How Many Events Make an Object? Improving Single-frame Object Detection on the 1 Mpx Dataset

Alexander Kugele[1,2], Thomas Pfeil[3], Michael Pfeiffer[1], Elisabetta Chicca[2]
[1] Bosch Center for Artificial Intelligence, Renningen, Germany
[2] University of Groningen, Groningen, Netherlands [3] Recogni GmbH, Munich, Germany
{alexander.kugele,michael.pfeiffer3}@de.bosch.com, tom@recogni.com, e.chicca@rug.nl

Abstract

Event cameras are promising novel vision sensors with higher dynamic range and higher temporal resolution compared to frame-based cameras. In contrast to images, single-frame detectors without memory perform poorly on event data. We analyze the distribution of event counts in the 2D bounding boxes in the 1 Mpx Dataset to find that the distribution is skewed towards few events, rendering it impossible to detect objects based only on current information. Memory layers like LSTM can alleviate this problem, but increase training time and inference costs. To bring the advantages of single-frame detectors to event camera data, we propose a data filtering mechanism and a novel bounding box memory. The filtering mechanism excludes labels with low event count during training, which improves performance on unfiltered test data. The bounding box memory memorizes bounding boxes until an event threshold is reached, which improves performance, has a low memory and latency footprint, and can be integrated into any object detector without retraining. Improvements are shown on a simulated dataset based on moving MNIST digits, as well as the 1 Mpx Dataset, the largest event camera object detection dataset to date, illustrating that our method scales to large datasets and works in a complex real-world setting.

1. Introduction

Autonomous driving is a topic of broad and current interest in computer vision. One of the key ingredients is a fast and accurate vision system to detect traffic participants like cars or pedestrians. Akin to human drivers, who almost exclusively use their visual system, it is believed that an autonomous driving agent should also be able to fulfill this task using only cameras. As a result, a lot of research and engineering time is invested to design and train better neural networks for frame-based camera input. However, frame-based cameras themselves already exhibit certain limitations. The low dynamic range leads to a performance degradation at night or in mixed-light scenarios like tunnels or alleys, while the fixed frame rate and absolute intensity measurements lead to motion blur and saturation. A high frame rate additionally leads to a lot of redundant data and a higher energy consumption.

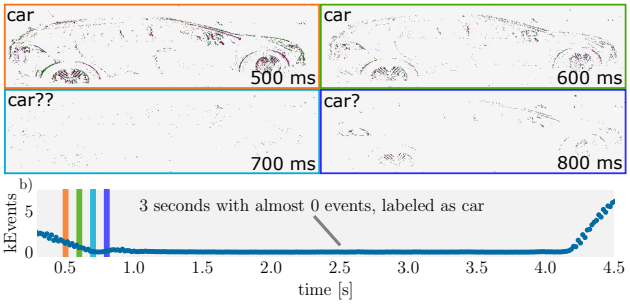

Figure 1. **a)** Four event volumes accumulated over a span of 33 ms showing a car that stops due to traffic. While the car can be clearly identified in the first two frames, it is unidentifiable in the third frame and hard to identify in the fourth. In the 1 Mpx Dataset, all of these are labeled as 'car', because labeling is done with a frame-based camera. **b)** Event count vs time for the object in a)

Event cameras have been invented almost 15 years ago [21] to be more similar to the sensory organ of the visual system, and their working principle improves significantly on the short-comings of traditional cameras. Each pixel measures changes in logarithmic light intensity and outputs a binary event when the intensity changed over a certain threshold compared to the intensity at the last event. Measuring relative logarithmic light intensity leads to a higher dynamic range and independence of global illumination [26]. Recording only intensity changes is data-efficient with peak bandwidth reductions of up to a factor of 1000 [29]. Independent updates of each pixel without global integration result in a high temporal resolution that

eliminates motion blur.

Multiple iterations of event cameras have been developed over the past decade [3, 7, 13, 17, 18, 21, 30–32], with the most recent generation exhibiting Megapixel resolution at low power (32 to 400 mW), low latency (8 to 150 μs) and high dynamic range (100 to 124 dB) [3, 7, 31]. For comparison, frame-based cameras have a latency of about 8000 μs at a dynamic range of about 60 dB while consuming about 1000 mW.

Although event cameras are superior in these aspects, they have not been used extensively in research or industry. This is arguably due to a lack of real-world datasets, which prohibits showing the advantage of event cameras in complex settings and impedes the development of algorithms for complex tasks. Recently, this changed with the release of multiple large-scale automotive datasets like DDD20 [14], DSEC [10], GEN1 [4] and the 1 Megapixel Automotive Detection Dataset [26]. Only a few publications made use of these large-scale datasets so far. In particular for the 1 Mpx Dataset, only results on a subset of the data have been published so far [20]. One potential issue is the considerable amount of compute that is needed to train large sequence models on vision data.

In this paper, we scrutinize the object detection capabilities of single-shot neural networks on event camera data and therefore focus on the 1 Megapixel Automotive Detection Dataset, or 1 Mpx Dataset for short [26]. In particular, we analyze the performance of a single-frame single-shot architecture, the most popular form of object detection network for image-like data. Single-frame denotes that each prediction only depends on a single frame (in contrast, to *e.g.* predictions based on a video sequence) and single-shot indicates the single-shot detector (SSD) architecture [24]. This type of detector is simple to train and fast during inference, making it a preferred choice. In comparison, the accompanying article to the 1 Mpx Dataset proposes a single-shot ANN architecture called RED (Recurrent Event-camera Detector). To integrate the temporal information over time, the internal state at multiple abstraction levels in the network is propagated through time with a convolutional Long Short-Term Memory network (ConvLSTM). This is more time- and memory-consuming than a single-frame detector, but increases detection performance in their experiments.

