# Better and Faster: Adaptive Event Conversion for Event-Based Object Detection

**Yansong Peng, Yueyi Zhang*, Peilin Xiao, Xiaoyan Sun, Feng Wu**

University of Science and Technology of China, Hefei, China, 230026
{pengyansong, peilinxiao}@mail.ustc.edu.cn, {zhyuey, sunxiaoyan, fengwu}@ustc.edu.cn

## Abstract

Event cameras are a kind of bio-inspired imaging sensor, which asynchronously collect sparse event streams with many advantages. In this paper, we focus on building better and faster event-based object detectors. To this end, we first propose a computationally efficient event representation Hyper Histogram, which adequately preserves both the polarity and temporal information of events. Then we devise an Adaptive Event Conversion module, which converts events into Hyper Histograms according to event density via an adaptive queue. Moreover, we introduce a novel event-based augmentation method Shadow Mosaic, which significantly improves the event sample diversity and enhances the generalization ability of detection models. We equip our proposed modules on three representative object detection models: YOLOv5, Deformable-DETR, and RetinaNet. Experimental results on three event-based detection datasets (1Mpx, Gen1, and MVSEC-NIGHTL21) demonstrate that our proposed approach outperforms other state-of-the-art methods by a large margin, while achieving a much faster running speed ($< 14\ ms$ and $< 4\ ms$ for $50\ ms$ event data on the 1Mpx and Gen1 datasets).

## Introduction

The event camera is a bio-inspired sensor to capture dynamic visual information, which brings a new paradigm to the sensing technology. Compared with traditional frame cameras, event cameras offer high temporal resolution (>10K fps) and high dynamic range (>120 dB), while maintaining low power consumption (<10 mW) (Gallego, Delbrück, and Orchard 2022). Object detection, which outputs the bounding box and classification for objects, is a field that may benefit from the advantages, especially in the scenes with low/high illumination and high moving speed.

After the birth of event cameras, there have been many attempts at event-based object detection (Liang et al. 2021; Messikommer et al. 2022; Li et al. 2022; Cordone et al. 2022; Bi et al. 2020). They can be roughly classified into two main categories according to the input representation. The first category attempts to detect objects directly from asynchronous events. For example, SNN-based methods (Cordone et al. 2022) and GNN-based methods (Schaefer et al. 2022), which directly operate the raw events, have been proposed. The second category of event-based object detection methods converts asynchronous events into image-like representations, such as event histogram (Moeys et al. 2016), time surface (Lagorce et al. 2017) and event volume (Hu et al. 2020). These representations are fed to frame-based object detectors.

Given a series of excellent frame-based object detectors have been proposed and achieved great success, in this paper, we attempt to exploit the potential of these frame-based object detectors and extend them to event-based data. In other words, we seek to achieve event-based object detection following the pipeline of the second category mentioned before. The frame-based object detectors originally cannot deal with event data. Thus, an efficient and effective conversion method is needed to transform the event data to event frames. A number of methods have been proposed for event conversion, but they either sacrifice performance or are very time-consuming, not suitable for efficient end-to-end event-based object detection.

To solve this problem, we first propose an event representation Hyper Histogram, which adequately preserves both the polarity and fine-grained temporal information of events. We also provide a fast implementation to convert events to Hyper Histograms. Then, we consider that a suitable input density is beneficial to event-based object detection. Thus, we devise an efficient Adaptive Event Conversion (AEC) module, which converts events into a Hyper Histogram according to the event density via an adaptive queue.

Similar to frame-based object detectors, data augmentation plays an important role in event-based object detection. However, the augmentation for event data is rarely explored. We propose a novel event-based data augmentation method Shadow Mosaic that can directly handle the raw asynchronous event streams. It can significantly increase the sample diversity by randomly changing the scale and density of the input event streams.

Our proposed methods can be easily combined with frame-based detectors. We test three mainstream object detection networks YOLOv5 (Jocher 2021), Deformable-DETR (Zhu et al. 2021) and RetinaNet (Lin et al. 2020). Experimental results on both the 1Mpx (Perot et al. 2020) and Gen1 (de Tournemire et al. 2020) datasets demonstrate that our proposed method significantly outperforms the current

*Corresponding author: Yueyi Zhang

state-of-the-art methods. We also directly test the trained model on the Nighttime Driving Detection dataset MVSEC-NIGHTL21 (Hu, Liu, and Delbruck 2021). The results indicate that the generalization ability of our model is superior.

The contributions are summarized in the following.

- We propose a computationally efficient event representation Hyper Histogram, which encodes both the polarity and fine-grained temporal information for events.
- We design an Adaptive Event Conversion module to convert raw events to Hyper Histograms according to event density, which improves the detection performance and reduces the runtime.
- We introduce a novel event-based augmentation method Shadow Mosaic, which enhances the generalization ability of detectors for event-based object detection.
- With above mentioned methods, we extend frame-based object detectors to events. Experimental results demonstrate superior detection performance on three datasets. The running time is also greatly reduced ($< 14\ ms$ and $< 4\ ms$ for 50 $ms$ event data on the 1Mpx and Gen1 datasets).

## Related Work

### Event Conversion

Due to the asynchronous nature of event data, converting events into a good spatio-temporal representation is fundamental to an event-based computer vision task.

