Low Cost and Latency Event Camera Background Activity Denoising

Shasha Guo and Tobi Delbruck, *Fellow, IEEE*

Abstract—Dynamic Vision Sensor (DVS) event camera output includes uninformative background activity (BA) noise events that increase dramatically under dim lighting. Existing denoising algorithms are not effective under these high noise conditions. Furthermore, it is difficult to quantitatively compare algorithm accuracy. This paper proposes a novel framework to better quantify BA denoising algorithms by measuring receiver operating characteristics with known mixtures of signal and noise DVS events. New datasets for stationary and moving camera applications of DVS in surveillance and driving are used to compare 3 new low-cost algorithms: Algorithm 1 checks distance to past events using a tiny fixed size window and removes most of the BA while preserving most of the signal for stationary camera scenarios. Algorithm 2 uses a memory proportional to the number of pixels for improved correlation checking. Compared with existing methods, it removes more noise while preserving more signal. Algorithm 3 uses a lightweight multilayer perceptron classifier driven by local event time surfaces to achieve the best accuracy over all datasets. The code and data are shared with the paper as DND21.

Index Terms—Dynamic vision sensor, background activity noise, denoising, hardware-friendly, ROC

1 Introduction

Neuromorphic event-based cameras ("silicon retinas") are inspired by the remarkable abilities of biological eyes. The first designs from almost 30 years ago [2] have matured to a stage where there are now commercially available event cameras. Event cameras come in various types [3], but all commercial types detect brightness changes using a Dynamic Vision Sensor (**DVS**) pixel [4]. The DVS camera output is a stream of binary signed *ON* and *OFF* brightness change events. Events signify that the log intensity changed by a critical threshold temporal contrast. The high dynamic range and sparse sub-millisecond output of DVS enables vision under poor lighting with quick responses [3].

The output of event cameras includes various types of noise including event jitter and quantization noise [4], [5], [6], [7]. Here we are concerned with Background Activity (**BA**) noise [4], [5], [8], [9]. These BA noise events come from pixels even in the absence of any scene activity and are thus noninformative. Under dim lighting, BA noise dominates the DVS output.

Robots and surveillance systems that use DVS cameras are increasingly driven by DVS activity event output, so their power consumption is proportional to activity [3], [10]. By reducing BA noise at the camera output, we can reduce the quiescent power consumption by orders of magnitude, particularly under low light conditions where noise increases dramatically. Existing denoising methods are not effective at reducing such high BA noise levels without also removing signal events.

Fig. 1 shows an example of this BA noise. A DVS looks down on the race track and signal events are generated by the moving car. The BA noise events shown in Fig. 1B obscure the signal events caused by the movement of the car. The DVS events are used to track and control the car [1]. Tracking the car using the original noisy Fig. 1B output results in noisy car velocity estimates, because the BA disturbs the tracking model updates, but denoising produces the Fig. 1C output. The only events left to process are those produced by the moving car, which makes tracking the car position and velocity very easy to initiate and precise to estimate. It also reduces the quiescent event data rate and hence the computational effort to practically zero when the car is not moving.

Prior work relied on simplified models of DVS pixel dynamics to predict idealized events and developed methods to retain only these events. The fundamental problem with this approach is that it discards informative events that do not follow the idealized model. By contrast, we developed DVS BA noise models from detailed measurements under low and high lighting conditions (Section 3), and devised a novel approach (Section 5) where we inject this accurately-modeled BA noise to desired clean DVS data. Because we know what is *signal* and what is *noise*, we can quantify the denoising accuracy.

Prior work quantified denoising using a single discrimination threshold, which prevents fair comparison between algorithms of denoising accuracy because any algorithm can filter out more noise simply by increasing the discrimination threshold. By contrast, we quantity the denoising accuracy by using a Receiver Operating Characteristic (**ROC**) curve that plots the True Positive Rate (**TPR**) and False Positive Rate (**FPR**) across discrimination threshold. Clean DVS events are

- *Shasha Guo is with the College of Computer Science and Technology, National University of Defense Technology, Changsha, Hunan 410073, China. E-mail: guoshasha13@nudt.edu.cn.*
- *Tobi Delbruck is with the Institute of Neuroinformatics, UZH-ETH, 8057 Zurich, Switzerland. E-mail: tobi@ini.uzh.ch.*

Manuscript received 5 Feb. 2021; revised 9 Aug. 2021; accepted 9 Feb. 2022. Date of publication 23 Feb. 2022; date of current version 5 Dec. 2022.
S. Guo's visit to Zurich was supported by Chinese Scholarship Council and her PhD advisors Weixia Xu and Lei Wang.
(Corresponding author: Shasha Guo.)
Recommended for acceptance by R. Cucchiara.
Digital Object Identifier no. 10.1109/TPAMI.2022.3152999

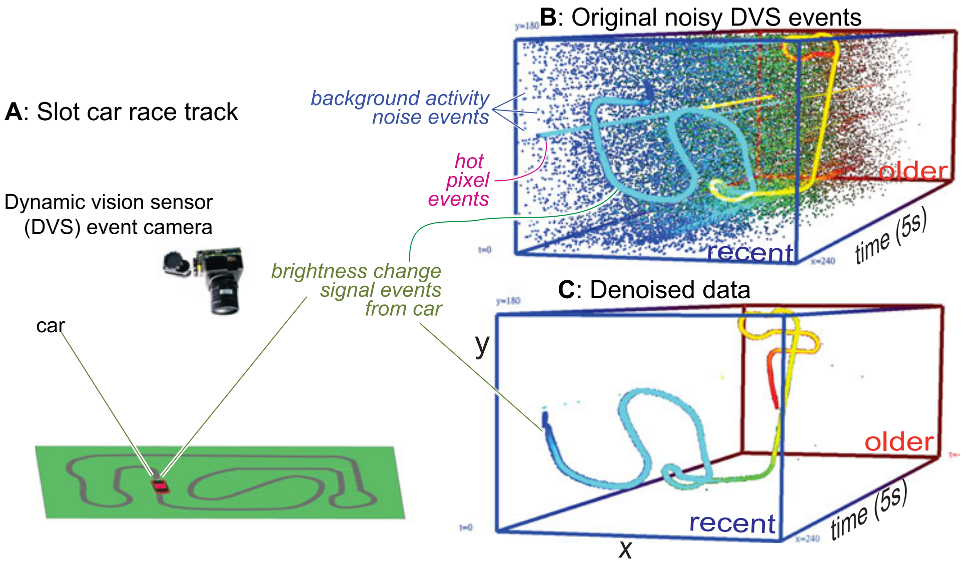
A: Slot car race track
Dynamic vision sensor (DVS) event camera
car
B: Original noisy DVS events
background activity noise events
hot pixel events
brightness change signal events from car
older
recent
time (5s)
C: Denoised data
y
x
older
recent
time (5s)
y=180
t=0
x=240

Fig. 1. Example of denoising event camera output. **A**: Slot car race track setup [1]. **B**: Sample DVS data from moving car corrupted by background activity noise events, including from “hot” pixels that fire at abnormally high rates all the time (Section 2.1.2). **C**: DVS data after denoising (using the STCF noise filter from Section 4.2 with $k = 4$ and $\tau = 10\text{ms}$).

generated from a simulated scene using a realistic but BA-free model of the DVS pixel, and then noise, either realistic synthetic BA noise or separately-recorded real BA noise (Section 3), is added to the signal DVS data. This process allows us to measure the effectiveness of denoising using a ROC curve. The ROC method allows a fair comparison of denoising ability independent of discrimination threshold.

Denoising is most beneficial for system power efficiency when done as early as possible, and recently reported event cameras can produce event rates ranging up to 1 GHz [6], [7], [11]. To denoise these high event rate streams in real time, we developed 3 new algorithms (Section 4) that have a small computational complexity and latency but good denoising accuracy.

Section 4.1 proposes a new pair of lightweight denoising filters called Fixed Window Filter (**FWF**) and Double Window Filter (**DWF**). They have a tiny memory footprint, but effectively filter out noise in sparse surveillance applications with stationary cameras.

Existing low-cost filters are not effective at denoising under high noise conditions in scenes produced by moving cameras. Section 4.2 proposes the SpatioTemporal Correlation Filter (**STCF**). This filter better preserves slowly moving features than previous methods, while reducing BA even with high noise rates.

For preserving the most possible signal events while discarding the most noise events, Section 4.3 proposes a lightweight Multilayer Perceptron denoising Filter (**MLPF**) that exploits structural cues in the local spatiotemporal window of past events to further increase the TPR and reduce the FPR. It is more than 10^4 times cheaper than prior machine learning denoising methods.

