IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 25 September 2022, accepted 2 November 2022, date of publication 14 November 2022,
date of current version 23 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3222330

RESEARCH ARTICLE

Lossless Compression of Neuromorphic Vision Sensor Data Based on Point Cloud Representation

MARIA MARTINI[1], (Senior Member, IEEE), JAYASINGAM ADHURAN[1], (Member, IEEE), AND NABEEL KHAN[1,2], (Member, IEEE)

[1]Wireless and Multimedia Networking Research Group, Faculty of Engineering, Computing, and the Environment, Kingston University London, KT12EE Kingston upon Thames, U.K.
[2]Department of Computer Science, University of Chester, Chester, CH1 4BJ Cheshire, U.K.

Corresponding author: Maria Martini (m.martini@kingston.ac.uk)

This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC) through "The Internet of Silicon Retinas: Machine to Machine Communications for Neuromorphic Vision Sensing Data (IoSiRe)" Grant EP/P022715/1.

ABSTRACT Visual information varying over time is typically captured by cameras that acquire data via images (frames) equally spaced in time. Using a different approach, Neuromorphic Vision Sensors (NVSs) are emerging visual capturing devices that only acquire information when changes occur in the scene. This results in major advantages in terms of low power consumption, wide dynamic range, high temporal resolution, and lower data rates than conventional video. Although the acquisition strategy already results in much lower data rates than conventional video, such data can be further compressed. To this end, in this paper we propose a lossless compression strategy based on point cloud compression, inspired by the observation that, by appropriately reporting NVS data in a (x, y, t) tridimensional space, we have a point cloud representation of NVS data. The proposed strategy outperforms the benchmark strategies resulting in a compression ratio up to 30% higher for the considered dataset.

INDEX TERMS Neuromorphic vision sensor (NVS), neuromorphic spike events, point cloud compression, geometric point cloud compression (GPCC), silicon retinas, spike encoding, data compression.

I. INTRODUCTION

Neuromorphic Vision Sensors (NVS) [1], [2], [3], also known as event-based sensors or dynamic vision sensors (DVS), are asynchronous sensors mimicking the human visual system, reporting only discrete changes of brightness in the observed scene, differently from frame based cameras, where frames are acquired at regular time intervals. NVS events are triggered whenever there is motion of the neuromorphic vision sensor, motion in the scene, or change of light conditions in the scene, with a potential time resolution of the order of a microsecond. Hence, no data is acquired for motionless vision sensors and static scenes, and in genenral with, no changes in the illumination of the scene. These features enable such sensors to achieve wide dynamic range, low-latency, and low-power requirements. Emerging applications of NVS can indeed be found in diverse scenarios, ranging from autonomous cars [4], [5] to robotics [6], [7] and unmanned aerial vehicles [8]. 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) scenarios [9].

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 change in brightness ($L = \ln(I)$ where I is the photocurrent) occurred, t is the firing time of the spike (timestamp), and p is the polarity of the event (increase or decrease of brightness).

The spike location can be seen as represented in three dimensions by the spatial coordinates and the timestamp information in the tuple, while the polarity flag indicates the direction of brightness change.

The associate editor coordinating the review of this manuscript and approving it for publication was Davide Patti.

In particular, the polarity p is

$$p = \begin{cases} +1 & if\ L(x,y,t) - L(x,y,t-\Delta t) > \theta \\ -1 & if\ L(x,y,t) - L(x,y,t-\Delta t) < \theta' \end{cases} \quad (1)$$

where Δt is the time interval between the last triggered event in (x, y) and the current one, and θ and θ' are the (adjustable) change of brightness thresholds for the generation of events. 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. The data rate produced depends on the event rate, depending in turn on the scene complexity and on the camera speed, as highlighted in [10] and [11], where a model for the estimation of such data rates is also presented.

In a previous work [12], to reduce the data rate we proposed an approach where accumulation of spike events is the key processing step: Time Aggregation based Lossless Video Encoding for Neuromorphic Vision Sensor Data (TALVEN). We proposed to arrange the asynchronous spike events data into a format leading to effective exploitation of temporal and spatial redundancy. The strategy utilizes the lossless compression mode of recent video encoding standards and achieves higher compression ratios than previous state-of-the-art approaches. Furthermore, the compressed NVS stream has the capability to adapt to diverse transport and networking layer protocols as a result of the utilization of the video encoding step. Although the compression part of this approach was lossless, time aggregation results in a reduction in time resolution that in some specific use cases, where a very high time resolution is required, may not be beneficial. A recent similar work [13] also utilizes the concept of time aggregation and video encoding.

