Adaptive Spatial-Temporal Window: Unlocking the Potential of Event Cameras in Heterogeneous Velocity Scenarios

Zhipeng Sui, Haiqing Hao, Weihua He, Seng-Hong Lee, Wenhui Wang

Department of Precision Instrument, Tsinghua University

CONTENT

1 Background and Motivation

2 Related Work

3 Adaptive Spatial-Temporal Window

4 Experiments and Results

Background and Motivation

• Neuromorphic Visual Perception

Concept: **A neuromorphic vision sensor**, also known as an **event camera**, is a visual sensor inspired by **biological vision systems**.

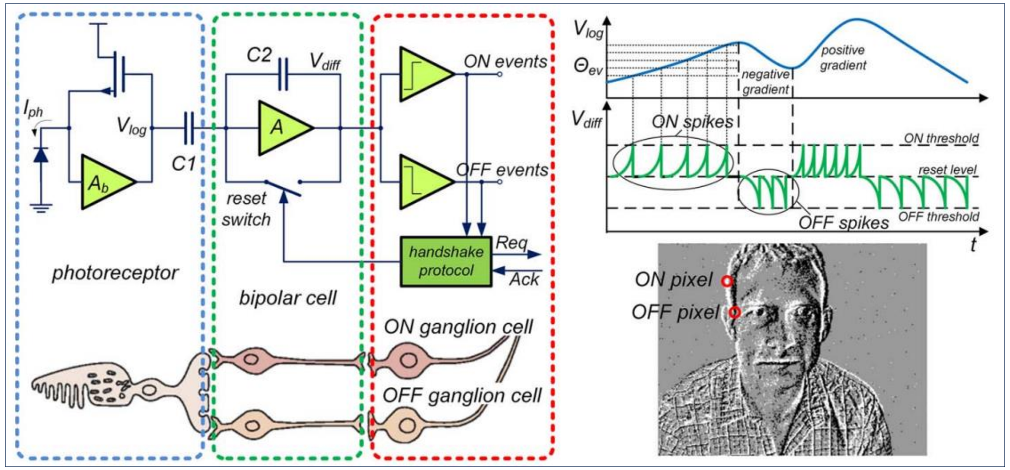
Imaging principle: An event camera **detects each pixel independently and asynchronously**. When the brightness change reaches a threshold, it outputs an event containing the **triggering time, pixel location, and brightness change polarity (ON or OFF).**

$$e_i = (t_i, x_i, y_i, p_i) \qquad\qquad I(x, y, t_k)$$

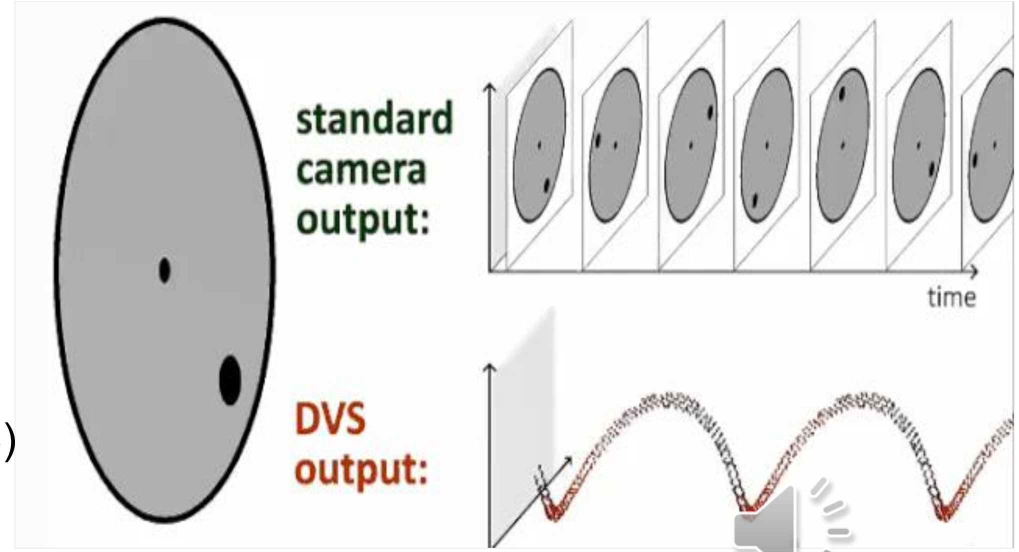
Sparse asynchronous data stream;
rich spatiotemporal information

Dense synchronous image;
color and texture information

- Advantages: high temporal resolution(μs), high dynamic range(140dB)
- Challenge: sparse asynchronous data format



event-camera circuits

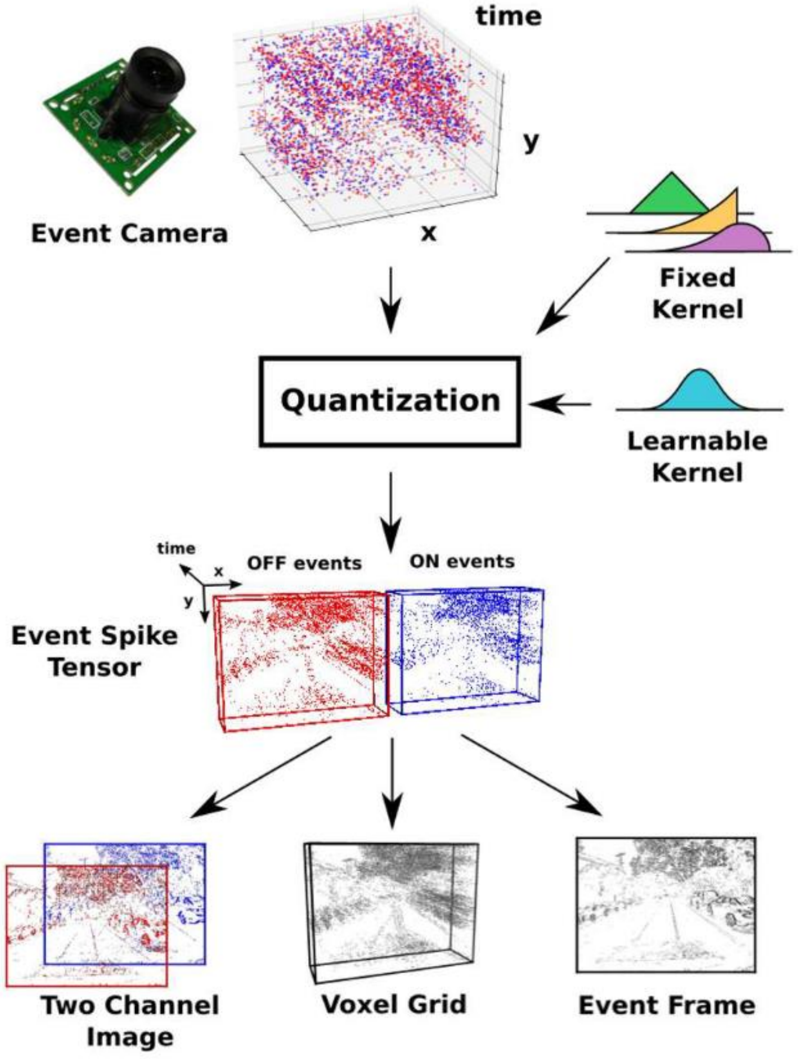


Comparison between event cameras and conventional cameras

[1] Event-based vision: A survey. TPAMI, 2020, 44(1): 154-180.
[2] High speed and high dynamic range video with an event camera. TPAMI, 2019, 43(6): 1964-1980.