We experimentally validate theoretical predictions of the FWF/DWF and STCF false-positive prediction rates (Sections 4.1.1 & 4.2.1). We present several new datasets for stationary and moving camera applications of DVS (Section 5.1), which we use to characterize denoising accuracy (Section 6). Our Supplementary Material (**SM**) includes additional methods and experiments.

In summary, our paper contributes the first detailed observations of DVS noise under low and high light intensities, and we use accurate models of this BA noise to develop more effective denoising methods. Our approach is unique by applying problem inversion: Previous work modeled ideal DVS and called anything not ”ideal” as noise; we acknowledge the complexity of DVS pixel dynamics and instead model the measured characteristic of BA noise, so we can then add this noise to clean DVS recordings to see how well we can remove it. We also present the first use of the ROC method to characterize denoising, to avoid the shortcomings of the previous work’s arbitrary choice of signal versus noise discrimination threshold.

2 Background

Fig. 2A shows how the DVS pixel asynchronously detects brightness changes, which are changes in the logarithmic intensity, exceeding a specified DVS event threshold. Each event consists of a (t, x, y, p) 4-tuple, where t is the timestamp in microseconds, x and y are the pixel address, and p is the signed +1 or -1 *polarity* of the brightness change, indicating *ON* or *OFF* events. Fig. 2B shows the DVS pixel circuit that generates these events. The main sources that create BA noise events are shown in red. They contribute BA events and variability to the pixel response characteristics. BA noise can be controlled by adjusting the DVS threshold and pixel bandwidth [12], but low illumination conditions always increase noise [4], [5], [7].

2.1 Using Spatiotemporal Correlation to Filter Out Noise

The most widely used BA denoising method for DVS is the Background Activity Filter (**BAF**) [13]. It is a Nearest Neighbor (NNb)-based filter which admits only events with local spatiotemporal correlation. Other NNb filters pass events

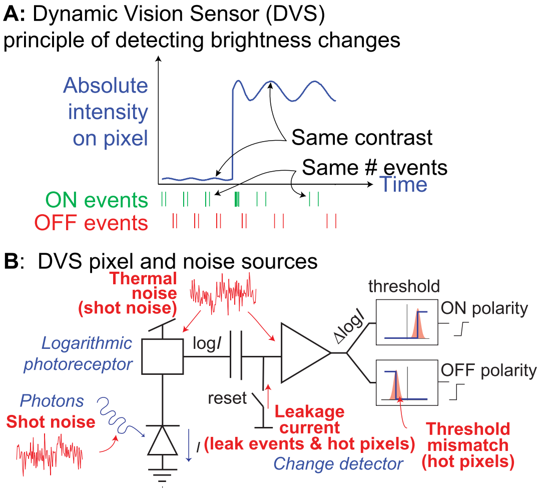
A: Dynamic Vision Sensor (DVS)
principle of detecting brightness changes
Absolute intensity on pixel
Same contrast
Same # events
Time
ON events
OFF events
B: DVS pixel and noise sources
Thermal noise (shot noise)
threshold
ON polarity
Logarithmic photoreceptor
log$I$
Δlog$I$
OFF polarity
Photons
Shot noise
reset
Leakage current
I (leak events & hot pixels)
Threshold mismatch (hot pixels)
Change detector

Fig. 2. **A**: DVS principle of detecting brightness change events [4]. **B**: Simplified DVS pixel architecture. The most important noise and mismatch variability sources (which cause *hot pixels*) are shown in red (see Section 2.1.2).

with alternating event polarity, or by enforcing a refractory period, but [14] showed that BAF is better at rejecting noise while preserving signal.

The BAF decides that an event e is a 'signal' if any past event has occurred within the spatiotemporal volume defined by the immediate nearest neighbors in space and the event timestamp condition $t_{\text{NNb}} - t_e < \tau$, where τ is the filter's *correlation time threshold*. If the condition is met, the event is classified as a *signal* event. Otherwise, it is classified as a *noise* event and is discarded.

Fig. 4A illustrates how t_e is the event timestamp and the t_{NNb} are the timestamps from the NNb pixels (x, y) that are defined by the conditions $|x_e - x| \leq 1$ and $|y_e - y| \leq 1$ where (x_e, y_e) is the address of the e event. The t_{NNb} come from a 2D Timestamp Image (**TI**) of the most recent event timestamps at each pixel[1]. Since τ typically is set between 1–100 ms, the TI could be quantized to 1 ms.

2.1.1 Increasing Neighborhood by Subsampling

By right-shifting the x and y event addresses by b bits before using them, each TI pixel holds the most recent event of $2^b \times 2^b$ DVS pixel blocks. Subsampling reduces TI memory by a factor of 2^b and enables weakly correlated signals over extended distances to pass through the filter with practically no additional computational cost.

2.1.2 Hot Pixel Filtering by Self-Exclusion

A so-called *hot pixel* has an abnormally high rate of noise events; see Fig. 1B for an example and Section 3 for statistics. They are caused by transistor and device mismatch. Hot pixel events are BA activity should not pass the filter but in bright lighting conditions they can form a large fraction of the BA output. If the neighborhood included the pixel itself, such hot pixels would pass their events through if they fire faster than the threshold correlation time. By excluding self-correlation, BAF automatically removes them. SM Section D.1.1 shows experimental results of self-exclusion on hot pixels.

TABLE 1
DVS Event Camera Denoising Algorithms

Filter	**Mem(#)**[a]	**Latency**[b] **Op/event**	**Time/event**
		Proposed new algorithms (Section 4)	
FWF/DWF Section 4.1	L[h]	$\approx 5L$[g]	$70 + 4.4L$ns[c]
STCF Section 4.2	N^2	≈ 25	44 ± 6ns[c]
MLPF Section 4.3	N^2 + MLP[j]	MLP (≈2 k)[j]	1250 ± 60ns[c]
		Existing hardware denoising (Section 2.2.2)	
BAF [13] Section 2.1	N^2	11	28 ± 3ns[c]
ONF [8] Fig. 4, Section 2.2.2	$4N$	40	34 ± 4 ns[c]
HashHeat [15]	L[h]	40	≈ 120ns[c]
EBBI(NN)OT [16], [17]	N^2	Binary image NNb median filter	
Ojeda [18]	N^2	Binary image NNb edge filtering	
Bose [19]	N^2	Binary image median filter ASIC	
Linares [20] BAF	N^2	Open BAF FPGA impl.	
		Offline handcrafted algorithms (Section 2.2.1)	
Feng [21]	$> N^2$	2 step density on event frames	
Afshar [22]	N^2	NA	2μs
Wu [23]	$\gg N^2$	PUGM w/ ICM	280μs
IE [24]	N^2	RANSAC optic flow consistency	
FSAE [25]	$\gg N^2$	LP optic flow consistency	
EV-Gait [26]	$\gg N^2$	LP optic flow consistency	
GEF [27]	NA	CM optic flow + APS filtering	
DBA [28]	NA	TI RANSAC optic flow + KNN	
		Offline learned algorithms (Section 2.2.1)	
FEAST [29]	NA	NA	NA
EDnCNN [30][e]	> 48M	167 M	30 ms
EventZoom [31]	NA	NA	APS+DVS 3D-UNet

[a] *Memory cells for $N \times N$ pixel DVS.*
[b] *Assuming operations and memory access are serial.*
[c] *AMD Ryzen 7 3700X 8-Core 3.59@3.99GHz, NVidia GeForce RTX2080 GPU, Windows 10 Pro 20H2 19042.685, Java 1.8.0.111, jAER 1.9.1, TensorFlow 1.5. MLPF computed in GPU batches of up to 4k events.*
[e] *6-layer CNN for each event. Mem/Op/runtimes measured by us.*
[g] *FWF/DWF operations over window L can be concurrent in hardware.*
[h] *Using a window memory of length L.*
[j] *The MLPF has 98-20-1 neurons.*

2.2 Relevant Work

This section reviews relevant work on DVS denoising[2]. Table 1 summarizes the algorithm implementation metrics. The memory and latency are relevant for software and hardware logic implementations. Latency and throughput are related: Low latency is desirable because it enables processing high event rates. Logic implementations can compute many operations in parallel, resulting in lower latency than listed. Most of the 'offline' algorithms lack reported cost and latency measurements.

