Event Probability Mask (EPM) and Event Denoising Convolutional Neural Network (EDnCNN) for Neuromorphic Cameras

R. Wes Baldwin¹, Mohammed Almatrafi², Vijayan Asari¹ and Keigo Hirakawa¹

¹Department of Electrical Engineering, University of Dayton
²Department of Electrical Engineering, Umm Al-Qura University
(baldwinr2, vasari1, khirakawa1)@udayton.edu, mmmatrafi@uqu.edu.sa

Abstract

This paper presents a novel method for labeling real-world neuromorphic camera sensor data by calculating the likelihood of generating an event at each pixel within a short time window, which we refer to as “event probability mask” or EPM. Its applications include (i) objective benchmarking of event denoising performance, (ii) training convolutional neural networks for noise removal called “event denoising convolutional neural network” (EDnCNN), and (iii) estimating internal neuromorphic camera parameters. We provide the first dataset (DVSNOISE20) of real-world labeled neuromorphic camera events for noise removal.

1. Introduction

Neuromorphic (a.k.a. event-based) cameras offer a hardware solution to overcome limitations of conventional cameras, with high temporal resolution (>800 kHz), low latency (20 µs), wide dynamic range (120 dB), and low power (10–30 mW) [28]. This is accomplished by a dynamic vision sensor (DVS), which reports the log-intensity changes (i.e. events) of each pixel in microseconds. However, performance of methods using neuromorphic cameras deteriorate with noise. This fact has been cited as a major challenge in recent research [18, 20]. Noise is noticeable in low light conditions, where events triggered by minor intensity fluctuations dominate over the usable signal in the scene. Currently, there is no reliable way to benchmark the denoising performance because the exact distribution of noise in DVS circuitry—which is environment-, scene-, and sensor-dependent—is still unknown. Since neuromorphic cameras generate millions of events each second, it is impractical to manually label each event. This has precluded machine learning approaches to event denoising until this point.

We propose the notion of an “event probability mask” (EPM) – a label for event data acquired by real-world neuromorphic camera hardware. We infer the log–likelihood probability of an event within a neuromorphic camera pixel by combining the intensity measurements from active pixel sensors (APS) and the camera motion captured by an inertial measurement unit (IMU). Our contributions are:

- **Event Probability Mask (EPM):** spatial-time neuromorphic event probability label for real-world data;
- **Relative Plausibility Measure of Denoising (RPMD):** objective metric for benchmarking DVS denoising;
- **Event Denoising CNN (EDnCNN):** DVS feature extraction and binary classifier model for denoising;
- **Calibration:** a maximum likelihood estimation of threshold values internal to the DVS circuitry; and
- **Dataset (DVSNOISE20):** labeled real-world neuromorphic camera events for benchmarking denoising.

Figure 1. Proposed EDnCNN denoising applied to “CheckerFast” sequence in DVSNOISE20 dataset. DVS dots are colored by time and overlaid on APS image. Red dots were classified by EDnCNN as noise. EDnCNN characterizes real-world noise distribution by learning the mapping between actual and noise-free DVS events.
2. Background and Related Work

2.1. Neuromorphic Cameras

An APS imaging sensor synchronously measures intensity values observed by the photodiodes to generate frames. Although APS is a mature technology generating high-quality video, computer vision tasks such as object detection, classification, and scene segmentation are challenging when the sensor or the target moves at high-speed (i.e., blurring) or in high dynamic range scenes (i.e., saturation). High-speed cameras rely on massive storage and computational hardware, making them unsuitable for real-time applications or edge computing.

A DVS is an asynchronous readout circuit designed to determine the precise timing of when log-intensity changes in each pixel exceed a predetermined threshold. Due to their asynchronous nature, DVS events lack the notion of frames. Instead, each generated event reports the row/column pixel index, the timestamp, and the polarity. Log-intensity change is a quantity representing relative intensity contrast, yielding a dynamic range far wider than a conventional APS. Typical event cameras have a minimum threshold setting of $15–50\%$ illumination change—with the lower limit determined by noise [20].

In this work, we make use of a dynamic active vision sensor (DAVIS) that combines the functionality of DVS and APS [11]. Two read-out circuits share the same photodiode, operating independently. One outputs the log-intensity changes; the other records the linear intensity at up to 40+ frames per second. In addition, the DAVIS camera has an IMU, operating at the 1kHz range with timestamps synchronized to the APS and DVS sensors. See Figure 3.

