CoL^{2}A: Convolution-free Local Linear Attention for SpatioTemporal Event Processing

Yusuke Sekikawa[†*] Jun Nagata[†] Itsumi Araki Andreu Girbau
DENSO IT Lab., Inc.

Abstract

Linear attention is sparse, recurrent, and GPU-parallel; these are essential features for processing sparse data from event-based cameras. We argue that locality is missing to efficiently model event-to-event relationships for continuous spatiotemporal perception. We propose CoL^{2}A by introducing locality into linear attention without using a computationally demanding convolution operation. The key idea for the convolution-free formulation is restricting the positional embedding local convolutional kernel into the special class which can be decomposed into two global positional embeddings that can be absorbed into query and key; this replaces convolution with a local sum. To the best of our knowledge, CoL^{2}A is the first to equip sparsity, recurrence, GPU parallelism and locality, simultaneously. We demonstrate CoL^{2}A's effectiveness on dense, high-temporal-resolution ($>$ 1000 fps) prediction task from events, demonstrating real-time capability while maintaining competitive results over the conventional method. https://github.com/DensoITLab/CoLA

1. Introduction

Vision signals exhibit substantial temporal redundancy: only a fraction of pixel intensities change over short time intervals. Consequently, capturing frames at fixed time intervals and processing every pixel—including those that remain unchanged—is computationally wasteful, leading to slow and resource-intensive perception systems. By contrast, sparse sensing and processing focus solely on these changes—an approach akin to biological vision—promising more efficient perception in continuous scenarios, such as autonomous driving. Event-based cameras, which use pixel arrays that asynchronously detect intensity changes, facilitate sparse and low-latency data acquisition, making them well-suited for efficient perception.

Numerous studies have tried to exploit the inherent *sparsity* in event signals to reduce both multiply-accumulate (MAC) count and processing time. Point-set-based methods treat events as sparse point clouds, modeling dependencies via point-wise feature embedding followed by permutation-invariant aggregation [46, 55, 65, 67]. Graph based methods represent events as nodes in sparse graphs and perform sparse graph operations [6, 7, 33, 50, 51]. EventFormer [30] introduced RNN with augmented memory, further reduced MAC by re-utilizing past computation by *recurrency*. Event-SSM [52] utilizes SSM, equivalently linear attention (LA) [14], for processing raw event; it is not lower in MAC but realized fast inference on GPU utilizing *parallel* capability of SSM. However, if one has to model event-to-event relation (e.g., for dense prediction task), we argue that an important property of "*locality*" is missing in LA/SSM.

To this end, we present *CoL^{2}A* (*C̄ōnvolution-free L̄ocal L̄inear Ātention*) which is linear attention incorporating locality without convolution. In the case of linear attention, locality could be incorporated by applying convolutional positional embedding (PE) on the key vector. However, the convolution consumes a substantial amount of MAC, prohibiting efficient execution. We find that the specific class of convolutional PE kernel (2D rotation) can be decomposed as global PE applied on query and key vectors. This decomposition replaces convolution into local sum

	Sparse	*Recurrent*	*Local*	*GPU Para.*
ConvRNN		✓	✓	
GNN	✓		✓	✓
PI aggregation	✓		✓	✓
PointRNN	✓	✓		
SNN	✓	✓	✓	
LA/SSM	✓	✓		✓
CoL^{2}A	✓	✓	✓	✓

Figure 1. **Key features of *CoL^{2}A***. *CoL^{2}A* embodies essential features for efficient AI. *ConvRNN*: Incorporates recurrence by RNN into CNN [53]. *GNN*: Graph Neural Network [46]. *PI aggregation*: Exploit Permutation invariant (PI) operation for point feature [46]. *PointRNN*: Point-wise embedding followed by RNN (with augmented memory) [30]. *SNN*: Spiking Neural Network [37]. *LA*: Linear Attention [32]. *SSM*: State Space Model [14].

*Corresponding author (sekikawa.yusuke@core.d-itlab.co.jp), † Equal contribution.

(Fig. 4). *CoL*^{2}*A* inherits all the favorable features of linear attention while achieving locality without convolution, thereby achieving *sparse*, *recurrent*, *GPU-parallel*, and *local* properties simultaneously for the first time (Fig. 1). Furthermore, we formulate an efficient scan algorithm for temporal contraction in *CoL*^{2}*A* which scales logarithmically with temporal length. As a result, *CoL*^{2}*A* makes the real-time processing of high temporal resolution signals feasible. We demonstrate the effectiveness of *CoL*^{2}*A* on dense prediction tasks—keypoint detection and video reconstruction—that benefit from the high fps output. In addition, we showcase its unique feature of resolution equivariance: increasing output rate without re-training.

2. Related Work

This section recaps literature on neural networks for events from the perspective of equipped functionality for efficient inference on dense prediction tasks (Fig. 1).

Convolutional RNN. A common strategy for handling event signals is to convert them into dense, frame-like representations [8, 21, 58], then process these frames with standard computer vision networks (e.g., CNNs and Transformers). This frame-based approach often achieves robust accuracy by leveraging well-established architectures designed for image-like data. Some studies introduce recurrency [20, 22, 23, 26, 53, 54], reducing MAC by reusing past computation. However, because these methods ignore the underlying sparsity of event streams, they still perform unnecessary computations in event-free regions—especially problematic for high temporal resolutions processing.

Graph Neural Network (GNN). GNNs offer a way to process event data in its native sparse form without needing to convert it into dense frames [6, 7, 33, 50, 51]. Because each graph node is defined only at event locations, this approach utilizes the sparsity of the data. However, updating the graph for each new event—removing outdated nodes and adding new ones—requires storing past events, which can become memory-intensive. Moreover, many existing GNN-based methods must downsample the event stream to keep computational and memory usage within a manageable size, potentially degrading performance.

Permutation Invariant (PI) Aggregation. In this approach, events are treated as 3D point clouds in spatiotemporal space, and point-based architectures like PointNet [45, 46] are used to capture dependencies among the sparse events [55, 65, 67]. While this strategy also leverages event sparsity, most models are not recurrent: every time new events arrive, the network must reprocess all events within the time window, leading to a heavy computational burden. EventNet [55] introduces a recursive update with an exponential decay mechanism, but this design cannot incorporate local memory while preserving recursion. The extension of the model, ALERT [65], improved accuracy via local memory but abandoned recurrency in the process.

Point RNN. This approach combines sparse, point-wise operations with recurrent neural networks (RNNs) to capture temporal dependencies among events. EventFormer [30] introduced an external memory module in place of the recurrent states of the conventional recurrent module, enhancing memory capacity with little overhead. Although it achieves state-of-the-art MAC/accuracy trade-offs, these methods are difficult to use for processing high temporal resolution data because their recurrent formulation does not allow parallel execution.

Spiking Neural Network (SNN). SNNs [37, 41, 76] communicate information via asynchronous spikes. Due to their sparse and asynchronous computation mechanism, SNNs are considered one of the promising models for event data. However, training SNNs is challenging due to the non-differentiability of binary spikes. Even approaches that mitigate this problem (e.g., Sigma-Delta model [42]) typically require specialized neuromorphic hardware [1, 3, 15]. Such hardware is still in its infancy [47] and generally supports too few neurons to match the performance of deep neural networks running on optimized hardware like GPUs.