2.2.1 Offline Denoising

A class of denoising algorithms considers event density. Feng *et al.* [21] proposed a 2-step event density matrix denoising method using fixed time windows. They offline estimated the discrimination threshold from the static parts of each scene and compared several BA denoising algorithms. The density concept was also used in Afshar's PhD thesis work [22], to control the discrimination threshold based on the average event density. Its large exponentially decayed neighborhood would be expensive for hardware implementation. The address subsampling of Section 2.1.1 also provides large spatial neighborhoods, but with negligible additional cost. Wu *et al.* [23] propose a Probabilistic

[1] The TI is the same as the Surface of Active Events (**SAE**) [3].

[2] The Event Based Vision pages at http://github.com/uzh-rpg/event-based_vision_resources provides a useful section on denoising.

Undirected Graph Model (**PUGM**) method using Iterated Conditional Modes (**ICM**) to sort the graph for signal and noise events. They hand craft an energy function that is minimized by spatiotemporal locality of events. The expensive and non-deterministic runtime of ICM makes it unsuitable for real-time denoising.

Another class of algorithms considers noise to be DVS events that do not represent an idealized instantaneous logarithmic intensity change. The Inceptive Event (**IE**) of [24] considers DVS output to an edge to consist of a single IE followed by trailing events that signify the edge contrast. These filters seek to label such IEs as particularly informative. IE uses an expensive iterative Local Plane (**LP**) optical flow estimation that only works well on sharp edges. The Filtered Surface of Active Events (**FSAE**) [25], *EV-Gait* [26], Guided Event Flow (**GEF**) [27], and Dynamic Background Activity noise filtering algorithm (**DBA**) [28] filter out events that do not like close to the LP in the TI; GEF combines Active Pixel Sensor (**APS**) frame information, and DBA includes K-nearest-neighbor clustering. However [30] showed that FSAE and IE are usually inferior to other methods and it is likely that *EV-Gait* and GEF have similar behavior, particularly under dim lighting where noise increases dramatically. *EV-Gait* is unique in showing superior denoising of events caused by flickering lighting.

DVS denoising by Deep Neural Network (**DNN**) have also been recently proposed [30], [31]. The Feature Extraction with Adaptive Selection Thresholds (**FEAST**) method from [29] learns features from DVS events using a competitive unsupervised learning method. It develops denoising ability, but the denoising accuracy was not evaluated. Several methods are closely related to IE and FSAE, in that they propose to estimate the ideal (noise free) DVS output and denoise by removing non-predicted events, except that here the prediction DNN are trained from labeled data. e.g., [30] posits (arbitrarily) that the goal of denoising is to maximize the sum of log positive and negative correct classification rates and formulates a theory and measurement of this goal. They combine simultaneously recorded APS gray frames and camera Inertial Measurement Unit (**IMU**) rate gyro output with a handcrafted probabilistic DVS event generation model to obtain a 2D bitmap containing the predicted events over a selected time interval. They trained an Event Denoising CNN (**EDnCNN**) to classify individual events as signals or noise using these binary maps as targets. For about half of their samples, the EDnCNN denoises with about the same quality as far simpler algorithms including BAF. The cost of running EDnCNN is over 150 million operations per event. *EventZoom* [31] also includes denoising with fusion of APS frames and DVS events.

The preceding methods are too costly and slow for in-camera denoising. The actual DVS pixel dynamics are more complex [5] than the event generation models of [26], [27], [30], [31], which also are not statistically validated against real DVS data, and which are not valid across light levels. These methods discard events that are informative but not "ideal" according to their simplified modeling of pixel dynamics.

2.2.2 Hardware Friendly Denoising

Samsung [6], [11] showed two DVS pixel circuits that reduce "leak" BA (see next section) produced in bright conditions, and Li *et al.* [32] demonstrated a pixel circuit that rejects uncorrelated events, but it is costly in pixel area and fill factor. Neither approach controls "shot" BA in dark conditions (see next section). Shot BA can be limited by feedback control of pixel bandwidth and threshold [12] but control actions create noise and it also limits speed and sensitivity.

Most work has focused on off-chip DVS noise filters. Liu *et al.* [33] proposed a mixed-signal BAF noise filter chip consuming 1 mW and flagging signal versus noise with 10ns-latency. Padala *et al.* [34] implemented correlation-based denoising using integrate and fire neurons on the IBM TrueNorth processor, which is a large neuromorphic spiking chip. Guo *et al.* [15] proposed the *HashHeat* filter that uses hashing functions to encode the spatiotemporal information of an event into a list, for checking whether the event is signal or noise. Like the Order(N) Filter (**ONF**) filter that follows, it is effective for sparse DVS data and they demonstrated an Field Programmable Gate Array (**FPGA**) implementation. Lee *et al.* [18] proposed a binary CNN with FPGA implementation that includes binary image vertical/horizontal edge prefiltering. The ONF filter of Khodamoradi *et al.* [8] was the first to propose a noise filter with memory cost scaling less than $O(N^2)$. It included the first formulas for BA filtering probabilities. They implemented this filter in FPGA logic and showed its effectiveness for sparse DVS data. The EBBIOT and EBBINNOT [16], [17] propose low-cost microcontroller-based object detection and tracking with DVS using binary event frames. The binary frames are easily corrupted by BA, so they use a median filter to remove it. The frame-based median filtering is less expensive than event-by-event BAF when the frames have a density greater than 10%. Bose *et al.* [19] report a energy-efficient silicon implementation of this approach. Linares-Barranco *et al.* [20] provides the complete logic design integrating BAF with simple object tracking in FPGA.

A main shortcoming of all previous denoising works is the lack of a ROC comparison. Lacking a discrimination threshold sweep, it is impossible to know how denoising threshold affects TPR and FPR as described further in Section 5.2.

3 Characteristics of DVS BA Noise

To design effective BA denoising algorithms, it is important to understand real BA noise characteristics. Fig. 3 shows measurements of DVS noise under high and low illumination conditions plotted as histograms of ISI between BA events. ISI—inherited from neuroscience—characterize the timing of spike events and their random variability in single pixels and across pixels. We collected this data from an iniVation Davis346 camera [35]. These measurements (and [5], [7, Fig. 5.10.5]) show that BA noise behaves very differently under bright and dark conditions.

When the scene is bright, *leak noise events* dominate the BA (Fig. 3A) [4], [9]. Leak events are created by junction leakage in the DVS change detector reset transistor (Fig. 2B) that charges the floating node and thus creates periodic *ON* activity. Here, the ISI averages about 5 s, corresponding to a rate of about 0.2 Hz/pixel, which is consistent with the nominal rates claimed in Table 1 of [3]. Above some intensity, leak event rate is proportional to light intensity [9]. Although it is periodic, each event caused by e.g. brightness change resets the period, and there is a large Fixed Pattern

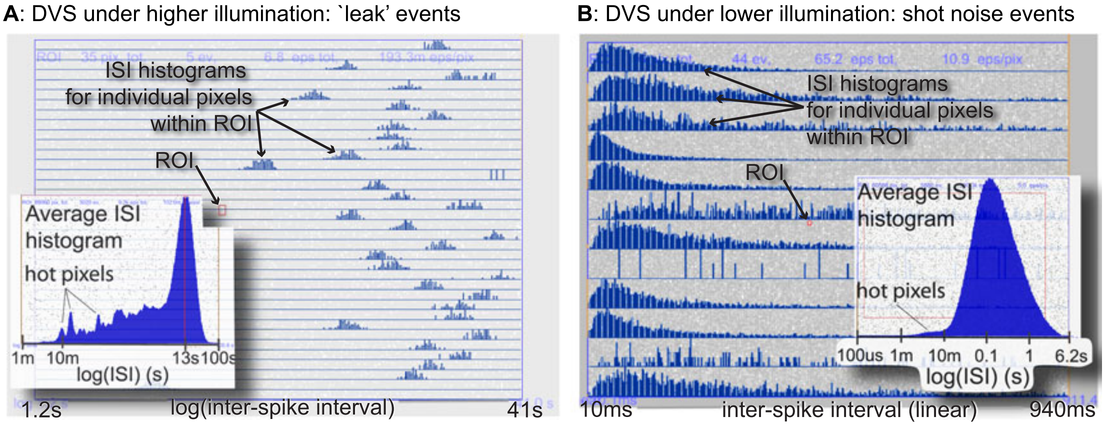

Fig. 3. *Observations of DVS BA leak and shot noise.* A and B both show a collection of histograms of Inter Spike Interval (**ISI**) between DVS events measured for pixels in a Region of Interest (**ROI**). The background image is a 20ms frame of accumulated ON (white) and OFF (black) events. **A**: BA noise under brighter illumination conditions (>100 lx scene illumination) is dominated by *leak noise* events that occur with fixed but noisy and highly variable rates (note logarithmic time scale of ISI histograms). **B**: BA noise under lower illumination conditions (<10 lx scene illumination) is dominated by *shot noise* events that have approximately Poisson statistics (note linear time scale of ISI histograms and exponential decay of waiting times). The insets shows that the average of ISI across all pixels are log-normal with a 1σ Coefficient Of Variation (**COV**) of about 1 decade and a significant tail of 'hot' pixels with higher firing rates (see Section 2.1.2).