Since their introduction, neuromorphic cameras have proven useful in simultaneous localization and mapping (SLAM) [37, 46], optical flow [2, 8, 48], depth estimation [14, 49], space applications [15, 17], tactile sensing [33, 39], autonomous navigation [30, 41], and object classification [4, 6, 10, 21, 34]. Many approaches rely on hand-crafted features such as [16, 26, 27, 32, 44], while other applications use deep learning architectures trained using simulated data [38, 42].

2.2. Denoising

There are largely four types of random noise in neuromorphic cameras. First, an event is generated even when there is no real intensity change. Referred to as “background activity” (BA), these false alarms severely impact algorithm accuracy and consume bandwidth. Second, an event is not generated, despite an intensity change (i.e., “holes” or false negatives). Third, the timing of the event arrival is stochastic. Lastly, although proportional to the edge magnitude (e.g., high contrast change generates more events than low contrast), the actual number of events for a given magnitude varies randomly.

Most existing event denoising methods are concerned with removing BA—examples include bioinspired filtering [5] and hardware-based filtering [24, 29]. Spatial filtering techniques leverage the spatial redundancy of pixel intensity changes as events tend to correspond to the edges of moving objects. Thus, events are removed due to spatial isolation [18, 19] or through spatial-temporal local plane fitting [7]. Similarly, temporal filters exploit the fact that a single object edge gives rise to multiple events proportional in number to the edge magnitude. Temporal filters remove events that are temporally redundant [3] or ambiguous. Edge arrival typically generates multiple events. The first event is called an “inceptive event” (IE) [4] and coincides with the edge’s exact moment of arrival. Events directly following IE are called “trailing events” (TE), representing edge magnitude. TE have greater ambiguity in timing because they occur some time after the edge arrival.

2.3. Neuromorphic Camera Simulation

Simulators such as ESIM [36] and PIX2NVS [9] artificially generate plausible neuromorphic events corresponding to a user-specified input APS image or 3D scene. The simulated neuromorphic events have successfully been used
in machine learning methods to perform tasks such as motion estimation [12, 43, 45] and event-to-video conversion [38]. However, the exact probabilistic distribution of noise within neuromorphic cameras is the subject of ongoing research, making accurate simulation challenging. To the best of our knowledge, we are unaware of any prior de-noising methods leveraging the exact and explicit characterization of the DVS noise distribution.

3. Event Probability Mask

We describe below a novel methodology of predicting the behavior of the DVS from APS intensity measurements and IMU camera motion. We derive the likelihood probability of an event in DVS pixels of a noise-free camera, a notion we refer to as “event probability mask” (EPM). EPM serves as a proxy for ground truth labels. For example, EPM identifies which of the real-world events generated by actual DVS hardware are corrupted by noise, thereby overcoming the challenges associated with modeling or simulating the noise behavior of DVS explicitly (see Section 2.3).

3.1. Proposed Labeling Framework

Let \( I : \mathbb{Z}^2 \times \mathbb{R} \rightarrow \mathbb{R} \) denote an APS video (the signal), where \( I(X, t) \) is the radiance at pixel \( X \in \mathbb{Z}^2 \) and at time \( t \in \mathbb{R} \). The log amplifier in DVS circuit yields a log-intensity video \( J : \mathbb{Z}^2 \times \mathbb{R} \rightarrow \mathbb{R} \), modeled as:

\[
J(X, t) := \log(a I(X, t) + b),
\]

where \( a \) and \( b \) are the gain and offset, respectively. In \textit{noise-free neuromorphic camera hardware}, idealized events are reported by DVS when the log-intensity exceeds a predefined threshold \( \varepsilon > 0 \):

\[
t_i(X) := \arg \min \{ t > t_{i-1}(X) \mid |J(X, t) - J(X, t_{i-1}(X))| \geq \varepsilon \},
\]

\[
p_i(X) := \text{sign}(J(X, t_i(X)) - J(X, t_{i-1}(X))).
\]

Ideally, each reported event from noise-free neuromorphic camera hardware provides the spatial location \( X \), precise timestamp \( t_i \) that \( J(X, t) \) crosses the threshold, and polarity \( p_i \in \{+1, -1\} \) indicating whether the change in the log-pixel intensity was brighter or darker respectively (see Figure 3).