In order to address the limitation above, in this paper we propose a completely lossless compression strategy. We leverage point cloud compression, inspired by the observation that appropriately reporting NVS data for each polarity value p in a (x, y, t) tridimensional space we have a point cloud representation of NVS data.

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

- A strategy to compress NVS data based on point cloud compression. This is the first work where a compression strategy specifically designed for volumetric data is used for NVS data. The proposed strategy is evaluated on diverse outdoor and indoor scenes.
- A comparison in terms of compression gains of the proposed strategy with the top two (as identified in our previous work [14]) state-of-the-art strategies for fully lossless compression (no time aggregation).
- A study on compression performance and complexity of the different modalities/options of the proposed strategy (in particular different point cloud sizes).

The rest of the work is structured as follows. The state-of-the-art lossless compression algorithms that can be adopted for NVS data are summarized in Section II. Our compression approach of utilizing the point cloud representation is proposed in Section III. Experiment setup for compression evaluation is reported in Section IV. We analyse the performance of the proposed and benchmark strategies in Section V, also studying the impact of the point cloud size on compression performance and complexity. Concluding remarks are reported in Section VI.

II. STATE-OF-THE-ART COMPRESSION APPROACHES

The benchmark algorithms to compress NVS data include NVS specific compression, discussed in Section II-A, as well as general purpose and IoT specific compression methodologies, reviewed in Section II-B. Figure 1 summarises qualitatively the performance of existing approaches in terms of trade-off between compression ratio and compression delay. Such algorithms are briefly reviewed below, while we suggest interested readers to refer to the detailed review and performance evaluation of the existing strategies in [14] and [15].

A. NVS SPECIFIC COMPRESSION STRATEGIES

The spike coding strategy in [16] exploits spatial correlation by projecting the spike events in a series of macrocubes, where the X and Y coordinates of the macrocube represent the spatial information of the spike event stream. Each macrocube is coded using either Address-Prior (AP) mode or Time-Prior (TP) mode, as shown in Figure 2. In AP mode, the macro-cube comprises spatial (X and Y location) and event count information. For instance, if 10 spike events occur at location X = 45, Y = 48, then the macro-cube at the specified location contains the event count of 10. Every spatial location comprises a timing information. Locations with multiple spike events, for instance event count of 4 (as highlighted by the point "A" in Figure 2) comprise 4 associated timestamps information. The timestamp field of the spike events is separately encoded via Delta coding, as shown in Figure 2. The AP yields good compression for spatially decentralized macro-cubes. In TP mode, the spike events are organized based on the timing information, i.e., events are projected in increasing order of their timestamps. Delta coding results in efficient compression of the timestamp field. In this mode, spike events are projected w.r.t. a centre point as shown in Figure 2. A centre point is a spatial location with multiple events across neighbouring pixels. The offset of spike events w.r.t. centre point results in compression of the spatial fields. The TP mode yields high compression gains for spatially centralized macro-cubes. In both modes, the residuals of the spatial and temporal fields are fed to a Context-adaptive binary arithmetic coding (CABAC) encoder. The polarity field is separately fed to the CABAC entropy encoder. The spike coding algorithm encodes spatial and temporal fields separately, hence the algorithm does not exploit any correlation between the spatial and temporal fields.

In order to address this limitation, we propose here a lossless compression strategy for NVS data based on point cloud compression, inspired by the observation that,