Noise (**FPN**) variability across pixels and timing jitter in the periodicity as shown by the Fig. 3A ISI histograms of individual pixels within the ROI; the rates vary by about a decade and the jitter is proportional to the ISI.

When the scene is dark, *shot noise events* dominate the BA (Fig. 3B). Shot noise is caused by fluctuations in the DVS pixel photoreceptor and change detector circuits that randomly exceed the pixel's event threshold. It has approximately Poisson statistics and creates roughly equal *ON* and *OFF* activity. Here, the shot noise rate is about 10 Hz/pixel, which is 50X higher than the leak noise rate. Shot noise also has large FPN with a log normal distribution spread of 0.5 to 1 decade across pixels, as shown by the ISI histograms.

Under low illumination, each pixel's BA ISI distribution is nearly Poisson. A Poisson model is not accurate for leak event BA, but the large pixel-to-pixel variability, jitter, and any visual signal input that causes log intensity change decorrelates the leak BA of neighboring pixels. Sections 4.1.1 and 4.2.1 show that our accurate simulation models of leak and shot noise lead to the same false-positive noise classification rates as real samples of recorded leak and shot noise.

4 New Denoising Methods

Fig. 4 illustrates the noise filters compared in this paper, classified by memory requirements. In addition to characterizing the BAF [13] and ONF [8] filters, we propose and characterize three new filters, FWF/DWF, STCF, and MLPF. All are available as source code[3] and MLPF includes training data and weights.

In Sections 4.1.1 and 4.2.1 we play recorded DVS noise through our FWF and STCF filters to validate equations that predict the FPR. Our SM Sections A&J.3 consider the dual question of how much signal is blocked by denoising.

4.1 Fixed and Double Window Filter (FWF & DWF)

For scenarios with severe limitations on memory and sparse DVS data, the proposed Fixed-Window Filter (**FWF**) checks the spatiotemporal correlation by only relying on a window of the past few events. We define an NNb *fixed window* to refer to the most recent L events that occurred before the current event. Fig. 4C illustrates the FWF operation for $L = 4$. The fixed window only stores the x, y spatial locations of the L events. The timing is implicit in the sequence of the event stream. The *spatial distance* between the current n^{th} event (e_n) and event j in the fixed window is denoted as $D_{n,n-j}$, where $j \in \{1, 2, \ldots, L\}$. The distance is calculated as the Manhattan distance $D_{n,n-j} = |x_n - x_{n-j}| + |y_n - y_{n-j}|$. Thus, for each event, we obtain a L-dimensional distance vector $\boldsymbol{D}$ from the temporal-related events, where $\boldsymbol{D} = (D_{n,n-1}, D_{n,n-2}, \ldots, D_{n,n-L})$. The minimum element $D_{\min}$ of $\boldsymbol{D}$ is the distance between the current event and the event closest to the current event in the L window. If the $D_{\min} < \sigma$, where the filter's *threshold pixel distance* is σ, it is regarded as a signal event, otherwise, as noise. In Fig. 4C, $\sigma = 2$ pixels.

If L is too small, then the window can be overwritten by only noise events. When L is too large, noise events may gain support too and pass through the filter.

4.1.1 Theory and Measurement of Noise Filtering by FWF

Our SM Section G.1 shows that if the DVS produces a total event rate of R_{t} events per second, which consists of a mixture of a noise event rate r_{n} per pixel and a total signal event rate R_s, then the false positive rate $r_{\text{FWF}}(L, \sigma)$ that escapes FWF denoising is given by

$$r_{\text{FWF}}(L, \sigma) = r_{\text{n}}\left(1 - e^{-\frac{r_{\text{n}} L}{R_{\text{t}}}(2\sigma^2 + 2\sigma + 1)}\right). \quad (1)$$

The effect of total event rate R_{t} is to fill the fixed window of length L and control the effective correlation time window; the faster the overall event rate, the briefer the time window.

Fig. 5 plots theoretical and measured $r_{\text{FWF}}(L, \sigma)$ versus L and σ for the Davis346 which has $M = 90\text{k}$ pixels. The measured values are from FWF applied to the experimental noise data of Fig. 3A, which has noise rate $r_{\text{n}} = 0.1\text{Hz}$ and hence total event rate $R_{\text{t}} = r_n \times M = 9\text{kHz}$ (there is no

[3] jaerproject.net: BAF, ONF, FWF/DWF, STCF, MLPF.

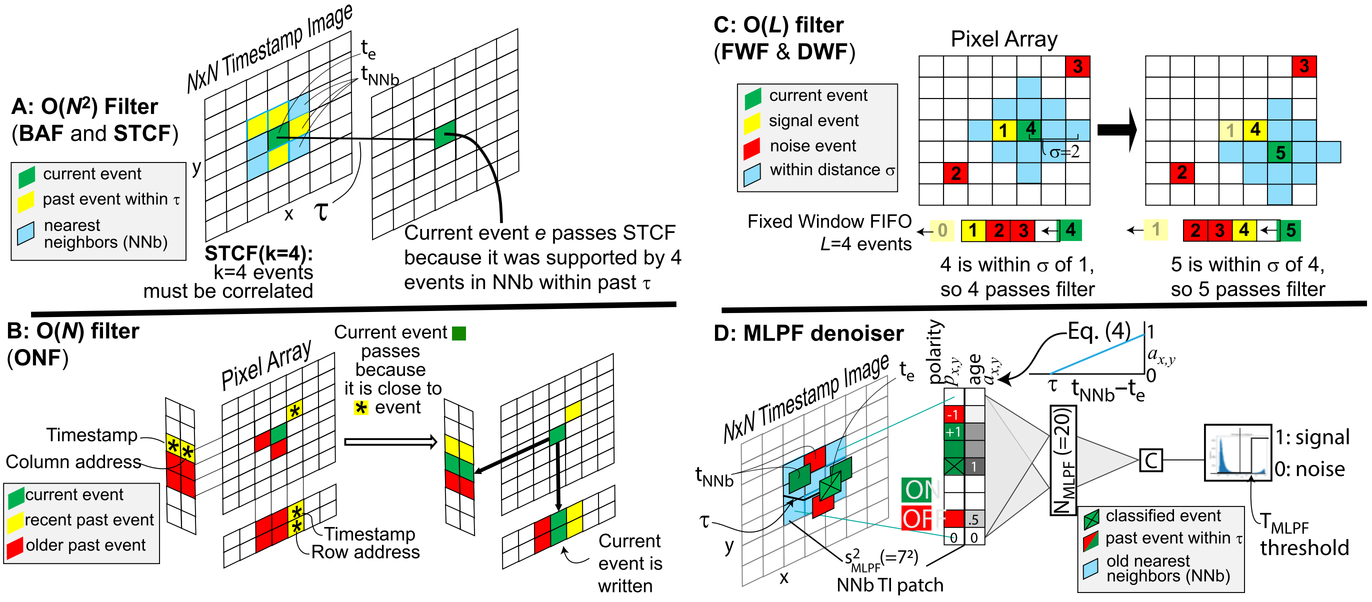

Fig. 4. *Overview of filters compared in this paper.* Green pixels means current event. Yellow pixel means past correlated event. Red pixel means past uncorrelated noise event. **A:** 2D-array filters with O(N^2) memory requirement, like BAF (Section 2) and STCF (Section 4.2). BAF requires only a single event and STCF requires more than one, here for example, four past events to appear within the correlation time τ within the NNb. **B:** 1D-array filter with $O(N)$ memory requirement (ONF). The left sub-figure shows the memory condition when the latest event is generated, that is, the spatial correlated row and column memory cells store the information of two older events and one yellow starred recent past event. Therefore, the current event (green) gets supports from the past yellow event and passes the filter. Then the right of B shows how it updates the filter memory array with the corresponding row and column cells with its information. **C:** Fixed-window Filter with $O(L)$ memory requirement like FWF and DWF (Sections 4.1.1&4.1.2). Events update the fixed memory First In First Out memory (**FIFO**) window of length L. The event order is indicated by number 1-5. 1 means the first event. Events 2 and 3 were blocked because they were too distant in space from event 1. Event 4 is the latest event, and it occupies the last entry of the window. Event 4 passes the filter because it is within distance σ to event 1. When new event 5 arrives, the window discards event 1. Event 5 also passes the filter because it is close to event 4.**D:** MLPF (Section 4.3) that classifies signal versus noise event using the small Multilayer Perceptron (**MLP**) shown here. The NNb TI patch input to the MLP is preprocessed by (4).

signal in this experiment). We see that the measurements very closely match the theory, which means we can use it to predict the effect of denoising on BA within 10% difference.

