Time Aggregation based Lossless Video Encoding for Neuromorphic Vision Sensor Data

Nabeel Khan, Khurram Iqbal, Maria G. Martini, *Senior Member, IEEE*

***Abstract*—Dynamic Vision Sensors (DVS) are emerging neuromorphic visual capturing devices, with great advantages in terms of low power consumption, wide dynamic range, and high temporal resolution in diverse applications such as autonomous driving, robotics, tactile sensing and drones. The capturing method results in lower data rates than conventional video. Still, such data can be further compressed. Recent research has shown great benefits of temporal data aggregation on event-based vision data utilization. According to recent results, time aggregation of DVS data not only reduces the data rate but improves classification and object detection accuracy. In this work, we propose a compression strategy, Time Aggregation based Lossless Video Encoding for Neuromorphic Vision Sensor Data (TALVEN), which utilizes temporal data aggregation, arrangement of the data in a specific format and lossless video encoding techniques to achieve high compression ratios. The detailed experimental analysis on outdoor and indoor datasets shows that our proposed strategy achieves superior compression ratios than the best state-of-the-art strategies.**

***Index Terms*—Dynamic Vision Sensor (DVS), Neuromorphic Spike Events, Silicon Retinas, Spike Encoding, Video Encoding, Data Compression, Time Aggregation.**

I. Introduction

Dynamic Vision Sensors (DVS) [1], [2] are based on the principle of biological sensing, *i.e.,* they report only the on/off triggering of brightness in the observed scene. Differently from frame based cameras, where frames are acquired at regular time intervals, DVS asynchronously acquire pixel-level light intensity changes, with a time resolution up to a microsecond. Events are triggered whenever there is either motion of the neuromorphic vision sensor or motion / change of light conditions in the scene or both. In other words, no data is transmitted for stationary vision sensors and static scenes. These unique properties enable neuromorphic vision sensors to achieve wide dynamic range, low-latency, and low-power requirements. The data rate produced by these sensors depends on the scene complexity and on the camera speed, as highlighted in [3], [4], where a model for the estimation of such data rates is also presented.

The neuromorphic silicon technology utilizes the Address Event Representation (AER) protocol for representing and exchanging spike data. According to the protocol, each event is represented by a tuple (x, y, p, t), where x and y are the coordinates of the pixel where a brightness change occurred, t is the firing time of the spike (timestamp), and p is the polarity of the event (increase or decrease of brightness). Spike location is represented in three dimensions by the spatial coordinates and the timestamp information of the tuple, while the polarity flag indicates the brightness change. Each tuple is represented by 64 bits, where the timestamp is represented by 32 bits and the remaining three fields are represented by 4 bytes.

Emerging applications of DVS can be found in diverse scenarios ranging from self-driving cars [5], [6] to robotics [7], [8] and drones [9]. Instead of constraining applications to on-board processing many scenarios, such as the coordination of multiple intelligent vehicles (cars, drones, etc.) require real-time data sharing and feedback. Even if the neuromorphic sensing technique provides an intrinsic compression, further compression of the produced data can be beneficial for transmitting such data in Internet of Things (IoT), Internet of Intelligent Vehicles (IoV), and Industrial Internet of Things (IIoT) scenarios. The data storage and transmission bandwidth are finite for on-board DVS processing and transmission respectively, therefore the compression of neuromorphic spike events is an open challenge demanding prompt solutions.

Time aggregation of spike events has been primarily used for practical reasons, such as interfacing event-based devices with other hardware systems and operating within memory and computational constraints. Recent works have shown great advantages of aggregation of spike events for object detection and classification in diverse scenarios ranging from action recognition tasks to tactile sensing. These applications have utilized different types of machine learning approaches, summarized in Section III, where the spike event stream is accumulated over fixed time intervals. The aggregation time interval varies (ranging from 1 ms to 50 ms) depending upon the target application and the desired accuracy. For instance, the spike accumulation is performed every 50 ms for the motion estimation task in autonomous driving, whereas the optimum interval is 10 ms for object detection applications. The higher the accumulation interval, the higher the compression gains. The state-of-the-art compression approaches, discussed in Section II-B, do not exploit accumulation of spike events, which limits the adaptability as well as compression gains of such strategies. Therefore, given the benefits of temporal aggregation, we propose a compression approach where accumulation of spike events is the key processing step. We organize the asynchronous spike events data into a format where the exploitation of temporal and spatial redundancy is maximized. The strategy utilizes the lossless compression mode of recent video encoding standards and achieves superior compression ratios as compared

Submitted 11 Feb 2020. This work is part of a project that has received funding from EPSRC via Grant EP/P022715/1 - The Internet of Silicon Retinas: Machine to machine communications for neuromorphic vision sensing data (IoSiRe). *(Corresponding author: Nabeel Khan.)*

The authors are with Wireless Multimedia & Networking Research Group, Kingston University London, UK. Email: {n.khan, khurram.iqbal, m.martini}@kingston.ac.uk

to the state-of-the-art approaches. Furthermore, the compressed DVS stream has the capability to adapt to diverse transport and networking layer protocols as a result of the utilization of the video encoding step. The state-of-the-art compression approaches lack such network friendly representation of the DVS data.

The contributions provided in this work are summarized in the following.

- A novel strategy, that we called TALVEN, to compress DVS data by accumulating spike events over a time interval. The proposed strategy is evaluated on diverse outdoor and indoor scenes.
- Comparison in terms of compression gains of the state-of-the-art strategies without time aggregation and our proposed strategy: DVS data comprise a multivariate stream of integers, therefore, it is important to analyze general purpose and integer based compression approaches on DVS data. We applied different types of compression algorithms, shown in Figure 1 [10], to DVS data and studied the compression gains on diverse outdoor and indoor scenes.
  Compression gains of the proposed and selected state-of-the-art strategies with time aggregation: We analysed and compared the achieved compression gains of the proposed and benchmark strategies when time aggregation is applied on the DVS data.

The remainder of this paper is structured as follows. The state-of-the-art lossless compression strategies are reviewed in Section II, where the suitability on DVS specific applications of such strategies and their potential disadvantages are also discussed. The benefits of spike events accumulation are discussed in Section III, whereas the compression approach is proposed in Section IV. Section V reports the simulation setup and the considered dataset for the evaluation of the proposed and benchmark compression algorithms. We analyse existing lossless data compression strategies performance without temporal aggregation in Section VI; whereas the compression gains, when temporal accumulation of spike events is employed, are analysed in Section VII. Finally. Section VIII concludes the paper.

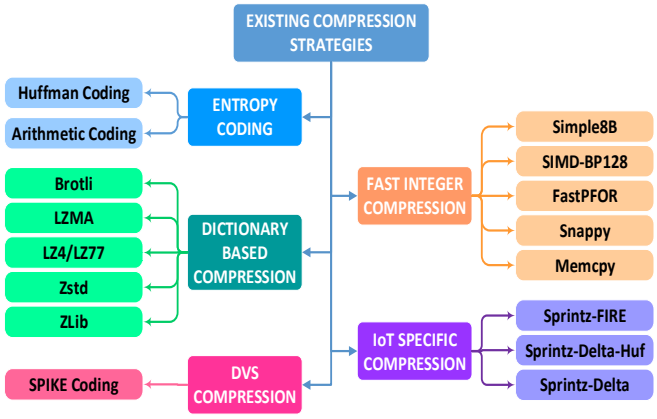

Figure 1: Benchmark methodologies to compress the DVS data [11].

II. Related work

Figure 1 reports the benchmark algorithms to compress the DVS data. These include DVS specific compression approach, discussed in Section II-A, as well as general purpose and IoT specific compression methodologies, reviewed in Section II-B.

A. DVS specific Spike Coding

The work in [12] proposed the first lossless compression strategy for the asynchronous spike event stream. The encoding method is derived from the spike firing model of the DVS. According to the spike generation mechanism, the DVS sequence exhibits spatial and temporal correlation. The spike coding method exploits the correlation by projecting the event stream into a sequence of three-dimensional macro-cubes. The two dimensions of the macro-cubes span the full spatial resolution of the sensor's pixel array, for instance, M = 180 (maximum Y -address) and N = 240 (maximum X-address) for DVS240B. The number of spike events is the third dimension of the macro-cube. There are two coding modes of the macro-cubes, namely Address-Prior (AP) and Time-Prior (TP) modes. These two modes exploit spatial redundancy within the macro-cube. The AP mode is designed for the scenarios where spike events are scattered over the entire spatial resolution. Conversely, the TP mode is designed for the case where spike events occur locally, *i.e.,* majority of the events occur over adjacent pixels. The spike coding algorithm computes prediction residuals for both the modes and the one yielding the maximum compression is chosen. In the final step, prediction residuals are entropy coded by utilizing CABAC (Context-adaptive binary arithmetic coding). The work in [13] extends the spike coding framework by introducing intercube prediction which exploits temporal correlation among the macrocubes. However, the compression gain achieved by the spike coding strategy is quite limited; for instance, the compression ratio for the intelligent driving dataset [13] varies between 2 and 3.