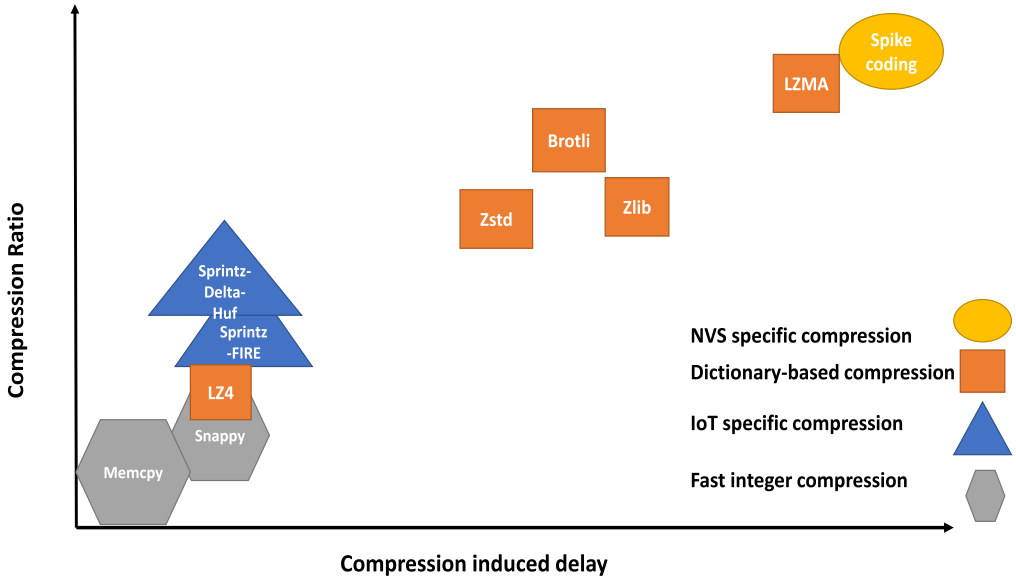


FIGURE 1. Qualitative comparative compression performance of state-of-the-art approaches on NVS data.

by appropriately reporting NVS data in a (x, y, t) tridimensional space, we have a point cloud representation of NVS data, that we can compress via specific point cloud compression strategies (see third column of Figure 2).

The work in [17] extends the spike coding framework by introducing intercube prediction which exploits temporal correlation among the macro-cubes. However, the compression gain achieved by the spike coding strategy is still quite limited; for instance, the compression ratio for the intelligent driving dataset [17] varies between 2 and 3.

In a recent work [12] we proposed to apply video encoding to an appropriately processed form of the asynchronous stream of NVS spike events. In order to utilize the benefits of video compression, in [12] we proposed to transform the event stream into a format mimicking video and exhibiting high spatial and temporal correlation. The approach outperforms state-of-the art approaches in terms of compression ratio, with a slight reduction in time resolution due to time aggregation of the events. Although the compression part of the TALVEN approach was lossless, time aggregation results in a reduction in time resolution that in some specific use cases, where a very high time resolution is required, may not be beneficial.

B. COMPRESSION STRATEGIES SUITABLE FOR NVS DATA

1) ENTROPY CODING

Entropy coders, such as Huffman and Arithmetic coding, are quite versatile, therefore they have the potential to be applied in diverse applications [18] of NVSs. They can be directly applied to the NVS data by treating each field of the spike event as an input symbol. However, 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 - in average - shorter codewords. Most of the dictionary-based strategies utilize a dynamic dictionary, whose content changes during the coding process. The advanced dictionary coders, such as Zstd [19], Zlib [20], LZMA [21] and Brotli [22], utilize multi-level encoding, where dictionary codewords are further compressed by entropy coding. The lack of repetitive pattern in the asynchronous stream of NVS data can limit the compression gains of the dictionary-based compression strategies.

3) IoT SPECIFIC COMPRESSION

The authors in [23] 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. The compression gains of the IoT specific approach on NVS data are not as high as for the data it was designed for.

4) FAST INTEGER COMPRESSION

Fast integer compression strategies are known for their excellent compression speed as they are specifically designed for encoding and decoding billions of arrays of integers for search engines and relational database applications. Simple8B [23], Memcpy [23], SIMD-BP128 [24], FastPFOR [24], and SNAPPY [25] are the most common fast integer compression algorithms. Fast integer compression approaches typically result in modest compression gains,