4.1.2 Double Window Filter (DWF)

A shortcoming of the FWF is its limited memory capacity in its window of L past events: The window is often occupied by noise events that also reject support for next-coming signal events. DWF improves the design of FWF by splitting the fixed window into an $L/2$ long *signal window*, for storing predicted signal events, and an $L/2$ long *noise window*, for storing predicted noise events. The DWF check process is similar to FWF: For each event, we check both windows and get $D_{\min}$. If $D_{\min} < \sigma$ then the current event is predicted as signal and updates the signal window. Otherwise, it updates the noise window.

The rationale for the DWF is that the detected noise events will update the noise window and the signal window can keep the correlated events. Section 6 shows that DWF outperforms FWF in a stationary camera scene with moving objects. For dense moving camera scenes, our experiments show that neither FWF and DWF are effective compared to $O(N^2)$ filters, because the relative movement between the camera and the objects easily makes the spatial distance larger than the threshold σ and thus will decrease the chance of signal events passing through the filter.

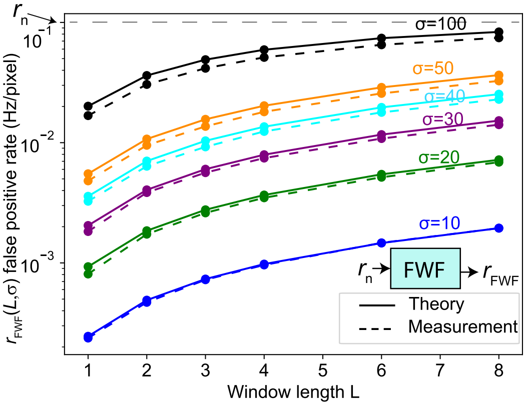

Fig. 5. FWF filter noise rate (false positive) theory of Eq. 1 compared with measurement using data of Fig. 3 A.

4.2 Spatiotemporal Correlation Filter (STCF)

The STCF is a generalization of the BAF that is inspired by [14], [22]. STCF filters out events that lack support from k past events within NNb and the correlation time τ rather than only 1. k can vary from 1 (which is functionally identical to the BAF) up to 8pixels for a neighborhood within 1 pixel.

Fig. 1 demonstrates how STCF can make a noisy DVS output essentially noise-free while completely preserving the interesting signal of the moving car.

4.2.1 Theory and Measurement of Noise Filtering by STCF

It is straightforward to compute the probability of BA noise events passing STCF (see SM Section G.3). If each pixel creates noise with Poisson rate r_n, then the probability p_0 of no

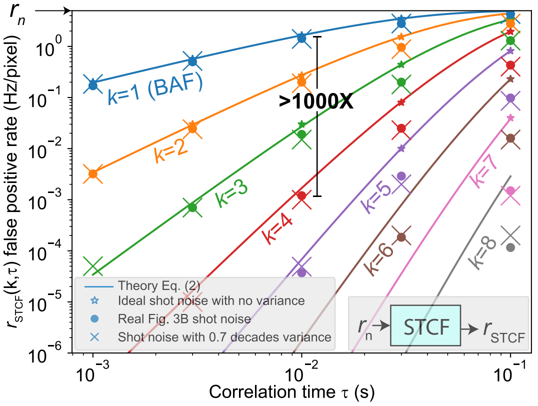

Fig. 6. STCF filter false positive rate of (2) compared with measurements using real and simulated shot noise.

event during the STCF correlation time τ is $p_0 = e^{-r_\text{n}\tau}$, and the probability p_{1+} of one or more events in τ is $p_{1+} = 1 - p_0 = 1 - e^{-r_\text{n}\tau}$. To pass a false positive STCF event with criterion k neighboring events during τ, there must be at least 1 event in each of k or more neighbors. Then the pixel's false positive rate $r_{\text{STCF}}(k,\tau)$ that escapes STCF denoising is

$$r_{\text{STCF}}(k,\tau) = r_\text{n} p_{k+} = r_\text{n} \sum_{x=k}^{8} \binom{8}{k} p_{1+}^{x} p_0^{8-x} \tag{2}$$

$$\rightarrow \begin{cases} r_\text{n} & \text{if } R\tau > 1 \\ r_\text{n}(r_\text{n}\tau)^k & \text{if } r_\text{n}\tau << 1 \end{cases} \tag{3}$$

which has a simple interpretation: When $r_\text{n}\tau << 1$, then $r_\text{n}\tau$ is the probability p_{1+} that a neighboring pixel had an event within the past τ, and k NNbs noise events are required for a false positive.

Fig. 6 plots (2) $r_{\text{STCF}}(k,\tau)$ versus τ and k (solid curves). STCF applied to simulated Poisson shot noise with $r_n = 5\text{Hz}$ and zero FPN matches exactly ($*$ points). However, STCF applied to the experimental noise data of Fig. 3B (• points) does not match; the $r_{\text{STCF}}(k,\tau)$ is less than predicted with the real shot noise input, especially for $\tau > 10\text{ms}$ ($R_n\tau > 1/20$) and $k > 2$. But the Fig. 3B real shot noise has pixel-to-pixel variance. When we repeat the simulation using simulated shot noise with a log normal variance ($\times$ points), we see that $r_{\text{STCF}}(k,\tau)$ matches very closely (• points nearly match $\times$ points). Pixels in the NNb with lower-than-average noise rate have a effect on the probability that reduces the FPR; e.g. if we take $k = 4$ and $p_{1+} = p \times (1 \pm \epsilon)$, then $p_{1+}^4 \approx p^4 \times (1 - 2\epsilon^2)$, so the log normal spread in p decreases the FPR.

The important thing is that increasing k can reduce BA noise by orders of magnitude. Fig. 6 shows that using the same $\tau = 10\text{ms}$, STCF with $k = 4$ reduces BA noise by a factor of more than a thousand compared with BAF.

4.3 Mulitlayer Perceptron Denoising Filter

Since the correlation-based denoising methods only count the number of recent events in the NNb, they cannot detect spatiotemporal structural cues that can be helpful in discriminating signal versus noise events. To study if a lightweight classifier trained on labeled data can achieve better denoising accuracy, we developed a DNN denoiser based on a simple MLP.

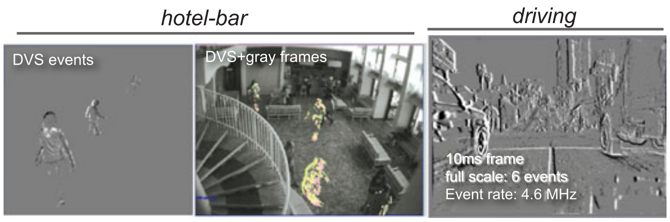

Fig. 7. Samples from the two datasets.

Fig. 4D shows how our MLPF uses a single hidden layer of N_{MLPF} neurons that is driven by a patch of s_{MLPF}^2 pixels of the TI from the NNb around the event that is to be classified. To determine if an event is signal or noise, it is important to know the age of the neighboring events; recent events are important and old events can be disregarded. The $a_{\text{x,y}}$ input channel encodes the age of NNb events as a type of Time Surface (**TS**) [3]: $a_{x,y}$ is calculated from each TI pixel by (4):

$$a_{x,y} = \begin{cases} 0 & \text{if } t_{x,y} < t_\text{e} - \tau \\ 1 - \frac{t_\text{e} - t_{x,y}}{\tau} & \text{otherwise} \end{cases} \tag{4}$$

where t_e is the timestamp of the event e that is to be classified, and $t_{x,y}$ is the timestamp of the most recent previous event stored in the TI. $a_{x,y}$ approaches one for recent events and decays to zero for older events. τ is a time window parameter that we set to 100 ms. (We also tried exponential decay but the accuracy was worse and the computations are more expensive.)

Polarity of past events is also informative, because a moving edge usually makes identical polarities. The $p_{\text{x,y}}$ input channel is formed from the signed NNb polarities, using -1 for OFF, +1 for ON, and 0 for events older than τ as well as those pixels with no event till t_e. The $p_{\text{x,y}}$ of central pixel is from the classified event, to provide the necessary information to determine if the classified event has the same polarity as past events in the NNb.