Event histogram (Maqueda et al. 2018), time surface (Lagorce et al. 2017) and event volume (Zhu et al. 2019) are commonly used image-like event representations with the shape $(2, W, H)$. They have been applied for a variety of event-based computer vision tasks. However, all these representations discard partial information of events, which leads to performance degradation. To retain more information of the event, inceptive time surface (Baldwin et al. 2019) with shape $(3, W, H)$ draws averaged time surface and event histogram together, but introduces more latency when generating them independently. There are also representations with shape $(B, W, H)$ like voxel grid (Zhu et al. 2019), which accumulates events within $B$ temporal windows. However, these sparse representations often suffer from poor convergence. There are also works that have been accomplished in designing learned representations via DNNs (Cannici et al. 2020; Messikommer et al. 2020), SNNs (Shrestha and Orchard 2018; Lee et al. 2020; Baldwin et al. 2022) or GNNs (Bi et al. 2020; Schaefer et al. 2022).

The representations for DNNs fail to retain adequate information while maintaining high conversion efficiency. The representations learned through DNNs bring too much latency. The asynchronous representations based on SNNs or GNNs cannot achieve comparable performance.

### Event-Based Object Detection

The early attempts of event-based object detection convert events into frames and send them to frame-based detectors (Lagorce et al. 2015; Cannici et al. 2019), which demonstrated the feasibility of event-based object detection. Integrating events into image-like planes (Lagorce et al. 2017; Hu et al. 2020) retains more spatio-temporal information. But due to the lack of large-scale event-based datasets, they have not yet utilized rich spatio-temporal information of events.

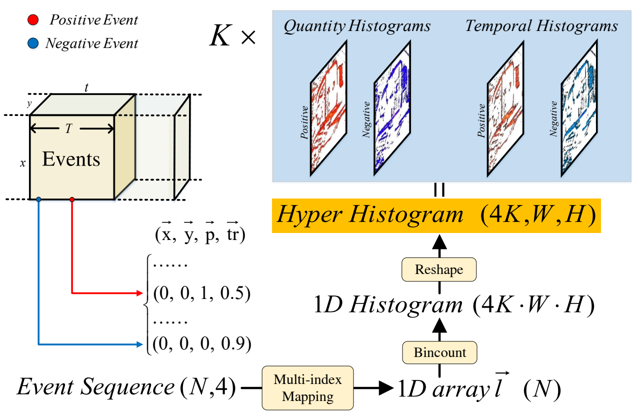


Figure 1: Hyper Histogram generation process. The $4K$ histograms are generated simultaneously by forming one 1D histogram with length $4K \cdot H \cdot W$ using the multi-index mapping method. A Hyper Histogram is equivalent to $K$ groups of quantity histograms and temporal histograms.

The benchmark real-world datasets of event-based object detection are the 1Mpx Detection dataset (Perot et al. 2020) and the Gen1 Detection dataset (de Tournemire et al. 2020). Several interesting networks have been proposed for these two datasets, such as (Perot et al. 2020; Liang et al. 2021; Li et al. 2022). These works modify specific baseline networks like SSD and RetinaNet by adding attention modules or designing memory mechanisms to improve the performance. However, almost all the reported works cannot achieve real-time object detection with high performance on both two datasets at a detection speed over the labeling frequency of 60 Hz.

More recently, novel SNN-based methods such as (Cordone et al. 2022) utilize the surrogate gradient learning method to detect objects from encoded voxel cube events. GNN-based methods like (Schaefer et al. 2022) generalize GNNs to process events as spatio-temporal graphs. However, their performance is still far behind that of DNN-based methods.

## Method

### Hyper Histogram

First of all, we describe a novel event representation Hyper Histogram, which is a 3D histogram with the dimension $(4K, W, H)$, where $K$ is the number of groups splitting the time window, $W$ and $H$ are the width and the height. The Hyper Histogram is a collection of the quantity and temporal histograms with different polarities, the generation of which is described below.

Traditional quantity event histogram records the event counts for each pixel within a time window (Maqueda et al. 2018). Taking the polarity $p_k$ into account, an event his-

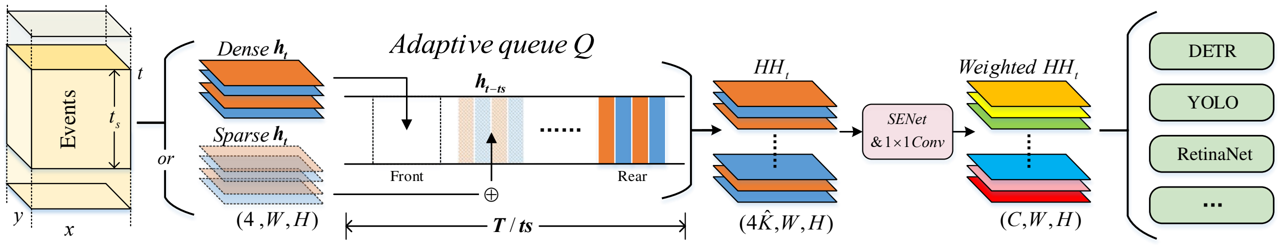


Figure 2: The flow chart of AEC module. The computed building blocks $\mathbf{h}_t$ within time $T$ is cached in an adaptive queue $Q$. The queue is adaptively updated at each timestamp based on the density of the incoming event stream within a short time $ts$. The queue ends up outputting the **dense** $HH$ with $4\hat{K}$ channels, where $\hat{K}$ is an adaptively changed value. Then, $HH$ is sent to an SENet block and an $1 \times 1$ convolution layer to get weighted $HH$ with unified channel number $C$. The weighted $HH$ is directly fed to frame-based object detectors.

togram can be divided into a positive quantity histogram and a negative quantity histogram. Taking the positive quantity histogram as an example, it can be described as

$$h^+(x, y) = \sum_{t_k \in T, p_k = 1} \delta(x - x_k, y - y_k). \quad (1)$$

where $\delta()$ denotes the two-dimensional impulse function.