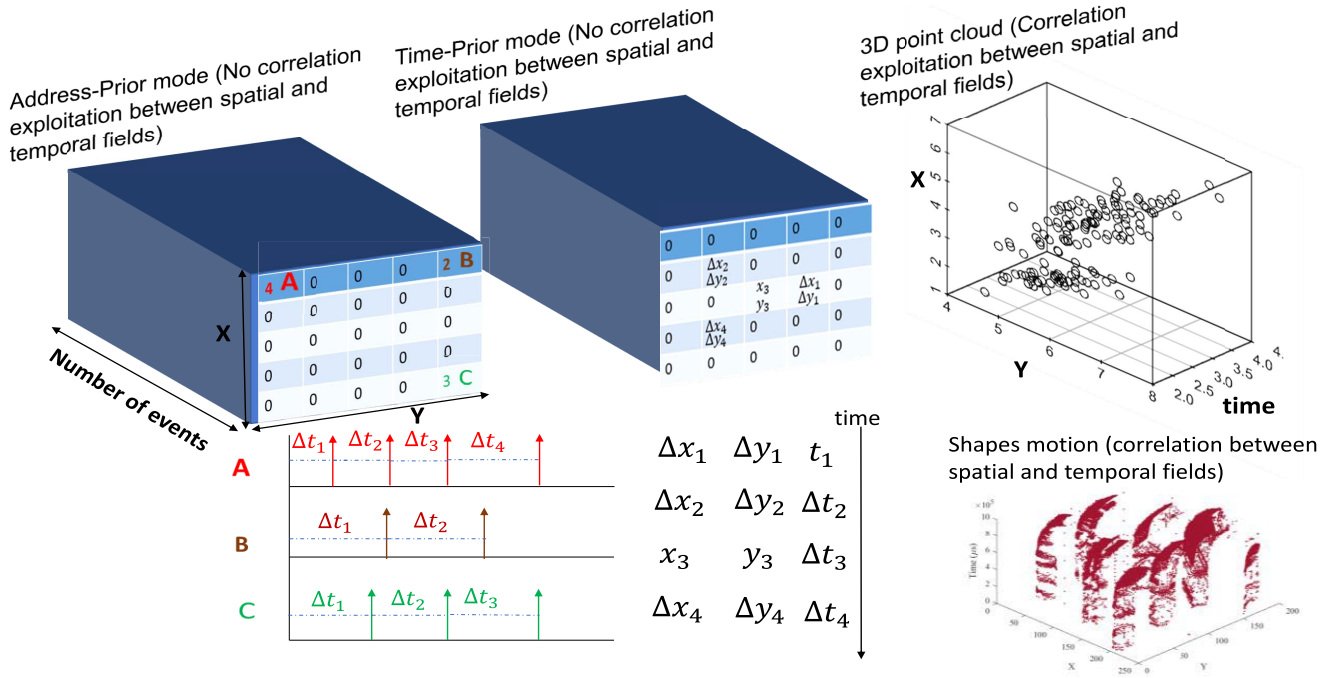


FIGURE 2. First column: Address-Prior mode of the spike coding strategy; Second column: Time-Prior mode of the spike coding strategy; Third column: spike event data as a 3D point cloud, as considered in our approach.

as their main design goal is compression and decompression speed at the expense of compression ratios, thus limiting their applicability.

C. POINT CLOUD COMPRESSION

A 3D point cloud is a set of points in the 3D space, possibly carrying attribute information (e.g., photometric properties). Point clouds can be static or dynamic. In the case of dynamic content, a different point cloud is considered at each time instance, representing the time-variation of a point cloud.

Most existing methods that compress point cloud geometry use octree coding [26], where voxelised blocks are partitioned until sub-cubes of dimension one are reached, and local approximations called ''triangle soups'' (trisoup) where geometry can be represented by a pruned octree plus a surface model [26], [27].

The Moving Picture Expert Group (MPEG) developed a standard point cloud encoder assuming data represented as coordinates in a 3D space (x, y, z) along with reflectance and RGB attributes associated with each point. Three different technologies were chosen as test models for the three different categories targeted: LIDAR point cloud compression (L-PCC) for dynamically acquired data; Surface point cloud compression (S-PCC) for static point cloud data; Video-based Point Cloud Compression (V-PCC) for dynamic content. Two main solutions were finally developed [28]: Video-based, equivalent to V-PCC, appropriate for point sets with a relatively uniform distribution of points; Geometry-based Point Cloud Compression (G-PCC), equivalent to the combination of L-PCC and S-PCC, appropriate for more sparse distributions. To encode the occupancy pattern of each octree node, G-PCC introduces many methods to exploit local geometry information and obtain an accurate context for arithmetic coding, such as Neighbour-Dependent Entropy Context, intra prediction, planar/angular coding mode.

Reviews on point cloud compression can be found in [28], [29], and [30], while a good overview of standardization activities is provided in [31]. Point cloud compression strategies have been designed for volumetric data, while in this paper we propose to adopt point cloud compression strategies on appropriately presented data from neuromorphic sensors.

III. PROPOSED STRATEGY

The global proposed compression scheme is reported in Figure 3 and we report in the following subsections the description of the key steps of the proposed strategy, further described in Section III-C.