Linear Attention and State Space Model. Modeling long sequences is a major challenge in machine learning, as it demands capturing dependencies across thousands of time steps. While RNNs have historically been popular for modeling temporal data, their sequential computation inhibits training on long sequences. Backpropagation through time (BPTT) often suffers from vanishing or exploding gradients, which increases memory usage and training costs. Attention mechanisms [66] offer parallel capabilities, successfully handling sequences longer than those typically feasible for conventional RNNs. It also demonstrates remarkable success in the computer vision area [17, 39, 73] However, the quadratic complexity of standard attention hinders efficient inference. To alleviate this problem, numerous *efficient* variants have been proposed, including linearized approaches such as Linear Transformers [32], RWKV [44], and RetNet [62]. These methods provide linear complexity in sequence length and support $\mathcal{O}(1)$ incremental updates by caching past computations.

State-space models (SSMs) are another promising framework for modeling long-range dependencies. In particular, Structured SSM (S4) [25] demonstrates remarkable performance on long-sequence tasks by introducing structure into the state transition matrix. Recent variants, including S5 [59] and Mamba-1,2 [14, 24], further refine these ideas. Mamba, for instance, incorporates input-dependent state transitions that enable tackling complex tasks that are often beyond the reach of basic SSMs. Refer

to [14] for a detailed analysis of the equivalences between certain forms of SSMs and linear attention.

Efforts to apply attention models to event data often utilize patch-based tokens (like ViT [17]), achieving strong results on various tasks [26, 29, 31, 70, 71, 78]. However, patch-based tokens underutilize the data's sparsity inherent in event streams; though efficient for high-level understanding, it is not an optimal architecture for capturing precise event-to-event relationships from high temporal resolution data. Event-SSM [52] and Event-Mamba [49] pioneered the use of SSM for processing sparse events without making dense features. Event-SSM adopts SOTA SSM called S5 [59] for directly processing each incoming event. It is not only sparse, but recurrent and GPU parallel, inherited from the base SSM. Yet, due to a lack of locality, it compresses all relevant spatiotemporal information in a single state vector; therefore, the state's dimension needs to be huge to store fine-grained spatial detail (Sec. 4.1), requiring infeasibly large amounts of MAC and memory.

There are several works realizing the *local* attention [68, 69, 72] aiming to improve the computational efficiency over the ViT. However, their formulation is for softmax attention; therefore not readily usable for linear attention for recurrent update. Furthermore, these methods rely on the MAC intensive convolution to incorporate locality. To realize the efficient AI for spatiotemporal data, we aim to incorporate locality into linear attention (or SSMs) without using convolution while preserving their recurrency.

3. Preliminaries

3.1. Input representation.

Event-based cameras consist of pixels that respond asynchronously to changes in brightness. Each event is defined as (x, y, p, t), where (x, y) is the pixel location, $p \in \{-1, 1\}$ indicates polarity, and t is the timestamp. Multiple events can be triggered when an edge crosses a pixel, resulting in millions of events per second. In this work, to remove redundant events, we adopted commonly used compressed events by trilinear voting [77] as follows:

$$\tilde{E}\,(x, y, t) = \sum_i p_i \max\left(0, 1 - |t - t_i^*|\right), \qquad (1)$$

where t_i^* is the normalized timestamp of the i-th event over the interval Δ_{in}. We sparsify this voxel grid to form a set of compressed events $E \in \mathbb{R}^{N\times 4}$, with each event represented as $\mathbf{e}_n = (x_n, y_n, \rho_n, t_n)$, where ρ_n is the accumulated polarity and N is the length of the compressed event stream.

3.2. Problem Formulation.

Given N sparse events $E \in \mathbb{R}^{N\times 4}$, we construct an input feature $X \in \mathbb{R}^{N\times d_f}$ by assigning a learnable d_f-dimensional vector scaled by each event's accumulated polarity ρ (Eq. (1)). Our objective is to formulate a neural network module, $\mathcal{R}$, which maps $X \in \mathbb{R}^{N\times d_f}$ to $Y \in \mathbb{R}^{N\times d_f}$ by modeling the causal dependencies between the N elements of the sparse input feature.

3.3. Linear Attention with input dependent decay.

Here we recap the linear attention model with input-dependent decay, i.e., Mamba-2 [14], which we'll incorporate locality in the next section.

Fully Parallel Form. Linear attention [32] is defined as:

$$Y, H_{N-1} = \mathcal{R}(X, \mathbf{0}) = (QK^\top \odot L)V, \qquad (2)$$

where $Q \in \mathbb{R}^{N\times d_{qk}}, K \in \mathbb{R}^{N\times d_{qk}}, V \in \mathbb{R}^{N\times d_v}$ are query, key, and value vector corresponding to the input $X \in \mathbb{R}^{N\times d_f}$, and $Y \in \mathbb{R}^{N\times d_v}$ is the output feature. This is known to be equivalent to a certain class of SSM [14]. L is the causal decay mask. The original linear attention [32] uses a causal mask without decay (lower triangular matrix of all $\mathbf{1}$s). RetNet [63] uses exponentially decaying causality with a pre-fix parameter. Mamba-2 [14] generalizes this to use the input-dependent decay. The decay mask $L \in \mathbb{R}^{N\times N}$ is an exponential of the accumulation of the input-dependent decay $D(X_n) \in \mathbb{R}^{d_{qk}}_-$:

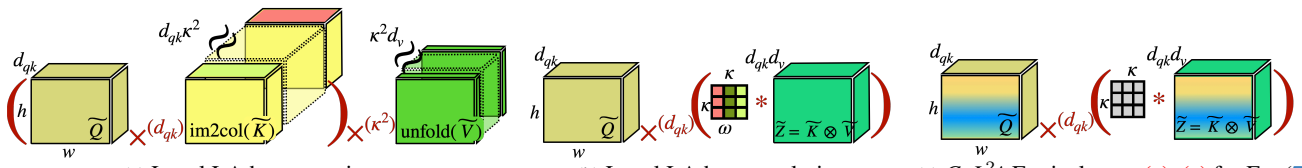
$$L_{nm} = \begin{cases} \exp(\Gamma_n)\ldots\exp(\Gamma_{n-m}), & n \geq m \\ 0, & n < m \end{cases}. \qquad (3)$$

When used for data having irregular timestamps, such as events, it can be incorporated using the difference of input events' timestamps as $\Gamma_n = D(X_n) \cdot (t_n - t_{n-1})$. The dot-product similarity of a query with all key of preceding inputs generates the attention map, where the attention map is discounted based on the causal decay mask L. The output is a projection of a value to this attention map.

Parallel-Recurrent Hybrid Form. Using the linearity, Eq. (2) can be rewritten as $(QK^\top \odot L)V = Q(LZ) = QH$, where $Z \in \mathbb{R}^{N\times d_{qk}\times d_v}$ is the pre-contraction key-value sequence, and $H \in \mathbb{R}^{1\times d_{qk}\times d_v}$ is the aggregated memory encoding the weighted key-value interactions for query projection. By inspection of Eq. (3), the memory H can be updated recurrently at each step as $H_n = \exp(\Gamma_n)H_{n-1} + Z_n$. Chunk of sequential data can be processed in parallel for efficiency, leading to parallel-recurrent hybrid update as:

$$\mathcal{R}(X_{n:}, H_{n-1}) = Q_{n:}H_{n:} = Q_{n:}(L_{n:}Z^\dagger_{n:}), \qquad (4)$$

where $Z^\dagger_{n:} = Z_{n:} + [\exp(\Gamma_n)H_{n-1}, \mathbf{0}]$; which incorporates previous states into the first element of $Z_{n:}$. This formulation allows partial state updates to be performed in parallel while leveraging previously computed memory states, significantly reducing computational costs over the fully parallel form of Eq. (2).