To equip a single-frame detector with similar capabilities, we identify and alleviate a specific problem that does not occur in frame-based data: The dependency of a representation of an object on the relative movements between camera and object. An example for this can be seen in Fig. 1. An event camera records a car that stops due to a traffic light. There is less and less relative movement between the car and camera, and therefore the car is less and less visible over time. However, the goal of the object detector is to detect the car at any time, because although the car vanishes in the event stream, it does not vanish in the real world. Therefore, it is imperative that an event camera object detector is able to detect objects even if they are not visible from recent events. To this end, we have the following contributions

- We identify that the distribution of events in 1 Mpx Dataset is skewed towards few events per label, rendering it impossible for single-frame detectors to perform well
- We simulate a toy dataset to investigate the effect of objects with a low event count in a dataset
- We propose a dataset filtering mechanism to improve performance of a single-frame detector during inference
- We propose a new, efficient memory that does not need to be trained and can be used with any object detector for event cameras
- We show that our filtering mechanism and memory improve the performance of a single-frame detector on the toy dataset and the 1 Mpx Dataset

2. Related Work

Event camera object detection Approaches for object detection on event streams can be divided into two streams. The first adapts existing ANNs for image data based on assumptions of the characteristics of event streams. In [28], a recurrent U-Net is proposed that can reconstruct a video from an event stream and the resulting frames can be fed into a frame-based object detector. An adapted version of SSD [24] is proposed in [26], which uses convolutional LSTM layers (ConvLSTM) in the neck of the detection network to propagate information about past detection to the current prediction. In [6], a frame-based detector is used on accumulated events and fused with the predictions of a frame-based detector on frames that are synchronized with the events. The authors of [23] use an SSD architecture as well, but propose a novel event representation called Temporal Active Focus, that aims to make the representation motion-independent. To model global and local dependencies, [19] use a transfomer network that aggregates features over multiple event tensors. Similarly, [11] propose a recurrent vision transformer that is optimized for event streams. In [9], the authors propose a graph neural network that processes a graph representation of the event stream and scale it up to more complex tasks by reducing computation.

A second approach to process event streams is the use of spiking neural networks (SNNs). Their promise is to process event data more efficiently, because neurons only communicate via binary signals and filter noise via a thresholding mechanism. Examples include [15], who adapt a YOLO network to a spiking neural network and [16] who propose

a Hybrid SNN-ANN architecture with an SNN feature extractor and an ANN head for object detection. To train a spiking neural network, typically many time steps have to be processed, which rendered it impossible to scale them up to larger datasets. From the above methods, only the method accompanying the 1 Mpx Dataset [26] trains and evaluates on the full dataset.

Datasets The largest object detection dataset for event camera data is the 1 Mpx Dataset. It provides multiple hours of high resolution event camera data together with 2D bounding boxes at a labeling rate of 60 Hz of an event camera mounted in a car driving in and around Paris. We discuss this dataset extensively in Sec. 3.1. The same authors previously released a similar dataset, GEN1 [4], with the main difference that the label frequency and resolution are lower (304×240 at 4 Hz). Because a high resolution is important for object detection to identify small objects, and because we want to train a realtime detector, we exclude this dataset from our analysis. Similar to the 1 Mpx Dataset, DSEC [10] provides a diverse set of driving scenarios. In addition to two high resolution event streams, the dataset also includes stereo frames, LIDAR and GPS, but no 2D bounding boxes. The biggest event camera dataset to date (51 hours), DDD20 [14], is recorded with a lower resolution event camera and also does not contain 2D bounding boxes.

3. Object Detection on Event Camera Data

Our goal in this paper is to understand the challenges of using event cameras in the complex real-world setting of 2D bounding box detection. To this end, we analyze the 1 Mpx Dataset with the surprising finding that a significant amount of objects is represented by only few events (Sec. 3.1). We are particularly interested to find out how a typical single-frame single-shot detector commonly used for images fares in this setting. We present our base architecture in Sec. 3.2 and compare it to RED [26], the architecture proposed in conjunction with the 1 Mpx Dataset. To equip our single-frame architecture with similar capabilities than a recurrent architecture, we propose a data filtering mechanism in Sec. 3.3 and a bounding box memory in Sec. 3.4. To understand the implications of each proposed method, we create a toy dataset of moving MNIST digits (frames and events), called Random-Movements-MNIST, or RM-MNIST, for short (Sec. 3.5). After showing that both methods improve results on RM-MNIST, data filtering and memory are applied to a bigger network trained on the 1 Mpx Dataset (Sec. 3.6), improving our single-frame architecture there as well.

3.1. The 1 Megapixel Automotive Detection Dataset

The 1 Megapixel Automotive Detection Dataset [26] is a large-scale, real-world event camera dataset for 2D object detection. Data is collected with a calibrated setup consisting of an event- and a frame-based camera mounted in a car. A commercial object detector is used on the frames to generate the labels for three categories: car, pedestrian and two-wheeler. With a sensor size of 1280x720 pixels and 25.8 million labels collected over 15.6 h of driving it is both the most high resolution and diverse dataset for event camera data to date. The fact that it is divided into about 700 sequences of approximately 60 seconds makes it possible to investigate the temporal behavior of event data for the first time.

Missing from the dataset are the RGB frames for labeling, night scenes and annotations, *e.g.* for different scene types (city, highway, ...) or weather conditions. Furthermore, due to the automatic labeling procedure, the labels are only as good as the frame-based object detector they used.