B. Existing compression methodologies tailored to the DVS data

1) Entropy coding: Entropy is a measure of uncertainty, i.e., the higher the uncertainty, the higher the entropy. In entropy coding, the most probable (frequent) symbols require shorter codewords. Huffman and Arithmetic are the most common entropy coding strategies. Huffman is prefix encoder, where each input data symbol is replaced by a variable length codeword. Arithmetic coding differs from Huffman as it encodes a group of input data symbols into a number.

Entropy coders are quite versatile, therefore they have the potential to be applied in diverse applications [14] of the DVS. They can be directly applied to the DVS data by treating each field of the spike event as an input symbol. Advance coding methods have entropy encoders as the final step mainly because these strategies have limited compression gains as a standalone compression algorithm.

2) Dictionary based compression: Dictionary coding strategies operate by replacing long strings, in the data to be compressed, with shorter codewords. These encoding methods

maintain a collection of strings in a data structure called dictionary. The text is encoded by replacing each string with a code that that acts as a pointer to the dictionary. Most of the dictionary-based strategies utilize a dynamic dictionary, one whose content changes during the coding process. The advanced dictionary coders, such as Zstd [15], Zlib [16], LZMA [17] and Brotli [18], utilize multi-level encoding, where dictionary codewords are further compressed by entropy coding.

Dictionary based compression can be applied to the DVS data by converting spike events into a multivariate stream of integers and then applying the concept of dictionary-based substitution. The higher the frequency of repeatable integers, the higher the compression gains. Dictionary based compression techniques can greatly ease the data rate and storage problems of DVS surveillance related applications [19] as these strategies have the potential to greatly reduce the size of the data. However, the lack of repetitive pattern in the asynchronous stream of DVS data can limit the compression gains of the dictionary-based compression strategies.

3) IoT specific compression: The authors in [20] proposed a compression strategy, called Sprintz, for resource constrained devices. The main design goal of the Sprintz algorithm is to achieve state-of-the-art compression gains without violating the memory and latency constraints of the IoT devices. The Sprintz compression approach exploits correlation among the successive samples of a multivariate stream. Sprintz-FIRE, Sprintz-Delta and Sprintz-Delta-Huf are the three main variants of the Sprintz strategy. Sprintz-FIRE is based on a forecaster called Fast Integer Regression (FIRE). The FIRE algorithm predicts the current sample based on information of the previous samples. The prediction residuals, *i.e.,* differences between the predicted and the actual samples is Huffman coded. In order to achieve superlative compression speed, Sprintz-Delta skips the Huffman coding and replaces the FIRE algorithm with delta coding. The third variant, Sprintz-Delta-Huf, achieves a trade off between compression ratio and compression speed by employing a combination of delta and Huffman coding.

The DVS data exhibit time-series characteristics, therefore, IoT specific compression strategies (such as Sprintz) can be directly applicable by converting the spike event stream into multivariate time-series integers. IoT specific compression can be useful in scenarios where higher compression gains and reduced power consumption are important performance criteria [14]. The compression ratio of the IoT specific approach is high for time series data generated by sensors like a temperature sensor which exhibits highly correlated data samples. Since the event stream generated by DVS sensors is asynchronous and presents less correlation than, for instance, data from temperature sensors, the compression gains of the IoT specific approach on DVS data are not expected to be as high as for the data it was designed for.

4) Fast Integer Compression: Fast integer compression strategies are known for their superlative compression speed as they are specifically designed for encoding and decoding billions of arrays of integers for search engines and relational database applications. Simple8B [20], Memcpy [20], SIMD-BP128 [21], FastPFOR [21], and SNAPPY [22] are the most common fast integer compression algorithms. Fast integer compression strategies have potential applications in scenarios where DVS data is transported to cloud storage and computing servers for the processing of visual data in order to perform event, action, person or object recognition/classification, and context awareness [23]. In such scenarios, Fast integer compression strategies can be applied to the DVS data by transforming spike events into a column major format (vector of integers). However, fast integer compression approaches may result in modest compression gains, as their main design goal is superlative compression and decompression speed at the expense of compression ratios, thus limiting their applicability.

The aforementioned state-of-the-art approaches do not consider the time aggregation of spike event data, which limits the potential compression gains of such strategies. In the subsequent section, we report the benefits of spike event aggregation.

III. Spike Event Aggregation

Since a single event carries little information and is subject to noise, it is important to process several events to yield a sufficient signal-to-noise-ratio (SNR) for the considered task. Recently several strategies [6], [7], [24], [25], [26], [27], [28], [29] have been proposed that operate on a group of events. These strategies aggregate the information present in the group of events to estimate the solution to the problem. The main advantage of event aggregation is the creation of temporal frames over a fixed duration. These frames, also called events frames [6], [24], [27], are created by aggregating the asynchronous stream of events over a fixed time window. Time-aggregation based neuromorphic event processing is beneficial because it increases efficiency by bringing together a group of distinct events into context with each other. In the following, we highlight some of the recent works showing great potential of time-aggregation based neuromorphic vision sensor data processing in diverse applications.

A. Time-aggregation based visual classification task

The authors in [26] explore the benefits of spatial and temporal downsampling of neuromorphic vision sensor data. The authors refer to temporal downsampling as what we call here ”spike events temporal aggregation”. The authors studied the classification task accuracy on an established neuromorphic dataset. The event based visual classification was done by utilizing the Synaptic Kernel Inverse Method (SKIM). The SKIM based classifier utilized 1000 hidden neurons and performed classification tasks on a wide variety of dataset such as N-MNIST, SpikingMNIST and N-Caltech101. According to the SKIM based experiments, the classification accuracy increases significantly when temporal aggregation of spike events is employed, for instance, the accuracy increases to 85.05 % with the temporal resolution of 8 ms on the N-MNIST dataset. Similarly the temporal resolution of 8 ms achieves the highest accuracy for the SpikingMNIST dataset. For the N-Caltech101 dataset, the classification accuracy increases up to the temporal resolution of 20 ms. The improved classification accuracy results from the fact that increase in temporal accumulation of spike events decreases the sparsity of the input pattern which results in more information in each time step.

B. Time-aggregation based motion estimation in autonomous driving

The authors in [6] proposed a deep neural network approach where DVS is employed to perform a challenging motion estimation tasks, *i.e.,* prediction of steering angle of a self-driving car. The strategy converts asynchronous spike events into frames over a specified frame rate. The process of event-to-frame conversion is done by aggregating spike events over a specified time interval. The resulting synchronous event frames are fed into a 50 layer deep residual network (ResNet-50) which predicts the steering angle of the autonomous car. According to the detailed experimental analysis, spike event frames with a frame rate of 20 fps (events accumulation time of 50 ms) produces the best prediction of the steering angle (root-mean-squared error of 9.74°). The authors also compared the event frame-based approach with that of the grayscale frames from conventional camera. According to the results, during a sunny day, the grayscale frame-based approach suffers from camera saturation and lack of temporal information which produces wrong steering angle prediction. Furthermore, poor illumination during night causes wrong motion prediction when conventional camera is utilised.

C. Time-aggregation based classification in tactile sensing

The authors in [27] proposed the first event based tactile sensing by utilizing the DVS camera. The authors utilized two time-series machine learning methods, Time Delay Neural Network (TDNN) and Gaussian Process (GP), to estimate the contact force in a grasp. Furthermore, the authors proposed Deep Neural Network (DNN) to classify object materials by using DVS based tactile sensing framework. According to several experimental studies to classify four different materials, spike data aggregation over 7 ms achieves the best accuracy, 79.17 %. Furthermore, the same time aggregated spike data successfully estimate the contact force with a mean square error of 0.16 N for TDNN and 0.17 N for GP. The framing of events over 7 ms intervals reduces the impact of noise and results in differentiation of meaningful events.

D. Time-aggregation based incipient slip detection

The authors in [7] proposed another novel industrial application of the DVS in tactile sensing. The authors proposed an approach to detect incipient slip based on the contact area between transparent silicone medium and different objects by utilizing the neuromorphic vision sensor. The authors proposed a fixed window cycle of 10 ms to construct a frame based on spike events produced during the grasping and releasing phase. In order to obtain meaningful features, the spike events triggered in the time period of 10 ms are aggregated. Therefore, each pixel of the event frame contains an accumulated event count. According to rigorous experiments, the results indicate an accurate detection of incipient slip, stress distribution and object vibration with very low latency.

E. Time-aggregation based object detection

In [28], the authors proposed a Convolutional Neural Network (CNN) architecture called YOLE for object detection by utilizing the neuromorphic vision sensor. The authors utilized a frame based model as an input to the CNN, where each pixel in the frame integrates neuromorphic spike events over time. The authors group events into batches of 10 ms with an input frame resolution of 128 $\times$ 128. According to the experimental analysis, the proposed method is not only able to detect objects but also their direction and position.

IV. Proposed strategy (TALVEN)

The aforementioned strategies process DVS data by aggregating events at fixed time intervals. A wide variety of machine learning strategies then process the time-aggregated events to perform different tasks in diverse scenarios. Therefore, it is worth investigating the compression gains achieved when time-aggregation of spike events is employed. In this section we propose a compression method which takes into account the time-aggregation of spike events.