Background and Motivation

- **Event-to-frame Conversion**



Event-to-frame conversion transforms **sparse asynchronous event streams** into **dense event frames**. This process involves two steps: **partitioning** and **representation**, which can be formally described as spatiotemporal integration.
(1) Partitioning: A continuous event stream is divided into discrete event packets. Common strategies include fixed temporal-window partitioning and fixed event-count partitioning.
(2) Representation: The events within each packet are transformed into a two-dimensional image sequence or a three-dimensional voxel sequence. Common representations include time surfaces, event counts, and voxel grids.

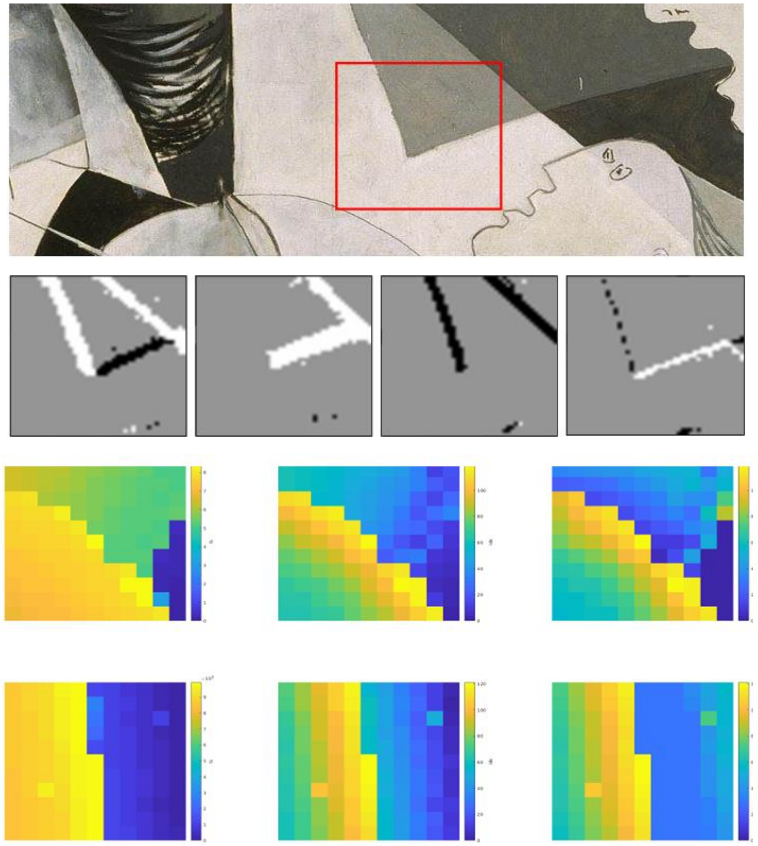
$$F(x,y)=\sum_{\underbrace{\boxed{e_k\in\mathcal{W}}}_{\text{integration interval}}}\underbrace{\boxed{\phi(x_k,y_k,t_k)}}_{\text{integration kernel}}\cdot\delta(x-x_k,y-y_k)$$

Partitioning　　　Representation

- **Heterogeneous velocity scenarios (HVS):** Object motion velocities vary across both **time** and **space**. Such scenarios are common in sports and frequently appear in traffic and athletic scenes.
- **Existing problem:** Event cameras are highly sensitive to motion velocity. Events generated by the same target under different motion speeds can differ substantially in quantity, leading to large **representational appearance variations** and **degrading downstream task performance**.

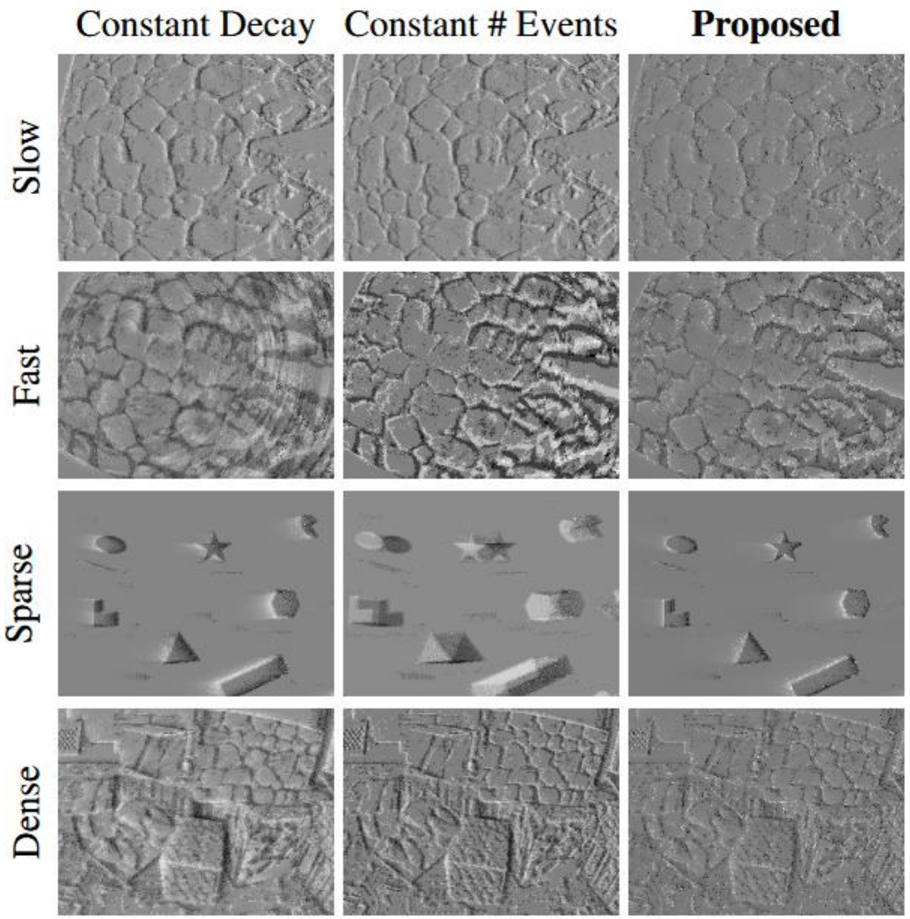
Related Work

- **Existing Representation Strategies**



Standard Time Surface | Normalization method of [3] | Speed Invariant Time Surface

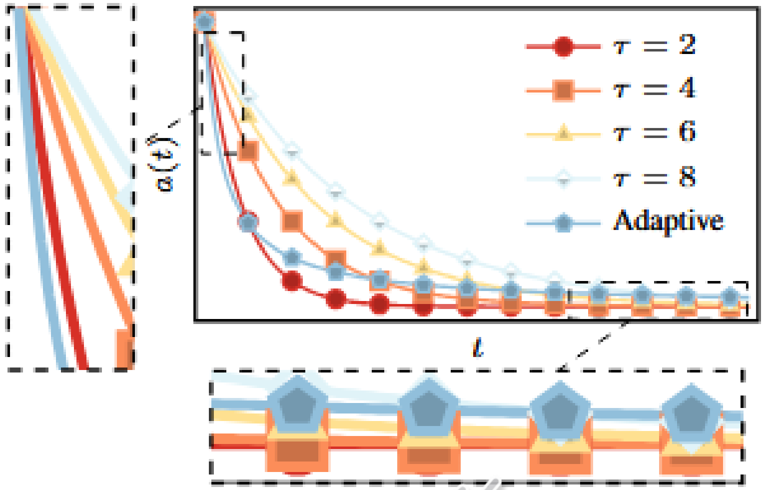
SITS
CVPR 2019: 10245-10254.