(a) Local LA by expansion (b) Local LA by convolution (c) *CoL*^{2}*A* Equivalent to (a), (c) for Eq. (7)

Figure 2. **Local attention: *CoL*^{2}*A*, compared with naive methods.** **(a)** Naive local attention expands $\tilde{K}$ and $\tilde{V}$ in a sliding-window fashion (unfold-like) to obtain a local attention map. **(b)** In the case of linear attention, it can be recast as a convolution. **(c)** *CoL*^{2}*A* applies a *global* positional embedding to both $\tilde{Q}$ and $\tilde{K}$ before forming $\tilde{Z}$, then compute local sum for contracting local feature by Eq. (8), followed by the multiplication by $\tilde{Q}$. *CoL*^{2}*A* is mathematically equivalent to the other two when kernel are in specific group (Eq. (7)), it is *multiplication-free* for the $\tilde{Z}$ contraction step and thus more efficient. Note: $\times^{(*)}$ indicates the contraction along $*$-dim in multiplication. Refer to Supp.2-Fig. S2 for intuitive derivation of *CoL*^{2}*A* (Eq. (8)) for Local LA by convolution (Eq. (6)).

4. Method

CoL^2*A* incorporate locality in linear attention while avoiding convolution; realizing *Sparse*, *Recurrent*, *Local*, and *GPU-Parallel* simultaneously (Fig. 1).

4.1. Why Does Locality Matter?

As explained in Sec. 3, linear attention (or equivalently SSM) could process events efficiently by its sparse, recurrent, and parallel capabilities; however, it is ill-suited for dense prediction tasks. Remember that new events interact through the single *memory* shared by all pixels. It means information from entire pixels is compressed into a single vector of $\mathbb{R}^{1\times d^{qk}\times d^{v}}$. It might be sufficient for tasks that do not require detailed spatiotemporal information, such as object classification; though , for the dense prediction task (such as keypoint detection), the dimension of the memory needs to be huge to store the details in the entire image, resulting in a substantial amount of compute (Tab. 1, GLA). Note that many low-level perception tasks do not necessarily require modeling the global dependency, but local information suffices. We extend the linear attention to use small local memory instead of a single high-dimensional memory to achieve better accuracy with little compute.

4.2. Local Linear Attention

In this section, for the clear explanation, let all the variables be in the sparse voxel format; i.e., the 1D sequence of input X, which corresponds to the event stream having the same discretized timestamp, would now be represented as $\tilde{X} \in \mathbb{R}^{(hw)\times d_f}$. Here for brevity, we omit the decay term L, and consider the case $d_{qk} = 2$, $d_v = 1$. In this sparse grid format, Q, K, V, Z, Y is represented as $\tilde{Q}, \tilde{K} \in \mathbb{R}^{(hw)\times 1\times 2}$, $\tilde{Z} \in \mathbb{R}^{(hw)\times 2\times 1}$, $\tilde{V}, \tilde{Y} \in \mathbb{R}^{(hw)\times 1}$. Extending the global linear attention of Eq. (3), local linear attention is

$$\tilde{Y} = \left(\tilde{Q}\,\mathrm{PE}(\bar{K}^\top)\right)\bar{V}, \tag{5}$$

$\bar{K} \in \mathbb{R}^{(hw)\times\kappa^2\times 2}$, $\bar{V} \in \mathbb{R}^{(hw)\times\kappa^2\times 1}$ represents the locally expanded $\tilde{K}, \tilde{V}$ (e.g., $\{\bar{K}, \bar{V}\} = \mathrm{unfold}(\{\tilde{K}, \tilde{V}\}, \kappa)$, It computes the attention map at each pixel by the dot product between the query and keys around, then aggregates values around with the map. When PE is identity mapping, the output remains invariant to permutations in the local region, thus being unable to extract spatial patterns.

Local PE by Local Expansion. By encoding local coordinates, the output depends on the order; enabling the extraction of spatial patterns (Fig. 2a). Here we embed the local position into the key as $\mathrm{PE}(\bar{K}^\top) = (\bar{K}\omega)^\top$ where ω is local PE matrix $\omega \in \mathbb{R}^{\kappa^2\times 2\times 2}$.

Local PE by Convolution. By recognizing the linearity in the attention map, we can reformulate it as a 2D convolution as follows:

$$\tilde{Y} = \tilde{Q}(\omega * \tilde{Z}). \tag{6}$$

Thought, this formulation is more memory efficient than the local expansion approach of Fig. 2a, it still consumes a significant amount ($2d_{qk}d_v\kappa^2$) of MAC for embedding the position; still, prohibiting the efficient use of local memory.

4.3. CoL^{2}A

Our main contribution in this paper is the derivation of the efficient convolution-free algorithm. The core idea for realizing this is to restrict the class of the kernel such that it can be decomposed as two global PEs which can later be absorbed into the query and keys to avoid coevolution. We consider the special PE kernel $\tilde{\omega}$ as follows:

$$\tilde{\omega}^{u,v,:,:} = \mathtt{R}(\theta u + \phi v) = e^{i(\theta u+\phi v)} \in SO(2), \tag{7}$$

where θ, ϕ are learnable scalars, u, v indicates the local position in the $\kappa\times\kappa$ kernel and $\mathtt{R}(\cdot)$ is a 2D rotation matrix. We decompose the contribution of the kernel $\mathtt{R}(\theta u + \phi v)$ into the global PE on Q and K. Finally, we get the *CoL*^{2}*A* mechanism (Fig. 2c) as follows:

$$\tilde{Y} = \left(\mathtt{R}(\theta\mathbf{X} + \phi\mathbf{Y})\cdot\tilde{Q}\right)(\mathbf{1} * \mathtt{R}(\theta\mathbf{X} + \phi\mathbf{Y})\cdot\tilde{Z}), \tag{8}$$

where $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{h\times w}$ is the horizontal and vertical global pixel coordinates, $\mathbf{1} \in \mathbb{R}^{\kappa\times\kappa}$ is all one matrix. Output $\tilde{Y}$ is

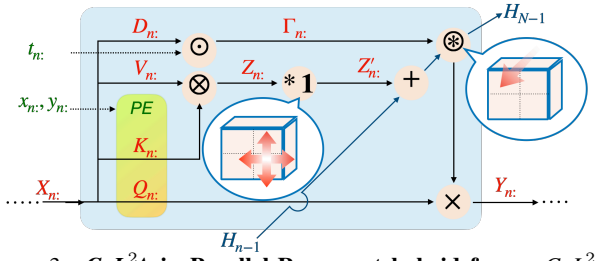


Figure 3. ***CoL²A* in Parallel-Recurrent hybrid form**. *CoL²A* incorporate locality into linear attention (Sec. 3.3) inheriting all the favorable features from them; it embodies *sparse*, *recurrent*, *GPU-parallel*, and *local* properties simultaneously. For the pre-contraction key-value matrices, $Z = K \otimes V$, corresponding to the latest T events, it first performs spatial contraction by local sum $*\mathbf{1}$ (Sec. 4.3, which is convolution-free), followed by the temporal contraction by associative scan $\circledast$ (Sec. 4.5). It processes $T = N - n$ events in parallel while re-utilizing the past compute through the low-dim local. The output is invariant for the chosen T; full parallelization when $T = N$, purely recurrent when $T = 1$.

computed as pixel-wise multiplication with $\tilde{Q}$ with global PE and local sum of $\tilde{Z}$ with global PE (Fig. 2c). The location information is embedded with a $2d_{qk}$ MAC per pixel. It is $d_v\kappa^2/2$ times less than the convolution-based formulation of Eq. (6). Note that with the specialization of local kernel Eq. (5)-(7) are equivalent to adopting RoPE [28, 61] for the local region so as *CoL²A* of Eq. (8).