Although quantity event histograms record the event distribution for pixels, they lose the temporal information. Thus, we additionally construct temporal histograms to utilize the chronological order. The temporal histogram for positive events is described as

$$h_t^+(x, y) = \sum_{t_k \in T, p_k = 1} \delta(x - x_k, y - y_k) \cdot \frac{t_k - t_0}{T} \quad (2)$$

where $t_0$ is the start time of the current time window. The negative histograms are generated similarly.

As shown in Fig. 1, Hyper Histogram is a matrix containing quantity and temporal histograms with different polarities. Positive and negative quantity histograms record the coordinate distribution and temporal histograms record the temporal distribution. The combination of four histograms is defined as $\mathbf{h}$, which is treated as the building block of Hyper Histogram.

It should be noted that the relative time only counts within a short period. A single building block of Hyper Histogram cannot describe the long-range temporal information. To solve this issue, we propose the Hyper Histogram with $K$ building blocks, which consists of $K$ groups of $\mathbf{h}$. The finer temporal subdivision of Hyper Histogram preserves more fine-grained information of events. This is especially important in the case of high-rate detection.

Another obstacle is that the channel-by-channel calculation of 2D histograms for $4K$ times is very time-consuming. In order to achieve efficient generation, we further provide a fast implementation. Specifically, we first cache a multi-index mapping matrix from 4D to 1D and multiply each event sequence with it to form a 1D array. After the Bincount and Reshape operations, the final three-dimensional Hyper Histogram with shape $(4K, W, H)$ is generated. Fig. 1 shows the generation process of Hyper Histograms.

Algorithm 1: Hyper Histogram Generation in AEC Module

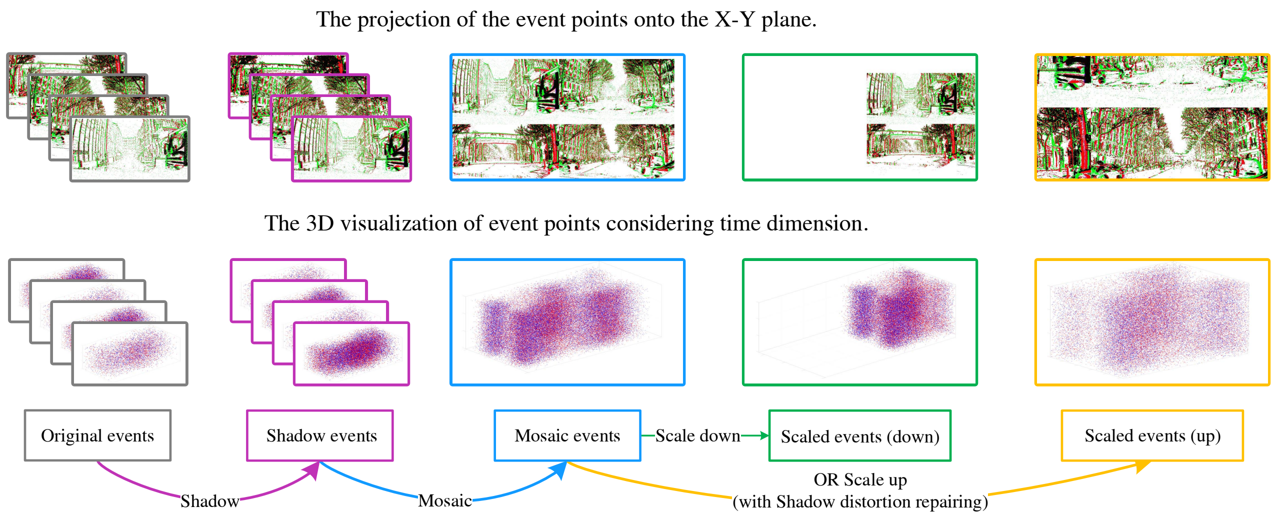
```
Input: queue $Q$, event stream $\overrightarrow{\mathbf{e}_t}$, length threshold $L$
Output: Hyper Histogram $HH_t$
1:  $L_s = 0$   ▷ Initialize $L_s$, denotes the total length of the
    sparse event streams during accumulation
2:  while $\overrightarrow{\mathbf{e}_t}$ is not empty do
3:      generate building block of $\overrightarrow{\mathbf{e}_t}$: $\mathbf{h}_t$
4:      $L_t$ = length($\overrightarrow{\mathbf{e}_t}$).
5:    ▷ Incoming sparse stream & Previous stream is dense
6:      if $L_t < L$ and $L_s = 0$ then
7:          Push $\mathbf{h}_t$ at the front of $Q$
8:          $L_s += L_t$
9:    ▷ Incoming dense stream || Accumulation is enough
10:     else if $L_t \geqslant L$ or $L_s \geqslant L$ then
11:         Push $\mathbf{h}_t$ at the front of $Q$
12:         $L_s = 0$
13:   ▷ Incoming sparse stream & Previous stream is sparse
14:     else
15:         Add $\mathbf{h}_t$ to the front of $Q$ (accumulation)
16:         $L_s = L_s + L_t$
17:     if $Q$ is full then
18:         Pop the rear of the queue
19:     $t = t + 1$
20: $HH_t$ = Stack(h in Q)   ▷ Generate Hyper Histogram
```

Compared with calculating 2D histograms for $4K$ times, our proposed fast implementation requires only one calculation for a 1D histogram, which can be achieved in a CUDA-accelerated way. According to statistics, the multi-index method improves the generating speed by over 14 times.