The authors filter out boxes with a diagonal smaller than 60 pixels and according to the data loading code provided with their paper [27], also boxes where at least one side is smaller than 20 pixels (both in the original 1280x720 resolution, *i.e.* before rescaling). We take this filtered version as the basis for our analysis and experiments.

To illustrate a potential failure case of the labeling strategy using frames, we cropped a single car from one of the training sequences and show it in Fig. 1a. While the car can be clearly identified in the first two frames (orange and green), it is not possible to see in frame three (turquoise) and hard to see in frame four (blue). This is of course related to the number of events in the specified time window. Switching to a fixed number of events to generate voxels instead of a fixed time window seems to resolve this problem, but the problem with this approach is that it is not known a priori where the objects are located on the sensor. Therefore, it is only possible to set a global threshold on the number of events. This however leads to the same problems as a fixed time window when more than one object is in the frame or when the sensor itself moves. Slow-moving objects and objects moving in the direction of the camera trajectory will still have a low number of events.

It can be seen in Fig. 1b that this is not a short stand-still for one or two frames, but over three seconds with almost no events, equaling 180 almost empty bounding boxes. To figure out if this is a general trend in the dataset, we look at the distribution of event counts and areas in the whole dataset in Fig. 2. The 2D distribution of normalized box area vs event count is shown in Fig. 2a. To normalize, we divide the area by $1280 \cdot 720 = 921600$ pixels squared. A logarithmic color

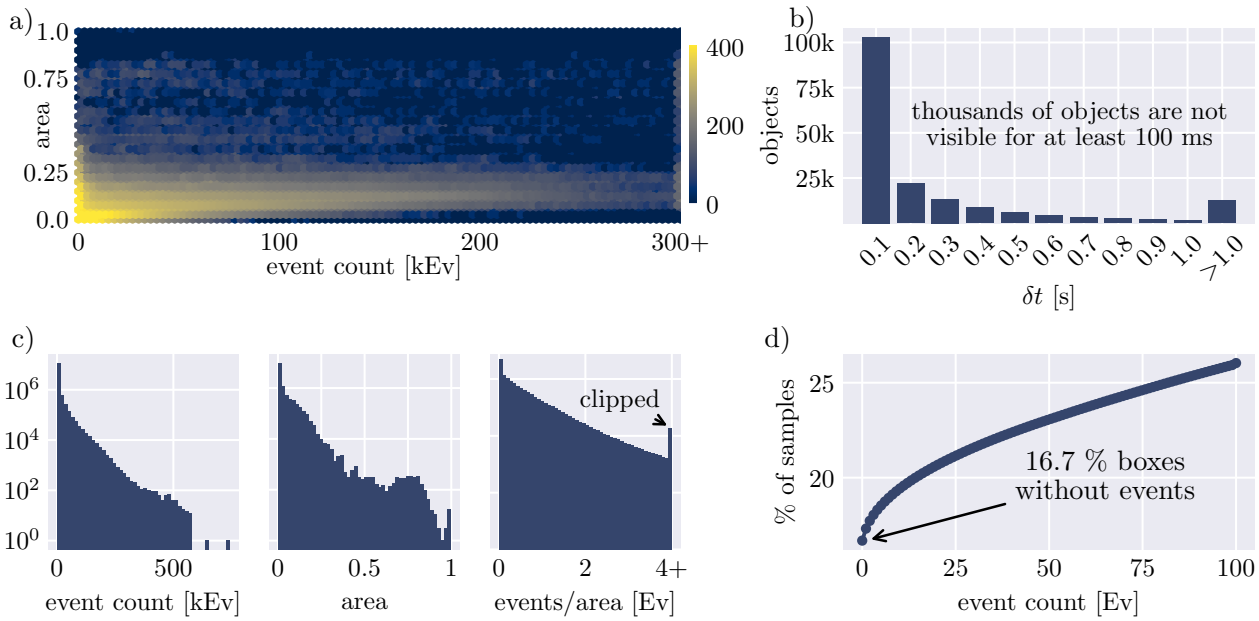

Figure 2. Distribution of events in bounding boxes and bounding box areas. **a)** Normalized bounding box area vs event count. The distribution is broad over the whole spectrum, with a concentration at small boxes. **b)** Number of objects that are below 100 events for at least δt seconds. **c)** Histograms over event counts, areas, and normalized event counts. **d)** Cumulative distribution of event counts per label. A significant amount of labels has almost no events.

code shows small amounts in blue and large amounts in yellow. We can see that the majority of bounding boxes is in the lower left corner, *i.e*. small boxes with only few events. In the area range from 0-0.25 however, the distribution has a long tail and there exist a significant amount of boxes at almost every hexagonal bin, from 0 up to 300 000 Events, where we capped the bins. Interestingly, the event count does not grow with the area of the bounding box, but instead for larger bounding boxes in the range from 0.25-0.75, the distribution is most dense between 0 and 75 000 Events. Unsurprisingly, only few bounding boxes fill the whole sensor area, and therefore the number of events is scarce in this region.

We are particularly interested to find out, if the dataset contains a lot of cases that are similar to what we have shown in Fig. 1. Therefore, we use the object tracking information in the dataset to analyze the event count over time for each object. Concretely, we define a threshold of 100 Events and measure the average time the event count of an object is below this threshold δt. The number of objects over δt is shown in Fig. 2b and it can be seen that a significant amount of objects is affected. Most of them (100 000) are not visible for about 100 ms, or about 6 frames (regarding the 60 Hz labeling frequency), but there are also about 10 000 objects not visible for over one second (at least 60 frames). Not accomodating this information during training and inference will impact model performance.