Lemma 1. *CoL²A (Eq. (8)) is equivalent to convolutional PE (Eq. (6)) with rotation kernel of Eq. (7).*

Proof. Eq. (8) on pixel (x, y) can be reformulated as:

$$\begin{aligned}\tilde{Y}^{x,y} &= e^{i(\theta x+\phi y)}\tilde{Q}^{x,y}\sum_{(u,v)\in\kappa} e^{i(\theta(x+u)+\phi(y+v))}\tilde{Z}^{x+u,y+v} \\ &= \tilde{Q}^{x,y}\sum_{(u,v)\in\kappa} e^{-i(\theta x+\phi y)}e^{i(\theta(x+u)+\phi(y+v))}\tilde{Z}^{x+u,y+v} \\ &= \tilde{Q}^{x,y}\sum_{(u,v)\in\kappa} e^{i(\theta u+\phi v)}\tilde{Z}^{x+u,y+v}.\end{aligned}$$

We use superscripts to index the spatial location, e.g., $\tilde{Q}^{x,y} \in \mathbb{R}^2$ extracts the vector at pixel (x, y). This equals Eq. (6) for the special kernel of Eq. (7). □

Demo. Figure 4 demonstrates *CoL²A* extracting local features, i.e., *edge*, *without using convolution*. When built into a neural network, Q, K, V projection and rotational coefficients θ, ϕ are optimized to extract local features.

4.4. Parallel-Recurrent Hybrid Form of CoL²A

CoL²A is compatible with the recurrent-parallel update of Eq. (4). After the spatial contraction of Z in Eq. (8), one can treat the resultant feature in each pixel as an independent temporal sequence; therefore, they can be processed in parallel using the same equation of Eq. (4) (Fig. 3).

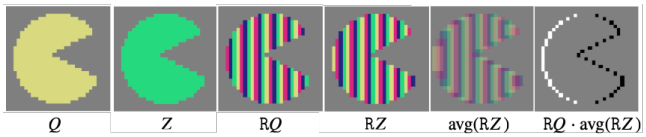


Figure 4. *CoL²A* demonstration for convolution free edge detection. Each pixel of Q and Z is $\mathbb{R}^2$; having unit norm on the colored region ($Q \perp Z$, color encodes angle), and 0 otherwise. Horizontal global location is embedded with $\theta = (\pi/2)$/pix as $\mathtt{R}Q$, and $\mathtt{R}Z$. Local average (1×3) is applied as avg($\mathtt{R}Z$). The dot product $\mathtt{R}Q \cdot$ avg($\mathtt{R}Z$) respond strongly on edges.

4.5. Linear-Time Temporal Contraction via Scan

CoL²A involves temporal contraction of the key–value term as $H{=}LZ$. Directly multiplying the decay term L leads to a quadratic $\mathcal{O}(T^2)$ complexity for token length T. Drawing inspiration from Heinsen's algorithm [27], we derive a stable, efficient algorithm (code: Fig. S1) as follows:

$$H = \exp\left(\Gamma^* + \log\left(Z^*\right)\right), \tag{9}$$

Where $\Gamma^* = \sum_n^{\text{cum}} \Gamma_n$, $Z^* = \sum_n^{\text{cum}} \exp(\log Z_n - \Gamma^*)$, and $\sum_n^{\text{cum}}$ denote a prefix (cumulative) sum. This formula reduces complexity to $\mathcal{O}(T \log T)$, a significant improvement over the original $\mathcal{O}(T^2)$ complexity. As the decay L is linear in Z, H can be decomposed as: $H = H_+ - H_-$, where $H_{\{+,-\}}$ is computed from positive and negative parts of Z. This further reduces latency by avoiding the logarithms of negative values, which involves complex numbers.

Compared to the chunk-wise formulation in RetNet [63] and Mamba-2 [14]—which first multiply query-key inside each chunk, leading to $\mathcal{O}(T^2)$ MAC and $\mathcal{O}(N^2)$ memory—our key-value first formulation is more MAC and memory efficient when feature dimension is small.

4.6. Computational Complexity Analysis

VS. other Linear Attention. Tab. 1 compared the MAC of *CoL²A* with other linear attention mechanisms. Global linear attention (GLA), e.g., RetNet [63], Mamba-2 [14], shares a single memory for all pixels; therefore, the memory dimension $d_{qk}^G d_v^G$ needs to be very large to store the spatial detail, making computation prohibitively expensive (GLA in Tab. 3-4. Sec. 4.1). With local memory by Eq. (6), each vector in the local memory could be small, however, it necessitates convolution for positional embedding, still consuming substantial computation. *CoL²A* addresses this by the convolution-free formulation.

VS. Conv-GRU. Table 2 compares the *span* (the longest chain of data-dependent operations that determines minimal parallel execution time [13]) between *CoL²A* and a de facto local nonlinear recurrency (Conv-GRU [5], Conv-LSTM [56]). While the nonlinear formulation requires $\mathcal{O}(T)$ sequential updates, *CoL²A* parallelizes this temporal contraction with Eq. (9), reducing the span to $\mathcal{O}(\log(T))$.

Table 1. MACs of *CoL²A*, compared with other linear attention (LA) for T=1, ignoring decay. **G**lobal **LA** (Mamba-2 [14], RetNet [63]) use single shared memory. **L**ocal **LA** use convolution.

	GLA	LLA	*CoL²A*
$K \otimes V \rightarrow Z$	$hwd_{qk}^G d_v^G$	$hwd_{qk}d_v$	$hwd_{qk}d_v$
Pos. Emb. (PE)	$hw4d_{qk}^G$	$hw2d_{qk}d_v\kappa^2$	$hw4d_{qk}$
Mem. size	$d_{qk}^G d_v^G$	$hwd_{qk}d_v$	$hwd_{qk}d_v$

Table 2. Parallel time complexity (span) of the core component of *CoL²A*, compared with nonlinear local recurrent block.