Now suppose \( \{t_i(X), p_i(X)\} \) refers to a set of events obtained from \textit{real, practical, noisy DVS hardware}. We consider formalizing the DVS event de-noising as a hypothesis test of the form:

\[
\begin{align*}
H_0 & : |J(X, t_i(X)) - J(X, t_{i-1}(X))| \geq \varepsilon \\
H_1 & : |J(X, t_i(X)) - J(X, t_{i-1}(X))| < \varepsilon.
\end{align*}
\]

That is, we would like to determine whether an event as described by \( (t_i, p_i) \) corresponds to an actual temporal change in the log-pixel intensity exceeding the threshold \( \varepsilon \). However, this formalism has a major disadvantage; the hypothesis test on \( (t_i, p_i) \) relies on another event, \( (t_{i-1}, p_{i-1}) \) which may also be noisy. Thus in this work, we revise the hypothesis test as follows:

\[
\begin{align*}
H_0 & : t_i(X) \in [t, t + \tau) \text{ for some } i \\
H_1 & : t_i(X) \notin [t, t + \tau) \text{ for all } i,
\end{align*}
\]

where \( \tau \) is a user-specified time interval (set to the integration window of APS in our work; see Theorem 1 below). Notice that the new hypothesis test decouples \( (t_i, p_i) \) from \( (t_{i-1}, p_{i-1}) \). Hypothesis test in (4) also abstracts away the magnitude and timing noises, while faithfully modeling BA and holes.

Define the event probability mask \( EPM \) \( M : \mathbb{Z}^2 \times [0, 1] \) as the Bernoulli probability of null hypothesis:

\[
M(X, t) := P[H_0] := P[\exists i \text{ s.t. } t_i(X) \in [t, t + \tau)].
\]

Intuitively, \textit{EPM quantifies the plausibility of observing an event within the time window \([t, t + \tau]\).} If an event occurred during \([t, t + \tau]\) but \( M(X, t_i) \approx 0 \), then this is an implausible event, likely caused by noise. On the other hand,
if \( M(X, t_i) \approx 1 \), we have a high confidence that the event corresponds to an actual temporal change in the log-pixel intensity exceeding \( \varepsilon \). In this sense, EPM is a proxy for soft ground truth label—the inconsistencies between EPM and the actual DVS hardware output identify the events that are corrupted by noise. EPM \( M: \mathbb{Z}^2 \times \mathbb{R} \rightarrow [0, 1] \) can be computed from APS and IMU measurements as explained in Theorem 1 below.

**Theorem 1** Let \( \theta(t) = (\theta_x(t), \theta_y(t), \theta_z(t))^T \) represent the instantaneous 3-axis angular velocity of camera measured by IMU’s gyroscope. Let \( A: \mathbb{Z}^2 \times \mathbb{Z} \rightarrow \mathbb{R} \) denote APS measurements with exposure time \( \tau \). Assume camera configuration with focal length \( f \), principal point \( c_x, c_y \), and skew parameter \( \kappa \). Then

\[
M(X, t) = \begin{cases} 
\frac{\tau |J_t(X, t)|}{\varepsilon} & \text{if } |J_t(X, t)| < \frac{\varepsilon}{\tau} \\
1 & \text{else.}
\end{cases} \tag{6}
\]

where \( J_t(X, t) \) is as described in (7) and (8).

Proof and derivation is provided in Appendix A. Examples of EPM are shown in Figure 4. Note that this method requires two parameters internal to the DAVIS camera, namely the threshold value \( \varepsilon > 0 \) and the offset value \( O \in \mathbb{R} \). Calibration procedure to obtain these values is described in Section 6.

### 3.2. Limitations

EPM calculation requires static scenes (i.e. no moving objects) and rotation-only camera motion (i.e. no translational camera movement) to avoid occlusion errors. We address this issue by acquiring data using a camera configuration as shown in Figure 5 (additional details in Section 7.1). We emphasize, however, this is only a limitation for benchmarking and network training (Section 5.3). These restrictions are removed at inference because spatially global and local pixel motions behave similarly within the small spatial window used by our denoising model. Our empirical results in Section 7.3 confirm robustness to such model assumptions. Additionally, Theorem 1 is only valid for constant illumination (e.g. fluorescent light flicker is detectable by DVS). Moreover, since the dynamic range of the DVS is much larger than the APS, it is not possible to calculate an EPM for pixels at APS extremes. Examples shown in Figure 4.