Algorithm 1 Event-based Adaptive Global Decay Process

Input: Event stream $\{e_i\}$, event activity a_0 and timestamp t_0.
Output: Event activity a_i, adaptive decay $\beta_{a,i}$.
Procedure:

1: **for** each event e_i **do**
2: Set $\beta_{a,i} \leftarrow \frac{1}{1+a_{i-1}(t_i - t_{i-1})}$, Eq. (15).
3: Set $a_i \leftarrow \beta_{a,i} a_{i-1} + 1$, Eq. (10).
4: **end for**

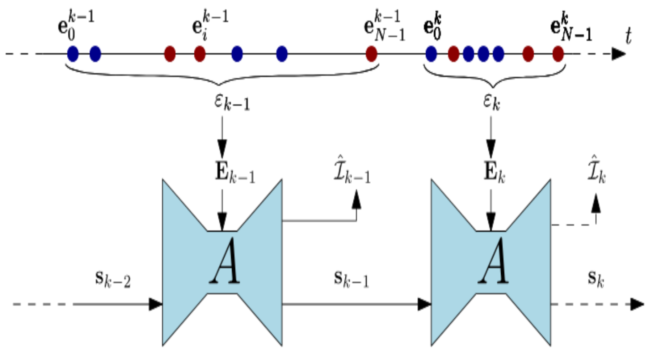


Adaptive Global Decay
CVPR 2023: 9771-9780.

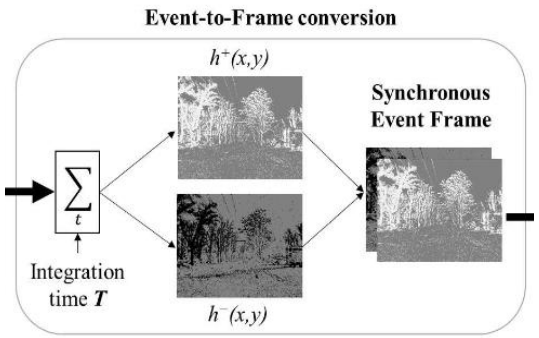
Limitation: Accumulation latency; ineffective handling of HVS

Related Work

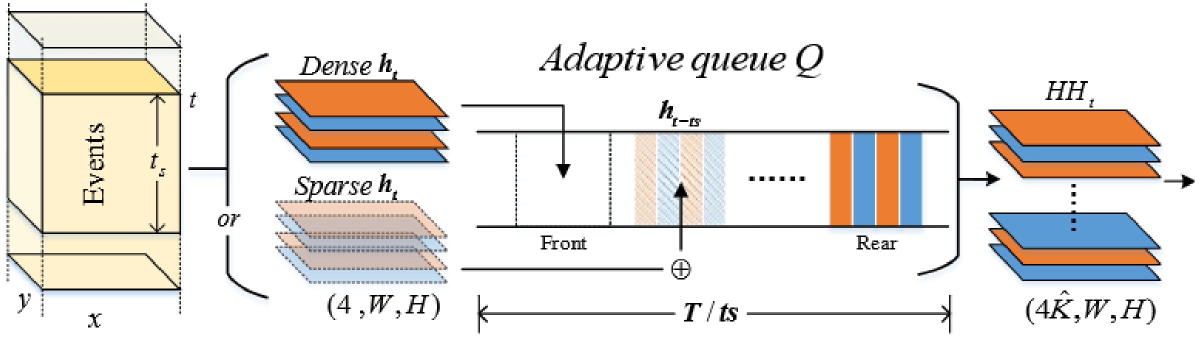
- **Existing Partitioning Strategies**



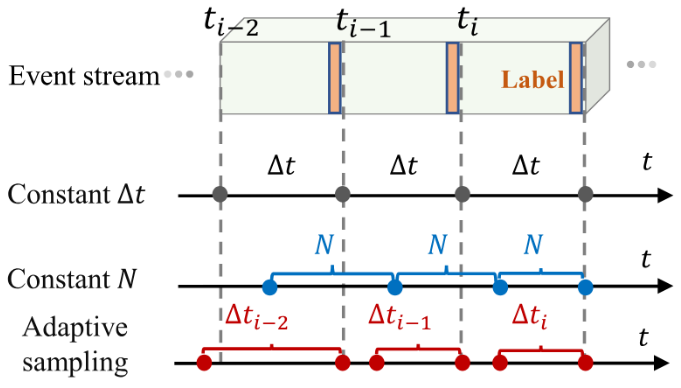
Fixed Number of Events
TPAMI, 2019, 43(6): 1964-1980.



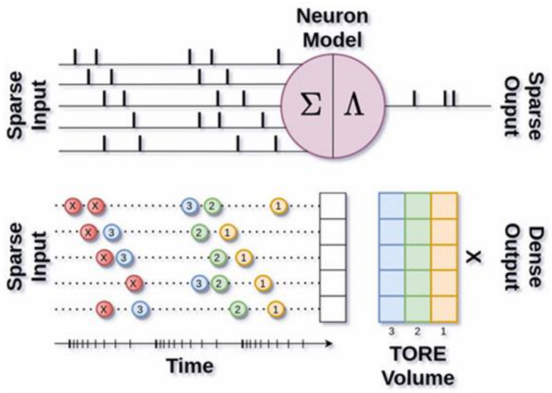
Fixed Time Window
CVPR, 2018: 5419-5427.