In Fig. 2c we show histograms for event count, normalized area and event count divided by normalized area. This last measure is particularly interesting, as it is a scale-independent measure for the event count of an object. As can be seen in the leftmost histogram, the event count linearly declines on a log-scale (*i.e*. it declines exponentially). Still, the distribution is very broad, ranging from 0 to 500 000 Events. The area shows a similar distribution with a small bump at around 0.75. Only few samples exist with a size larger than 0.85. The right-most histogram shows the distribution of events per (non-normalized) area, *i.e*. a value of 1 means that on average there is one event per pixel in the bounding box. As large changes in the input can lead to more than one event per pixel in the given time frame, this does not necessarily mean that the event volume is saturated, in practice even objects with 4 events per pixel can still be identified most of the time. The data has a similar, but much broader distribution. Two peaks are visible on the left and right edges, where the right peak just comes from clipping the data. The left peak indicates that there is a significant amount of bounding boxes without any events in it.

The cumulative number of samples (in percent) over the event count is shown in Fig. 2d. One sixth of the provided labels have no events at all, and almost every fourth label

has fewer than 75 events. We therefore believe that this has to be considered explicitly when optimizing. In Sec. 3.3 we explain our filtering strategy and how it helps the optimization process. In Sec. 3.4 we introduce a novel memory mechanism for 2D bounding boxes that does not have to be optimized and can be used with off-the-shelf object detectors for event camera data.

3.2. Object detection network

Single-frame single-shot detectors are the most common architecture for image data. We want to build on their success and find a suitable architecture for event streams, which is able to handle few events as discussed in Sec. 3.1. RED [26], which was proposed in conjunction with the 1 Mpx Dataset includes ConvLSTM layers in the network neck, which potentially helps in the cases of few events. However, it is unclear how long the network can remember objects and we have already seen that a significant amount of cases require remembering objects over multiple seconds. In addition, introducing recurrence makes networks more sophisticated to train, they need more memory during runtime and have an increased latency and energy consumption. Therefore, we present a single-frame detector, that we will equip with additional functionality to handle few events.

Events are transformed to an event volume of shape (bins, 2, 720, 1280) in fixed time steps of $16.67\,\mathrm{ms}$, which matches the frame rate of ground truth labels of $60\,\mathrm{Hz}$. Using a fixed time step instead of a fixed event count, as it is done in [28], ensures that the predictions are always in sync with the ground truth labels and also makes it easier to build downstream tasks on the basis of predicted labels. As the temporal information can be controlled by the number of bins of the event volume, the effect of motion blur for fast moving objects is reduced, compared to other representations (histogram, timesurface, frame-like). Similar to [26], we downscale the input shape of $(720, 1280)$ by two and rescale the event volumes to a square shape of $(360, 360)$ pixels.

Multiple heads with pre-defined prior boxes are used to predict bounding boxes at different scales. The focal loss [22] is used for classes, smooth L1-loss to regress the locations. During training, prior boxes are set to match a ground truth box, if $\mathrm{IoU} >= 0.5$, and background if they do not match any ground truth bounding box. During inference, non-maximum suppression (NMS) is used to merge close predictions. To evaluate the performance of, we use CoCo mean average precision (mAP) [12].

3.3. Filtering labels

We have seen in Sec. 3.1 that the 1 Mpx Dataset dataset has a lot of labels with only few events. This can have multiple effects on the training and validation of object detectors. To make use of these labels during training, a mechanism is needed that does not only utilize the current events but also information from the past. The authors of [26] use ConvLSTM layers to propagate the internal state over time, which solves this problem but leads to a larger storage footprint during training and validation and longer training times because the network has to be unrolled during training and the error signal has to be backpropagated through time. As shown in Fig. 1b and Fig. 2b, the time to unroll has to be considerably long (more than 180 steps) to cover extreme cases where an object is not visible for multiple seconds. Our goal is to perform object detection on single event volumes, and therefore we treat labels with a low event count as bad labels and filter them out. This can be viewed as label cleaning, which is a key step to increase performance [8, 33].

During validation, the labels with low event count cannot be detected with a single-frame detector. Therefore, we test them under three scenarios: First, we filter training, validation and test data to more accurately evaluate the actual performance on single frames, where we expect the single-frame detectors to perform as well as detectors with memory. Second, we filter only training data but test on unfiltered data to evaluate the gain of reducing these 'noisy' labels. Third, we add a memory mechanism to the first or second scenario to evaluate the gain in performance with memory.

But how many events make an object? By filtering at different event thresholds, we are going to determine the best trade-off between discarding labels and filtering noise (see Sec. 3.6).

3.4. The bounding box memory

As we have seen in Sec. 3.1, a significant amount of labels has only few events. To detect objects under this condition during inference, it is absolutely necessary to memorize the past, because the information of the present is not sufficient to detect the object. [26] propose ConvLSTM layers in the detector neck, which boost the detection performance significantly. We want to propose a simpler mechanism, that does not need to be trained, has a low memory footprint and can be used with any event camera object detector. The key idea is, that after a successful detection from a single-frame detector, the object cannot have moved out of the 2D bounding box without generating events. By remembering the locations of detected objects, we do not need to detect the objects again until a significant amount of events occurs in that region.

In the following, we describe the process in more detail. To start, all predicted bounding boxes above the confidence threshold T_c are put into the memory. At each time step, bounding boxes are removed from memory if the event count N crossed a threshold T_e and there is a box in the pre-

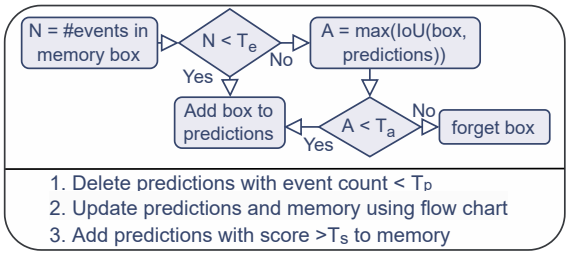

Figure 3. Memory update process. Boxes are memorized as long as the event count in that area is below a threshold.