Nonlinear local recurrent unit		Linear local recurrent unit: *CoL²A*	
GRU/LSTM	$\mathcal{O}(T)$	Scan (Eq. (9))	$\mathcal{O}(\log(T))$
Convolution	$\mathcal{O}(1)$	Linear (FFN etc.)	$\mathcal{O}(1)$
		Sum (Eq. (8))	$\mathcal{O}(1)$

5. Experiments

We compare the efficiency of *CoL²A* with the recurrent unit capable of modeling the dense feature at the pixel resolution (without patching). Conv-GRU [5], a convolutional variant of the GRU [12] is de facto in this category. Actually, FireNet [54], which is built on Conv-GRU, processes features at the input camera resolution without spatial pooling, realizing fine-detailed feature extraction with small compute. It significantly outperforms non-recurrent approaches, like E2Vid [48], in terms of the accuracy/MAC.

In addition to the intensive comparison with the Conv-GRU baseline, we also evaluate the network adopting 1) linear attention with global memory (*GLA*), and 2) linear attention with local memory with convolution (*LLA*). *GLA* is essentially the upgraded variant of Event-SSM [52] adopting Mamba-2 [14] in place of S5 which incorporates input-dependent decay. LLA-Net replaces the average operation of *CoL²A* with convolution (Eq. (6)). Both serve as ablation for the *local* memory mechanism by *CoL²A*, which is the core contribution of this study.

Main experiments focus on two dense tasks—keypoint detection and video reconstruction—both benefit from the high temporal resolution offered by event-based cameras. Low-latency keypoint detection enables robust tracking with minimal computational overhead, while high-speed video reconstruction from compressed events represents another compelling application. FireNet is currently the de facto choice for these tasks on resource-limited devices. We consider, *CoL²A* is inherently better suited to these scenarios due to its efficient handling of sparse and high temporal resolution data. We'll verify this in the experiments.

5.1. Experimental Setup

We aim to compare the accuracy/computation tradeoff that originated solely from the differences in their core recurrent mechanism, ConvGRU, GLA, and *CoL²A* to provide a clear view of *CoL²A*'s advantages in sparse, high-resolution event processing. To do this, we carefully designed the experimental setting as follows:

Table 3. Corner tracking performance comparison on ATIS Corner dataset [38] (for 100ms interval) (Table S2 for more results.).

	Arc [2]	SILC[38]	GLA/-L	FireNet[10]	*CoL²A*
Reprj. (pix) ↓	7.22	3.68	NA/NA	11.1[1]	**4.98**[1]
Track (sec) ↑	0.91	1.12	0.0/0.0	8.84	**13.0**

Table 4. Image reconstruction accuracy comparison on the HQF dataset [60]. ET-Net using 1,000x more parameters (refer to Tab. 5) is provided only for reference. (Table S3 for more results.)

	ETNet [71]	GLA	LLA	FireNet [10]	*CoL²A*
MSE ↓	0.0634	0.0787	**0.0780**	0.0784	0.0787
SSIM ↑	0.4571	0.3610	**0.4110**	0.4100	0.4102
LPIPS ↓	0.3279	0.3461	0.3425	**0.3418**	**0.3418**

Table 5. Memory footprint comparison. A number of learnable model parameters and a size of states memory (180×240 input).

	ETNet [71]	GLA	LLA	FireNet [10]	*CoL²A*
Param.	22.2M	16.1K	17.2K	27.1K	16.1K
Memory	4.72M	0.29K	958K	1.01M	958K

Network design. Baseline (FireNet) consists of two Conv-GRU blocks (each with a hidden dimension of $d = 12$) for feature extraction, followed by a prediction head [9]. *CoL²A*-Net follows the same overall architecture, using two *CoL²A* blocks (Supp.4-Fig. S3) for sparse feature extraction. GLA-Net and LLA-Net replace *CoL²A* with global linear attention (GLA) and convolution-based local linear attention (LLA), respectively, keeping the rest unchanged. We match the spatial dimensions of input/output and set d_f=d_{qk}=d_v=12 with two heads mirroring the baseline configuration. The spatial size of local memory is set to $(h/5, w/5)$, which consumes a similar memory footprint for recurrent states as the baseline (Tab. 5).

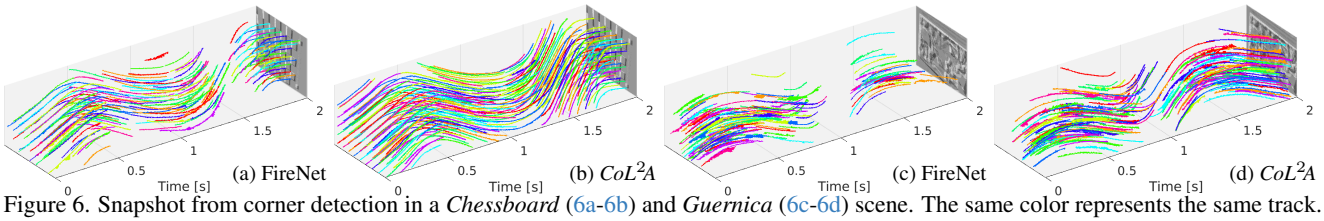
Training data. All the models are trained on simulated events; moving a virtual camera in 6-DOF motion observes random images from MS-COCO [34]. Following practices to narrow the simulation-to-real gap, we utilized multiple types of noise (e.g., timestamp jitter) using the simulator from OpenEB [9]. We feed dense voxelized inputs to the baseline FireNet and convert these into sparse streams (Sec. 3) for GLA, LLA, and *CoL²A*, ensuring all networks receive identical input and utilize the same loss functions.

Input/Output Rate. We quantize event timestamps in $\Delta_{\text{in}} = 1$ms. We use $100\Delta_{\text{in}}$ sequence during training with gradient flow. FireNet uses a nonlinear recurrency; it is infeasible to train the fine-grained temporal signal due to the enormous memory demands and unstable gradients during BPTT. So, following the original protocol [54]), we concatenate 10 time steps along the feature dimension ($\Delta_{\text{out}} = 10\Delta_{\text{in}}$) to reduce the recurrent step. Contrarily, *CoL²A*, GLA, and LLA allow training on the fine-grained time resolution ($\Delta_{out} = \Delta_{in}$) by their parallel mechanism.

[1] Larger error over SILC is due to smoothing effect. Also report in [10].

(a) FireNet (Δ_{in}=1ms) (b) FireNet (Δ_{in}=10μs) (c) GLA (Δ_{in}=1ms) (d) GLA-L (Δ_{in}=1ms) (e) *CoL*^{2}*A* (Δ_{in}=1ms)

Figure 5. Image Reconstruction on the High-Speed and HDR data set [48]. (Figure S5 for more result.)



(a) FireNet (b) *CoL*^{2}*A* (c) FireNet (d) *CoL*^{2}*A*

Figure 6. Snapshot from corner detection in a *Chessboard* (6a-6b) and *Guernica* (6c-6d) scene. The same color represents the same track.

Loss Functions and optimization. For keypoint detection, we follow [11], where the network directly outputs a cornerness heatmap. We use binary cross-entropy between the ground truth and the predicted heatmap. For video reconstruction, we adopt the protocol of [20], in which the network predicts image gradients and then integrates them to reconstruct intensity images using the Frankot–Chellappa algorithm [19, 75]. The loss is a combination of mean squared error (MSE), structural similarity index measure (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS) [74]. Since the output temporal resolution of *CoL*^{2}*A*/GLA/LLA-Net is 10× finer than FireNet, we interleave their output to use the same loss. We optimized all the networks using the same protocol, which was primarily based on the baseline method. We trained for 10 epochs using the AdamW optimizer [36] with a learning rate of 10^{-3}.