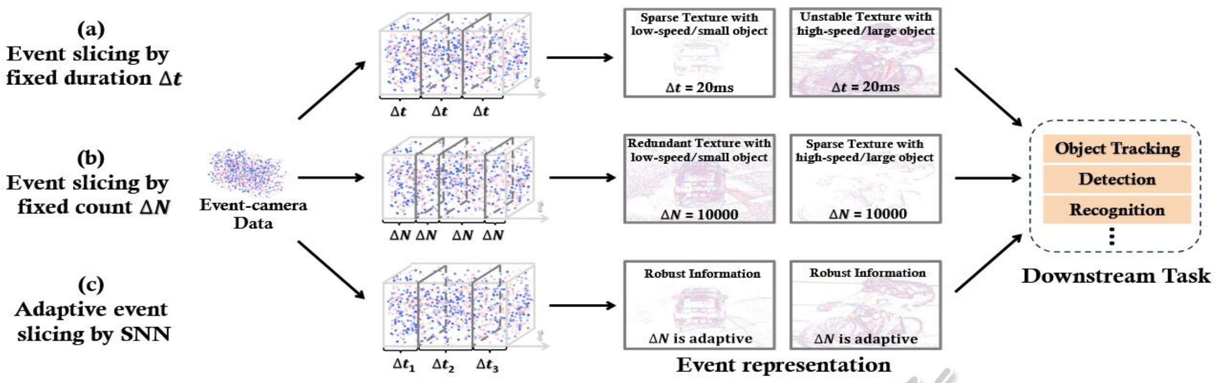
Adaptive Event Conversion
AAAI, 2023, 37(2): 2056-2064.



Adaptive Temporal Sampling
TIP, 2022, 31: 2975-2987.



TORE
TPAMI, 2022, 45(2): 2519-2532.



SpikeSlicer
NeurIPS, 2024, 37: 75064-75094.

Limitation: Difficulty in balancing temporal adaptivity and spatial locality

Related Work

• Summary of Existing Partitioning Strategies

To fundamentally address **event-to-frame conversion in HVS**, it is essential to achieve **motion-adaptive event partitioning** (adaptive “exposure”). However, existing partitioning strategies often struggle to balance **temporal adaptivity** and **spatial locality**, and may introduce substantial **computational overhead and latency**.

Comparison of different partitioning strategies

Partitioning Strategy	Temporal Adaptivity	Spatial Locality
Fixed Time Window	✗	✗
Fixed Number of Events	✗	✗
ATSLTD	✓	✗
Adaptive Temporal Sampling	✓	✗
Adaptive Event Conversion	✓	✗
Adaptive Global Decay	✓	✗
SpikeSlicer	✓	✗
TORE	✗	✓
Event Lifetime	✓	✓
ASTW (Ours)	✓	✓

Adaptive Spatial-Temporal Window

- Existing **partitioning strategies** do not fully account for the **spatiotemporal heterogeneity** in motion velocity.

Visualization of Heterogeneous Velocity Scenarios

Temporal Heterogeneity

Temporal variations in target velocity

Temporal Adaptivity

Spatial Heterogeneity

Velocity differences across spatial locations

Spatial Locality

Adaptive Spatial-Temporal Window

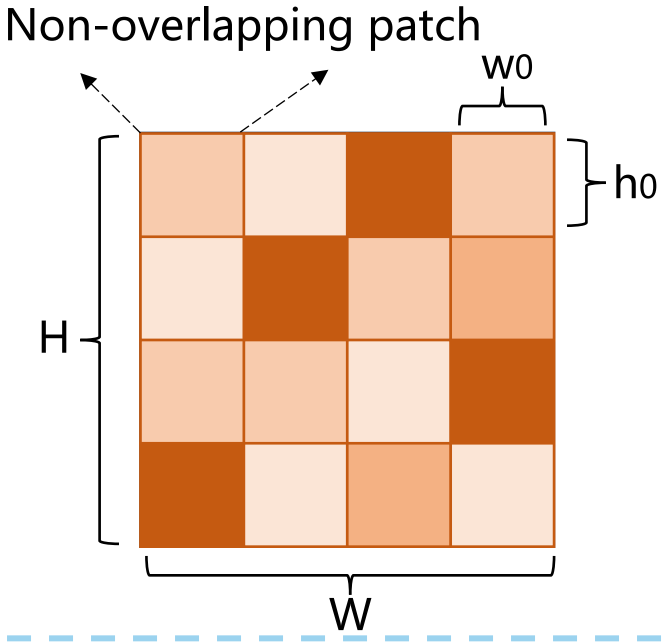
1. How to achieve **spatially differential** processing?
2. How to determine the **temporal window**?
3. How to enable **efficient computation**?

Key Technical Challenges

Adaptive Spatial-Temporal Window

◆ **Research Idea:** We propose an **efficient partitioning** strategy that accounts for **spatial locality** and **temporal adaptivity**.

Dividing the image plane into **spatial patches**



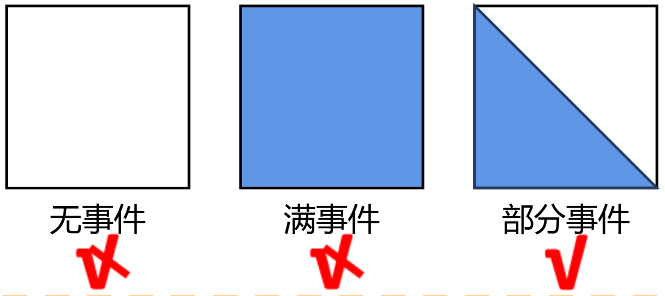
Determining suitable time windows via **information maximization**

1. **Information entropy** within each spatial block

$$H_{ij} = -p_{ij} \cdot \log_2 p_{ij} - (1 - p_{ij}) \cdot \log_2 (1 - p_{ij})$$

2. **Local information maximization and spatial-consistency constraint**

$p_{ij} = 1/2$ -> proportion of non-zero pixels



Efficient computation based on **statistics** and **vectorization**

1. **Event density statistics**

(1) Define the event density:

$$N(\Delta t) = c \cdot L \cdot v \cdot \Delta t$$

$$D_{ij} = \frac{N_{ij}}{\Delta t_{\mathrm{ref}} \cdot s^2} = \frac{c \cdot L_{ij} \cdot v_{ij}}{s^2}$$

(2) Local time window expression:

$$\frac{c \cdot L_{ij} \cdot v_{ij} \cdot \Delta t_{ij}}{s^2} = c \cdot \frac{1}{2} = \gamma$$

$$D_{ij} \cdot \Delta t_{ij} = \gamma$$

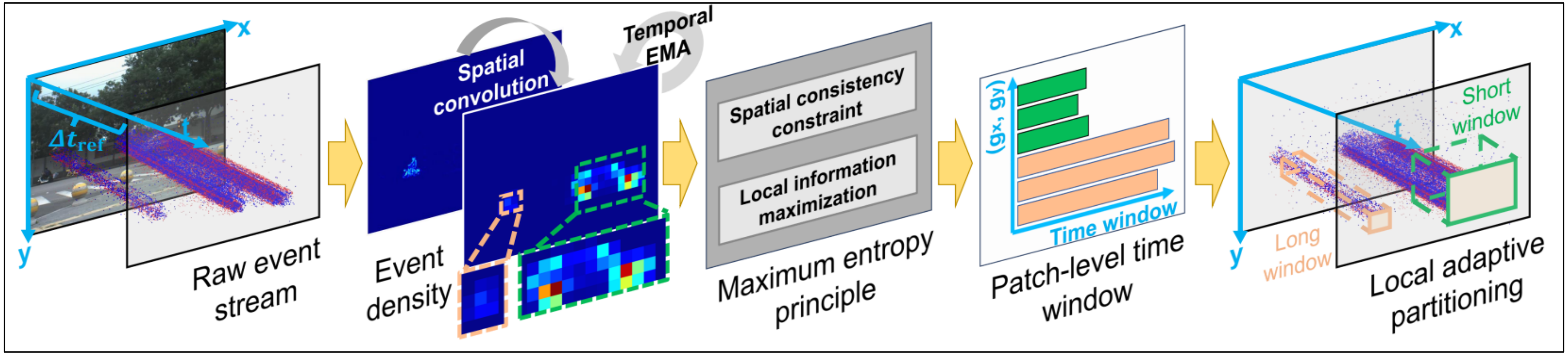
2. **Vectorized computation**

Enabling $\mathcal{O}(N)$ –complexity partitioning.