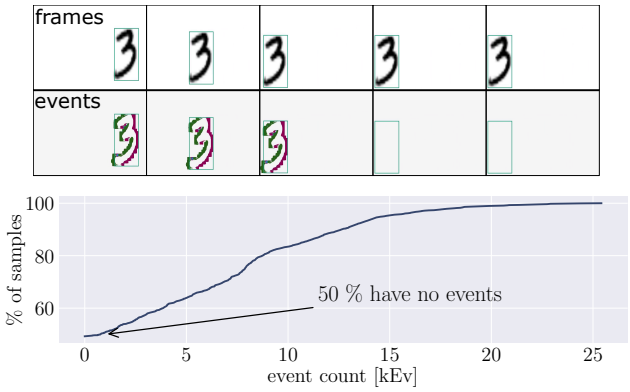

Figure 4. RM-MNIST. **Top**: Example sequence. Stationary objects vanish under event representation. **Bottom**: Cumulative event counts per object. Objects stand still 50 % of the time.

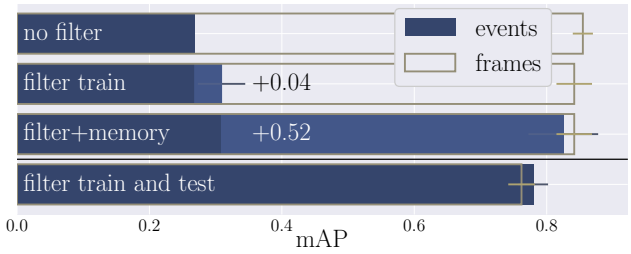

Figure 5. RM-MNIST results. *no filter*: single-frame baseline. *filter train*: Boxes without events are filtered from the train dataset. *filter + memory*: The memory is added, improving mAP significantly. *filter train and test*: Stationary objects are also filtered from the test dataset; Results are the same for frames and events, showing that only stationary objects cause the difference in mAP. Mean and error of the mean over two experiments are reported.

dicted boxes that overlaps with the given box from memory by at least T_a (see Fig. 3). All remaining bounding boxes are added to the predicted boxes. Checking the overlap T_a enables to correct for missed detections. To update the memory, predicted boxes are filtered by confidence and event count, and only boxes with $N > T_\mathrm{p}$ are put into memory. This ensures that predictions from only a small number of events are discarded, because they are unlikely to be correct. As the scene changes while the event camera itself is moving, boxes in memory are deleted regularly without any explicit intervention.

Our memory can be implemented efficiently even for a large number of predictions by using Integral Images, introduced in the Viola-Jones detector [34]. The Integral Image allows calculating event counts for all predictions quickly and to create it, the event volume at the current time step has to only be summed in the channel dimension to get a gray scale image equivalent. We measure an overhead of about $2.12\,\mathrm{ms}$ per call with Python cProfile [5] on a CPU when using our implementation of this memory mechanism. We see a maximum of about 50 bounding boxes stored in memory at a time, equalling $0.8\,\mathrm{kByte}$.

Due to its simplicity, this memory has multiple failure cases. Foremost, it is dependent on the performance of the object detector. False positives could be remembered for a long time because boxes are only deleted from memory when the threshold T_e is crossed. The object could move very slightly, which we could account for by estimating the optical flow from the events but estimating the optical flow is non-trivial [1,2,35,36]. Occluding objects and egomotion generate events which clears the memory, although some objects might not have moved. Despite its simplicity, we will see in Sec. 3.5, that this memory is highly effective on a simulated dataset.

3.5. RM-MNIST

So far, we have observed that there is a skew towards objects with few events in the 1 Mpx Dataset (Sec. 3.1) and proposed methods to improve training and inference (Secs. 3.3 and 3.4). We investigate the effectiveness of these methods first on a smaller dataset where almost $50\,\%$ of labels are empty.

We simulate the frame- and event-based Random-Movements-MNIST, or RM-MNIST dataset, with the MNIST digits 3 and 6 moving in random directions on a white background. Directions and speeds are drawn at random, but digits always move in a straight line. As digits randomly stop, no events are generated during these times. We simulate 50 train, 6 validation and 10 test sequences of 5 seconds each at a resolution of 1280x720 pixels using the Event Camera Simulator [25]. Frames are rendered at $1\,\mathrm{Hz}$ to ensure that the simulation is accurate. Frames and labels are saved at $60\,\mathrm{Hz}$ to match the rate of the 1 Mpx Dataset.

In Fig. 4 an exemplary sequence is shown. As the number comes to a stand-still, it is not visible anymore in the event representation, but can be clearly identified in the frames. As we simulate the dataset such that all digits stop at some point, almost $50\,\%$ of labels do not contain any

events (Fig. 4). If the object is moving on the other hand, a lot of events are generated because of the high contrast between the black digit and white background. When filtering, we select all objects with non-zero event count. In Fig. 5, we compare the performance on the unfiltered dataset ('no filter') to the performance when filtering the train set ('filter train'), filtering the train set and using the memory introduced in Sec. 3.4 and filtering train and test set ('filter train and test'). When using the memory, we tune the thresholds on the validation set. All results are reported on the test set.