## Adaptive Event Conversion Module

With the novel event representation Hyper Histogram, we design an Adaptive Event Conversion (AEC) module to reduce extra time consumption from converting continuous event streams to Hyper Histograms and make full use of event information for accurate detection.

In AEC, the channel number $4\hat{K}$ of Hyper Histogram is adaptively changed according to the density of incoming



The projection of the event points onto the X-Y plane.
The 3D visualization of event points considering time dimension.
Original events
Shadow events
Mosaic events
Scale down
Scaled events (down)
Scaled events (up)
Shadow
Mosaic
OR Scale up
(with Shadow distortion repairing)

Figure 3: The flowchart of the Shadow Mosaic augmentation method on raw events. The current event sample and 3 randomly selected event samples are loaded each time. The Shadow method is performed on 4 samples first to generate shadow events with different spatio-temporal density. Then 4 shadow event samples are merged into one large mosaic event sample. The coordinates of merged events are randomly scaled up or down. The Shadow method is also used to repair the event distortion after scaling. Finally, event cropping is performed.

event streams. If the event stream is super dense, then $\hat{K}$ is large and vice versa. We also leverage the channel attention and weighted dimension-reducing to further improve performance.

First, an adaptive queue $Q$, with the capacity $T$, is initialized, which caches building blocks. At timestamp $t$, the module loads the event stream of a short time $ts$ and generates its building block $\mathbf{h}_t$. At the same time, the queue chooses how to handle $\mathbf{h}_t$ according to the length of the incoming event stream (reflecting its density) and generate the Hyper Histogram $HH_t$. The main steps are summarized in Algorithm 1.

The queue outputs $HH_t$, which consists of $4\hat{K} \leqslant \frac{4T}{ts}$ groups of dense histograms. This design prevents over-sparse event histograms being fed to the following network. The generated histograms are then weighted by a SENet block (Hu, Shen, and Sun 2018). The channel number is reduced from $4\hat{K}$ to a pre-defined number $C$ through a $1 \times 1$ convolution layer.

The squeeze and excitation process of SENet makes full use of the global information of each channel. The $1 \times 1$ convolution layer can be deemed as the weighted dimension-reducing process. The combination of the two operations can better distinguish events in different time windows and avoid the network convergence problem due to the unfixed input channel number.

The adaptive queue in the AEC module requires the input to be sequential, so it can only be used in the detection of continuous event streams. In the training process, we did not utilize continuous event streams, so the adaptive queue is not applicable for the training phase. We only utilize the adaptive queue in the testing phase, where the testing stream is continuous. In order to improve robustness of our method, during training, the $K$ is randomly selected in the range $[1, \frac{T}{ts}]$.

## Shadow Mosaic Augmentation

Data augmentation is essential in object detection tasks to improve the performance and generalization ability of the model by diversifying the dataset. Previous event-based object detection methods seldom discuss the data augmentation. In order to achieve high performance and robust end-to-end event-based object detection, our method needs effective data augmentation methods that can be directly applied to raw event data. We design the Shadow Mosaic for event-based data augmentation. Fig. 3 shows the procedures of Shadow Mosaic augmentation, which mainly has four procedures: Shadow, Mosaic, Scaling and Cropping.

**Shadow** The spatio-temporal density of event streams is usually unbalanced, which is probably due to different moving speeds, illumination and etc. This inspires us to propose the Shadow method, which simulates events of different densities based on the original events.

Sparse shadow events are simulated by random sampling the original events. Dense shadow events are simulated by replicating events in the three-dimensional domain. When the brightness of a certain X-Y coordinates changes rapidly in a short period, event cameras will generate denser events over the time dimension. Similarly, intense jitter in a short period generates denser noisy events in nearby X-Y coordinates. Otherwise, relatively sparse events are generated.

**Mosaic** In object detection, detection of small objects is always challenging. To improve the detection accuracy on small objects, we refer to the classic image augmentation method Mosaic with scale jittering (Bochkovskiy, Wang, and Liao 2020; Simonyan and Zisserman 2015), and extend the Mosaic method for events.

The first step in Mosaic is to merge four shadow event samples into one large sample. The original coordinates of each shadow event $(x, y) \in \mathbb{R}^{W \times H}$ are transformed to new coordinates $(x' + \Delta x_1, y' + \Delta y_1) \in \mathbb{R}^{2W \times 2H}$, with a random offset $(\Delta x_1, \Delta y_1)$. The events belonging to each shadow event sample occupy a separate range of $W \times H$ in the new coordinate system, and do not overlap.

**Scaling** The merged events are then scaled up or down according to a ratio $k \in [0.5, 2]$, and another random offset $(\Delta x_2, \Delta y_2)$ is added. The coordinates of the scaled mosaic event become

$$(k(x' + \Delta x_1) + \Delta x_2, k(y' + \Delta y_1) + \Delta y_2). \tag{3}$$

One problem is that there is event distortion after scaling. Image scaling is usually accomplished by image interpolation, such as nearest, bilinear and bicubic interpolations. If the traditional image interpolation is directly used on scaled events, or no interpolation is used, the generated events usually suffer from pixel loss or distortion. Fig. 4 shows the visualizations of the scaled events after using Shadow and traditional interpolation methods.