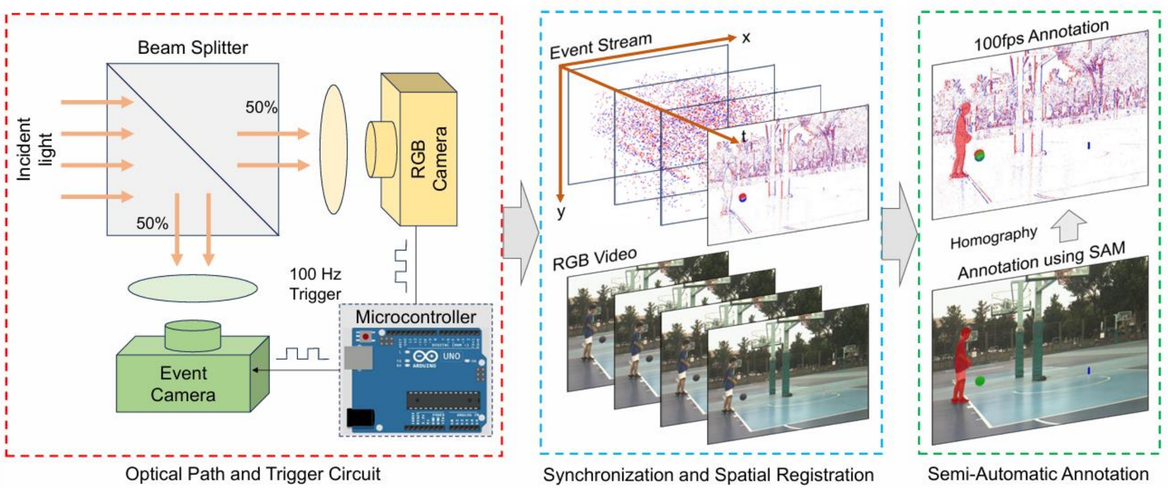
Improving robustness to spatial scale and velocity variations through **spatio-temporal smoothing**

Adaptive Spatial-Temporal Window

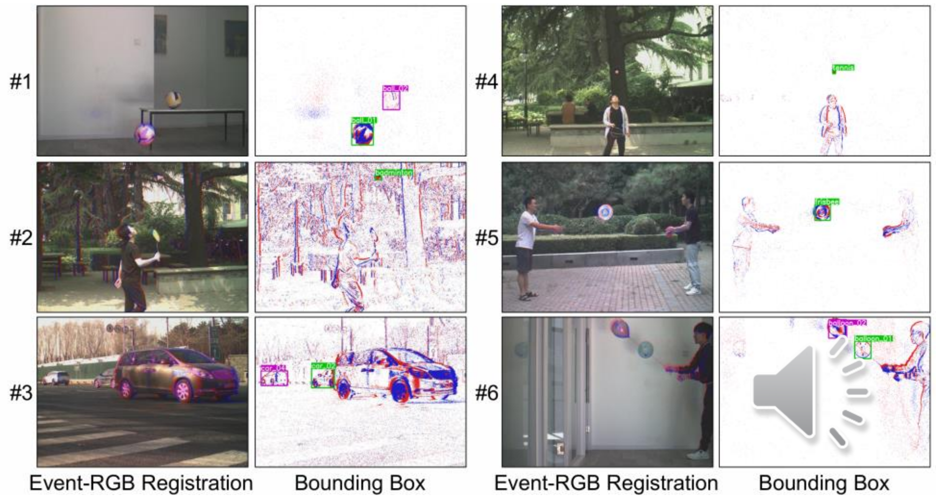
◆ ASTW Workflow



Workflow of ASTW strategy



Acquisition workflow of the HetVel dataset



Visualization of the HetVel dataset

Adaptive Spatial-Temporal Window

◆ Algorithm Implementation and Vectorization Acceleration

Algorithm Implementation

- Compute the event density for each local region
- Compute the local adaptive time window
- Efficiently partition events in a vectorized manner

Vectorization Acceleration

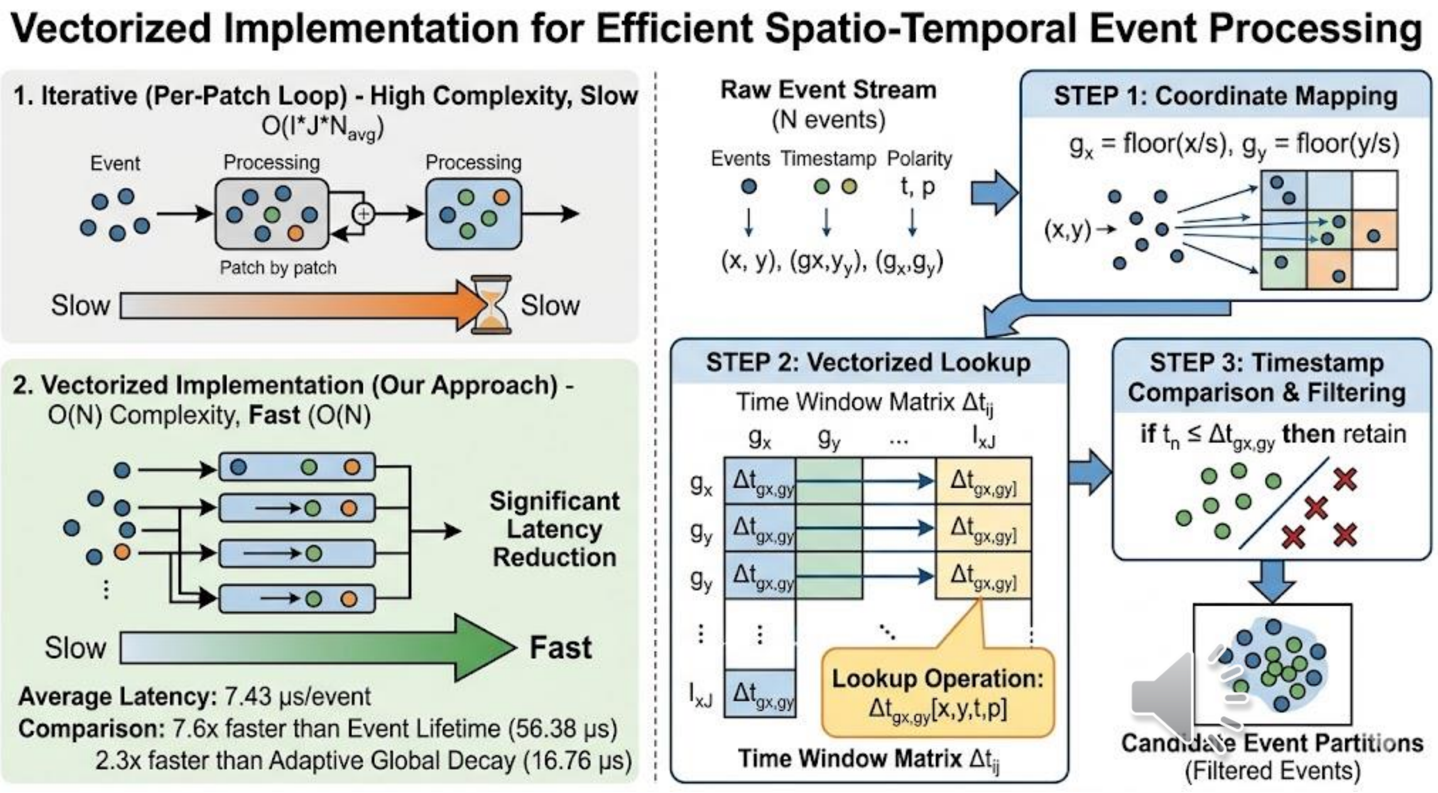
- Coordinate mapping: Map event pixel coordinates to spatial-block indices
- Windowed search: Search for events within the candidate temporal window
- Temporal comparison: Retain only events within the current temporal window