According to the mathematical analysis in [12], [13], the DVS spike firing mechanism has the following important features:

- A linear increase and decrease of luminance intensity on a certain pixel produces temporal correlation between the consecutive events.
- Adjacent pixels receive almost the same luminance intensities simultaneously which indicates the existence of spatial redundancies.

The aforementioned features show that a DVS spike sequence exhibits spatial and temporal correlation. Video compression is a form of source coding where spatial, temporal and statistical redundancies are exploited to store the relevant information more compactly. Therefore, we propose to apply video encoding to an appropriately processed form of the asynchronous stream of DVS spike events. In order to utilize the benefits of video compression, we transform the event stream into a format mimicking video and exhibiting high spatial and temporal correlation. Figure 2 shows the basic blocks of the proposed strategy, where the DVS spike event sequence is converted into synchronous video frames. In the following, we analyse the basic blocks of the proposed strategy.

A. Event Frame

The advantages of reduction in data size through temporal aggregation of events can be exploited by projecting the DVS spike event stream into a sequence of frames, where each frame has the full resolution ($M \times N$) of the pixel array. The projection of DVS spike sequence in a frame to the XY plane is done by recording the location histogram count, *i.e.*, recording the number of event count at each pixel. For instance, Figure 3 shows the location histogram count in four scenarios of different temporal resolution factors of 1 ms, 2 ms, 5 ms and 10 ms. According to the figure, time resolution of 1 ms represents 36 events with 36 bytes, *i.e*, each pixel has an event and each event count is represented by a byte. The number of bytes required to represent 222 events, for a resolution of 10 ms, remains the same. Therefore, the projection of spike sequence as a frame representing event count at each pixel inherently reduces the size of the data.

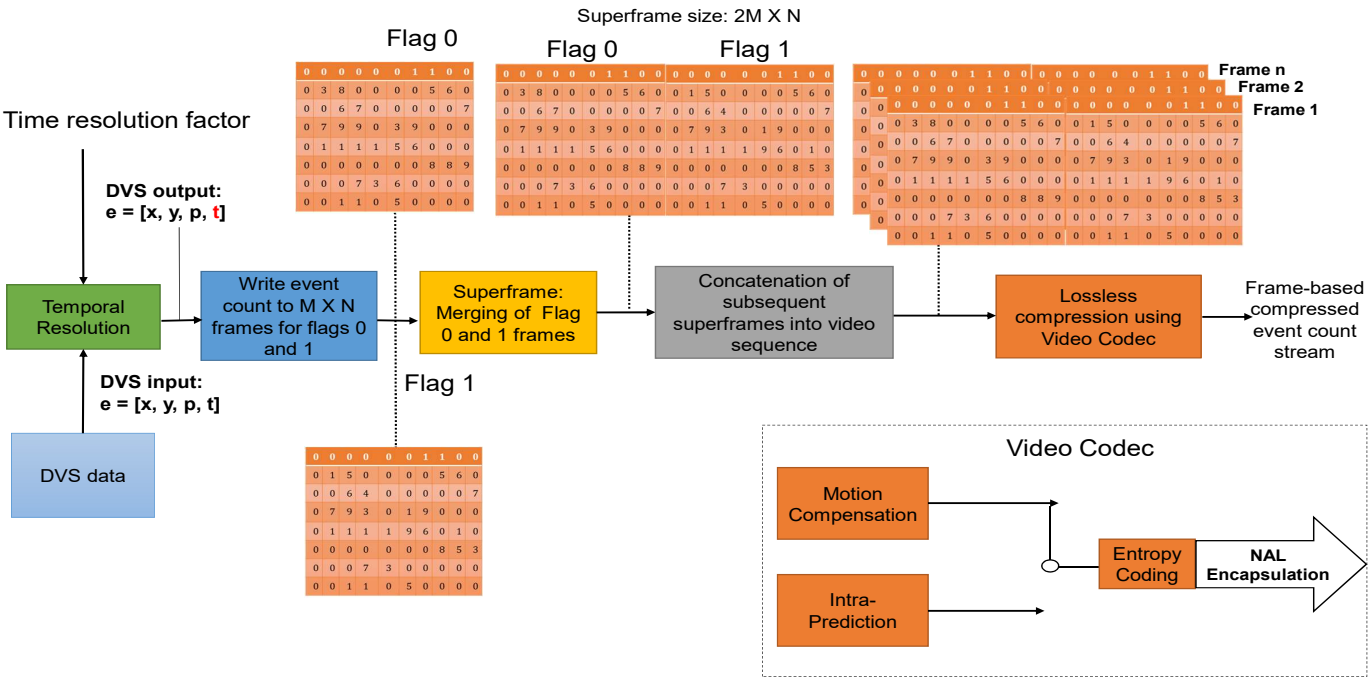

Figure 2: Block diagram of Time Aggregation based Lossless Video Encoding for Neuromorphic Vision Sensor Data (TALVEN).

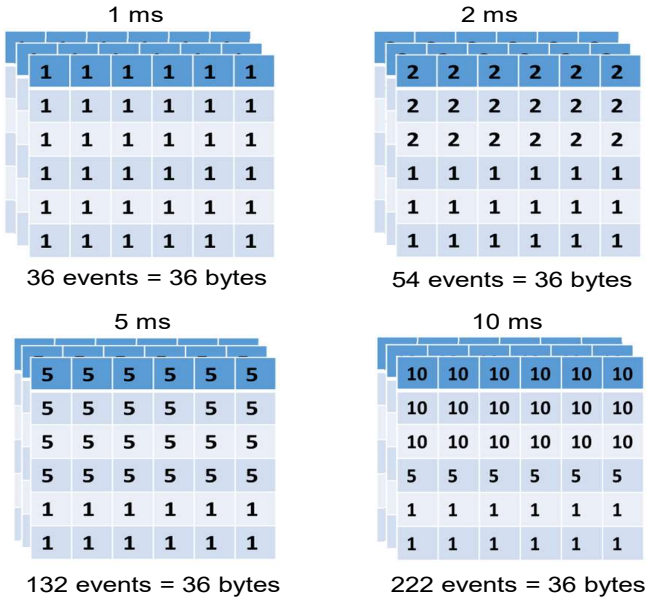

Figure 3: Event frames containing the event count at four different time resolutions.

B. Separate Event Frames for Each Flag

We propose to divide the spike sequence into two separate frames, one associated to positive increase of luminance intensity (flag 1) and one for decrease of luminance (flag 0). As highlighted in the mathematical model proposed in [12], [13], there exists a strong correlation between the polarities of co-located spike events. For instance, if the flag of the previous event is one (or zero) on a pixel, there is a high probability that the next event polarity will be one (or zero) on the same pixel. This is mainly because of the smooth change in luminance. For feature extraction, the authors in [30] divide the events in spatio-temporal surfaces of flag zero and one mainly because there exists a strong correlation between time surfaces of the same polarity. Therefore, we propose to record event count separately for each polarity with each frame having the full resolution of the pixel array. This would increase the temporal correlation between the frames of the same polarity, which can then be exploited by applying the video compression concept of interframe coding.

C. Arranging data in "superframes"

We propose to merge the frames of each polarity with the same timestamp into one single *superframe* composed of the "0 polarity flag" frame on the left and the "1 polarity flag" frame on the right. The main rationale behind the creation of such "superframes" is the requirement to arrange data in a way resulting in high interframe correlation, that can be exploited by a video encoder. For instance consider the three possible alternative scenarios shown in Figure 4; subsequent frames report events of opposite polarity in scenario one, whereas in scenario two consecutive frames report events with either the same or inverse polarity. With these data arrangement, interframe correlation is low, hence resulting in inefficient compression if a video encoder is used on these frames. In scenario 3, where both the polarity frames are merged into one *superframe*, subsequent frames have high interframe correlation that can be effectively exploited by a video encoder used on a "video sequence" composed of these *superframes*. This is mainly because of the increase in probability of finding the best matching block as the neighbouring frame in time now includes both the polarity frames. It is important to note that

each frame requires extra header information (for instance, frame number, frame type, etc.); therefore, another advantage of the creation of such *superframes* is the lower proportion of header data through reduction of frame rate. For instance, without it the temporal resolution of one millisecond produces 2000 fps, whereas with frame concatenation, the frame-rate reduces to 1000 fps.

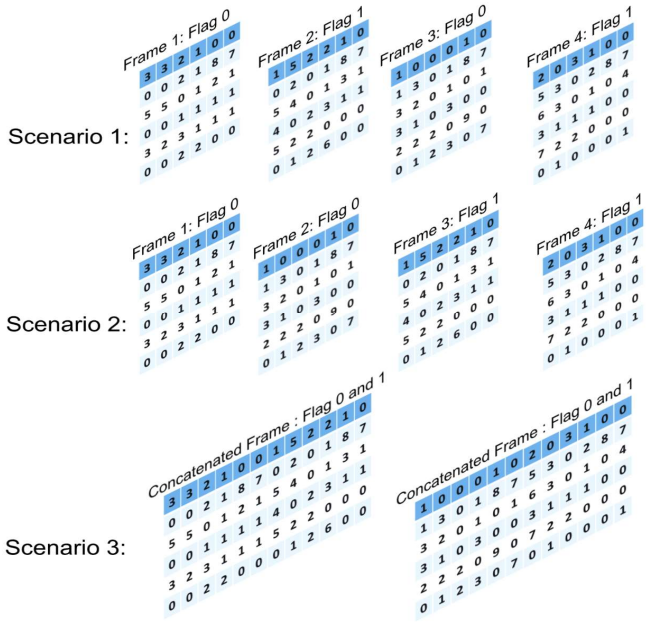

Figure 4: Scenario 1: Neighbouring event frames have inverse polarity. Scenario 2: Neighbouring event frames have same polarity. Scenario 3: Neighbouring concatenated frames (superframes) have both the polarities.