We evaluated a variety of s_{MLPF} and $N_{\text{MLPF}} = 20, 100$ values, with and without the $p_{\text{x,y}}$ input channel. We also tried perceptrons with only input and output layers (no hidden layer) and MLPs with 2 hidden layers. SM Section F.4 shows that denoising accuracy is good for nearly any choice, but using polarity is helpful. Here we report an accurate but very low cost MLP that uses $s_{\text{MLPF}} = 7$ pixels and $N_{\text{MLPF}} = 20$ neurons, which uses about 2k weights and Multiply-Accumulate (**MAC**) per event. The hidden neurons use a Rectified Linear Unit (**ReLU**) activation function, and the final output C uses a sigmoidal 0-1 activation function. We threshold C against T_{MLPF} to form the final binary classification signaling that the event is S (signal) or N (noise). SM Section F describes the MLPF training and additional experiments.

5 Methodology

To test denoising accuracy, we start with a clean event stream, either by transforming a frame-based video to its event format using *v2e* [5] based on an accurate DVS pixel model, or by aggressively denoising a recording. Next, we insert BA noise events into the event stream. Simulated noise is generated from a frozen log normal FPN rate distribution. The shot noise is drawn from a Poisson process, and the leak

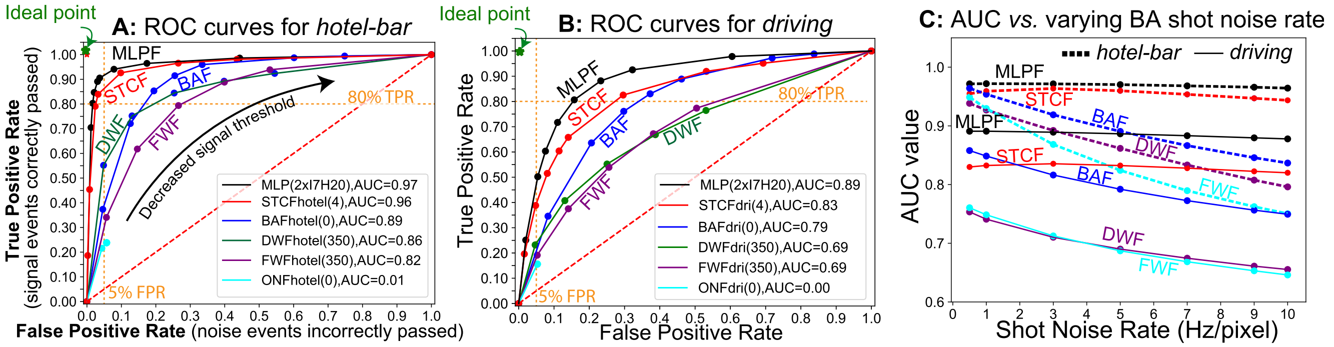

Fig. 8. ROC curves and summarized AUC denoising accuracy for both datasets. **A**: *hotel-bar* ROC curves. **B:** *driving* ROC curves. **C:** AUC for both datasets as a function of BA shot noise rate.

events are generated periodically, with observed event timing jitter. To check theory, we also inject the Fig. 3 prerecorded noise. Then we denoise using one of the algorithms and collect the output, while tracking signal and noise events. Finally, we calculate the desired metrics based on the collected output and input. This algorithm is implemented as the *NoiseTesterFilter* class[4] (see SM Section H).

5.1 Datasets

Hotel-bar (Fig. 7) is a 12 minute 19M event surveillance recording from a DAVIS346 camera looking down on the bar area of a hotel. It represents typical stationary camera surveillance applications of DVS. To measure denoising, we selected a 6.1s 610k event segment with people moving around. The recording was aggressively denoised, so it represents clean DVS data. Its median sparsity[5] is 99%.

Driving (Fig. 7) is a 6s 3.9M event simulated scene captured from the dashboard of a car driving through a city. We generated using *v2e* [5] realistic DVS events representative of applications where a moving DVS produces events from a significant fraction of DVS pixels during any short time. Its median sparsity[5] is 90.5%.

5.2 Denoising Metrics

Denoising performance should consider accuracy, computational complexity, and memory footprint. There is often a cost-accuracy trade-off, requiring striking a balance between cost and accuracy.

We regard the denoising as a binary classification of signal or noise for each event. A *positive classification means that the filter classifies an event as a signal event*. We measure the denoising accuracy by the ROC and Area Under the Curve (**AUC**). The ROC method plots the TPR and FPR over all thresholds, providing a clear picture of the effect on signal and noise discrimination. Ideal denoising achieves zero FPR (noise misclassified as signal) and perfect TPR=1 (signal correctly classified as signal) and AUC=1. Higher AUC means a better classifier. Increasing TPR always increases FPR. The optimum TPR and FPR depends on the application: An always-on surveillance system might favor low FPR noise suppression while a mobile robot might favor high TPR signal retention.

Denoising metrics like Signal to Noise Ratio (**SNR**) depend on discrimination threshold; an infinite SNR is easily produced by aggressive denoising, but it would also remove many informative signal events. The AUC is likewise an arbitrary measure of accuracy, but unlike metrics that combine TPR and FPR at a selected discrimination threshold, the AUC removes bias by a particular choice of threshold.

6 Denoising Accuracy Results

Using the methodology, we measured the ROC curves for the five filters for the *hotel-bar* and *driving* scenes. We used synthetic shot noise of 5 Hz/pixel with a COV of 0.5 decades FPN. To generate each ROC curve, we swept the filter threshold parameter. For BAF, ONF, and STCF we swept the correlation time τ. For FWF/DWF, we first swept the window length L and set value of L to be 350 considering the trade-off between accuracy and computational cost (Section J.8). Then we swept the distance parameter σ. For the STCF we set the number of correlated neighbors to $k = 4$ (SM Section J.6). For the MLPF we swept the classification threshold T_{MLPF} from 0 to 1. (See SM Sections K and F.)

Fig. 8 shows all ROC curves and Table 2 summarizes denoising accuracy using AUC and two selected TPR/FPR operating points. Fig. 8A shows the results for the*hotel-bar* sequence and Fig. 8B shows results for the *driving* sequence. Fig. 8C summarizes AUC accuracy results for both datasets versus shot noise rate.

MLPF achieves significantly higher AUC than any of the handcrafted methods. It particularly improves the TPR, probably by filtering in events with weak correlation but with signal-like spatiotemporal structure. For example, in *driving* at FPR of 20%, MLPF has a TPR of 85% compared with STCF's 67%.

Among the hand-crafted methods, STCF has the highest accuracy by every metric. Its ROC curve shows higher TPR

TABLE 2
Denoising Accuracy Comparison

	hotel-bar			*driving*		
Filter	AUC	FPR @TPR 80%	TPR @ FPR 5%	AUC	FPR @ TPR 80%	TPR @ FPR 5%
ONF	0.01	NA	22	0.01	NA	17
DWF	0.86	23	55	0.69	60	26
BAF	0.89	20	35	0.79	35	21
STCF	**0.96**	**3**	**85**	**0.83**	**30**	**39**
MLPF	**0.97**	**2**	**92**	**0.89**	**16**	**46**

[4] NoiseTesterFilter.java on github. Using NoiseTesterFilter on YouTube.

[5] *Sparsity* is the fraction of nonactive pixels during 20 ms windows.

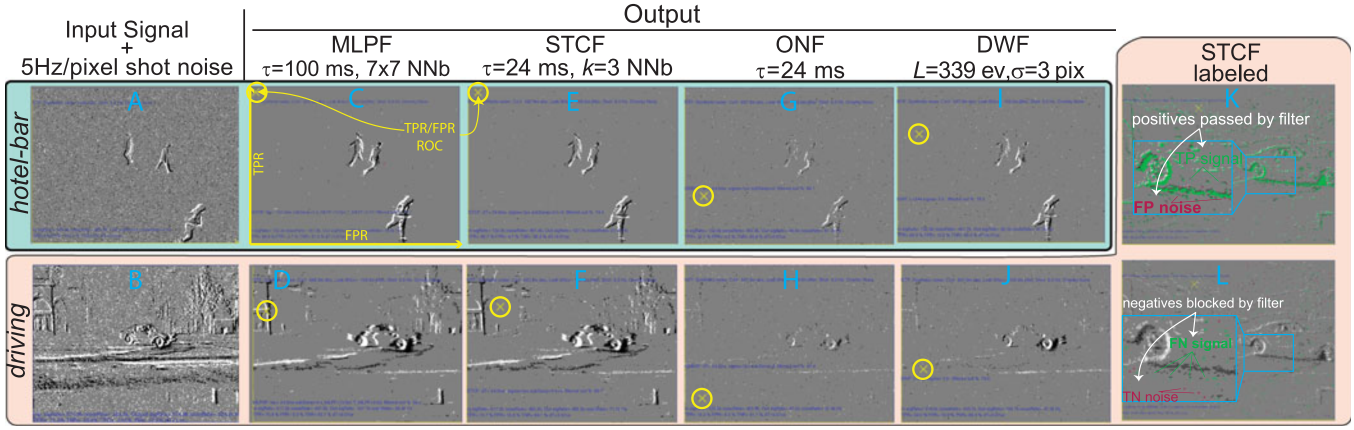

Fig. 9. Denoising comparisons. The small blue overlay statistics text are generated by *NoiseTesterFilter*. *Hotel-bar* shows 20ms frame at time 1606.903s. *Driving* shows 20ms frame at time 4.355s. Both use full scale 3 ON and OFF event rendering. Yellow crosses show ROC TPR and FPR point as in Figs. 8 A and 8 B. Panels K and L label classifications from STCF.

and lower FPR for all discrimination thresholds, and its AUC is higher than the other filters, though still far from ideal for the more dense and complex *driving* segment.