5.2. Results

Compute and Memory Complexity. Table 5 compares memory footprint. Table 6 compares computational complexity–MACs and wall-clock latency–in five scenarios, event-wise update, $\{1, 10, 100, 1000\}$ frames parallel update. In the event-wise update scenario, our method requires very few MAC because an input event only interacts with local memory (MAC for a single event is the same for different resolutions). In contrast, Conv-GRU must process all pixels, even when there are little or no events, resulting in a significantly higher MAC requirement than our method. In batch processing scenarios, the MAC count of both methods increases almost linearly with the number of parallel processing T. There is a notable difference in the wall clock latency. Thanks to the *CoL*^{2}*A*'s parallel mechanism (Eq. (9)), it can compute multiple timestep output in a single forward pass for any T, resulting in faster inference. Conversely, nonlinear recurrent units must update the output sequentially, preventing them from leveraging the GPU's parallel capabilities for the temporal dimension. The results reflect the *parallel time complexity* in Tab. 2.

Keypoint detection. We evaluate each model's precision by track length, following the protocols in [11] on the ATIS Corner dataset [38]. We apply non-maximum suppression (7×7 kernel with a threshold of 10^{-6}) to the network's predictions to identify corners, then employ a simple nearest-neighbor tracking algorithm to form keypoint tracks. For each keypoint, we search for a 5×5 spatial neighbor within the past 10 ms, updating the tracks accordingly. Table 3 presents the quantitative comparison with Arc [2], SILC [38], and FireNet [10]. Figure 6 provides qualitative results. *CoL*^{2}*A* significantly outperforms the baseline while using a fraction of MAC and model parameters.

Video reconstruction. We evaluate each model on the High Quality Frames (HQF) dataset [60] using MSE, SSIM, and LPIPS, following the prior work [71]. Table 4 presents the quantitative results, which also include ET-Net [71] here as a reference—this model achieves SoTA in terms of accuracy but has notably higher compute and memory. Additionally, we provide qualitative comparisons on the High-Speed and HDR (HSHDR) dataset [48] in Fig. 5. *CoL*^{2}*A* produces comparable reconstruction quality as FireNet with substantially lower compute. Moreover, unlike prior methods that degrade when the input rate is changed (e.g., to increased FPS) without re-training, our formulation mathematically guarantees invariant output.

Ablation on event data. GLA and LLA serve as an ablation on the local mechanism, *CoL*^{2}*A*, our main contribution. GLA-Net, which replaces *CoL*^{2}*A*'s local memory mechanism with the global linear attention from Mamba-2, fails on both tasks (Tab. 3-4, Fig. 5). The results verify that spatiotemporal detail cannot be compressed into the single memory (Sec. 4.1, GLA-L adopting 4 times larger d_{qk} also fails). *CoL*^{2}*A* performs comparably well as ones using convolution-based LLA, which consumes κ^2 times more compute for embedding the local position. These two ablations verify locality is essential for modeling fine-grained spatial relationships while the restricted kernel of *CoL*^{2}*A* suffices to capture local patterns.

Table 6. Computational complexity comparison with Conv-GRU block ($d=12$) and *CoL²A* block ($d_f{=}d_{qk}{=}d_v{=}12$ with two head). For the batch processing scenario, we assume that the sparsity of the input tensor is 90%. Value in each cell represents MAC count/latency [ms]. The latency is evaluated on NVIDIA A6000 GPU (N/A: Could not run due to the out of memory).

Resolution	Single event				$T=1$ frame				$T=10$ frame				$T=100$ frame				$T=1000$ frame			
	Conv-GRU		*CoL²A*		Conv-GRU		*CoL²A*		Conv-GRU		*CoL²A*		Conv-GRU		*CoL²A*		Conv-GRU		*CoL²A*	
240×180	0.51G	0.36	15K	0.04	0.51G	0.36	12M	0.14	5.1G	1.02	0.12G	0.34	50G	16	1.2G	3.8	0.51T	568	0.12T	44
346×260	1.1G	0.38	15K	0.04	1.1G	0.38	25M	0.15	10G	2.19	0.25G	0.66	0.11T	30	2.5G	9.0	1.1T	N/A	0.25T	98
640×480	3.6G	0.70	15K	0.04	3.6G	0.70	86M	0.23	36G	9.5	0.86G	2.6	0.36T	107	8.6G	34	3.6T	N/A	0.86T	N/A
1280×720	10G	2.6	15K	0.04	10G	2.6	0.26G	0.47	0.11T	32	2.6G	9.0	1.1T	N/A	25G	N/A	10T	N/A	0.26T	N/A

Table 7. Image classification on ImageNet1K.

Accuracy	Softmax	GLA	LLA	*CoL²A*
Train (%) ↑	37.58	35.56	39.90	38.51
Test (%) ↑	71.48	70.79	73.94	73.58

Table 8. Foundation model (DINOv3) distillation on ImageNet1K.

MSE	Softmax	GLA	LLA	*CoL²A*
Train ↓	1.016	1.275	0.926	0.955
Test ↓	1.124	1.504	0.952	0.976

Ablation on image data. We also ablate the contribution of *CoL²A* in a minimal and basic setting in terms of network architecture and data[2]. We designed two tasks: 1) image classification and 2) foundation model's (DINOv3 [57]) dense feature distillation. For both tasks, we used ImageNet1K [16] dataset and ViT-Tiny [64] network. Different from our main experiments, these tasks require global high-level understanding, We evaluate accuracy by replacing the softmax attention in the original ViT-Tiny with different attention mechanisms, including *CoL²A*. Results are reported in Tab. 7-8. On both tasks, *CoL²A* outperforms not only GLA (Mamba-2–like) but also softmax attention by a non-negligible margin. The gain is more pronounced on the dense feature distillation task. We also compare with LLA (Eq. (6)), local linear attention adopting the convolution. Unlike *CoL²A*, it can represent the arability kernel at the cost of increased computation (Tab. 1). Despite the strong restriction on kernel patterns, *CoL²A* achieves comparable test accuracy with LLA on both tasks.

5.3. Summary and Discussion

CoL²A delivers accuracy on par with—or even better than—its Conv-GRU-based counterpart (Tab. 3-4) while also providing more efficient computation (Tab. 6). Comparison with the global memory model (GLA) demonstrates the significant benefit of incorporating locality in memory (by *CoL²A*) for fine-grained dense prediction tasks. In addition, ablation on the dense task demonstrates the effectiveness of *CoL²A* on different data, tasks, and architecture.

[2]We developed *CoL²A* for sparse temporal data such as event streams. The core innovation of our algorithm—convolution-free local memory—is also compatible with dense, non-temporal architectures and data, e.g., image classification with ViT [18]. In this setting, as the number of tokens is small (hundreds vs. millions in the main experiments), we can compare with softmax attention. Refer to Supp.3 for detailed experimental settings.