The second row of Fig. 4 illustrates the visualizations after interpolating the scaled event directly, which introduces considerable spatial noise due to the asynchronous nature of events. The traditional interpolation methods are only suitable for the generated image-like planes, but they destroy the relationship between channels, as shown in the third row, which also loses many spatial details. Only Shadow can repair the scaled events and restore more realistic original events. Therefore, to ensure that the augmented data are consistent with the original data, the Shadow method is performed once again after scaling up.

**Cropping** Finally, event cropping is performed on the scaled events. Referring to the imaged-based cropping operation (Simonyan and Zisserman 2015), we design event cropping to eliminate events outside the cropping area while keeping the intersecting coordinates of four event samples inside the cropping area. The cropping areas are shown by colored boxes in Fig. 3 with the crop size $2W \times 2H$.

## Experiments

### Datasets

We evaluate our proposed methods on three representative datasets, which are the 1Mpx Detection dataset (Perot et al. 2020), the Gen1 Detection dataset (de Tournemire et al. 2020) and the Nighttime Driving Detection dataset MVSEC-NIGHTL21 (Hu, Liu, and Delbruck 2021).

The 1Mpx Detection dataset contains a total of 14.65 hours of events, of which 11.19 hours are for training, 2.21 hours are for validation, and 2.25 hours are for testing. The sample resolution is $1280 \times 720$. There are 7 object classes, but only 3 classes are chosen for performance comparison. The total number of bounding boxes is 25 M+. The labeling frequency is 60 Hz.

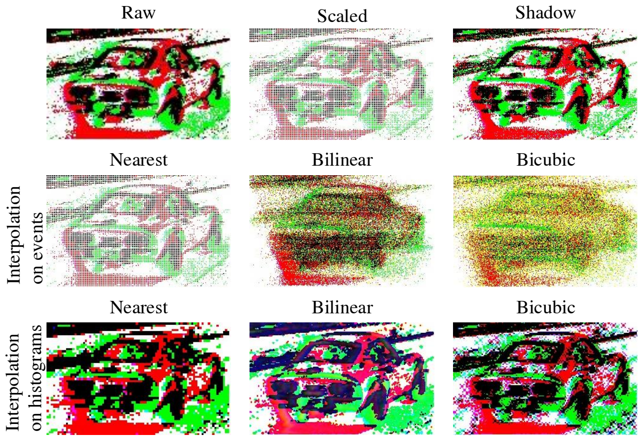


Figure 4: Visualizations of event distortion repaired by Shadow and other interpolation methods.

The Gen1 Detection dataset is collected with a relatively low resolution of 304×240, and there are only 2 classes of objects. It consists of 39 hours of events, of which 22.63 hours are for training, 6.59 hours for validation, and 10.10 hours for testing. The labeling frequency is 20 Hz.

The MVSEC-NIGHTL21 dataset is a derived dataset of the Multi Vehicle Stereo Event Camera dataset (Zhu et al. 2018) with moving vehicle detection labels. There are 386 labeled event streams with a resolution of 346×260. The entire dataset is collected at night with low illumination, and the objects are barely discernable from the RGB frames provided by the authors. The labeling frequency is 12 Hz.

### Models

The frame-based models we choose for event-based object detection are YOLOv5 (Jocher 2021), Deformable-DETR (Zhu et al. 2021) and RetinaNet (Lin et al. 2020). The model depth and layer channel multiples of YOLOv5 are both 1.0. The backbones of RetinaNet and Deformable-DETR are both ResNet50 (He et al. 2016).

### Implementation Details

For fair comparisons, we choose the most commonly used $50\ ms$ as the time interval. The accumulated time $T$ of the queue in AEC is also set to $50\ ms$. $C$ and $ts$ are set as 4 and $\frac{50\ ms}{8}$. Since the interval between two GTs may be greater than $ts$, during testing, the GT will not be updated until arrives the next GT timestamp. The values of the threshold $L$ on three datasets are chosen as 100000, 10000, and 5750 respectively.

We follow the dataset filtering of previous works (Perot et al. 2020; Li et al. 2022). The resolution of the 1Mpx dataset is reduced to 640×360. When training the models, we utilize 8 NVIDIA GTX 1080Ti GPUs for 100 epochs. When testing, only one NVIDIA GTX 1080Ti GPU is used.

### Experimental Results

In the experiment, we utilize two metrics to evaluate the detection performance. One is COCO mAP (mAP@0.5:0.95)

| Methods | mAP (%) | runtime (ms) |
|---|---|---|
| SAM (Liang et al. 2021) | 23.9 | - |
| UDA (Messikommer et al. 2022) | 48.0$^\star$ | - |
| ASTMNet (Li et al. 2022) | 48.3$^\diamond$ | 72.3 |
| RED (Perot et al. 2020) | 43.0 | 39.3 |
| ITS + RetinaNet | 35.0 / 36.9* | 90.3 |
| ITS + Deformable-DETR | 37.8 / 39.8* | 88.0 |
| ITS + YOLOv5 | 39.4 / 41.5* | 84.0 |
| Ours + RetinaNet | 41.8 / 44.1* | 22.0 |
| Ours + Deformable-DETR | 45.9 / 48.3* | 20.7 |
| Ours + YOLOv5 | **48.4 / 51.0*** | **13.8** |