For the sparse *hotel-bar segment, DWF is significantly more accurate than FWF.* DWF uses less memory than ONF but the TPR is higher than that of ONF considering the same FPR for both datasets. Because DWF uses an NNb window of the most recent events, for some moving object scenarios, DWF/FWF requires only a tiny memory of a few tens of cells to achieve similar or superior accuracy compared with O(N^2) filters (See SM Sections J.1 and J.2).

Fig. 8C shows that MLPF and STCF maintain high AUC accuracy when the noise rate varies by a factor of more than ten. The other denoising filters degrade with higher noise rates.

6.1 Denoising Samples

Fig. 9 compares denoising by STCF, DWF, MLPF, ONF, and BAF on dataset samples. The **A** and **B** panels show the input with added shot noise of 5 Hz/pixel and 0.7 decades of FPN pixel to pixel variance.

MLPF followed by STCF (**C–F**) produces the cleanest output (most signal, least noise) for both datasets. For *driving*, MLPF (**D**) preserves as much signal as STCF while reducing noise FPR from 16% for STCF (**F**) to 6% for MLPF. ONF (**G–H**) blocks noise, but also a most of the signal events. DWF (**I**) works well on *hotel-bar* but in the denser *driving*, it blocks too much signal.

The **K** and **L** frames show the *driving* STCF positive output events (in gray), with overlaid signal and noise event labels. **K** shows labeled TP and FP events that passed STCF. The few FP noise events are closely mixed with signal events. **L** shows labeled negative events blocked by STCF. The indicated FN signal events are clearly part of the moving edge. These FN signal events are some of the events that MLPF classifies correctly by inferring that they are likely caused by real features.

7 Discussion

ONF and DWF: The ONF is limited to sparse activity scenarios. Its event memory is easily overwritten by events unrelated to current events, because the ONF stores entire rows or columns of events from the pixel array to a single memory location. Even though DWF uses less memory than ONF, it has better denoising accuracy.

STCF versus BAF: Our STCF generalization of BAF significantly improves the True Negative Rate (**TNR**) (detecting noise events correctly) without degrading the TPR (i.e. incorrectly filtering out signal). Thus STCF is very effective at removing uncorrelated noise, but signal is hardly degraded because it tends to produce multiple events from nearby pixels. STCF can remove high levels of noise without requiring a short correlation time threshold, which means it still passes slowly moving features that would be blocked by BAF (see SM Sections A&J.3). Considering that the memory cost of STCF which is one timestamp memory cell per pixel is identical to BAF it makes sense to implement STCF rather than BAF. In surveillance applications, using STCF can reduce the quiescent event rate by many orders of magnitude (see SM Section G.3.1).

MLPF: The STCF filter accurately rejects noise when it is isolated, but classifies signal versus noise only by counting the number of events in the spatiotemporal neighborhood. Our MLPF takes inspiration from the ultra quick classifiers that have been developed for high energy particle physics. They compute a highly quantized fully connected 3-layer network with ≈ 8k operations in less than 100 ns, with about 10 nJ of energy [36]. Our MLPF uses only a few thousand MAC operations per event and thus we expect that its real-time implementation would be practical by hardware parallelization within the logic circuits of an event camera. Because it can exploit structural cues, the MLPF achieves significantly better AUC accuracy than the handcrafted methods and and it is more than 10^4 times cheaper than previous DNN denoising architectures [30], [31] (see Table 1).

8 Conclusion

We proposed three new BA noise filters (DWF, STCF, and MLPF) that provide low-resource and high accuracy denoising. We also derived false positive BA filter equations which provide a principled computation of the filter correlation parameter to limit BA noise.

To quantify BA denoising accuracy, we introduced a novel framework to compare denoising using ROC. It quantifies BA denoising accuracy by the single scalar AUC metric, which removes the prior work limitations of particular threshold choice. By contrast to the complex DVS event generation models of some Section 2.2.1 algorithms, our method relies only on accurate BA noise modeling: Any clean recording of DVS activity can be used together with either recorded or realistic synthetic BA noise to evaluate denoising accuracy. Any clean DVS data combined with known BA noise can be used to train an MLPF denoiser. We used the framework to compare 5 types of BA noise filters in stationary and moving camera DVS scenes. Our STCF achieves the highest AUC among all hand-crafted filters , and our MLPF achieves even more ideal AUC with the tradeoff of higher computational cost.

BA denoising has great value for stationary camera scenes, where denoising can reduce the quiescent activity (and hence system level power consumption) by a factor of more than 100X while preserving most of the signal events. In moving camera, dense activity scenes, it is more difficult to distinguish signal from noise, but denoising increases sparsity[6], which upcoming DNN accelerators will exploit for improved energy efficiency.

Our SM includes many additional observations, datasets, and experiments. DND21 code and datasets are available at https://sites.google.com/view/dnd21/home.

Acknowledgments

The authors acknowledge the contributions of M. Milde, D. Karpul, C. Bartolozzi, E. Chicca, and G. Orchard toward discussions leading to STCF, and L. Wang for discussions leading to DWF. We thank A. Khodamoradi and R. Kastner for sharing source code and data for ONF. The authors gratefully acknowledge the anonymous reviewers, S.C. Liu, T. Aarrestad, G. Gallego, D. Joubert, Z. Wang, and Y. Hu for helpful comments. The authors thank the 2017 and 2018 Telluride and CapoCaccia Neuromorphic Engineering Workshops for opportunities to develop these ideas.

References

[1] T. Delbruck *et al.*, "Human versus computer slot car racing using an event and frame-based DAVIS vision sensor," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2015, pp. 2409–2412. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7169170

[2] M. Mahowald, "VLSI analogs of neuronal visual processing: A synthesis of form and function," Ph.D. dissertation, Dept. Comput. Sci., California Institute of Technology, Pasadena, CA, USA, May 1992. [Online]. Available: https://resolver.caltech.edu/CaltechCSTR:1992.cs-tr-92–15

[3] G. Gallego *et al.*, "Event-based vision: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. vol. 44, no. 1, pp. 154–180, Jan. 2022.

[4] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128x128 120dB 15us latency asynchronous temporal contrast vision sensor," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, Feb. 2008. [Online]. bAvailable: https://ieeexplore.ieee.org/abstract/document/4444573/

[5] Y. Hu, S.-C. Liu, and T. Delbruck, "v2e: From video frames to realistic DVS events," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2021, pp. 1312–1321.

[6] Y. Suh *et al.*, "A 1280x960 dynamic vision sensor with a 4.95um pixel pitch and motion artifact minimization," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2020, pp. 1–5.

[7] T. Finateu *et al.*, "5.10 a 1280x720 Back-Illuminated stacked temporal contrast event-based vision sensor with 4.86um pixels, 1.066GEPS readout, programmable Event-Rate controller and compressive data-formatting pipeline," in *Proc. IEEE Int. Solid-State Circuits Conf.*, 2020, pp. 112–114.

[8] A. Khodamoradi and R. Kastner, "O(N)-Space spatiotemporal filter for reducing noise in neuromorphic vision sensors," *IEEE Trans. Emerg. Top. Comput.*, vol. 9, no. 1, pp. 15–23, First Quarter 2018.

[9] Y. Nozaki and T. Delbruck, "Temperature and parasitic photocurrent effects in dynamic vision sensors," *IEEE Trans. Electron Devices*, vol. 64, no. 8, pp. 3239–3245, Aug. 2017.