A. ARRANGEMENT OF SPIKE EVENTS IN A TRIDIMENSIONAL SPACE, WITH SEPARATE POINT CLOUDS FOR EACH POLARITY

The first step of our approach consists in the arrangement of spike events in tridimensional space to enable correlation exploitation between the spatial and temporal fields of the spike events. According to [32], there exists a strong correlation within spatio-temporal surfaces of same polarity. In other words, spike events with the same polarity have high spatial correlation. Therefore, we propose to consider separately the events with positive and negative polarity, that we represent with two separate point clouds, one for

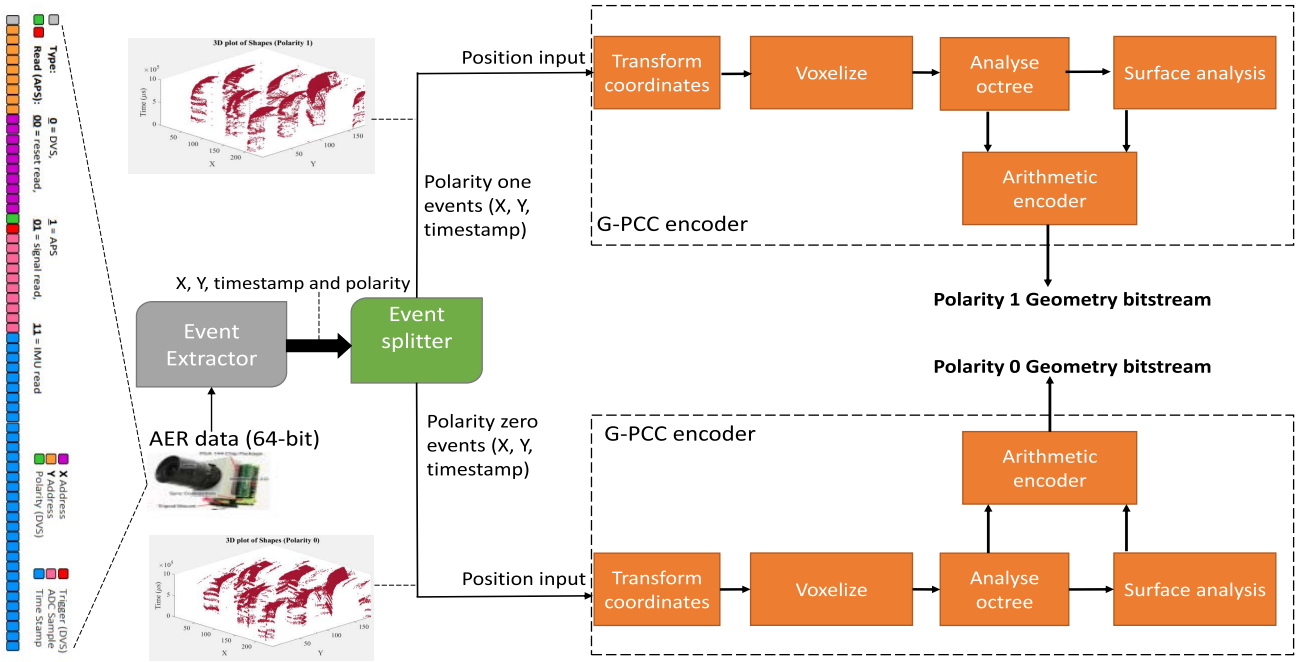
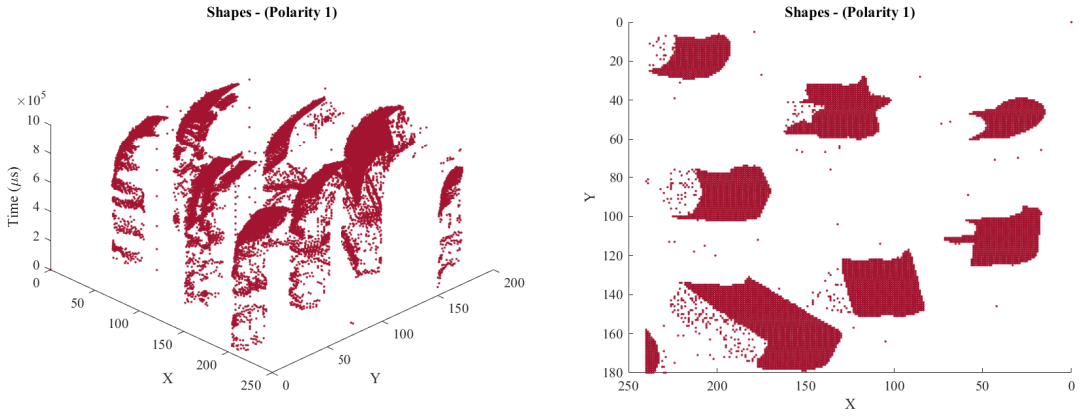


FIGURE 3. Proposed compression method.