Table 1: Performance comparison with the state-of-the-art methods on the 1Mpx Detection dataset. $*$: The confidence threshold is 0.3. $\diamond$: The reported result is in COCO AP50.

| Methods | mAP (%) | runtime (ms) |
|---|---|---|
| AEGNN (Schaefer et al. 2022) | 16.3 | - |
| SNN-SSD (Cordone et al. 2022) | 18.9 | - |
| SAM (Liang et al. 2021) | 35.5 | - |
| FS (Crafton et al. 2021) | 39.6$^\diamond$ | 41.2 |
| RED (Perot et al. 2020) | 40.0 | 16.7 |
| ASTMNet (Li et al. 2022) | 46.7* | 35.6 |
| ITS + RetinaNet | 36.5 / 44.1* | 34.2 |
| ITS + Deformable-DETR | 37.9 / 45.8* | 32.6 |
| ITS + YOLOv5 | 38.2 / 46.1* | 25.6 |
| Ours + RetinaNet | 43.1 / 51.9* | 9.4 |
| Ours + Deformable-DETR | 44.5 / 53.5* | 7.7 |
| Ours + YOLOv5 | **47.0 / 56.5*** | **3.9** |

Table 2: Performance comparison with the state-of-the-art methods on the Gen1 Detection dataset. $*$: The confidence threshold is 0.3. $\diamond$: The reported result is in COCO AP50.

calculated using YOLOv5 utilities, with a confidence threshold of 0.3 as the state-of-the-art method ASTMNet (Li et al. 2022). We also provide results calculated using the official COCO API (The confidence threshold is 0) for a fair comparison. The other is runtime, which includes the event conversion time and the network inference time. For a comprehensive comparison, we compare our method with several state-of-the-art methods and baselines utilizing ITS (Inceptive Time Surface) representation (Baldwin et al. 2019).

**1Mpx Detection Dataset** From Table 1, we can see significant performance gains for all three models on the 1Mpx dataset. Among three detection models, YOLOv5 shows the best performance. Compared with the state-of-the-art network ASTMNet (Li et al. 2022), the mAP is increased from 48.3% to 51.0%, while maintaining a very short runtime of 13.8 $ms$, which is only 19% of ASTMNet's and 35% of RED's. Compared with our baselines, the gains are (7.2% mAP, 68.3 $ms$ runtime), (8.5% mAP, 67.3 $ms$ runtime) and (9.5% mAP, 70.2 $ms$ runtime), respectively.

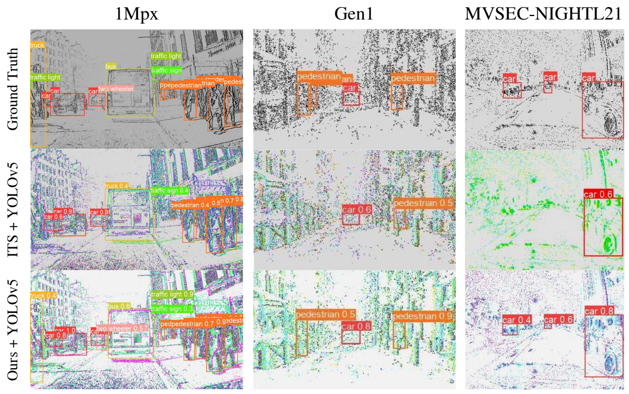


Figure 5: Visualization of detection results on 3 datasets. Detector with our methods detects the most objects and the bounding boxes are more accurate.

| Methods | Training dataset | mAP (%) |
|---|---|---|
| ITS + RetinaNet | 1Mpx / Gen1 | 30.7 / 30.1 |
| ITS + Deformable-DETR | 1Mpx / Gen1 | 35.4 / 30.9 |
| ITS + YOLOv5 | 1Mpx / Gen1 | 37.3 / 31.7 |
| Ours + RetinaNet | 1Mpx / Gen1 | 36.5 / 34.6 |
| Ours + Deformable-DETR | 1Mpx / Gen1 | 44.3 / 37.0 |
| Ours + YOLOv5 | 1Mpx / Gen1 | **46.8 / 38.8** |

Table 3: Detection Performance on the MVSEC-NIGHTL21 dataset. It should be noted that the models we use for testing are pretrained on the 1Mpx / Gen1 datasets.

**Gen1 Detection Dataset** The results of the comparison on the Gen1 dataset are shown in Table 2. Our YOLOv5-based version achieves the largest improvement over ASTMNet in terms of mAP, from 46.7% to 56.5%. Leveraging the compact structure of YOLOv5, the runtime is only 3.9 $ms$, which is only 11% of ASTMNet's and 23% of RED's. Compared with our baselines, the gains are (7.8% mAP, 24.8 $ms$ runtime), (7.7% mAP, 24.9 $ms$ runtime) and (10.4% mAP, 21.7 $ms$ runtime), respectively.

**MVSEC-NIGHTL21 Dataset** We further provide the generalization ability test results in Table 3 by directly evaluating trained models on the MVSEC-NIGHTL21 dataset. The performance of our YOLOv5-based version trained on two datasets is increased by 9.5% and 7.1%. While Deformable-DETR achieves increments of 8.9% and 6.1%. RetinaNet increases by 5.8% and 4.5%.