[10] S. Liu, B. Rueckauer, E. Ceolini, A. Huber, and T. Delbruck, "Event-driven sensing for efficient perception: Vision and audition algorithms," *IEEE Signal Process. Mag.*, vol. 36, no. 6, pp. 29–37, Nov. 2019.

[11] B. Son *et al.*, "4.1 a 640x480 dynamic vision sensor with a 9um pixel and 300Meps address-event representation," in *Proc. IEEE Int. Solid-State Circuits Conf.*, 2017, pp. 66–67. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7870263/

[12] T. Delbruck, R. Graca, and M. Paluch, "Feedback control of event cameras," in *Proc. 3rd Int. Workshop Event-Based Vis.*, 2021, Art. no. 8. [Online]. Available: https://arxiv.org/abs/2105.00409

[13] T. Delbruck, "Frame-free dynamic digital vision," in *Proc. Int. Symp. Secure-Life Electron. Adv. Electron. Qual. Life Soc.*, 2008, pp. 21–26. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.192.2794&rep=rep1&type=pdf

[14] D. Czech and G. Orchard, "Evaluating noise filtering for event-based asynchronous change detection image sensors," in *Proc. 6th IEEE Int. Conf. Biomed. Robot. Biomechatronics*, 2016, pp. 19–24.

[15] S. Guo, Z. Kang, L. Wang, S. Li, and W. Xu, "HashHeat: An O(C) complexity hashing-based filter for dynamic vision sensor," in *Proc. 25th Asia South Pacific Des. Automat. Conf.*, 2020, pp. 452–457.

[16] J. Acharya *et al.*, "EBBIOT: A low-complexity tracking algorithm for surveillance in IoVT using stationary neuromorphic vision sensors," in *Proc. 32nd IEEE Int. Syst.-On-Chip Conf.*, 2019, pp. 318–323.

[17] D. Singla, V. Mohan, T. Pulluri, A. Ussa, B. Ramesh, and A. Basu, "EBBINNOT: A hardware efficient hybrid event-frame tracker for stationary neuromorphic vision sensors," May 2020. [Online]. Available: http://arxiv.org/abs/2006.00422

[18] F. C. Ojeda, A. Bisulco, D. Kepple, V. Isler, and D. D. Lee, "On-device event filtering with binary neural networks for pedestrian detection using neuromorphic vision sensors," in *Proc. IEEE Int. Conf. Image Process.*, 2020, pp. 3084–3088.

[19] S. K. Bose, D. Singla, and A. Basu, "A 51.3 TOPS/W, 134.4 GOPS in-memory binary image filtering in 65nm CMOS," *IEEE J. Solid-State Circuits*, vol. 57, no. 1, pp. 323–335, Jan. 2022.

[20] A. Linares-Barranco *et al.*, "Low latency event-based filtering and feature extraction for dynamic vision sensors in real-time FPGA applications," *IEEE Access*, vol. 7, pp. 134926–134942, 2019.

[21] Y. Feng, H. Lv, H. Liu, Y. Zhang, Y. Xiao, and C. Han, "Event density based denoising method for dynamic vision sensor," *NATO Adv. Sci. Inst. Ser. E Appl. Sci.*, vol. 10, no. 6, Mar. 2020, Art. no. 2024. [Online]. Available: https://www.mdpi.com/2076–3417/10/6/2024

[22] S. Afshar, "High speed event-based visual processing in the presence of noise," Ph.D. dissertation, Int. Centre Neuromorphic Syst., MARCS Institute for Brain, Behaviour and Develop., Western Sydney Univ., Penrith, Australia, 2020. [Online]. Available: https://researchdirect.westernsydney.edu.au/islandora/object/uws:56384/

[23] J. Wu, C. Ma, L. Li, W. Dong, and G. Shi, "Probabilistic undirected graph based denoising method for dynamic vision sensor," *IEEE Trans. Multimedia*, vol. 23, pp. 1148–1159, 2020.

[24] E. Mueggler, C. Forster, N. Baumli, G. Gallego, and D. Scaramuzza, "Lifetime estimation of events from dynamic vision sensors," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 4874–4881.

[25] E. Mueggler, C. Bartolozzi, and D. Scaramuzza, "Fast event-based corner detection," in *Proc. Brit. Mach. Vis. Conf.*, 2017, pp. 1–8. [Online]. Available: http://www.bmva.org/bmvc/2017/papers/paper033/index.html

[26] Y. Wang *et al.*, "EV-Gait: Event-based robust gait recognition using dynamic vision sensors," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6358–6367. [Online]. Available: http://openaccess.thecvf.com/content_CVPR_2019/html/Wang_EV-Gait_Event-Based_Robust_Gait_Recognition_Using_Dynamic_Vision_Sensors_CVPR_2019_paper.html

[6] e.g., MLPF increases sparsity (see Section 5.1) from 84% to 97% for *driving* with 5 Hz/pixel noise, a decrease of active pixels by 4X.

[27] Z. W. Wang, P. Duan, O. Cossairt, A. Katsaggelos, T. Huang, and B. Shi, “Joint filtering of intensity images and neuromorphic events for high-resolution noise-robust imaging,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 1609–1619. [Online]. Available: http://openaccess.thecvf.com/content_CVPR_2020/html/Wang_Joint_Filtering_of_Intensity_Images_and_Neuromorphic_Events_for_High-Resolution_CVPR_2020_paper.html

[28] S. A. S. Mohamed, J. N. Yasin, M.-H. Haghbayan, J. Heikkonen, H. Tenhunen, and J. Plosila, “DBA-Filter: A dynamic background activity noise filtering algorithm for event cameras” in *Intelligent Computing*. Berlin, Germany: Springer, 2022, pp. 685–696.

[29] S. Afshar, N. Ralph, Y. Xu, J. Tapson, A. van Schaik, and G. Cohen, “Event-Based feature extraction using adaptive selection thresholds,” *Sensors*, vol. 20, no. 6, Mar. 2020, Art. no. 1600.

[30] R. W. Baldwin, M. Almatrafi, V. Asari, and K. Hirakawa, “Event probability mask (EPM) and event denoising convolutional neural network (EDnCNN) for neuromorphic cameras,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 1698–1707. [Online]. Available: https://ieeexplore.ieee.org/document/9156496/

[31] P. Duan, Z. W. Wang, X. Zhou, Y. Ma, and B. Shi, “EventZoom: Learning to denoise and super resolve neuromorphic events,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 12819–12828. [Online]. Available: http://ci.idm.pku.edu.cn/CVPR21a.pdf

[32] C. Li, L. Longinotti, F. Corradi, and T. Delbruck, “A 132 by 104 10um-Pixel 250uW 1kefps dynamic vision sensor with Pixel-Parallel noise and spatial redundancy suppression,” in *Proc. Symp. VLSI Circuits*, 2019, pp. C216—C217.

[33] H. Liu, C. Brandli, C. Li, S. Liu, and T. Delbruck, “Design of a spatiotemporal correlation filter for event-based sensors,” in *Proc. IEEE Int. Symp. Circuits Syst.*, 2015, pp. 722–725.

[34] V. Padala, A. Basu, and G. Orchard, “A noise filtering algorithm for Event-Based asynchronous change detection image sensors on TrueNorth and its implementation on TrueNorth,” *Front. Neurosci.*, vol. 12, Mar. 2018, Art. no. 118.

[35] G. Taverni *et al.*, “Front and back illuminated dynamic and active pixel vision sensors comparison,” *IEEE Trans. Circuits Syst. Exp. Briefs*, vol. 65, no. 5, pp. 677–681, May 2018. [Online]. Available: http://dx.doi.org/10.1109/TCSII.2018.2824899

[36] C. N. Coelho *et al.*, “Automatic heterogeneous quantization of deep neural networks for low-latency inference on the edge for particle detectors” *Nature Mach. Intell.*, vol. 3, pp. 675–686, Jun. 2021. [Online]. Available: https://www.nature.com/articles/s42256–021-00356-5

Shasha Guo received the BS degree in information security from National University of Defense Technology, Changsha, China, in 2017. She is currently working toward the PhD degree in computer science and technology with the same university. Her research interests include computer vision and neuromorphic computing.

Tobi Delbruck (Fellow, IEEE) received the BSc degree in physics from the University of California in 1986 and the PhD degree from Caltech in 1993. Currently, he is a professor of Physics and Electrical Engineering with the Institute of Neuroinformatics, University of Zurich and ETH Zurich, where he has been since 1998. The Sensors group, which he co-directs with SC Liu, currently focuses on neuromorphic sensory processing, robotics, and efficient hardware AI.