Another limitation is that the value of \( M(X) \) diminishes when the camera motion is very slow. This is due to the fact that the events are infrequently generated by DVS, reducing the probability of observing an event within a given time window \([t, t + \tau]\). While EPM captures this phenomenon accurately (limited only by the IMU sensitivity), it is difficult to discriminate noisy events from events generated by extremely slow motion.

![Figure 5. Camera setup for DVSNOISE20 collection. Gimbal limits camera motion while centering the focal point at the origin.](image)

### 4. Application: Denoising Benchmarking

EPM is the first-of-its-kind benchmarking tool to enable quantitative evaluation of denoising algorithms against real-world neuromorphic camera data. Given a set of events \( (t_i, p_i) \), let \( E: \mathbb{Z}^2 \rightarrow \{0, 1\} \) denote an event indicator:

\[
E(X, t) = \begin{cases} 
1 & \text{if } t_i(X) \in [t, t + \tau) \text{ for some } i \\
0 & \text{if } t_i(X) \notin [t, t + \tau) \text{ for all } i.
\end{cases} \tag{9}
\]

Then if EPM is known, the log-probability of the events \( (t_i, p_i) \) is explicitly computable:

\[
\log Pr[E] = \sum_{X \in \mathbb{Z}^2} E(X) \log M(X) + (1 - E(X)) \log(1 - M(X)). \tag{10}
\]

(Proof: for each pixel \( X \in \mathbb{Z}^2 \), \( \log Pr[E(X)] = 1 \) = \( M(X) \) and \( \log Pr[E(X)] = 0 \) = \( 1 - M(X) \).) This log-probability can be used to assess the level of noise present in real, practical, noisy DVS hardware. On the other hand, if \( (t'_i, p'_i) \) denotes the outcome of an event denoising method, then the corresponding log-probability \( \log Pr[E'] \) is an objective measure of the denoising performance. The improvement from noisy events \( (t_i, p_i) \) to denoised events \( (t'_i, p'_i) \) can be quantified by \( \log Pr[E'] - \log Pr[E] \).

The objective of denoising methods, then, is to yield a set of events \( (t'_i, p'_i) \) to maximize \( \log Pr[E'] \). In fact, the theoretical bound for best achievable denoising performance is computable. It is

\[
\max_{E: \mathbb{Z}^2 \rightarrow \{0, 1\}} \log Pr[E] = \sum_{X \in \mathbb{Z}^2} \log \max(M(X), 1 - M(X)), \tag{11}
\]

which is achieved by a thresholding on \( M(X) \)

\[
E_{opt}(X) = \begin{cases} 
1 & \text{if } M(X) > 0.5 \\
0 & \text{if } M(X) \leq 0.5.
\end{cases} \tag{12}
\]
defined as called “relative plausibility measure of denoising” (RPMD), we propose an objective DVS quality metric for denoising (Proof: if \( M(X) \leq 1 - M(X) \) then \( M(X) \leq 0.5 \)). Thus we propose an objective DVS quality metric for denoising called “relative plausibility measure of denoising” (RPMD), defined as

\[
\text{RPMD} := \frac{1}{N} \log \frac{Pr[E_{opt}]}{Pr[E]} \quad (13)
\]

where \( N \) is the total number of pixels. Lower RPMD values indicate better denoising performance with 0 representing the best achievable performance. Benchmarking results using RPMD are shown in Figures 7 and 9.

5. Application: Event Denoising CNN

Event denoising for neuromorphic cameras is a binary classification task whose goal is to determine whether a given event corresponds to a real log-intensity change or noise. We propose EDnCNN, an event denoising method using convolutional neural networks. EDnCNN is designed to carry out the hypothesis test in (4), where the null hypothesis \( H_0 \) states that the event \( t_i(X) \) is expected to be generated by DVS within a short temporal window \([t, t+\tau]\) due to changes in log-intensity.

The input to EDnCNN is a 3D vector generated only from DVS events. The training data is composed of DVS and the corresponding EPM label. Once trained, EDnCNN does not require APS, IMU, stationary scenes, or restricted camera motion. By training on DVS data from actual hardware, EDnCNN benefits from learning noise statistics of real cameras in real environments.