Algorithm 1 ASTW Strategy

Input: Event stream $\mathcal{E}$ ($e_k \in \mathcal{E}, k = 1, 2, \ldots, N$), global timestamp T_{global}, list of labels $\mathcal{L}$ ($L_m \in \mathcal{L}, m = 1, 2, \ldots, M$)

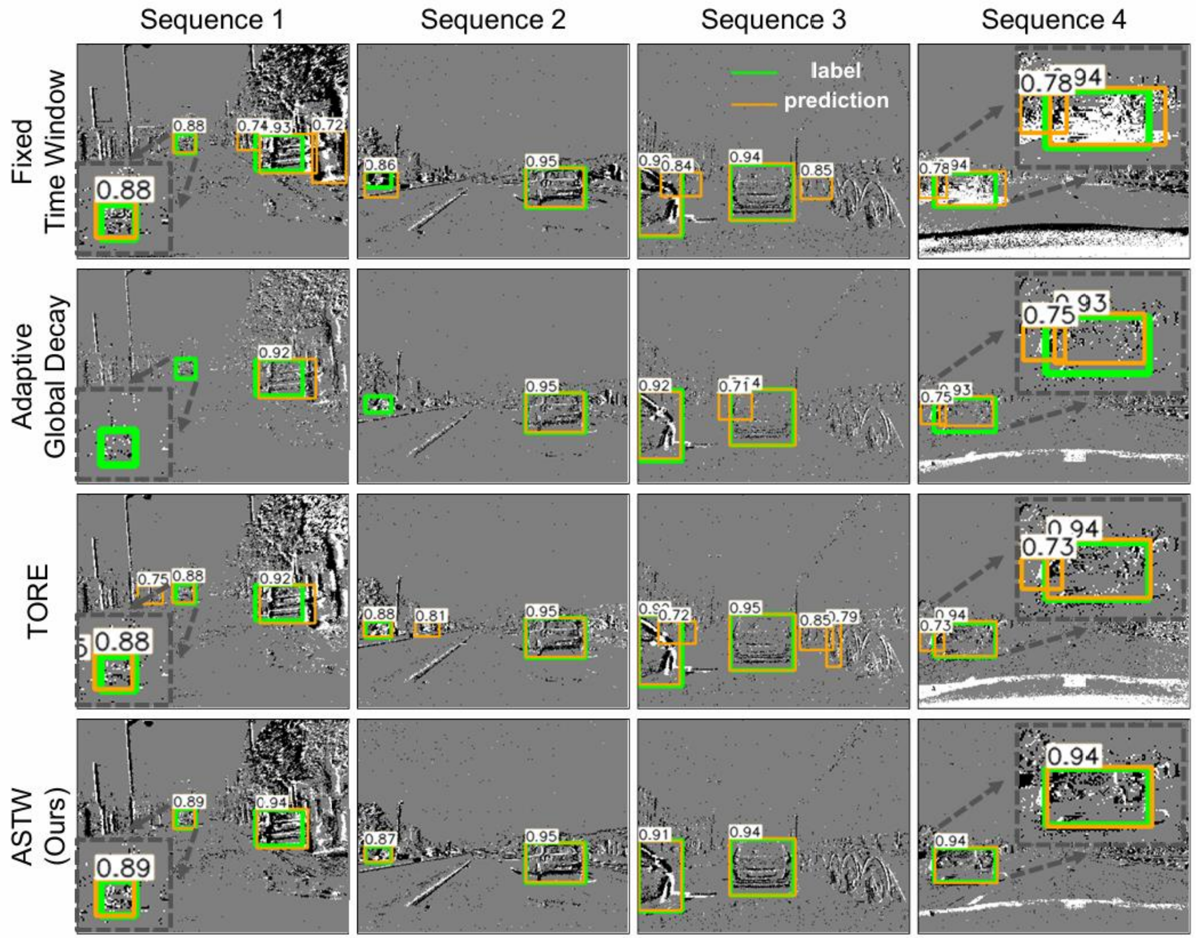
Output: List of partitioned events $\mathcal{P}$ ($P_m \in \mathcal{P}, m = 1, 2, \ldots, M$)

```
Set reference time window Δt_ref and patch size (s, s):
while T_global < e_N(t) or not EOF do
    Calculate D_ij within [T_global − Δt_ref, T_global), Eq. 4
    Calculate D̃_ij and D̄_ij, Eq. 5
    Calculate time window Δt_ij for each patch, Eq. 6
    Extract candidate partition P_cand from ℰ using Δt_ij
    Choose minimum time window and update T_global:
        Δt_step = min(Δt_ij), T_global ← T_global + Δt_step
    if T_global ≤ L_{m+1}(t) and T_global > L_m(t) then
        P_m ← P_cand, m ← m + 1
        Save partitioned events P_m and label L_m
    end if
end while
```