For all experiments, we train the same network on frames and events to compare the performance. An ideal event object detector should match the performance of a frame-based detector. As expected, we see a large difference in mAP between frames and events of 0.59 on the full dataset, because the labels with zero event counts are not detected. When filtering only the training data, we achieve an absolute mAP increase of 0.04. This shows that learning improves when filtering out the non-informative labels, even in this simple case. Adding our memory improves results drastically and is almost as good as the frame detector (0.824 ± 0.053 vs 0.842 ± 0.026). As a control experiment, we also test the performance when filtering train and test set. As expected, both frame and event detectors achieve a similar mAP.

To compare the performance of our memory to an architecture with learnable memory, we trained a network similar to the one in [26]. We add a block of three ConvLSTM layers after the backbone. An SSD head is connected to the last ConvLSTM layer with the same configuration as our SSD network. To learn temporal dependencies, we unroll the network for four time steps during training. As we have to pass a vector of size (batch, 10, 360, 360) to each input, we cannot unroll for more than four time steps due to the GPU memory, although it would help the training process. As with the other experiment, we first evaluate on the frames and measure $(0.769 \pm 0.026)\,\mathrm{mAP}$, comparable to our single-frame detector. When using events, the performance drops to $(0.263 \pm 0.061)\,\mathrm{mAP}$, *i.e.* the detector does not manage to learn to memorize the location in the absence of events. This could be due to the fact that the times, where the object stands still can span several seconds. To see how these results translate to the 1 Mpx Dataset, see Sec. 3.6.

3.6. Results on the 1 Mpx Dataset

We have seen in the previous sections, that it is possible to build a highly effective memory for bounding boxes and that filtering objects with a low event count out of the training data improves mean average precision. In this section, we want to see how the results from RM-MNIST translate to the large, real-world 1 Mpx Dataset.

Our network is similar to the one described in Sec. 3.5, but we replace the ResNet-18 backbone with a ResNeXt-50 backbone. The ResNeXt architecture improves on the original ResNet by splitting each residual block into multiple, independent blocks that can learn different transformations. We believe that this architecture is particularly suited for the event volume representation, as different transformations in one block can learn to focus on different time slices and therefore the time information can be kept in the channels even in deeper parts of the network. The whole test set is used to generate the final results.

architecture	mAP
single-frame (SF)	$0.180\,03 \pm 0.000\,38$
SF + dataset filtering (DF)	$0.204\,45 \pm 0.000\,85$
SF + DF + memory	$0.213\,95 \pm 0.000\,78$
SF + ConvLSTM	0.1604 ± 0.0042
RED [26]	0.45

Table 1. Results on the 1 Mpx Dataset. Our dataset filtering and memory improve over single-frame and ConvLSTM, but do not surpass RED. Mean and error of the mean over 2 runs.

Results are shown in Fig. 6 and Tab. 1. We plot a few samples with different event counts from the validation set to understand how to choose the event count threshold for the training filter. We see that for all three classes, labels with fewer than 100 events are unidentifiable. In the 200-500 events regime, outlines can already be identified but no details. For more than 1000 events, some details can be identified, especially in the pedestrian class. Objects with more than 5000 events are as detailed as an image.

In a first experiment, we train and test on a filtered version, shown in Fig. 6b on the upper half. Already filtering out all labels with 0 events increases results by 0.045, with minor improvements when filtering at 10 or 100 events. Filtering 1000 or $10\,000$ events further increases mAP on the filtered test set. As filtering at these high event counts inevitably leads also to filtering small bounding boxes, the increase in mAP compared to smaller thresholds can therefore not only be attributed to the increased identifiability of the objects, but also to the objects being larger on average, and therefore easier to detect.

Based on the visual inspection in Fig. 6a and the results when training on filtered versions with different thresholds in Fig. 6b, we chose 100 events as our threshold for further experiments. When training on the dataset with samples with less than 100 events filtered out, but testing on the unfiltered test set, we see an increase of $0.024\,\mathrm{mAP}$. Adding the memory presented in Sec. 3.4 additionally increases mAP by 0.010. The thresholds we chose are $T_\mathrm{p} = 0.02$, $T_\mathrm{e} = 0.05$, and we deactivate the IoU threshold (set to -1), which we found by random search on $50\,\%$ of the validation set. The increase in mAP is not as high as it was on