5.1. Input: Event Features

There exist a wide array of methods to extract features from neuromorphic camera data \([3, 10, 13, 21, 22, 25, 26, 34, 44]\). These methods are designed to summarize thousands or even millions of events into a single feature to carry out high-level tasks such as object tracking, detection, and classification. However, event denoising is a low-level classification task. Denoising requires inference on pixel-level signal features rather than high-level abstraction of scene content. For example, there is a high likelihood that an isolated event is caused by noise, whereas spatially and temporally clustered events likely correspond to real signal \([35]\). For this reason, event denoising designed to discriminate IE, TE, and BA would benefit from features that faithfully represent the local temporal and spatial consistency of events.

In denoising, we take inspiration from PointConv \([47]\), a method for generating nonlinear features using local coordinates of 3D point clouds. Unlike PointConv, designed for three continuous spatial domains, a DVS event is represented by two discrete spatial and one continuous temporal dimension. We leverage the discrete nature of the spatial dimensions by mapping the temporal information from the most recently generated event at each pixel to construct a time-surface \([26]\). This is similar to FEAST \([1]\), which extracts features from a spatial neighborhood of time-surface near the event-of-interest. However, the temporal history of recent events at each pixel is averaged into a single surface, obfuscating the spatial consistency of event timing useful to denoising.

Combining ideas from PointConv and FEAST, we propose to encode the events within the spatial-temporal neighborhood of the event-of-interest \((t_i(X), p_i(X))\). The EDnCNN input is a feature vector \( Q \in \mathbb{R}^{m \times m \times k \times 2} \). Here, \( m \times m \) refers to the size of the spatial neighborhood centered at pixel \( X \) where the event-of-interest occurred. At each pixel within this spatial neighborhood, we wish to encode \( k \) most recently occurring events (i.e. before \( t_i(X) \)) of polarities \( p_i = -1 \) and \( p_i = 1 \) (hence the dimension 2 in \( Q \)). Note that the temporal neighborhood is not thresholded by time but by the number of events. This allows for automatic adaptation to pixel velocity—events corresponding
to a slower moving edge require a longer temporal window than a fast moving edge with very frequent event reporting.

When an event \( t_i(X), p_i(X) \) is received, we populate \( Q(:, 1, -1) \) and \( Q(:, 1, +1) \) with \( m \times m \) relative time-surfaces formed by the difference between the time stamp of event-of-interest \( t_i(X) \) and the time stamps of the most recent events at each neighborhood pixel with the polarities \(-1\) and \(+1\) respectively. We repeat the construction of time-surfaces using the second most recent events at each neighborhood pixel, which is stored into \( Q(:, 2, -1) \) and \( Q(:, 2, +1) \) etc., until \( k \) most recent events at every pixel are encoded into \( Q \). See Figure 6. As a side note, encoding of EDnCNN input feature is very memory efficient. Each time-surface is the size of the DVS sensor, meaning the overall memory requirement is \( M \times N \times k \times 2 \) where \((M, N) \in \mathbb{Z}^2\) is the spatial resolution of the DVS sensor, storing the most recent \( k \) events. In our implementation, \( m = 25 \), \( k = 2 \), and the resolution of DAVIS346 is \((M, N) = 346 \times 260\).

5.2. Network Architecture

Let \( \hat{E}_\phi : \mathbb{Z}^2 \rightarrow \{0, 1\} \) be the output of the EDnCNN binary classifier, where the network coefficients is denoted \( \phi \). The output \( \hat{E}_\phi(X) = 1 \) implies that an event \((t_i, p_i)\) corresponds to a real event. Testing showed a shallow network could quickly be trained using the features described in Section 5.1, which is ideal for high-performance inference. EDnCNN is composed of three \( 3 \times 3 \) convolutional layers (employing ReLU, batch normalization, and dropout) and two fully connected layers. The learning was performed via Adam optimizer with a decay rate of 0.1 and a learning rate of 1E-4. The network is trained on EPM-labeled DVS events generated during APS frame exposure. Once trained, EDnCNN can classify events at any time. Due to the fact that small local patches are primarily scene independent, EDnCNN can perform well against new scenes and environments and requires no tuning or calibration at inference.