1) Impact of superframes on compression ratio: The arrangement of spike event data in superframes doubles the frame size. In the following, we discuss the three important benefits of the superframes; due to these, the increase in frame size has minimal impact on the potential compression gains.

- Superframes have better temporal correlation: Superframes have concatenated $M \times N$ spatial regions each for flag zero and one. The components with the same polarity in successive frames have high correlation. The interframe coding algorithm of the video encoding exploits this correlation to maximize the compression gains.
- Superframes have built-in information about polarity of events: For instance, if we consider a frame size of $M \times N$ (instead of a superframe resolution of $2 \times M \times N$) then the per-pixel count would represent cumulative events of flag zero and one. In this scenario, not distinguishing between the polarity of the spike events. In such a case, the information about the polarity of the events should be separately encoded and then integrated in the compressed stream. The separate encoding and integration would limit the compression gains.
- Superframes double the maximum limit of the event count: Furthermore, a frame size of $M \times N$ would represent $2^n - 1$ (where n is the bit depth of the encoder, *i.e.*, 255 limit for 8-bit encoder) number of cumulative events for polarity zero and one. On the other hand, superframe doubles the maximum limit of the event count to $2^n + 2^n - 2$ (510, for 8-bit encoder), *i.e.*, $2^n - 1$ each for polarity zero and one events. Experimental analysis of the TALVEN strategy on diverse dataset shows that with superframe resolution and 8-bit video encoder, the maximum limit is never exceeded.

D. Video Encoding

Next, we propose to utilize video encoding by representing each *superframe* as a video frame. The *superframe* contains all the spike sequence information as shown in Figure 4 (scenario 3). The frame number field in the video header represents the timing information, *i.e.,* if the temporal aggregation is 10 ms then the timing information for all the events recorded in frame number 125 is 1250 ms.

In the following, we discuss the main encoding steps of compression exploiting spatial, temporal and statistical correlation among the event frames associated to the DVS sequence.

1) Interframe coding: In conventional video coding, interframe coding is the key technique in achieving high compression ratios. The arrangement of data of a spike sequence as an event frame, where each pixel stores the event count, provides the opportunity to exploit the most important concept in video compression. In interframe coding, a reference frame is used to predict the current frame. Frame prediction is done through a concept called block-based motion compensated prediction (MCP). A frame is divided into blocks, and for each block in the current frame, a motion vector is identified based on the address of the matching block in the reference frame. Figure 5 shows how interframe coding is performed on neuromorphic event count by utilizing MCP. In the example shown in the figure, the temporal correlation between adjacent blocks is low. However, the flexibility of MCP to find the best matching block over the entire pixel array of a frame, and over several neighbouring frames, has the potential of achieving higher compression.

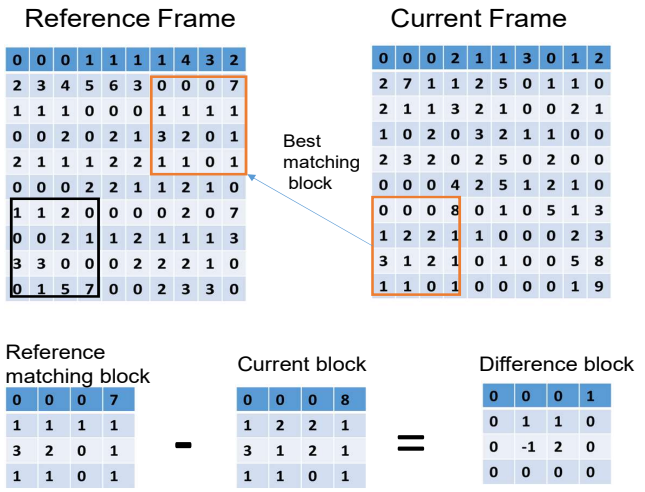

Figure 5: Interframe coding application to the event frames. Adjacent blocks between the two frames show low correlation. MCP finds the best matching block.

2) Intraframe coding: In video compression, intraframe coding exploits spatial redundancy by exploiting the correlation among neighbouring pixels. According to the DVS spike generation mechanism, adjacent pixels receive almost the same luminance intensities simultaneously. In other words, if a pixel receives a polarity zero (or one) spike then there is a high probability that the neighbouring pixels have undergone a polarity zero (or one) spike recently. The proposed arrangement of DVS data in *superframes* results in higher spatial correlation because the adjacent pixels within a *superframe* have the same polarity. The same polarity spike event count, among the neighbouring pixels, is exploited by different prediction modes of intraframe coding. For instance, there are 9 prediction modes for intraframe coding in the H.264 video coding standard, whereas in HEVC (H.265) the number of prediction modes increases to 35. These modes exploit spatial redundancy within a frame by computing prediction values through extrapolation.

3) Entropy coding: Interframe coding results in motion vector (address pointing the position of the best matching block) and prediction residuals (difference between the two matched blocks). On the other hand, intraframe coding results in spatial residual prediction by utilizing the neighbouring pixels. In order to achieve further compression, these residuals and motion vectors are entropy coded. The entropy coding utilized by the most recent standards, such as H.264 and HEVC, is a lossless compression technique called Context Adaptive Binary Arithmetic Coding (CABAC).

Algorithm 1 TALVEN

Input: Temporal resolution T_r [s]
Input: DVS spike sequence duration T_{seq} [s]
Input: Total frames to encode $T_{frames} = \frac{T_{seq}}{T_r}$

for $t = 1$ to T_{frames} **do**
  for $t = 1$ to T_r **do**
    Write event count in an $M \times N$ frame for flag 0
    Write event count in an $M \times N$ frame for flag 1
  end for
  Merge both the event frames into a superframe F for video encoding
  if $F ==$ frame type I **then**
    Apply intraframe coding
  else
    Apply interframe coding
  end if
  Apply entropy coding to the residuals of interframe or intraframe coding
  Apply NAL encapsulation to the compressed frame F
end for
Output: Compressed event frame sequence

4) NAL units: One of the advantages of utilizing a video encoding format for neuromorphic vision sensor data is the incorporation of Network Abstraction Layer (NAL) units in the video coding standards. NAL units provide a network friendly representation of neuromorphic vision sensor data. The increasing and diverse applications of DVS, ranging from autonomous driving to drones, call for a flexible representation of the compressed data. The main advantage of NAL units is their flexibility to map the coded data to diverse transport and network layer protocols such as RTP/IP, and TCP/IP. Furthermore, NAL provides different ways to pack a NAL unit stream ranging from packet-stream format to byte-stream format, where the former is used for transmission applications while the latter is used for storage applications. Another important design goal of NAL is to provide robustness against data loss by providing different modes of transmission such as out-of-band and in-band transmission.

E. Pseudocode of the proposed strategy

Algorithm 1 summarizes all the steps of the TALVEN strategy. The total number of frames encoded, T_{frames}, depends upon the DVS spike sequence duration T_{seq} and the event aggregation time interval T_r (temporal resolution). There are three types of encoded frames in video compression. An I-frame is the result of intraframe coding and is the least compressible frame. B and P frames are the result of interframe coding, where a P frame is predicted from a single reference frame and a B frame is predicted from two neighbouring frames. The residuals of the encoded frames are entropy coded, followed by the NAL encapsulation step as shown in Algorithm 1.

V. Simulation Setup

A. Dataset

The compression performance of the proposed and benchmark strategies is evaluated by utilizing Dynamic and Active-pixel Vision Sensor (DAVIS) dataset [31]. The dataset was produced by a hybrid sensor technology, DAVIS, which outputs an asynchronous event output stream as well as classical frame based intensity images with a spatial resolution of 180×240. Furthermore, the dataset also includes the motion speed information of every sequence. The dataset is comprised of outdoor and indoor scenes as shown in Figure 6. These scenes are captured in different conditions including indoor, outdoor and different types of motion (angular, linear etc.). In order to asses the compression gains of the proposed and benchmark strategies, we extract sequences with different scene complexity and motion speed as shown in Table I. According to the table, the extracted *Boxes* sequence has a very high event rate, approximately 4.3 Mega-events/s. This is mainly because of the high scene complexity and very high motion speed of the sensor. On the other hand, the *Shapes* sequence has a low event rate because of the low scene complexity and low sensor speed as shown in Table I. The *Dynamic* sequence is captured in an office environment with camera motion as well as a person moving in the scene. The *Outdoor* sequences in the dataset were acquired in an urban environment with camera at both running and walking speed. We extracted the *Running* sequence with three different running speeds generating different event rates as shown in Table I. The *Urban* sequence has slow walking speed in a dense urban environment.