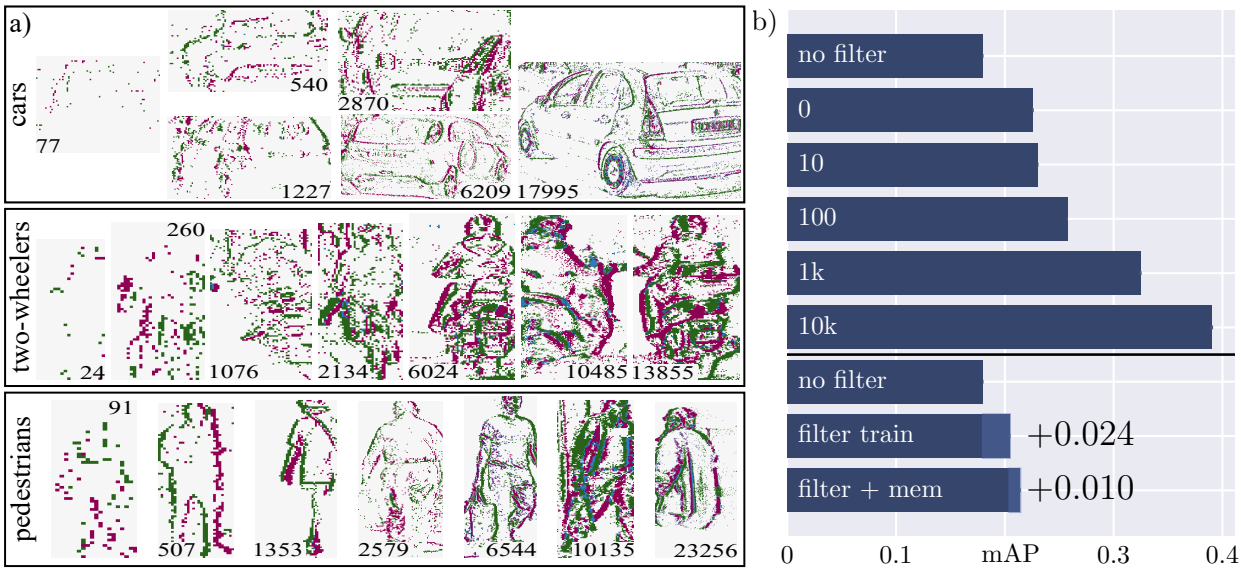

Figure 6. Results on 1 Mpx Dataset. **a)** Samples for the three classes car, two-wheeler and pedestrian for different event counts in the validation set. Labels below 100 events are not identifiable. There is no difference in detail for objects above 5000 events. **b)** mAP for different training configurations. The top half shows performance when filtering train and test with a certain event count threshold. Jumps in mAP are at 0, 1000 and 10 000 events. mAP increases when training on a filtered version with all labels below 100 events removed. Adding the memory further increases mAP.

the RM-MNIST dataset. We discuss different failure cases of our memory in the appendix, with the conclusion that a main driving factor is the performance of the single-frame detector: If the detector produces a lot of false positives, the memory tends to increase the effect by memorizing the false positives over multiple time steps.

The best result in [26] is 0.43 mAP, approximately two times our best result. On the other hand, their single-frame detector Events-RetinaNet is on par with our single-frame detector without memory and filtering (0.18 mAP vs $(0.180\,03 \pm 0.000\,38)$ mAP). This begs the question, if the difference comes only from the memory mechanism (ConvLSTM vs our algorithm). To this end, we also train a network with three ConvLSTM layers after our ResNeXt-50 backbone to get (0.1604 ± 0.0042) mAP, even worse than the results of the single-shot detector. This could be due to our small rollout size of four time steps which leads to worse predictions from non-zero initial states.

4. Conclusion

In this paper we have identified a problem when frame information is used to create labels for event camera data, at the example of the 1 Mpx Dataset. Labelled objects can appear with only few events, because the object and/or the event camera are not moving. After a careful analysis of the event distribution of the large, real-world 1 Mpx Dataset, we simulate a frame- and event-based dataset (RM-MNIST), to investigate the effect of empty labels during training and validation. Our findings show that filtering labels during training improves performance. During validation, a novel bounding box memory mechanism is used to remember bounding boxes of empty labels over a theoretically infinite amount of time, boosting validation mAP. For long sequences without events, this memory improves significantly over a ConvLSTM network with built-in memory.

These insights are translated to the 1 Mpx Dataset, where we filter labels at different event thresholds to answer the question how many events make an object: Approximately 100 events. Filtering samples below 100 events increases mAP on the non-filtered test set. Adding our novel bounding box memory further increases mAP, although not as significantly as for RM-MNIST. A ConvLSTM SSD network similar to RED that we train ourselves is worse than our single-frame architecture. By releasing our code upon publication, we invite other researchers to try to improve upon the single-frame or the ConvLSTM approach to set a new state-of-the-art on the 1 Mpx Dataset.

5. Acknowledgments

The authors would like to acknowledge the financial support of the CogniGron research center and the Ubbo Emmius Funds (University of Groningen).

References

[1] Ryad Benosman, Sio-Hoi Ieng, Charles Clercq, Chiara Bartolozzi, and Mandyam Srinivasan. Asynchronous frameless event-based optical flow. *Neural Networks*, 27:32–37, 2012. 6

[2] Tobias Brosch, Stephan Tschechne, and Heiko Neumann. On event-based optical flow detection. *Frontiers in Neuroscience*, 9, 2015. 6

[3] Shoushun Chen and Menghan Guo. Live demonstration: CeleX-V: A 1M pixel multi-mode event-based sensor. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1682–1683, 2019. 2

[4] Pierre de Tournemire, Davide Nitti, Etienne Perot, Davide Migliore, and Amos Sironi. A large scale event-based detection dataset for automotive. *arXiv preprint arXiv:2001.08499*, 2020. 2, 3

[5] Python Docs. Python cProfile. Checked 23/02/2022, 2022. 6

[6] Zaid El Shair and Samir Rawashdeh. High-temporal-resolution event-based vehicle detection and tracking. *Optical Engineering*, 62(3):031209, 2022. 2

[7] Thomas Finateu, Atsumi Niwa, Daniel Matolin, Koya Tsuchimoto, Andrea Mascheroni, Etienne Reynaud, Pooria Mostafalu, Frederick Brady, Ludovic Chotard, Florian LeGoff, Hirotsugu Takahashi, Hayato Wakabayashi, Yusuke Oike, and Christoph Posch. 5.10 a 1280 × 720 back-illuminated stacked temporal contrast event-based vision sensor with 4.86μm pixels, 1.066GEPS readout, programmable event-rate controller and compressive data-formatting pipeline. In *2020 IEEE International Solid- State Circuits Conference - (ISSCC)*, pages 112–114, 2020. 2

[8] Benoit Frenay and Michel Verleysen. Classification in the presence of label noise: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5):845–869, 2014. 5

[9] Daniel Gehrig and Davide Scaramuzza. Pushing the limits of asynchronous graph-based object detection with event cameras. *arXiv preprint arXiv:2211.12324*, 2022. 2