(a) Global volumetric view (45°, 45°)

(b) X-Y view (180°, 90°)

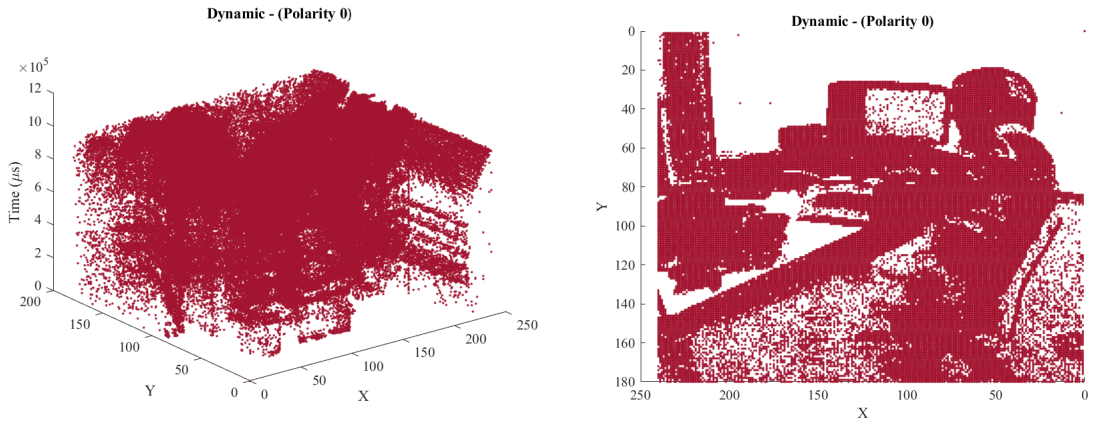
FIGURE 4. Examples of (x, y, t) point cloud representations for the shapes sequence with polarity 1. Shapes image is shown in Figure 6.

each polarity. This results in 3D spatio-temporal point clouds with high spatial correlation which will be exploited by the 3D point cloud encoder. We note that for each position in the space-time volume there are three possibilities: no event, event with polarity 1, event with polarity 0, hence the two clouds associated to the different polarities are not complementary, and one cannot be obtained from the other.

Examples of point cloud representations for the considered scenes are reported in Figures 4 and 5 for the *Shapes* and *Dynamic* sequences respectively. Frame based images of these sequences are shown in Figure 6.

B. GEOMETRY-BASED POINT CLOUD ENCODER

We adopt geometry-based point cloud compression for encoding the point clouds generated, composed of NVS data. We observe that the possible point cloud compression alternative mentioned earlier, V-PCC, employs 3D to 2D conversion techniques to obtain a 2D + t representation (with possible projection distortions) that can be compressed via a 2D video encoder. In contrast, G-PCC supports lossless point cloud coordinate representation in the 3D space as input [31], [33]. Therefore, we have used the services of G-PCC, using the z-axis for time rather than space information, as it enables us to fully exploit "volumetric redundancy" the



(a) Global volumetric view (45°, 45°)

(b) X-Y view (180°, 90°)

FIGURE 5. Point cloud representation (polarity 0) of the "dynamic" sequence.

(x, y, t) 3D space (see NVS point clouds in the left panels of Figures 4 and 5).

The first step of geometry coding is to perform a coordinate transformation followed by voxelization and then by geometry analysis using the octree or trisoup scheme. The octree scheme was used here. Finally, the resulting structure is arithmetically encoded. We only encode the geometry of the point cloud here, with no need for attributes. Therefore, only the position input of the encoder is utilized, whereas the attributes input of the encoder is disabled. We propose to split in the time domain the data of the scene in multiple clouds in order to reduce the encoding delay. We observe in fact that we expect that the bigger the point cloud in the time domain, the higher the compression efficiency (as we can exploit more temporal redundancy), but the higher the compression delay (as we have to wait to build the point cloud before starting the encoding process and in addition the encoding process itself is more complex for bigger point clouds).