In the following, we report the simulation setup for the benchmark and proposed strategies. Section V-B reports the AER data format of the asynchronous event stream. Furthermore, Section V-B also reports the simulation setup for all the

Figure 6: Different types of scenes in the considered DAVIS dataset [31]. The sensor moves with different types of motion (angular, linear, etc.) in front of the indoor scenes, whereas for the outdoor scenarios the body-mounted sensor moves with different walking and running speeds.

considered benchmark strategies discussed in Section II and reported in Figure 1.

Table I: Extracted dataset for experimental analysis.

	Sequence	Event Rate (kev/s)	Sequence Duration (s)	Scene Complexity	Speed
Indoor	Boxes	4288.65	5 (45-50)	High	High
	Poster	4021.1	5 (45-50)	High	High
	Dynamic	1077.73	20 (1-20)	Medium	Medium
	Slider	336.78	3 (1-3)	Medium	Low
	Shapes	245.61	20 (1-20)	Low	Low
Outdoor	Running3	1525.5	20 (40-60)	Medium	High
	Running2	1229.4	20 (20-40)	Medium	Medium
	Running1	713.8	20 (1-20)	Medium	Medium
	Urban	503.04	10 (1-10)	High	Low
	Walking	342.2	20 (1-20)	Medium	Low

B. Simulation set-up for the benchmark strategies

The simulation setup for the benchmark strategies, discussed in Section II-B, is shown in Figure 7. According to the figure, the spike event stream of neuromorphic vision sensors is represented by the AER data format, where each event is 8-byte long. The least significant four bytes (1-32 bits) of the AER data represents the timestamp information, whereas the retinomorphic vision sensor type, Asynchronous Time-based Image Sensor (ATIS) or DVS, is represented by the most significant bit (64^{th} bit). The ATIS has built in temperature, gyroscope and acceleration sensors. Analog-to-Digital converted samples of these integrated sensors are conveyed by 10-bit (33-42 bits). The polarity flag and trigger (trigger bit is switched on when polarity changes) information is conveyed by the 43^{rd} and 44^{th} bit of the AER representation respectively. Finally, X and Y spatial addresses are represented by 10-bit (43-52 bits) and 9-bit (53-61 bits) respectively.

We convert a series of 64-bit AER data into a multivariate stream with seven columns (each 8-bit long). The conversion is performed by extracting the spatial addresses, timestamp, and polarity flag information from the AER data, as shown in Figure 7. According to the figure, the four columns constitute the timestamp information of the spike events. The spatial addresses (X and Y) and the polarity flag comprise of the remaining three columns. Finally, the multivariate stream is transformed into row-major and column-major formats. IoT specific compression strategies employ the row-major format [20], whereas the fast integer compression, entropy and dictionary based algorithms utilize the column-major format.

For spike coding strategy, the size of the macro-cube is 180×240 in spatial and 32768 (number of events) in temporal [13].

C. Simulation set-up for TALVEN

In order to asses the time-aggregation based compression gains on the considered dataset, we utilize an H.265 (HEVC) video encoder wrapper (x265) in FFmpeg library libx265. In HEVC lossless encoding, DCT (Discrete Cosine Transform) and quantization are bypassed which results in low complexity implementation of the encoder. The main goal of the encoder is to find the optimal intracoding and intercoding predictions and then losslessly encode the residuals. The preset option selects the compression ratio and speed tradeoff, *i.e.*, the higher the preset the higher the compression gains. We selected the default medium preset for compression speed. Furthermore, we selected 6 different (1 ms, 5 ms, 10 ms, 20 ms, 40 ms and 50 ms corresponding to 6 different frame rate) temporal resolutions for each of the considered scenes.

D. Key performance metrics

- *End-to-end compression ratio* (compression ratio *w.r.t* total number of events). The performance of the considered and benchmark strategies is evaluated by computing the compression ratio: $\frac{N_{event} \times 64}{\gamma}$, where γ is the size (in bits) of the compressed output stream and N_{event} is the total number of spike events, with each event equal to 64 bits.
- *Video encoder compression ratio* (compression ratio *w.r.t* input frame size). Since video encoding is one

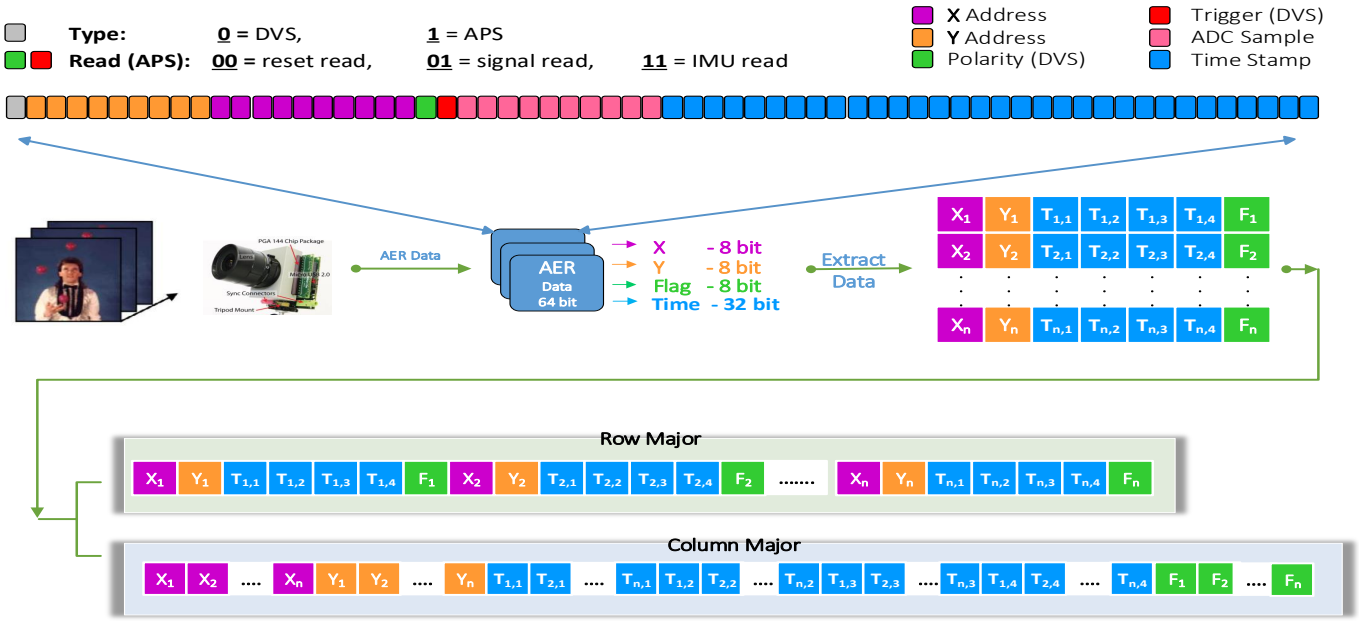

Figure 7: Experiment setup for the evaluation of benchmark strategies [10]. The top part of the figure shows the AER data format.

of the key steps of TALVEN, the performance of the video encoder is evaluated by computing the ratio: $\sum_{1}^{T_{frames}} \frac{\text{size of uncompressed frame [bits]}}{\text{size of compressed frame [bits]}}$, where T_{frames} is the total number of encoded frames.

VI. Compression Performance of the Existing Strategies Without Time Aggregation of Spike Event Data

The end-to-end compression performance (*w.r.t* total number of events) of all the considered benchmark lossless compression strategies is reported in Table II. The cube-based spike coding mechanism achieves the best compression ratio as shown in the table. Instead of compressing DVS sequence as a multivariate stream of integers, spike coding strategy projects DVS spike sequence into multiple macrocubes. Spatial redundancy within the macrocube is exploited by considering the correlation among the neighbouring pixels (intracube prediction). On the other hand, temporal correlation among the neighbouring macrocubes is exploited through intercube prediction. The residuals of the both the prediction types are fed to the CABAC thus achieving further compression.

Dictionary based compression strategies yield the second best compression performance as shown in Table II. Among the considered dictionary based compression strategies, LZMA, Brotli, Zlib and Zstd show better compression gains. These strategies utilize a huge variable size dictionary (up to 4GB) with the output of the dictionary-based compression further processed by an entropy coding step. For instance, LZMA utilizes a binary arithmetic encoder which encodes the output stream of dictionary phrases on a bit-by-bit basis, thus yielding impressive compression gains. On the other hand, Brotli, Zlib and Zstd use Huffman coding as the final compression step. In order to achieve superior decompression speed, Zstd utilizes Finite State Entropy (FSE) before the Huffman coding which results in a slight decrease in compression ratio as compared to Zlib and Brotli. On the other hand, Brotli uses second order context modelling before the Huffman coding step which improves its compression performance as compared to Zlib and Zstd.