[10] Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza. DSEC: A stereo event camera dataset for driving scenarios. *IEEE Robotics and Automation Letters*, 2021. 2, 3

[11] Mathias Gehrig and Davide Scaramuzza. Recurrent vision transformers for object detection with event cameras. *arXiv preprint arXiv:2212.05598*, 2022. 2

[12] Github. CoCo python api. Checked 01/02/2022, 2015. 5

[13] Menghan Guo, Jing Huang, and Shoushun Chen. Live demonstration: A 768× 640 pixels 200Meps dynamic vision sensor. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–1. IEEE, 2017. 2

[14] Yuhuang Hu, Jonathan Binas, Daniel Neil, Shih-Chii Liu, and Tobi Delbruck. DDD20 end-to-end event camera driving dataset: Fusing frames and events with deep learning for improved steering prediction. In *The 23rd IEEE International Conference on Intelligent Transportation Systems*, 2020. 2, 3

[15] Seijoon Kim, Seongsik Park, Byunggook Na, and Sungroh Yoon. Spiking-YOLO: spiking neural network for energy-efficient object detection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 11270–11277, 2020. 2

[16] Alexander Kugele, Thomas Pfeil, Michael Pfeiffer, and Elisabetta Chicca. Hybrid SNN-ANN: Energy-efficient classification and object detection for event-based vision. In Christian Bauckhage, Juergen Gall, and Alexander Schwing, editors, *43rd DAGM German Conference (GCPR)*, volume 13024, pages 297–312. Springer, 2021. 2

[17] Chenghan Li, Christian Brandli, Raphael Berner, Hongjie Liu, Minhao Yang, Shih-Chii Liu, and Tobi Delbruck. Design of an RGBW color vga rolling and global shutter dynamic and active-pixel vision sensor. In *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 718–721, 2015. 2

[18] Chenghan Li, Luca Longinotti, Federico Corradi, and Tobi Delbruck. A 132 by 104 10μm-pixel 250μw 1kefps dynamic vision sensor with pixel-parallel noise and spatial redundancy suppression. In *2019 Symposium on VLSI Circuits*, pages C216–C217. IEEE, 2019. 2

[19] Zichen Liang, Hu Cao, Chu Yang, Zikai Zhang, and Guang Chen. Global-local feature aggregation for event-based object detection on eventkitti. In *2022 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 1–7, 2022. 2

[20] Zichen Liang, Guang Chen, Zhijun Li, Peigen Liu, and Alois Knoll. Event-based object detection with lightweight spatial attention mechanism. In *2021 6th IEEE International Conference on Advanced Robotics and Mechatronics (ICARM)*, pages 498–503, 2021. 2

[21] P. Lichtsteiner, C. Posch, and T. Delbruck. A 128×128 120 db 15μs latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-State Circuits*, 43(2):566–576, Feb 2008. 1, 2

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 5

[23] Bingde Liu. Motion robust high-speed light-weighted object detection with event camera. *arXiv preprint arXiv:2208.11602*, 2022. 2

[24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. *Lecture Notes in Computer Science*, pages 21–37, 2016. 2

[25] Elias Mueggler, Henri Rebecq, Guillermo Gallego, Tobi Delbruck, and Davide Scaramuzza. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM. *The International Journal of Robotics Research*, 36(2):142–149, Feb 2017. 6

[26] Etienne Perot, Pierre de Tournemire, Davide Nitti, Jonathan Masci, and Amos Sironi. Learning to detect objects with a 1 megapixel event camera. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 16639–16652. Curran Associates, Inc., 2020. 1, 2, 3, 5, 7, 8

[27] Prophesee. Prophesee automotive dataset toolbox. Checked 01/02/2022, 2020. 3

[28] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. Events-to-video: Bringing modern computer vision to event cameras. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 5

[29] 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 (CVPR)*, June 2018. 1

[30] Bongki Son, Yunjae Suh, Sungho Kim, Heejae Jung, Jun-Seok Kim, Changwoo Shin, Keunju Park, Kyoobin Lee, Jinman Park, Jooyeon Woo, Yohan Roh, Hyunku Lee, Yibing Wang, Ilia Ovsiannikov, and Hyunsurk Ryu. 4.1 a 640×480 dynamic vision sensor with a 9μm pixel and 300Meps address-event representation. In *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, pages 66–67, 2017. 2

[31] Yunjae Suh, Seungnam Choi, Masamichi Ito, Jeongseok Kim, Youngho Lee, Jongseok Seo, Heejae Jung, Dong-Hee Yeo, Seol Namgung, Jongwoo Bong, Sehoon Yoo, Seung-Hun Shin, Doowon Kwon, Pilkyu Kang, Seokho Kim, Hoonjoo Na, Kihyun Hwang, Changwoo Shin, Jun-Seok Kim, Paul K. J. Park, Joonseok Kim, Hyunsurk Ryu, and Yongin Park. A 1280×960 dynamic vision sensor with a 4.95μm pixel pitch and motion artifact minimization. In *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5, 2020. 2

[32] Gemma Taverni, Diederik Paul Moeys, Chenghan Li, Celso Cavaco, Vasyl Motsnyi, David San Segundo Bello, and Tobi Delbruck. Front and back illuminated dynamic and active pixel vision sensors comparison. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 65(5):677–681, 2018. 2

[33] Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge Belongie. Learning from noisy large-scale datasets with minimal supervision. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6575–6583, 2017. 5

[34] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I, 2001. 6

[35] Alex Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. EV-FlowNet: Self-supervised optical flow estimation for event-based cameras. *Robotics: Science and Systems XIV*, Jun 2018. 6

[36] 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 (CVPR)*, June 2019. 6