An additional benefit of *CoL²A* is its equivariance to temporal resolution, which offers practical advantages. For instance, in a keypoint detection and tracking task, rapid camera motion causes large keypoint displacement, making the nearest neighbor-based association difficult. Increasing the input temporal resolution (easily done by reducing Δ_{in}, it is an advantage of event camera) may alleviate this. However, dense nonlinear recurrent unit (e.g., Conv-GRU) encounters two primary challenges. First, changing input temporal resolution alters the feature maps extracted by convolution-based nonlinear recurrency, which degrades performance unless re-trained (Fig. 5c). Second, the sequential dependence of nonlinear recurrency prevents parallel execution (Tab. 2). *CoL²A* addresses both. Linear recurrency ensures the same output for the number of timesteps being processed in parallel. Sparse capability keeps the computational cost invariant to the temporal resolution (except for the temporal contraction in Eq. (9)).

6. Conclusion

We introduce *CoL²A*, an efficient local linear attention mechanism for continuous perception from sparse spatiotemporal data. Our formulation achieves locality while avoiding computationally expensive convolution. Experiments show that *CoL²A* matches or surpasses existing methods while requiring significantly fewer MACs and parallel time complexity. Furthermore, ablation in dense and non-temporal settings indicates the possibility of using *CoL²A* on diverse data and architecture.

Limitations and future work. The kernel restriction in Eq. (8), while enabling convolution-free local attention, significantly reduces flexibility compared to convolutional parameterizations. Although we did not observe a severe accuracy drop in our ablation studies, broader evaluation across input modalities (e.g., events, images, point clouds) and tasks (e.g., detection, segmentation, tracking) is left for future research. From a different perspective, the convolution-free formulation of *CoL²A* for extracting the local pattern potentially equips useful spatial equivariance; for example, *CoL²A*'s output is consistent for up-sampled/down-sampled input without re-training (Supp.5). Extending *CoL²A* to incorporate spatial equivariance, such as rotation, scale or deformation, presents another promising direction for future work.

References

[1] Filipp Akopyan, Jun Sawada, Andrew Cassidy, Rodrigo Alvarez-Icaza, John Arthur, Paul Merolla, Nabil Imam, Yutaka Nakamura, Pallab Datta, Gi-Joon Nam, Brian Taba, Michael Beakes, Bernard Brezzo, Jente B. Kuang, Rajit Manohar, William P. Risk, Bryan Jackson, and Dharmendra S. Modha. Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(10):1537–1557, 2015. 2

[2] Ignacio Alzugaray and Margarita Chli. Asynchronous corner detection and tracking for event cameras in real time. *IEEE Robotics and Automation Letters*, 3(4):3177–3184, 2018. 6, 7, 5

[3] Arnon Amir, Brian Taba, David Berg, Timothy Melano, Jeffrey McKinstry, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, Jeff Kusnitz, Michael Debole, Steve Esser, Tobi Delbruck, Myron Flickner, and Dharmendra Modha. A low power, fully event-based gesture recognition system. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7388–7397, 2017. 2

[4] R. Wes Baldwin, Mohammed Almatrafi, Jason R. Kaufman, Vijayan Asari, and Keigo Hirakawa. Inceptive event time-surfaces for object classification using neuromorphic cameras. In *Image Analysis and Recognition*, pages 395–403, Cham, 2019. Springer International Publishing. 4

[5] Nicolas Ballas, Li Yao, Chris Pal, and Aaron C. Courville. Delving deeper into convolutional networks for learning video representations. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. 5, 6

[6] Yin Bi, Aaron Chadha, Alhabib Abbas, Eirina Bourtsoulatze, and Yiannis Andreopoulos. Graph-based object classification for neuromorphic vision sensing. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 491–501, 2019. 1, 2

[7] Yin Bi, Aaron Chadha, Alhabib Abbas, Eirina Bourtsoulatze, and Yiannis Andreopoulos. Graph-based spatio-temporal feature learning for neuromorphic vision sensing. *IEEE Transactions on Image Processing*, 29:9084–9098, 2020. 1, 2

[8] Marco Cannici, Marco Ciccone, Andrea Romanoni, and Matteo Matteucci. A differentiable recurrent surface for asynchronous event-based data. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 136–152. Springer, 2020. 2

[9] P.W.D. Charles. Openeb-core: Generic algorithms for visualization, event stream manipulation. `github.com/prophesee-ai/openeb`, 2025. 6

[10] Philippe Chiberre, Etienne Perot, Amos Sironi, and Vincent Lepetit. Detecting stable keypoints from events through image gradient prediction. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1387–1394, 2021. 6, 7, 5

[11] Philippe Chiberre, Etienne Perot, Amos Sironi, and Vincent Lepetit. Long-lived accurate keypoints in event streams. *ArXiv*, abs/2209.10385, 2022. 7, 4

[12] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In Dekai Wu, Marine Carpuat, Xavier Carreras, and Eva Maria Vecchi, editors, *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar, Oct. 2014. Association for Computational Linguistics. 6

[13] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009. 5

[14] Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024. 1, 2, 3, 5, 6, 4

[15] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, Yuyun Liao, Chit-Kwan Lin, Andrew Lines, Ruokun Liu, Deepak Mathaikutty, Steven McCoy, Arnab Paul, Jonathan Tse, Guruguhanathan Venkataramanan, Yi-Hsin Weng, Andreas Wild, Yoonseok Yang, and Hong Wang. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018. 2

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 8, 2

[17] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2, 3

[18] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 8, 2, 3

[19] Robert T. Frankot and Rama Chellappa. A method for enforcing integrability in shape from shading algorithms. *IEEE Transactions on pattern analysis and machine intelligence*, 10(4):439–451, 1988. 7, 4

[20] Pablo Rodrigo Gantier Cadena, Yeqiang Qian, Chunxiang Wang, and Ming Yang. Sparse-E2VID: A Sparse Convolutional Model for Event-Based Video Reconstruction Trained with Real Event Noise . In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4150–4158, Los Alamitos, CA, USA, June 2023. IEEE Computer Society. 2, 7, 4

[21] Daniel Gehrig, Antonio Loquercio, Konstantinos G. Derpanis, and Davide Scaramuzza. End-to-end learning of representations for asynchronous event-based data. In *Int. Conf. Comput. Vis. (ICCV)*, October 2019. 2

[22] Daniel Gehrig, Michelle Ruegg, Mathias Gehrig, Javier Hidalgo-Carrió, and Davide Scaramuzza. Combining events and frames using recurrent asynchronous multimodal networks for monocular depth prediction. *IEEE Robotics and Automation Letters*, 6, 2021. 2

[23] Mathias Gehrig and Davide Scaramuzza. Recurrent vision transformers for object detection with event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[24] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2024. 2

[25] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021. 2

[26] Ryuhei Hamaguchi, Yasutaka Furukawa, Masaki Onishi, and Ken Sakurada. Hierarchical neural memory network for low latency event processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22867–22876, June 2023. 2, 3

[27] Franz A. Heinsen. Efficient parallelization of a ubiquitous sequential computation, 2023. 5, 1

[28] Byeongho Heo, Song Park, Dongyoon Han, and Sangdoo Yun. Rotary position embedding for vision transformer. *arXiv preprint arXiv:2403.13298*, 2024. 5, 4

[29] Zhou Jiazhou, Chen Kanghao, Zhang Lei, and Wang Lin. Path-adaptive spatio-temporal state space model for event-based recognition with arbitrary duration. *arXiv [cs.CV]*, Sept. 2024. 3