IoT specific compression strategies yield low compression performance. This is mainly because these strategies encode only 8 rows of data, *i.e.*, compression is performed on 8 consecutive spike events. This type of encoding works well for slow changing time series data, for instance temperature measurements, where the correlation among the consecutive data samples is very high. However, the correlation among the 8 consecutive spike events is generally low mainly because the probability of consecutive events having similar spatial addresses and time stamps is low. Sprintz-Delta-Huf and Sprintz-FIRE show approximately the same compression performance. This shows that both Delta coding and Sprintz forecasting algorithm FIRE result in similar residuals of the consecutive spike events. The performance of Sprintz-Delta is the worst among the variants of the IoT specific approach. This is mainly because the residuals of the Delta coding are not Huffman coded. This shows that Huffman coding as the final encoding step enhances the compression gains of the Sprintz strategy. It is important to note that Huffman encoding as a standalone compression approach yields low compression gains, only 1.87. The combination of Huffman coding with other encoding frameworks (Dictionary and IoT specific strategies) improves the compression performance as reported in Table II. Fast integer compression strategies (Memcpy, Simple8B, SIMD-BP128 and FastPFOR) yield the worst compression gains. The main design goal of these strategies is the fast compression and decompression speed.

It is important to note that the higher event rate sequences result in better compression performance as compared to the low event rate sequences as shown in Table II. For instance,

Table II: End-to-end compression ratio (*w.r.t* total number of events) comparison.

	Sequence	Spike Coding	LZMA	Brotli	Zlib	Zstd	LZ4	Sprintz Delta-Huf	Sprintz FIRE	Sprintz Delta	Huffman	Snappy	FastPFOR	SIMDBP128	Simple8B	Memcpy
Indoor	Boxes	4.95	4.92	4.38	4.21	4.13	3.03	3.83	3.72	2.83	1.96	2.98	1.4	1.38	1.25	1.12
	Dynamic	3.85	3.34	3.19	3.13	3.07	2.46	2.68	2.63	2.33	1.89	2.44	1.31	1.28	1.15	1.12
	Poster	4.88	4.77	4.26	4.12	4.02	2.97	3.7	3.6	2.76	1.96	2.92	1.4	1.37	1.24	1.12
	Slider	3.84	3.19	2.85	2.89	2.76	2.3	2.65	2.62	2.36	1.79	2.26	1.41	1.36	1.23	1.12
	Shapes	3.78	3.04	2.78	2.8	2.67	2.19	2.46	2.43	2.26	1.79	2.21	1.34	1.31	1.17	1.12
Outdoor	Running 1	3.68	3.25	3.09	3.05	2.96	2.39	2.6	2.58	2.33	1.87	2.37	1.35	1.32	1.18	1.12
	Running 2	3.92	3.41	3.26	3.19	3.13	2.51	2.7	2.67	2.35	1.89	2.48	1.3	1.27	1.15	1.12
	Running 3	3.97	3.49	3.32	3.26	3.21	2.56	2.75	2.72	2.36	1.9	2.53	1.29	1.26	1.14	1.12
	Urban	3.45	3.13	2.93	2.91	2.83	2.32	2.58	2.54	2.31	1.83	2.31	1.36	1.35	1.22	1.12
	Walking	3.54	3.11	2.89	2.88	2.8	2.24	2.53	2.52	2.3	1.84	2.24	1.35	1.31	1.18	1.12
Total Average		**3.99**	**3.57**	**3.30**	**3.24**	**3.16**	**2.50**	**2.85**	**2.80**	**2.42**	**1.87**	**2.47**	**1.35**	**1.32**	**1.19**	**1.12**

the *Poster* sequence exhibits very high event rate (4.01 Mega-events/s), whereas the *Shapes* sequence has the lowest event rate (0.245 Mega-events/s). Since the *Shapes* sequence has very low scene complexity and low speed of the sensor, intuitively this sequence should achieve high compression gains. However, if we compare the compression performance of both the sequences, then the *Shapes* sequence achieves the lowest compression ratio, whereas *Poster* sequence yields higher compression gain. In order to find the rational behind this key observation, we computed histogram and entropy of high and low event rate sequences data as reported below.

A. Histogram and entropy computation of different fields of the spike event data

The histogram and entropy of the spatial addresses (X and Y) and time stamp (delta coded) fields of the *Shapes* and *Poster* sequences are reported in Figure 8. The higher the entropy, the higher the uncertainty in the data, which results in low lossless compression ratio. The spatial address is the least compressible field of the DVS spike sequence as shown by the entropy computation of spatial addresses X and Y in Figure 8 (the entropy is above 7 for both the sequences for both X and Y, being $log_2(180) = 7.492$ the entropy of the equivalent "uncompressible source" for Y and $log_2(240) = 7.907$ for X). Hence, both the sequences results in low compression gains for the spatial address field. The timestamp of each spike event consumes 32 bits; however, the entropy of the delta-coded time stamp is very low for both the sequences. This shows that this field is highly compressible as compared to the spatial address field. The range of values (histogram bins) for the *Poster* delta coded time stamp is only 0 and 1, whereas for the *Shapes* sequence the series of intervals ranges from 0 to 20. This is mainly because a spike event is elicited every microsecond for the high event rate sequence as shown by the delta-coded time stamp histogram. This shows that the time stamp information for the *Poster* sequence can only be represented by one bit (0 to 1) only, whereas the *Shapes* sequence requires at least 5 bits (0 to 31). Therefore, the compression performance of high event rate sequences is better than the low event rate sequences as reported in Table II.

The compression ratio achieved by all the considered benchmark strategies is very limited. For instance, the average compression performance of the Spike coding is 3.99 which means that on average, $\frac{64}{3.99}$, 16.04 bits are required for each spike event. The compressed data rate is still high; for instance, the bit rate of the *Boxes* sequence for spike coding is 55.44

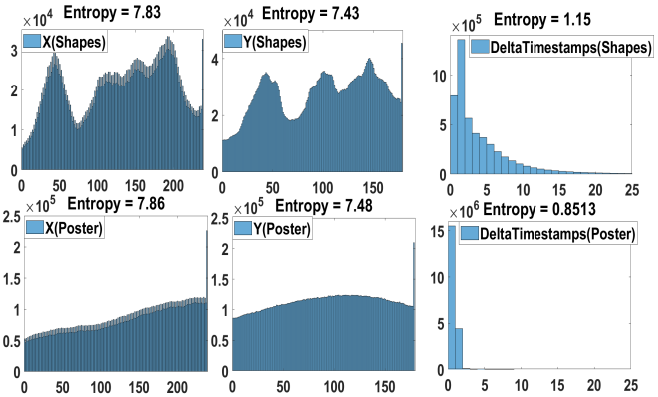

Figure 8: First Row: Histogram of the spatial address (X and Y) integers and delta-coded time stamps of the *Shapes* scene. Second Row: Histogram of the spatial address and delta-coded time stamps of the *Poster* sequence.

Mbps, whereas the low complexity *Shapes* sequence results in 4.158 Mbps according to the compression ratios reported in Table II. In the subsequent section, we analyze the compression gains achieved when time aggregation of spike events is employed.

VII. Comparative performance analysis of the proposed and benchmark strategies

A. TALVEN performance for different contents and comparison with benchmarks with no time aggregation

Figure 9 presents the compression performance of the TALVEN approach for the considered outdoor and indoor sequences, detailing the performance of the subsequent steps in the TALVEN strategy.

The global compression performance, i.e., *w.r.t* the number of spike events, is reported in the upper part of the figure. According to the first row of figure, a massive increase in compression performance is observed for increasing time aggregation of spike events (from 1ms to 50ms). If we compare the results of TALVEN (hence with time aggregation) with the results of the strategies in Table II (without time aggregation), we observe that the compression ratio of the *Boxes* sequence is increased from 4.95 (compression performance of the best strategy in Table II) to 9.717 (TALVEN compression performance) under time aggregation of 1 ms. Higher time aggregation intervals result in even higher performance increases: the compression ratio increases to 21.74 (339 % increase) and 77.28 (1461

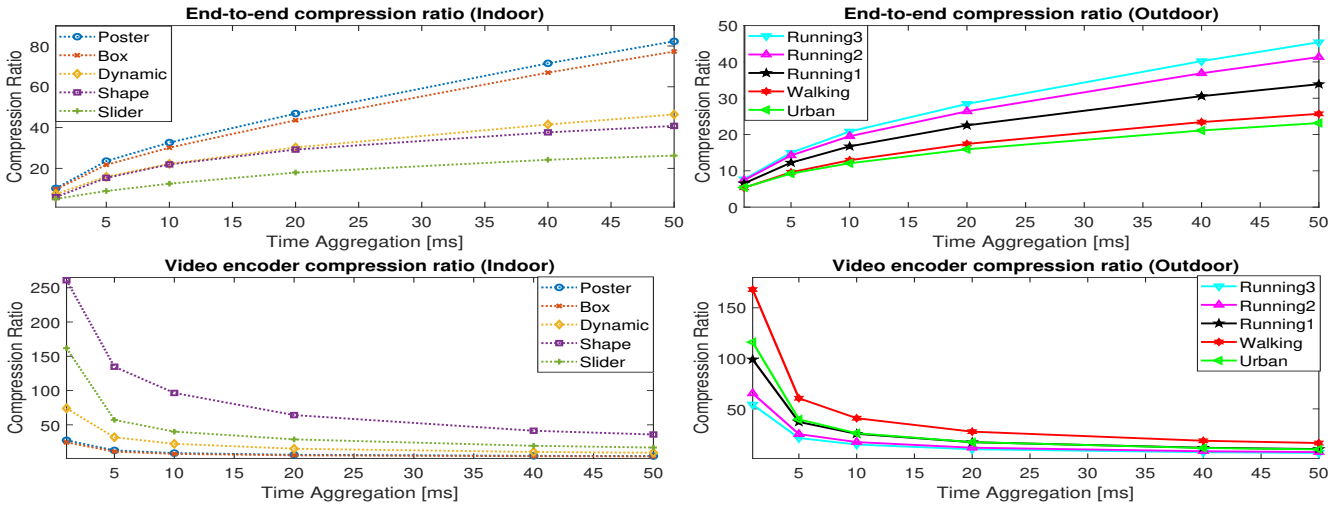

Figure 9: First row: end-to-end compression gains (i.e. *w.r.t* total event count) of the TALVEN strategy. Second row: Compression ratio of the video encoding step of the TALVEN.

