0.4983 ± 0.0197	**0.5610 ± 0.0098**
	DETR-DC5-Resnet50	0.3423 ± 0.0305	0.5211 ± 0.0148	**0.5824 ± 0.0147**	0.4326 ± 0.0347	0.5070 ± 0.0301	**0.5833 ± 0.0102**

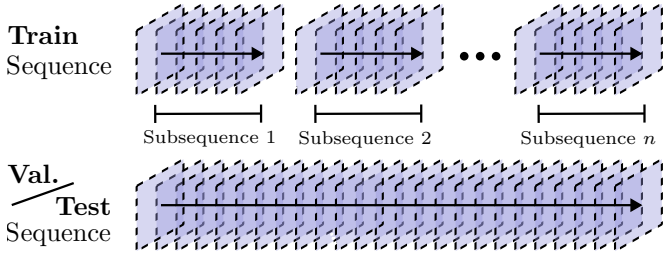


Fig. 3. Splitting of the sequence of tensors $\boldsymbol{R}^n$ for training and validation/test. The training sub-sequences are fed randomly to the algorithm during the training phase.

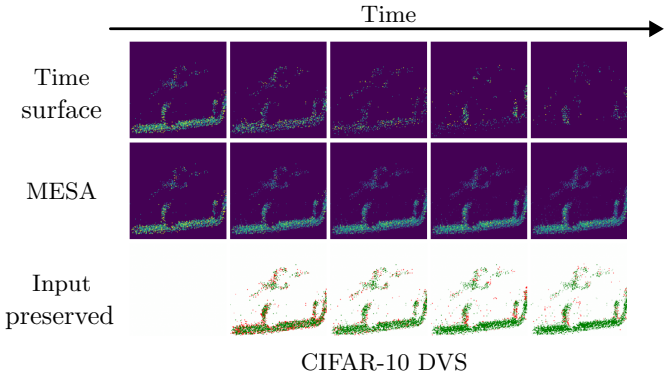


Fig. 4. Sequence of time surfaces from the CIFAR-10 DVS dataset (top), the corresponding MESA memory states (center), and the retained information using MESA (bottom; green=preserved, red=forgotten).

C. Training

The training sets of the selected datasets are composed of multiple event stream recordings. We partition each event stream into small chunks of duration Δ_t and generate a representation tensor $\boldsymbol{R}^n$ for each chunk, resulting in a sequence of tensors. The sequence from the training set is then split into multiple sub-sequences, which are batched and randomly fed to the pre-processing/estimator pair for optimization. On the other hand, validation and test sets are not split into sub-sequences, each entire sequence is inferred to the algorithm in a chronologically ordered way to emulate the behavior of the system. Fig. 3 shows the training vs val/test sequence splitting.

In our experiments, all models were trained for 20 epochs. The training sub-sequences are composed of 20 samples, their overlap is set to 3 samples, $\Delta_t = 5\,\text{ms}$, $\tau = 3\,\text{ms}$ for time surfaces, and $B = 3$ for voxel grid representations. All experiments are conducted with short Δ_t values to simulate high-throughput application scenarios. This choice, consistent across all datasets, naturally leads to lower absolute performance compared to results in works using longer temporal windows, but better reflects the intended deployment conditions. The batch size was set to 32 for regression, 512 for classification and 258 for object detection. Given the large number of sequences resulting from the small Δ_t, data augmentation was not applied. The optimizer used was AdamW, with an initial learning rate and weight decay set to $1\text{e}{-4}$.

IV. RESULTS

A. Accuracy

We assess the effectiveness of the proposed dynamic pre-processing pipeline in comparison to the direct use of the representation tensor $\boldsymbol{R}^n$. We also perform a further comparison with a static pre-processing pipeline (static memory), as in [12], where $\boldsymbol{K}^n$ in (3) is replaced by a fixed value $k = 0.5$. Fig. 4 shows a sequence of time surfaces from the CIFAR-10 DVS dataset, highlighting the ability of MESA to maintain salient features over event representations.

Table I confronts the performance of the different pre-processing strategies for regression (Euclidean distance), classification (accuracy metric) and object detection (mean average

© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Circuits and Systems II: Express Briefs. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCSII.2025.3621345

TABLE II
ABSOLUTE MEMORY (MB) AND COMPUTATIONAL OVERHEAD (GFLOPS) INTRODUCED BY THE MESA MODULE AND THE REQUIREMENTS OF EACH USED BACKBONE ALONE IN TERMS OF BOTH MEMORY AND COMPUTE.

	Module/model	*Memory (MB)*	*GFLOPs*
MESA (preprocessing)		0.55	0.009
backbones	ResNet-18	44.59	1.819
	MobileNet-v3s	9.70	0.060
	MobileViT-v2s	10.93	0.819
	DETR-based	158.32	5.458
	DETR-C5-based	158.32	9.084

precision, $\text{mAP}_{.5:.95}$ metric) tasks, respectively. The use of a dynamic pre-processing pipeline proves to be highly effective, with MESA consistently emerging as the best option. In the regression task, MESA can reduce the Euclidean distance by up to 36% compared to using only simple representation. For classification, it can achieve accuracy gains exceeding 10%, and for object detection, it can improve the $\text{mAP}_{.5:.95}$ by up to three times compared to the other solutions.