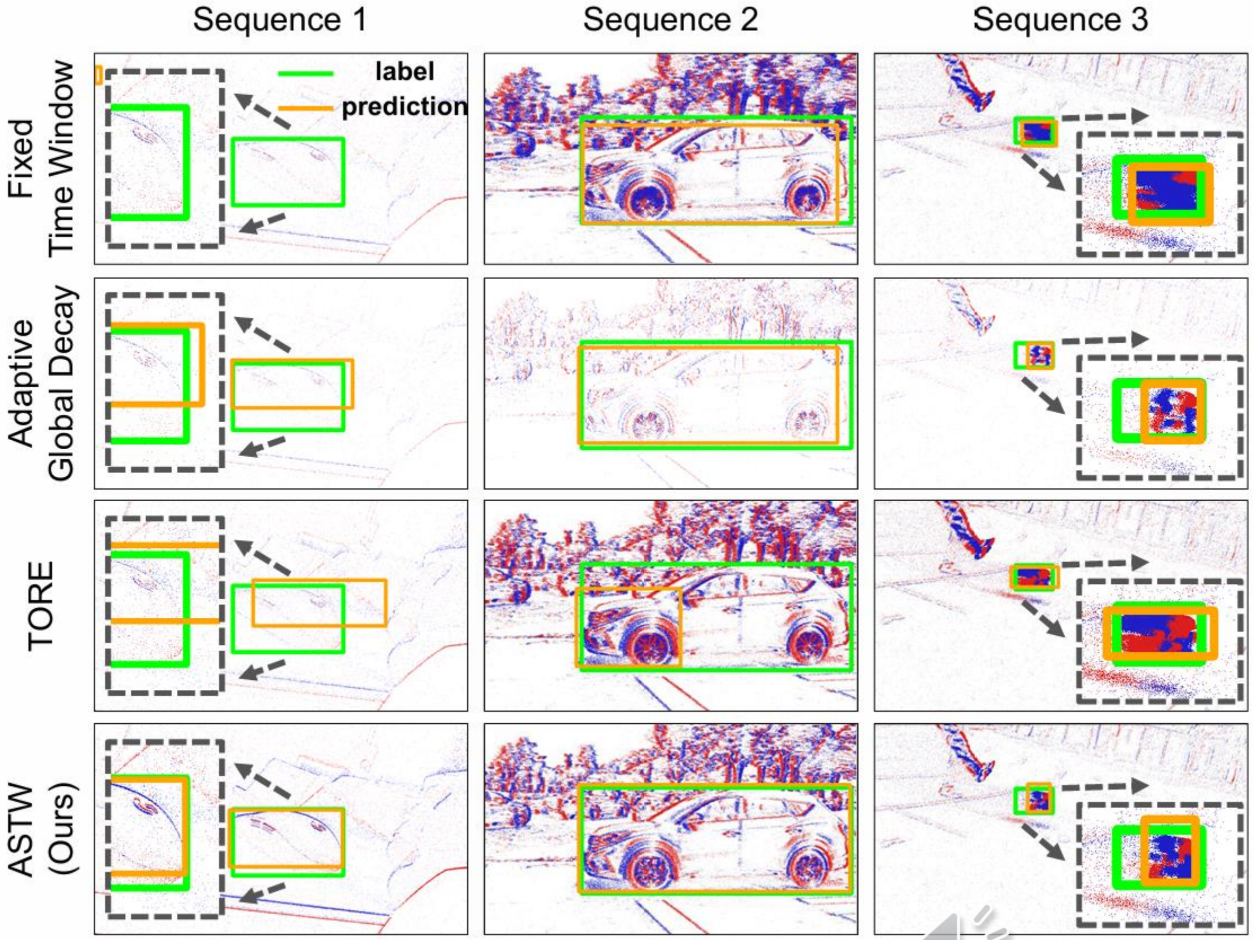


Experiments and Results

◆ **Validation on Object Detection and Tracking Tasks**



Object detection results (Gen1 dataset)



Object Tracking Results (EventVOT dataset)

- ASTW effectively suppresses **motion blur** and outperforms other partitioning methods in both **visualization quality** and **task performance**.

Experiments and Results

◆ Performance Comparison between Our Method and Other Methods

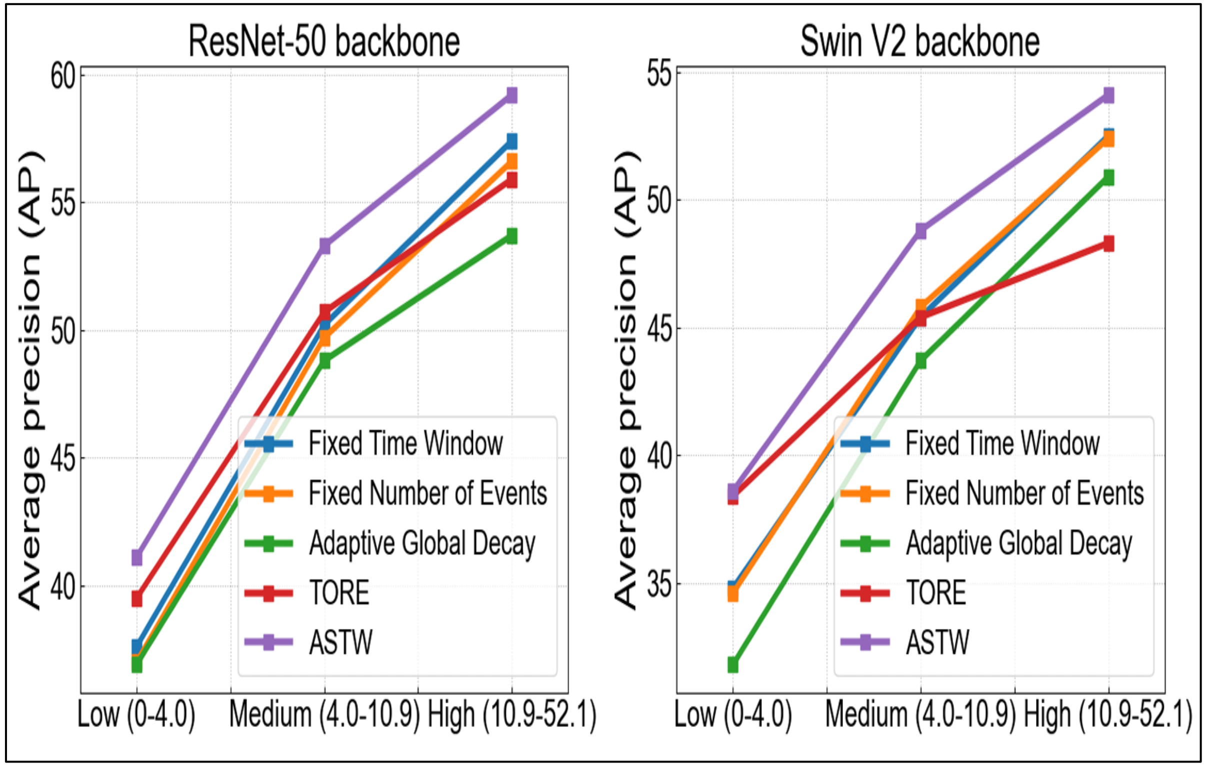
Task performance comparison of different partitioning strategies