C. DETAILS OF THE PROPOSED STRATEGY

The summary of all the steps to encode spike events using a point cloud encoder is shown in Algorithm 1. Spike events are represented via the AER protocol, where each spike event is 64-bit long. Therefore, the first step is to extract the spatial, temporal and polarity information from the AER data stream. The next step is to split the spike events based on the polarity flag. The spatial and temporal information for each polarity is then fed to the position input of a geometric-based point cloud compressor (G-PCC). It is important to note that the step described above can be implemented concurrently for the two point clouds corresponding to different polarities (as shown in Figure 3). The parallel implementation will reduce the time required to compress the spike event stream.

D. DISCUSSION

In the proposed method, the spatial and temporal coordinates of spike event data are the coordinates of 3D points (voxels).

Algorithm 1 Point Cloud Compression (PCC) of Spike Events

Input: Spike event data stream, AER.
Total number of events extracted from input stream: N_{events}
Number of input bits: $N_{input\ bits} = N_{events} \times 64$
Step 1: Extract spatio-temporal coordinates (X, Y, timestamp), and polarity fields from the AER data stream
Step 2: Split the data into spike event 0 and 1 streams
Step 3: Apply G-PCC encoding to the stream with polarity 1
Step 4: Apply G-PCC encoding to the stream with polarity 0
Output: Polarity 1 Compressed geometric bitstream, with size N_{G1} (bits)
Output: Polarity 0 Compressed geometric bitstream, with size N_{G0} (bits)
Output: Total size (bits) of the output stream, $\gamma = N_{G1} + N_{G0}$
Output: Compression-ratio $= \frac{N_{input\ bits}}{\gamma}$

The 3D point cloud geometric encoder utilizes the concept of octree decomposition. In octree decomposition, the 3D point cloud cube is decomposed recursively. At each octree decomposition level, a cube is divided into subcubes until the predefined threshold decomposition level (tree depth) is reached or there are no voxels in the subcubes. The encoder finds the best matching subcubes. The 3D point cloud encoding strategy of neuromorphic data utilizes the correlation between the spatial and temporal coordinates, which is not the case in the spike coding algorithm, as shown in Figure 2. According to the figure (second row and third column), the 3D plot of the spatial and temporal fields shows the motion trajectory of different objects. Furthermore, the proposed algorithm divides the spike sequence into two separate 3D point cloud, one associated to flag 0 and one for flag 1. As highlighted

TABLE 1. Extracted dataset for Experiment 1.

	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

in [32], there exists a strong correlation between the spatio-temporal surfaces with same polarities. The construction of a separate 3D point cloud for each polarity enables the exploitation of this correlation.

IV. PERFORMANCE EVALUATION SETUP

A. DATASET

We assessed the compression performance of the proposed and comparative benchmark strategies on the Dynamic and Active-pixel Vision Sensor (DAVIS) dataset [3]. The scenes in the dataset were acquired via a hybrid sensor technology, DAVIS, which outputs a spike event stream as well as conventional frames (intensity images) with a spatial resolution of 180×240. For each scene, the dataset also includes information on the vision sensor speed. The dataset is composed of outdoor and indoor scenes as shown in Figure 6. The details of the experiments are discussed in the subsequent sections.

B. EXPERIMENT 1

To evaluate the compression performance of the proposed and state-of-the-art approaches, the extracted sequences with varied complexity (in terms of scene and camera speed) is shown in Table 1. According to the table, the *Boxes* sequence has the highest event rate because of high complexity (textured scene and high camera speed), whereas the *Shapes* sequence is the least complex.

We selected the best performing benchmark strategies (Spike Coding and LZMA) based on our previous study [14]. We note that, although the TALVEN approach we proposed earlier provides better compression performance than Spike Coding and LZMA, the time aggregation step used in TALVEN results in a reduction in time resolution and this strategy is recommended for specific scenarios where the accumulation of events is beneficial. We highlight that in our previous paper, in order to make the comparison between TALVEN and the other strategies fair, we accumulated events also for the benchmark strategies. Differently from the previous one, this paper addresses the case where keeping the time resolution extremely high is beneficial. For this reason, the TALVEN strategy, which is not fully lossless in the time domain, is not considered in the comparison since this would not be fair.

TABLE 2. Extracted dataset for Experiment 2 (analysis of impact of point cloud size on compression performance).