B. Computational overhead

From a computational point of view, MESA employs channel reduction (both max and average) followed by a lightweight convolutional layer for the spatial attention module, an element-wise multiplication of $\boldsymbol{K}^n$ with $\boldsymbol{M}^{n-1}$ and an element-wise addition with $\boldsymbol{R}^n$. Under the assumption of a fixed number of channels and convolutional kernel size, all these operations exhibit a complexity of $O(HW)$, making their computational cost almost negligible compared to that of most neural estimator. This observation is validated by Table II, that presents the memory footprint and computational complexity of MESA in comparison to each estimator considered in this work, for an input size of $H = W = 224$. It follows naturally that any overhead introduced by MESA in terms of power consumption, on-chip area, or latency is almost negligible, particularly given that its operations can be readily implemented using standard hardware already required by the estimators. The most notable exception is MobileNet-V3-Small, whose lightweight nature makes the 15% increase in computational complexity introduced by MESA more pronounced. Nevertheless, the associated increase in memory footprint remains negligible, while the resulting accuracy improvement is significant.

V. CONCLUSION

In this work, we propose a dynamic input pre-processing pipeline to enhance state-of-the-art neural models on event-based vision tasks with high throughput requirements. A spatial attention mechanism updates a memory tensor that aggregates successive event representations over time, dynamically retaining the most relevant spatiotemporal information while discarding redundant content. The lightweight design is compatible with diverse neural architectures. Experiments on multiple datasets and estimators show consistent accuracy gains highlighting the benefit of integrating dynamic memory and attention for event-based visual processing.

REFERENCES

[1] K. He *et al.*, "Deep Residual Learning for Image Recognition," in *2016 Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778. doi:10.1109/CVPR.2016.90

[2] A. Howard *et al.*, "Searching for MobileNetV3," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, Oct. 2019, pp. 1314–1324. doi:10.1109/ICCV.2019.00140

[3] N. Carion *et al.*, "End-to-End Object Detection with Transformers," in *16th European Conf. on Comput. Vis. (ECCV 2020)*. Berlin, Heidelberg: Springer-Verlag, Aug. 2020, pp. 213–229. doi:10.1007/978-3-030-58452-8_13

[4] S. Mehta and M. Rastegari, "MobileViT: Light-weight, general-purpose, and mobile-friendly vision transformer," in *Proceedings of 2022 International Conference on Learning Representations (ICLR)*, 2022.

[5] G. Gallego *et al.*, "Event-Based Vision: A Survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 1, pp. 154–180, Jan. 2022. doi:10.1109/TPAMI.2020.3008413

[6] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. doi:10.1162/neco.1997.9.8.1735

[7] G. Bellec *et al.*, "A solution to the learning dilemma for recurrent networks of spiking neurons," *Nature Communications*, vol. 11, no. 1, p. 3625, Jul. 2020. doi:10.1038/s41467-020-17236-y

[8] X. Lagorce *et al.*, "HOTS: A Hierarchy of Event-Based Time-Surfaces for Pattern Recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1346–1359, Jul. 2017. doi:10.1109/TPAMI.2016.2574707

[9] A. Z. Zhu *et al.*, "Unsupervised Event-Based Learning of Optical Flow, Depth, and Egomotion," in *2019 Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 989–997. doi:10.1109/CVPR.2019.00108

[10] A. Sabater, L. Montesano, and A. C. Murillo, "Event Transformer. A sparse-aware solution for efficient event data processing," in *2022 Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2022, pp. 2676–2685. doi:10.1109/CVPRW56347.2022.00301

[11] Z. Wang, Y. Hu, and S.-C. Liu, "Exploiting Spatial Sparsity for Event Cameras with Visual Transformers," in *2022 IEEE International Conference on Image Processing (ICIP)*, Oct. 2022, pp. 411–415. doi:10.1109/ICIP46576.2022.9897432

[12] C. Boretti *et al.*, "Memory in Motion: Exploring Leaky Integration of Time Surfaces for Event-based Eye-tracking," in *2024 IEEE Biomed. Circ. and Syst. Conf. (BioCAS)*, Oct. 2024, pp. 1–5. doi:10.1109/BioCAS61083.2024.10798345

[13] S. Woo *et al.*, "CBAM: Convolutional Block Attention Module," in *15th European Conf. on Comput. Vis. (ECCV 2018)*. Berlin, Heidelberg: Springer-Verlag, Sep. 2018, pp. 3–19. doi:10.1007/978-3-030-01234-2_1

[14] R. W. Baldwin *et al.*, "Time-ordered recent event (TORE) volumes for event cameras," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 2, pp. 2519–2532, 2023. doi:10.1109/TPAMI.2022.3172212

[15] G. Lenz *et al.*, "Tonic: Event-based datasets and transformations." Zenodo, Jul. 2021. doi:10.5281/zenodo.5079802

[16] M. Jaderberg *et al.*, "Spatial Transformer Networks," in *Advances in Neural Information Processing Systems*, vol. 28. Curran Associates, Inc., 2015.

[17] Z. Wang *et al.*, "Event-based eye tracking. ais 2024 challenge survey," in *2024 Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2024, pp. 5810–5825.

[18] G. Orchard *et al.*, "Converting Static Image Datasets to Spiking Neuromorphic Datasets Using Saccades," *Frontiers in Neuroscience*, vol. 9, Nov. 2015. doi:10.3389/fnins.2015.00437

[19] H. Li *et al.*, "CIFAR10-DVS: An Event-Stream Dataset for Object Classification," *Frontiers in Neuroscience*, vol. 11, 2017. doi:10.3389/fnins.2017.00309

[20] L. Fei-Fei, R. Fergus, and P. Perona, "Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories," in *2004 Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW'04)*, ser. CVPRW '04. USA: IEEE Computer Society, Jun. 2004, p. 178.

[21] C. Boretti *et al.*, "PEDRo: An Event-based Dataset for Person Detection in Robotics," in *2023 Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2023, pp. 4065–4070. doi:10.1109/CVPRW59228.2023.00426

© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.