5.3. Training

We present three strategies for training EDnCNN and prove that these strategies are statistically equivalent, given sufficient training data volume. The first approach aims at minimizing the RPMD by maximizing a related function:

\[
\phi_{opt1} = \arg \max_{\phi} \sum_X P_r[\hat{E}_\phi(X)]
\]

\[
\sum_X \hat{E}_\phi(X)M(X) + (1 - \hat{E}_\phi(X))(1 - M(X)).
\]

Strictly speaking, (14) is not equivalent to RPMD minimization. Maximizing \( P_r[\hat{E}_\phi] \) implies multiplying \( P_r[\hat{E}_\phi(X)] \) over \( X \in \mathbb{Z}^2 \). However, (14) is equivalent to the \( L^1 \) minimization problem:

\[
\phi_{opt2} = \arg \min_{\phi} \sum_X \left| M(X) - \hat{E}_\phi(X) \right|. \quad (15)
\]

(Proof: Penalty for choosing \( \hat{E}_\phi(X) = 1 \) is \( 1 - M(X) \) in (15), which is equivalent to reward of \( M(X) \) in (14).)

Lastly, consider a minimization of classification error:

\[
\phi_{opt3} = \arg \min_{\phi} \sum_X \left| E_{opt}(X) - \hat{E}_\phi(X) \right|, \quad (16)
\]

where \( E_{opt} \) is the theoretically optimal classifier defined in (12). Given sufficient data, (16) is statistically equivalent to (14) and (15). Proof is provided in Appendix A.

6. Application: Calibration

A key to calculating the event likelihood from APS and IMU data is knowing the log contrast sensitivity \( \varepsilon \) in (2). In theory, this parameter value is controlled by registers in neuromorphic cameras programmed by the user. In practice, the programmed register values change the behavior of the DVS sensors, but the exact thresholding value remains unknown [12]. Similarly, the gain and offset values \( a, b, \alpha, \beta \) in DVS (1) and APS are not easily observable and are difficult to determine precisely. It is of concern because the offset value \( O \) in (8) is \( O = \beta + ab/a \) (see Appendix A). Offset allows mapping the small linear range of the APS to the large dynamic range of the DVS.

In this work, we make use of the EPM to calibrate the thresholding value \( \varepsilon \) and the offset \( O \) from the raw DVS data. Recalling (5) and (10), the probabilistic quantities \( P_r[H_0] \) and \( \log P_r[E] \) are parameterized in part by \( \varepsilon \) and \( O \). Hence, rewriting them more precisely as \( P_r[H_0|\varepsilon, O] \) and \( \log P_r[E|\varepsilon, O] \), respectively, we formulate a maximum likelihood estimate as follows:

\[
(\hat{\varepsilon}, \hat{O}) = \arg \max_{(\varepsilon, O)} P[E|\varepsilon, O], \quad (17)
\]

where \( E : \mathbb{Z}^2 \rightarrow \{0, 1\} \) denotes the event indicator for the unprocessed, noisy DVS data. Solution to (17) is not required for network inference. These estimated values are only used to compute EPM for use in benchmarking (Section 4) and denoising (Section 5).

7. Experiments

7.1. Dataset: DVSNOISE20

Data was collected using a DAVIS346 neuromorphic camera. It has a resolution of 346 \( \times 260 \) pixels, dynamic range of 56.7 and 120dB for APS and DVS respectively, a latency of 20\( \mu \)s, and a 6-axis IMU. As discussed in Section 3.2, Theorem 1 is valid in absence of translational camera motion and moving objects. Movement of the camera
was restricted by a gimbal (Figure 5), and the IMU was calibrated before each collection. Only stationary scenes were selected, avoiding saturation and severe noise in the APS.

The APS framerate (41-56 fps; $\eta$ range 17 to 24 ms) was maximized using a fixed exposure time ($\tau$ range 0.13 ms to 6 ms) per scene. Since EPM labeling is only valid during the APS exposure time, benchmarking (Section 4) and training EDnCNN (Section 5.3) are restricted to events occurring during this time. However, a large data volume can be acquired easily by extending the length of collection. In addition, we calibrated APS fixed pattern noise and compensated for the spatial non-uniformity of pixel gain and bias.