% increase) under aggregation time of 10 ms and 50 ms respectively as shown in Figure 9.

1) Video encoding compression gains of TALVEN: The second row compares the compression gains of the video encoder section for the considered scenes, *i.e.,* size after compression *w.r.t* the input frame size. We observe that the best compression gain in this step is achieved for the *Shapes* sequence. This is mainly because the *Shapes* sequence exhibits low scene complexity and low sensor speed. Therefore, the subsequent event frames have very high correlation, which is exploited by the interframe coding. The compression ratio achieved for the *Shapes* sequence is close to 100 (encoder compression performance) at the time aggregation interval of 10 ms, as reported in Figure 9. On the other hand, the *Boxes* sequence yields the lowest compression gain; for instance the compression ratio is only 7.59 at the temporal downsampling factor of 10 ms. The *Boxes* sequence exhibits very high scene complexity, coupled with high speed of the vision sensor, which leads to low temporal correlation between the subsequent frames. Similarly for the outdoor sequence of *Running3*, the compression gain is the minimum; whereas the *Walking* sequence achieves the best compression performance among the considered outdoor scenarios.

Another important observation is the decrease of video encoding compression gain of all the considered strategies with the increase of temporal downsampling interval, as shown in Figure 9. At lower time aggregation intervals (1 ms achieves 1000 fps), the temporal correlation between the frames is very high which leads to high temporal redundancy and hence high compression gains. The increase in size of the time aggregation interval decreases the temporal correlation, which results in lower compression gains for all the considered scenes.

2) End-to-end compression gains of TALVEN: The end-to-end compression performance, reported in the upper part of Figure 9, is the lowest at temporal downsampling factor of 1ms. This is mainly because the number of events per frame is very low, *i.e.,* the spike events (per second) are projected over 1000 fps. On the other hand, time aggregation of 50 ms yields the best compression gains *w.r.t* total spike events because only 20 frames (per second) carry the same number of spike events. However, this does not imply that the higher the event rate of the sequence, the higher the compression gain *w.r.t* the number of spike events. For instance, the compression ratio (*w.r.t* the total number of spike events) of the *Shapes* sequence is higher than the *Slider* one, even though the event rate of the *Slider* (336.78 Kilo-events/s) sequence is 37.12 % higher than the *Shapes* sequence (245.6 Kilo-events/s). Similarly *Poster* achieves a better compression ratio *w.r.t* total spike events as compared to the *Boxes* sequence, which highlights the fact that TALVEN exploits temporal redundancy efficiently thus yielding high compression gains. This is in contrast to the compression results for the benchmark strategies without time aggregation, reported in Table II, where the higher the event rate, the higher the compression gains, irrespective of the complexity of the scene.

B. Comparative results with benchmark strategies with time aggregation

Comparative results with the benchmark strategies are reported in Figure 10, showing the end-to-end compression performance (*w.r.t* total number of events) when neuromorphic events are aggregated over constant time intervals of 1 ms, 5 ms, 10 ms, 20 ms, 40 ms and 50 ms. As benchmark strategies, we selected Spike Coding, LZMA and Brotli, since they are the three best performing strategies as highlighted in the previous section and in our tests done under different time aggregation intervals, not reported here for brevity.

A summary of the key observations from the comparative results is reported below.

1) Marginal increase for LZMA and Brotli under time aggregation: The increase in compression performance with increasing time aggregation for LZMA and Brotli is marginal. For instance, the compression performance of LZMA and Brotli for the *Boxes* sequence increases to 5.5 and 5 (under time

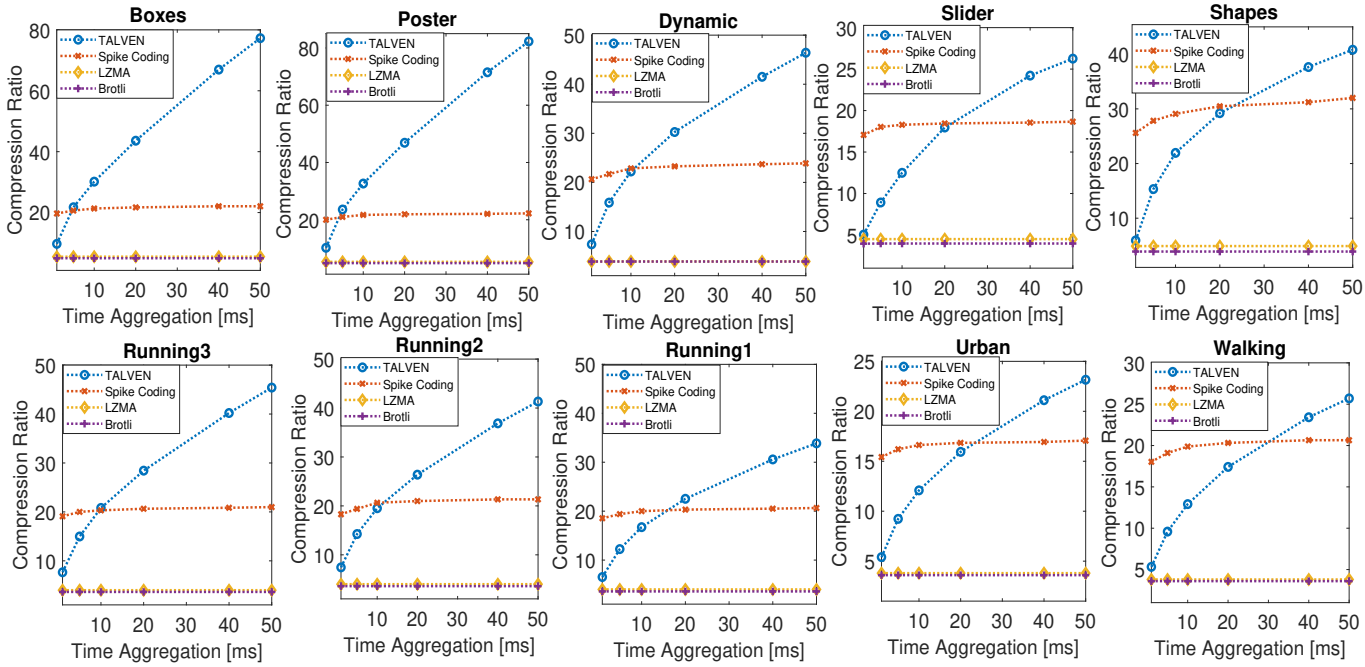

Figure 10: End-to-end time aggregation based compression gains (i.e. *w.r.t* total number of events) of TALVEN and benchmark strategies for the considered Indoor (first row) and Outdoor (second row) sequences.

aggregation of 1 ms) from 4.92 and 4.38 respectively (without time aggregation). We have observed a similar marginal improvement in compression gain for all the other benchmark strategies. A further increase in the time aggregation window size does not improve the compression performance, as shown in Figure 10. This is mainly because time aggregation of the spike events has no impact on the spatial address field (X and Y), *i.e.*, it remains unchanged. The spatial address field exhibits very high entropy as shown in Figure 8, therefore, this field achieves the same low compression performance even after the time stamp field has undergone downsampling. Hence, no further compression gain is achieved for aggregation of the spike events above 1 ms.

2) Spike coding achieves top comparative compression gains at lower time aggregation intervals: In the proposed TALVEN strategy, a frame of spike event data is created by accumulating event count over the fixed time interval (the obtained frames are equally spaced in time). On the other hand, in spike coding [12] a macro-cube (similar to the event frame in TALVEN) is created, where each macro-cube contains approximately the same number of spike events, which results in a variable time window. In this case, the number of macro-cubes to encode remains constant irrespective of the temporal downsampling interval. Each macro-cube contains 32768 events, therefore the *Boxes* sequence (with an event rate of 4288.645 Kilo-events/s) results in approximately 132 macro-cubes per second. The same number of macro-cubes are encoded at every time aggregation interval. Hence for the spike coding strategy, the compression gain increases marginally with the increase in time aggregation interval. On the other hand, the number of frames to encode in TALVEN decreases with an increase in the time aggregation interval. For instance, the temporal downsampling factor of 10 ms results in 100 fps for the TALVEN strategy. Therefore the higher the temporal downsampling, the higher the event aggregation per frame, which results in higher compression gains as reported in Figure 10. However for low event rate sequences (*Shapes* and *Slider*), the number of events aggregated per frame (at lower temporal downsampling intervals) is very low. Therefore, TALVEN results in lower compression gains as compared to the spike coding strategy at time aggregation below 10 ms, as shown in Figure 10.