Partitioning Strategy	Object Detection						Object Tracking					
	ResNet-50		Swin V2		RVTs		EventVOT Dataset			HetVel Dataset		
	mAP	AP_{50}	mAP	AP_{50}	mAP	AP_{50}	SR	PR	NPR	SR	PR	NPR
Fixed Time Window [21]	47.9	74.5	43.9	73.2	46.9	73.4	61.7	60.2	84.3	47.7	56.1	81.6
Fixed Number of Events [26]	47.5	74.5	44.0	73.1	46.2	72.4	60.1	56.7	82.4	48.1	55.9	82.2
Adaptive Temporal Sampling [17]	47.4	74.3	44.0	72.6	45.9	72.1	60.8	57.5	83.5	48.2	55.7	82.4
Adaptive Event Conversion [24]	45.0	71.1	41.1	69.7	43.6	69.2	58.0	52.2	80.3	44.9	51.9	78.6
Adaptive Global Decay [23]	46.0	73.9	41.6	69.9	47.1	73.6	60.8	57.7	83.5	48.4	56.1	82.2
SpikeSlicer [6]	46.9	73.4	42.9	72.8	45.6	71.5	61.9	60.5	84.4	48.4	56.0	82.4
TORE [2]	48.3	75.2	43.6	73.1	47.3	73.8	62.1	58.7	84.8	48.6	56.3	82.7
Event Lifetime [22]	45.2	72.0	40.8	72.1	44.7	69.8	58.7	53.4	81.1	46.0	53.4	79.8
ASTW (ours)	**50.6**	**76.0**	**46.6**	**75.2**	**48.3**	**75.4**	**62.3**	**60.5**	**85.1**	**50.7**	**58.5**	**84.9**

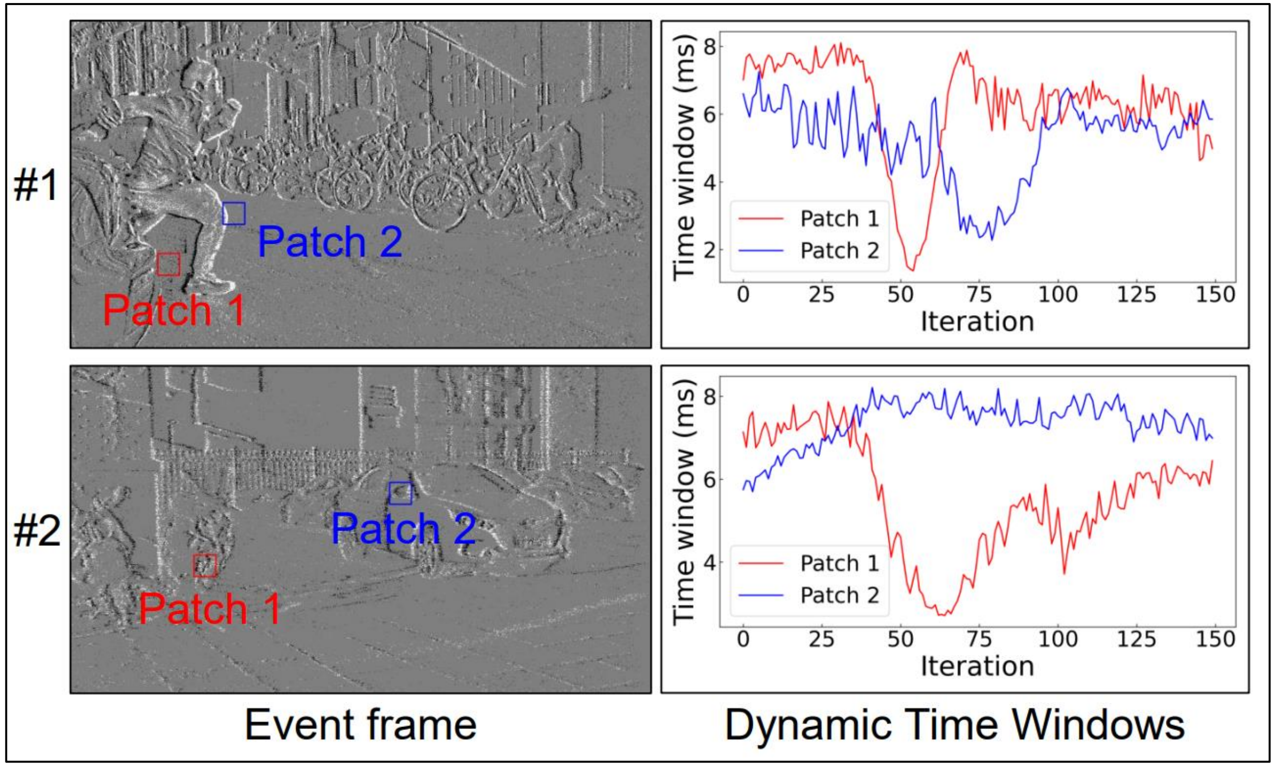
- The method is validated on different network architectures, including **feedforward and recurrent** networks, as well as backbone networks based on **CNNs and Transformers**.
- ASTW achieves the best overall performance. Compared with the second-best results, ASTW improves object detection by **2.6 mAP** and **2.0 AP_{50}** , and improves object tracking by **0.2 SR** and **0.3 NPR**, while achieving **comparable PR**.

Experiments and Results

◆ Detection Performance and Case Study



Detection performance of different methods under different motion velocities



Sudden pedestrian appearances in traffic scenarios and rapid saccadic motion in robot scenarios

- ASTW strategy demonstrates superior task performance across **different motion velocities**.
- The visualizations intuitively validate the **spatial locality** and **temporal adaptivity**.
- ASTW enables **800-1000 partitions** per second and effectively tracks moving targets with speeds up to **2400 pixels/s**.

Experiments and Results

◆ Ablation Study and Parameter Sensitivity Analysis

Component analysis of ASTW strategy

Patch Division	Spatial Smoothing	Temporal Smoothing	Causal Consistency	mAP
✗	✓	✓	NA	46.6
✓	✗	✓	✓	50.1
✓	✓	✗	✓	50.3
✓	✓	✓	✗	49.4
✓	✓	✓	✓	**50.6**

Sensitivity of the ASTW strategy to (Δt_{min}, Δt_{max})

$(\Delta t_{\min}, \Delta t_{\max})$	(2.5, 250)	(5, 250)	(10, 250)	(25, 150)
mAP	50.0	50.1	**50.6**	49.5
$(\Delta t_{\min}, \Delta t_{\max})$	(25, 200)	(50, 150)	(50, 200)	(75, 150)
mAP	49.6	47.8	48.7	48.8

Experiments on different event representations

Representation	Time Surface [30]	Event Count [23]	Voxel Grid [38]
Baseline	51.3	50.1	47.6
ASTW	**53.0**	**52.1**	**49.0**

Sensitivity of ASTW strategy to patch size

Patch Size	1	2	4	8	16	32	64
mAP	49.9	50.4	**50.6**	50.3	50.0	49.3	48.9

Sensitivity of ASTW strategy to information constant

Information Constant	1.0	1.3	1.5	1.7	2.0
mAP	50.4	50.5	50.3	**50.6**	50.0

Sensitivity of ASTW strategy to reference time window

Reference Time Window	50	100	250	500	750
mAP	49.4	50.1	**50.6**	**50.6**	50.0

- The ablation study demonstrates the effectiveness of each component and further verifies the importance of **spatial locality**.
- ASTW can be integrated with different representations to improve **task performance**.

Thank you for your attention!