We obtained 16 indoor and outdoor scenes of noisy, real-world data to form DVSNOISE20. Examples are shown in Figure 4. Each scene was captured three times for $\approx 16$ seconds, giving 48 total sequences with a wide range of motions. The calibration procedure outlined in Section 6 was completed for each sequence. The estimates of internal camera parameters were repeatable and had mean/standard deviation ratios of 21.44 ($O$), 13.58 ($\epsilon_{pos}$), and 13.27 ($\epsilon_{neg}$). The DVSNOISE20 dataset, calibration, and code are available at: http://issl.udayton.edu.

7.2. Results

To ensure fair evaluation, EDnCNN was trained using a leave-one-scene-out strategy for DVSNOISE20. In Figure 7, the performance of EDnCNN is assessed by RPMD. EDnCNN improved the RPMD performance of the noisy data by an average gain of 148 points. Improvement from denoising was significant in all scenes except for the Alley and Wall sequences. These scenes have highly textured scene contents, which are a known challenge for neuromorphic cameras, and thus represent the worst case performance we expect from any denoising method. The RPMD performance of EDnCNN in these two scenes was no better or worse than noisy input.

EDnCNN was benchmarked against other state-of-the-art denoising methods: filtered surface of active events (FSAE [32]), inceptive and trailing events (IE & IE+TE [4]), background activity filter (BAF [19]), and nearest neighbor (NN & NN2 [35]). RPMD scores are reported in Figure 7 for each scene. FSAE and IE did not improve significantly over noisy input, but did reduce total data volume. IE+TE improved RPMD performance while reducing data volume and generated a top score in the LabFast sequence. BAF, NN, and NN2 work and behave similarly, outperforming EDnCNN in 3 of 16 scenes. However, the performances of these methods were significantly more sensitive than EDnCNN. EDnCNN outperformed other denoising methods in 12 of 16 scenes, with a statistically significant $p$-value of 0.00248 via the Wilcoxon signed-rank test.

Figure 8 shows examples of denoised DVS events (superimposed on APS image for visualization). Qualitatively, IE+TE, BAF, and NN2 pass events that are spatially isolated, making it more difficult to distinguish edge shapes compared to EDnCNN. EDnCNN removes events that do not correspond to edges, and enforces a strong agreement with the EPM label as designed (see Figure 4).

7.3. Robustness to Assumptions and Dataset

To test for robustness, we also benchmarked on simulated DVS data from ESIM [36]. ESIM is an event camera simulator allowing user-specified 3D scenes, lighting, and camera motion. In our experiments, we interpret DVS data simulated from a virtual scene as an output from a noise-free neuromorphic camera. We then injected additional random events (i.e. BA noise) into the scene. Figure 9 shows the result of RPMD benchmarking on synthetic data as a function of BA noise percentage. As expected, the RPMD performance of noisy data scales linearly as the percentage of noise increases. Since EDnCNN was not trained against noise-free data, it has a slightly lower performance than some other methods at 0% BA noise level. By contrast, IE+TE, BAF, NN, and NN2 underdetect noise as event count increases, deteriorating performance.

Figure 10 shows example sequences with non-rotational camera motion [40] and multiple moving objects [31]. Qualitatively, EDnCNN denoising seems consistent with
stationary scene performance in Figure 8. Additional analysis on other datasets and further examples may be found in supplementary material and Appendix A.

8. Conclusion

Contrast sensitivity in neuromorphic cameras is primarily limited by noise. In this paper, we present five major contributions to address this issue. We rigorously derived a method to assign a probability (EPM) for observing events within a very short time interval. We proposed a new benchmarking metric (RPMD) that enables quantitative comparison of denoising performance on real-world neuromorphic cameras. We developed EDnCNN, a neural network-based event denoising method trained to minimize RPMD. We showed that internal camera parameters can be estimated based on natural scene output. We collected a new benchmarking dataset for denoising along with the EPM labeling tools (DVSNOISE20). Quantitative and qualitative assessment verified that EDnCNN outperforms prior art. EDnCNN admits higher contrast sensitivity (i.e. detection of scene content obscured by noise) and would vastly enhance neuromorphic vision across a wide variety of applications.
References


[29] Hongjie Liu, Christian Brandli, Chenghan Li, Shih-Chii Liu, and Tobi Delbruck. Design of a spatiotemporal correlation...