3) TALVEN achieves better temporal redundancy exploitation: Similar to the motion compensation concept in video encoding, intercube prediction exploits temporal correlation in the spike coding strategy [13], where previous coding cubes are used as references to predict the current coding cube. However, the prediction residuals achieved through the intercube prediction are only for the spike event count because spike polarity and timestamps are encoded separately. Since each macro-cube contains an equal number of spike events in the spike coding strategy, the timestamp associated with each spike event is also encoded. For instance, if a pixel receives an event count of 5, then the timestamp of all the five spikes fired at the pixel must be recorded. The spike coding strategy encodes the timestamps of the spike events by applying delta coding, *i.e.,* timestamps are differential coded one after another. Similarly polarity of each event is separately coded in the spike coding strategy. Since luminance increases or decreases in a steady state, the temporal correlation between the polarities of the events on a pixel is very high. Hence, the previous spike polarity is taken as the context of the current spike polarity which is fed to the context based entropy encoder.

On the other hand, timestamps of each spike event are not required for TALVEN, instead the frame number field represents

the downsampled timestamp of all the spike events in the frame. Similarly the polarity information is embedded within the super-frames. Prediction residuals computed through video encoding (interframe coding) exploit temporal redundancy among all the attributes (event count per pixel, spike event polarities and time stamp information). Therefore, the embedding of polarity and time stamp information within a frame allows TALVEN to achieve better exploitation of temporal redundancies. For instance, consider the *Running3* sequence with 20 ms time aggregation in Figure 10, where Spike coding encodes 47 macro-cubes/s ($\frac{1525.5}{32.7}$) and TALVEN encodes 50 frames/s. Both the strategies approximately encode a similar number of event frames, however, the compression ratio achieved by TALVEN is approximately 40 % better than the spike coding strategy. Similarly 20 ms of time aggregation for the *Shapes* sequence yields approximately similar compression gains for the spike coding (7.5 macro-cubes/s) and TALVEN (50 fps) strategies as shown in Figure 10.

VIII. Conclusions

Recently several studies have shown the benefits of time aggregation of spike event data in diverse scenarios, ranging from steering angle prediction in intelligent driving to different object detection and classifications tasks in tactile sensing, robotics and computer vision. Motivated by these studies, in this paper we proposed to aggregate over fixed time windows the spike event stream and to represent such data in an appropriate form compatible with video encoding to compress the spike event data generated from the DVS. The proposed strategy transforms the asynchronous stream of DVS data into synchronous event frames, formatted in way that ensures that the subsequent video encoder can exploit the spatial and temporal redundancies in the data. According to the experimental analysis, the proposed strategy shows excellent compression gains as compared to the benchmark strategies. Increases in the temporal aggregation window size further increase the compression gains *w.r.t* the spike event rate.

References

[1] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 x 128 120 dB 15 μs Latency Asynchronous Temporal Contrast Vision Sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[2] ——, "A 128x128 120 dB 30mW asynchronous vision sensor that responds to relative intensity change," in *IEEE International Solid-State Circuits Conference (ISSCC)*, San Francisco, USA, February 2006.

[3] N. Khan and M. G. Martini, "Bandwidth modeling of silicon retinas for next generation visual sensor networks," *Sensors*, vol. 19, no. 8, p. 1751, 2019.

[4] ——, "Data rate estimation based on scene complexity for dynamic vision sensors on unmanned vehicles," in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Bologna, Italy, September 2018.

[5] J. Li, S. Dong, Z. Yu, Y. Tian, and T. Huang, "Event-based vision enhanced: A joint detection framework in autonomous driving," in *IEEE International Conference on Multimedia and Expo (ICME)*, Shanghai, China, July 2019, pp. 1396–1401.

[6] A. I. Maqueda, A. Loquercio, G. Gallego, N. Garcia, and D. Scaramuzza, "Event-based vision meets deep learning on steering prediction for self-driving cars," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake city, USA, June. 2018.

[7] A. Rigi, F. Baghaei Naeini, D. Makris, and Y. Zweiri, "A Novel Event-Based Incipient Slip Detection Using Dynamic Active-Pixel Vision Sensor (DAVIS)," *Sensors*, vol. 18, no. 2, pp. 1–17, 2018.

[8] D. Tedaldi, G. Gallego, E. Mueggler, and D. Scaramuzza, "Feature detection and tracking with the dynamic and active-pixel vision sensor," in *IEEE International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP)*, Krakow, Poland, June 2016.

[9] E. Mueggler, B. Huber, and D. Scaramuzza, "Event-based, 6-dof pose tracking for high-speed maneuvers," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, Chicago, USA, September 2014.

[10] K. Iqbal, N. Khan, and M. G. Martini, "Performance comparison of lossless compression strategies for dynamic vision sensor data," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, May. 2020.

[11] N. Khan, K. Iqbal, and M. G. Martini, "Lossless compression of data from static and mobile dynamic vision sensors-performance and trade-offs," *IEEE Access*, vol. 8, pp. 103 149–103 163, 2020.

[12] Z. Bi, S. Dong, Y. Tian, and T. Huang, "Spike coding for dynamic vision sensors," in *IEEE Data Compression Conference (DCC)*, Snowbird, Utah, USA, 2018, pp. 117–126.

[13] S. Dong, Z. Bi, Y. Tian, and T. Huang, "Spike coding for dynamic vision sensor in intelligent driving," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 60–71, Feb 2019.

[14] G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza, "Event-based vision: A survey," *arXiv:1807.09480*, 2020.

[15] Y. Collet and E. M. Kucherawy, "Zstandard - real-time data compression algorithm," July 2018, available at http://facebook.github.io/zstd/.

[16] P. Deutsch and J.-L. Gailly, "Zlib compressed data format specification version 3.3," RFC 1950, May, Tech. Rep., 1996.

[17] A. Lempel and J. Ziv, "Lempel -– Ziv — Markov chain algorithm," 1996.

[18] J. Alakuijala and Z. Szabadka, "Brotli compressed data format," *Internet Engineering Task Force*, 2016.

[19] M. Litzenberger, B. Kohn, A. N. Belbachir, N. Donath, G. Gritsch, H. Garn, C. Posch, and S. Schraml, "Estimation of vehicle speed based on asynchronous data from a silicon retina optical sensor," in *IEEE Intelligent Transportation Systems Conference (ITSC)*, Toronto, Canada, September. 2006.

[20] D. Blalock, S. Madden, and J. Guttag, "Sprintz: Time Series Compression for the Internet of Things," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 2, no. 3, p. 93, 2018.

[21] D. Lemire and L. Boytsov, "Decoding billions of integers per second through vectorization," *Software - Practice and Experience*, vol. 45, no. 1, pp. 1–29, 2015.

[22] S. H. Gunderson, "Snappy: a fast compressor/decompressor," April 2015, available at https://github.com/google/snappy.

[23] M. G. Martini, N. Khan, Y. Bi, Y. Andreopoulos, H. Saki, and M. S. Bahaei, "Challenges and perspectives in neuromorphic-based visual IoT systems and networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, May. 2020.

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

[25] M. Liu and T. Delbruck, "Adaptive Time-Slice Block-Matching Optical Flow Algorithm for Dynamic Vision Sensors," in *British Machine Vision Conference (BMVC)*, Newcastle, UK, September 2018, pp. 1–12.

[26] G. Cohen, S. Afshar, G. Orchard, J. Tapson, R. Benosman, and A. van Schaik, "Spatial and temporal downsampling in event-based visual classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 10, pp. 5030–5044, Oct 2018.

[27] F. B. Naeini, A. Alali, R. Al-Husari, A. Rigi, M. K. AlSharman, D. Makris, and Y. Zweiri, "A novel dynamic-vision-based approach for tactile sensing applications," *IEEE Transactions on Instrumentation and Measurement*, pp. 1–12, 2019.

[28] M. Cannici, M. Ciccone, A. Romanoni, and M. Matteucci, "Asynchronous convolutional networks for object detection in neuromorphic cameras," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, Long Beach, California, US, June 2019.

[29] H. Rebecq, G. Gallego, E. Mueggler, and D. Scaramuzza, "EMVS: Event-Based Multi-View Stereo—3D Reconstruction with an Event Camera in Real-Time," *International Journal of Computer Vision*, vol. 126, no. 12, p. 1394–1414, Dec 2019.

[30] X. Lagorce, G. Orchard, F. Galluppi, B. E. Shi, and R. B. Benosman, "Hots: A hierarchy of event-based time-surfaces for pattern recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 7, pp. 1346–1359, July 2017.

[31] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose

estimation, visual odometry, and slam," *International Journal of Robotics Research*, vol. 36, no. 2, pp. 91–97, 2017.