These encouraging results indicate that we can achieve high-performance end-to-end event-based object detection on the 1Mpx and Gen1 datasets, quicker than any previous work. With some lighter and quicker detectors, the detection speed can be further improved. Meanwhile, with our proposed methods, the model trained on large-scale datasets has sufficient generalization ability to be transferred to other datasets collected under different conditions.

| | 1 Mpx Detection dataset | | Gen1 Detection dataset | |
|---|---|---|---|---|
| Methods | mAP (%) | runtime (ms) | mAP (%) | runtime (ms) |
| Baseline (ITS + YOLOv5) | 39.4 | 84.0 | 38.2 | 25.6 |
| + Mosaic (w/o distortion repair) | 44.3 (+4.9) | 83.7 | 43.3 (+5.1) | 26.0 |
| + Shadow Mosaic | 46.5 (+7.1) | 84.5 | 45.0 (+6.8) | 26.2 |
| + Shadow Mosaic + Hyper Histogram | 47.4 (+7.9) | 22.6 | 46.6 (+8.4) | 9.2 |
| + Shadow Mosaic + Hyper Histogram + AEC | **48.4 (+9.0)** | **13.8** | **47.0 (+8.8)** | **3.9** |

Table 4: Performance of YOLOv5 with different modules.

| Interpolation Methods | Applied on | mAP (%) |
|---|---|---|
| Nearest | Events | 43.6 |
| Bilinear | Events | 39.8 |
| Nearest | ITS | 44.5 |
| Bilinear | ITS | 43.7 |
| Shadow | Events | **45.0** |

Table 5: Performance of YOLOv5 with Shadow and different interpolation methods on the 1Mpx Detection dataset.

**Visual Detection Results** We present some visual detection results of different datasets and different methods in Fig. 5. It can be observed that YOLOv5 with our proposed AEC module and Shadow Mosaic successfully detects the most objects. The regression of bounding boxes is also more accurate. Especially for the MVSEC-NIGHTL21 dataset with different scales and event densities, the models pre-trained on the 1Mpx dataset can still achieve accurate detection.

## Ablation Studies

We perform a series of ablation experiments on the proposed methods. These experiments are conducted on the 1Mpx dataset and the Gen1 dataset. The baseline we utilize is ITS+YOLOv5. The ablation results are shown in Table 4.
**Mosaic** The Mosaic augmentation method enriches the scale diversity of event samples. This leads to increases of 4.9% and 5.1% in mAP for the two datasets.
**Shadow** The Shadow augmentation method increases the sample diversity by randomly changing the density of input events. It also repairs the distortion after scaling. This leads to increases of 2.2% and 1.7% in mAP for two datasets. We also compare the ability of Shadow with other interpolation methods to repair the distortion. Different methods are used with Mosaic augmentation. We report the optimal mAP in Table 5. The Shadow method outperforms any other interpolation methods.
**Hyper Histogram** The experimental results prove that the representation Hyper Histogram achieves optimal mAP increments of 0.8% and 1.6% for the two datasets. The multi-index mapping method accelerates the runtime by 3.7 and 2.8 times on the two datasets.
**AEC** The AEC module further reduce the runtime by 6.1 and 6.6 times on two datasets, with mAP increments of 1.1% and 0.4%, respectively.

We also conduct ablation experiments on the event representation. We start with comparing the Hyper Histogram with other commonly used representations, which are event histogram, time surface, event volume and ITS. We also compare the performance of models with different values of $K$. It should be noted that in these ablation experiments, no augmentation methods are utilized for a fair comparison.

| Representations | mAP (%) | runtime (ms) |
|---|---|---|
| Event histogram (Moeys et al. 2016) | 34.0 | 34.7 |
| Time surface (Lagorce et al. 2017)$*$ | 35.1 | 48.2 |
| Event volume (Hu et al. 2020)$*$ | 35.9 | 44.0 |
| ITS (Baldwin et al. 2019)$*$ | 36.3 | 84.0 |
| Hyper Histogram (K=1) | 35.9 | 21.2 |
| Hyper Histogram (K=2) | 36.4 | 22.0 |
| Hyper Histogram (K=4) | 37.0 | 22.4 |
| Hyper Histogram (K=8) | 36.9 | 22.7 |
| Hyper Histogram (K=16) | 35.7 | 23.0 |
| Hyper Histogram (AEC) | **37.2** | **13.8** |

Table 6: Performance of YOLOv5 with different event representations on the 1Mpx Detection dataset. $*$: Preprocessing is needed.

The quantitative results are shown in Table 6. It can be seen that the Hyper Histogram with a fixed $K = 4$ yields a maximum mAP of 37.0%. The mAP performance does not continue to increase with $K$ consistently. The introduction of AEC achieves 37.2% mAP, and the running time is reduced to only 16.4% of models using ITS. The 13.8 $ms$ runtime also includes a 9.4 $ms$ inference time, which means that the event conversion time is only 4.4 $ms$.

## Conclusion

In this paper, we propose a new event representation Hyper Histogram, which keeps the quantity and temporal information. We also design an Adaptive Event Conversion module to achieve efficient end-to-end event-based object detection. In addition, we propose an event-based augmentation method Shadow Mosaic, which significantly improves the event sample diversity and enhances the generalization ability of the model. We equip our proposed modules on three representative object detection models: YOLOv5, Deformable-DETR and RetinaNet. The experimental results on three event-based detection datasets (1Mpx, Gen1 and MVSEC-NIGHTL21) provide credible evidence that our proposed approach achieves superior performance over state-of-the-art methods with much faster running speed.