[30] Uday Kamal, Saurabh Dash, and Saibal Mukhopadhyay. Associative memory augmented asynchronous spatiotemporal representation learning for event-based perception. In *The Eleventh International Conference on Learning Representations*, 2023. 1, 2

[31] Uday Kamal and Saibal Mukhopadhyay. Efficient learning of event-based dense representation using hierarchical memories with adaptive update. In *European Conference on Computer Vision*, 2024. 3

[32] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR, 2020. 1, 2, 3

[33] Yijin Li, Han Zhou, Bangbang Yang, Ye Zhang, Zhaopeng Cui, Hujun Bao, and Guofeng Zhang. Graph-based asynchronous event processing for rapid object recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 934–943, 2021. 1, 2

[34] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. *CoRR*, abs/1405.0312, 2014. 6

[35] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *CoRR*, abs/2201.03545, 2022. 2

[36] Ilya Loshchilov, Frank Hutter, et al. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*, 5, 2017. 7

[37] Wolfgang Maass. Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997. 1, 2

[38] Jacques Manderscheid, Amos Sironi, Nicolas Bourdis, Davide Migliore, and Vincent Lepetit. Speed invariant time surface for learning to detect corner points with event-based cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10245–10254, 2019. 6, 7, 5

[39] Sachin Mehta and Mohammad Rastegari. Mobilevit: lightweight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178*, 2021. 2

[40] Elias Mueggler, Chiara Bartolozzi, and Davide Scaramuzza. Fast event-based corner detection. 2017. 5

[41] Manish Nagaraj, Chamika Mihiranga Liyanagedera, and Kaushik Roy. Dotie - detecting objects through temporal isolation of events using a spiking architecture. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4858–4864, 2023. 2

[42] Peter O'Connor and Max Welling. Sigma delta quantized networks. In *International Conference on Learning Representations*, 2016. 2

[43] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 1

[44] Bo Peng, Eric Alcaide, Quentin Gregory Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Nguyen Chung, Leon Derczynski, et al. Rwkv: Reinventing rnns for the transformer era. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. 2

[45] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2

[46] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 1, 2

[47] Nitin Rathi, Indranil Chakraborty, Adarsh Kosta, Abhronil Sengupta, Aayush Ankit, Priyadarshini Panda, and Kaushik Roy. Exploring neuromorphic computing based on spiking neural networks: Algorithms to hardware. *ACM Comput. Surv.*, 55(12), Mar. 2023. 2

[48] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. High speed and high dynamic range video with an event camera. *IEEE transactions on pattern analysis and machine intelligence*, 43(6):1964–1980, 2019. 6, 7, 5

[49] Hongwei Ren, Yue Zhou, Jiadong Zhu, Xiaopeng Lin, Haotian Fu, Yulong Huang, Yuetong Fang, Fei Ma, Hao Yu, and Bojun Cheng. Rethinking efficient and effective point-based networks for event camera classification and regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(8):6228–6241, 2025. 3

[50] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal

graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020. 1, 2

[51] Simon Schaefer, Daniel Gehrig, and Davide Scaramuzza. Aegnn: Asynchronous event-based graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12371–12381, 2022. 1, 2

[52] Mark Schöne, Christian Mayr, and David Kappel. *Scalable Event-by-event Processing of Neuromorphic Sensory Signals With Deep State-Space Models*, volume 1. Association for Computing Machinery, 2024. 1, 3, 6, 4, 5

[53] Cedric Scheerlinck, Henri Rebecq, Daniel Gehrig, Nick Barnes, Robert E Mahony, and Davide Scaramuzza. Fast Image Reconstruction with an Event Camera. Technical report. 1, 2

[54] Cedric Scheerlinck, Henri Rebecq, Daniel Gehrig, Nick Barnes, Robert E Mahony, and Davide Scaramuzza. Fast image reconstruction with an event camera. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, Mar. 2020. 2, 6, 4, 5

[55] Yusuke Sekikawa, Kosuke Hara, and Hideo Saito. Eventnet: Asynchronous recursive event processing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3887–3896, 2019. 1, 2

[56] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Convolutional lstm network: a machine learning approach for precipitation nowcasting. In *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 802–810, Cambridge, MA, USA, 2015. MIT Press. 5

[57] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3, 2025. 8, 2

[58] Amos Sironi, Manuele Brambilla, Nicolas Bourdis, Xavier Lagorce, and Ryad Benosman. Hats: Histograms of averaged time surfaces for robust event-based object classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1731–1740, 2018. 2, 4

[59] Jimmy T. H. Smith, Andrew Warrington, and Scott W. Linderman. Simplified state space layers for sequence modeling, 2023. 2, 3, 4

[60] Timo Stoffregen, Cedric Scheerlinck, Davide Scaramuzza, Tom Drummond, Nick Barnes, Lindsay Kleeman, and Robert Mahony. Reducing the sim-to-real gap for event cameras. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16*, pages 534–549. Springer, 2020. 6, 7, 5

[61] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2021. 5

[62] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. July 2023. 2

[63] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models, 2023. 3, 5, 6

[64] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers distillation through attention. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10347–10357. PMLR, 18–24 Jul 2021. 8, 2

[65] Carmen Martin Turrero, Maxence Bouvier, Manuel Breitenstein, Pietro Zanuttigh, and Vincent Parret. Alert-transformer: Bridging asynchronous and synchronous machine learning for real-time event-based spatio-temporal data. In *Forty-first International Conference on Machine Learning*, 2024. 1, 2

[66] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, L ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 2

[67] Qinyi Wang, Yexin Zhang, Junsong Yuan, and Yilong Lu. Space-time event clouds for gesture recognition: From rgb cameras to event cameras. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1826–1835. IEEE, 2019. 1, 2

[68] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvtv2: Improved baselines with pyramid vision transformer. *CoRR*, abs/2106.13797, 2021. 3

[69] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 548–558, 2021. 3

[70] Zhong Wang, Zengyu Wan, Han Han, Bohao Liao, Yuliang Wu, Wei Zhai, Yang Cao, and Zheng-jun Zha. Mambapupil: Bidirectional selective recurrent model for event-based eye tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5762–5770, 2024. 3, 5

[71] Wenming Weng, Yueyi Zhang, and Zhiwei Xiong. Event-based video reconstruction using transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 3, 6, 7, 5

[72] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021. 3

[73] Kun Yuan, Shaopeng Guo, Ziwei Liu, Aojun Zhou, Fengwei Yu, and Wei Wu. Incorporating convolution designs into visual transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 579–588, 2021. 2

[74] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 7

[75] Hong-Kai Zhao, Stanley Osher, and Ronald Fedkiw. Fast surface reconstruction using the level set method. In *Proceedings IEEE workshop on variational and level set methods in computer vision*, pages 194–201. IEEE, 2001. 7

[76] Yajing Zheng, Zhaofei Yu, Song Wang, and Tiejun Huang. Spike-based motion estimation for object tracking through bio-inspired unsupervised learning. *IEEE Transactions on Image Processing*, 32:335–349, 2023. 2

[77] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised event-based learning of optical flow, depth, and egomotion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 989–997, 2019. 3

[78] Nikola Zubic, Mathias Gehrig, and Davide Scaramuzza. State space models for event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5819–5828, June 2024. 3