## Acknowledgments

This work is in part supported by the National Key R&D Program of China under Grant 2020AAA0108600, the National Natural Science Foundation of China under Grants 61901435 and 62032006, and the University Synergy Innovation Program of Anhui Province under Grant GXXT-2019-025.

## References

Baldwin, R.; Almatrafi, M.; Kaufman, J. R.; Asari, V.; and Hirakawa, K. 2019. Inceptive event time-surfaces for object classification using neuromorphic cameras. In *International Conference on Image Analysis and Recognition*, 395–403.

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

Bi, Y.; Chadha, A.; Abbas, A.; Bourtsoulatze, E.; and Andreopoulos, Y. 2020. Graph-Based Spatio-Temporal Feature Learning for Neuromorphic Vision Sensing. *IEEE Transactions on Image Processing*, 29: 9084–9098.

Bochkovskiy, A.; Wang, C.-Y.; and Liao, H.-Y. M. 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv:2004.10934.

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

Cannici, M.; Ciccone, M.; Romanoni, A.; and Matteucci, M. 2020. A Differentiable Recurrent Surface for Asynchronous Event-Based Data. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 1–17.

Cordone, L.; et al. 2022. Object Detection with Spiking Neural Networks on Automotive Event Data. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, 1–8.

Crafton, B.; Paredes, A.; Gebhardt, E.; and Raychowdhury, A. 2021. Hardware-Algorithm Co-Design Enabling Efficient Event-based Object Detection. In *2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 1–4.

de Tournemire, P.; Nitti, D.; Perot, E.; Migliore, D.; and Sironi, A. 2020. A Large Scale Event-based Detection Dataset for Automotive. arXiv:2001.08499.

Gallego, G.; Delbrück, T.; and Orchard, G., et al. 2022. Event-Based Vision: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1): 154–180.

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

Hu, J.; Shen, L.; and Sun, G. 2018. Squeeze-and-Excitation Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7132–7141.

Hu, Y.; Liu, S.-C.; and Delbruck, T. 2021. v2e: From Video Frames to Realistic DVS Events. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1312–1321.

Hu, Y.; et al. 2020. Learning to Exploit Multiple Vision Modalities by Using Grafted Networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 85–101.

Jocher, G. 2021. ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support. *Zenodo*.

Lagorce, X.; Meyer, C.; Ieng, S.-H.; Filliat, D.; and Benosman, R. 2015. Asynchronous Event-Based Multikernel Algorithm for High-Speed Visual Features Tracking. *IEEE Transactions on Neural Networks and Learning Systems*, 26(8): 1710–1720.

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

Lee, C.; Kosta, A.; Zhu, A. Z.; Chaney, K.; Daniilidis, K.; and Roy, K. 2020. Spike-FlowNet: Event-based Optical Flow Estimation with Energy-Efficient Hybrid Neural Networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 366–382.

Li, J.; Li, J.; Zhu, L.; Xiang, X.; Huang, T.; and Tian, Y. 2022. Asynchronous Spatio-Temporal Memory Network for Continuous Event-Based Object Detection. *IEEE Transactions on Image Processing*, 31: 2975–2987.

Liang, Z.; Chen, G.; Li, Z.; Liu, P.; and Knoll, A. 2021. Event-based Object Detection with Lightweight Spatial Attention Mechanism. In *Proceedings of the IEEE International Conference on Advanced Robotics and Mechatronics (ICARM)*, 498–503.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2020. Focal Loss for Dense Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2): 318–327.

Maqueda, A. I.; Loquercio, A.; Gallego, G.; García, N.; and Scaramuzza, D. 2018. Event-Based Vision Meets Deep Learning on Steering Prediction for Self-Driving Cars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5419–5427.

Messikommer, N.; Gehrig, D.; Gehrig, M.; and Scaramuzza, D. 2022. Bridging the Gap between Events and Frames through Unsupervised Domain Adaptation. *IEEE Robotics and Automation Letters*, 7(2): 3515–3522.

Messikommer, N.; Gehrig, D.; Loquercio, A.; and Scaramuzza, D. 2020. Event-based asynchronous sparse convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 415–431.

Moeys, D. P.; Corradi, F.; Kerr, E.; Vance, P.; Das, G.; Neil, D.; Kerr, D.; and Delbrück, T. 2016. Steering a predator robot using a mixed frame/event-driven convolutional neural network. In *Proceedings of the International Conference* *on Event-based Control, Communication, and Signal Processing (EBCCSP)*, 1–8.

Perot, E.; de Tournemire, P.; Nitti, D.; Masci, J.; and Sironi, A. 2020. Learning to Detect Objects with a 1 Megapixel Event Camera. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, 16639–16652.

Schaefer, S.; et al. 2022. AEGNN: Asynchronous Event-based Graph Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 12371–12381.

Shrestha, S. B.; and Orchard, G. 2018. SLAYER: Spike Layer Error Reassignment in Time. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1412–1421.

Simonyan, K.; and Zisserman, A. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 1–14.

Zhu, A.; Yuan, L.; Chaney, K.; and Daniilidis, K. 2019. Unsupervised Event-Based Learning of Optical Flow, Depth, and Egomotion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 989–997.

Zhu, A. Z.; Thakur, D.; Özaslan, T.; Pfrommer, B.; Kumar, V.; and Daniilidis, K. 2018. The Multi Vehicle Stereo Event Camera Dataset: An Event Camera Dataset for 3D Perception. *IEEE Robotics and Automation Letters*, 3(3): 2032–2039.

Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; and Dai, J. 2021. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 1–16.