Sequence	Event Rate (kev/s)	Sequence Duration (s)	Scene Complexity	Cloud-size (s)
Boxes	3094.82	60 (1-60)	High	1, 5, 10, 20, 30, 60
Poster	2822.50	60 (1-60)	High	1, 5, 10, 20, 30, 60
Dynamic	1188.74	60 (1-60)	Medium	1, 5, 10, 20, 30, 60
Shapes	385.44	60 (1-60)	Low	1, 5, 10, 20, 30, 60
Walking	488.84	60 (1-60)	Medium	1, 5, 10, 20, 30, 60
Running	1156.23	60 (1-60)	Medium	1, 5, 10, 20, 30, 60

In experiment 1, a single point cloud is considered for each of the considered sequences, *i.e.,* the size of the point cloud is equal to the duration of the sequence. For instance, the Boxes sequence with a duration of 5 seconds is encoded in a single point cloud. Similarly all the spike events of the 10 second Urban sequence are encoded as a single point cloud.

C. EXPERIMENT 2

The size of the point cloud can impact the compression ratio and for this reason we studied how dividing a scene in multiple point clouds of different sizes can impact the compression ratio. In order to compare the impact of the point cloud size on the compression performance, we extracted from the original dataset six sequences with the same duration (60s), then split in different chunk sizes. Details on these sequences are reported in Table 2. As shown in the table, we selected 60 second long sequences (Boxes, Poster, Dynamic, Shapes, Walking and Running) and six different point cloud sizes (1s, 5s, 10s, 20s, 30s, and 60s) for each of the considered sequences. The point cloud size of 1 second implies that the 60 second long sequence is encoded in 60 point clouds, whereas the cloud size of 60 means that each sequences is encoded in one large point cloud of 60 seconds.

D. G-PCC CONFIGURATION

We follow common testing conditions for octree based lossless geometry coding. Table 3 reports the selected coding parameters for the G-PCC encoder. The same coding parameters are used in both the experiments. It is important to note that the attributes are not utilized in the experiments.

The enforceLevelLimits flag, limiting the maximum size of the point cloud when set to 1, was set to 0 in order to curb the G-PCC codec from reducing the number of points of large point clouds generated from sequences such as Boxes. Moreover, sliceMaxPoints, i.e., the maximum number of points in a slice, is set to be higher than all the point cloud sizes in our dataset. Additionally, positionBaseQp is set to 0 for lossless coding of the 2-D coordinates and the timestamp. Other configuration parameters tabulated in Table 3 follow the common testing conditions for octree based lossless geometry coding.

FIGURE 6. Different types of scenes in the considered DAVIS dataset [3]. 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.

TABLE 3. Selected coding parameters for G-PCC.

Position related parameters	
mode	0
mergeDuplicatedPoints	0
srcResolution	0
outputResolution	0
positionQuantizationScale	1
trisoupNodeSizeLog2	0
neighbourAvailBoundaryLog2	8
intra_pred_max_node_size_log2	6
inferredDirectCodingMode	1
maxNumQtBtBeforeOt	4
minQtbtSizeLog2	0
planarEnabled	1
planarModeIdcmUse	0
convertPlyColourspace	1
transformType	0
numberOfNearestNeighborsInPrediction	3
levelOfDetailCount	12
intraLodPredictionSkipLayers	0
interComponentPredictionEnabled	0
positionBaseQp	0
sliceMaxPoints	107000000
enforceLevelLimits	0

E. KEY PERFORMANCE METRICS

We describe in the following the considered performance metrics.

- *Compression Ratio*. The performance of the proposed 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 represented with 8 bytes (64 bits) in the uncompressed mode.
- *Number of iterations.* The complexity of the octree structure in G-PCC can be measured using the number of iterations that the octree divisions undergoes until it meets no voxels in the subcubes or the maximum depth of the tree. This can also provide an indication of the distribution of points across the entire cloud. The software used for point cloud compression was G-PCC Test Model Category 13 (TMC13) [34] developed by MPEG and the computer used for performing compression was an Intel i7 2.6 GHz system with 16 GB RAM running 64 bit Windows operating system.

V. RESULTS

A. EXPERIMENT 1: COMPARATIVE PERFORMANCE ANALYSIS WITH BENCHMARK STRATEGIES

Figure 7 reports the comparison in terms of compression ratio between the proposed strategy and the best performing benchmarks strategies (as assessed in [14]). We remind that the comparison is done with strategies which are fully lossless (also in the time domain). Our previous approach [12], even if lossless after time aggregation, does not fall in this category due to the time-aggregation step.

The proposed strategy outperforms the benchmarks in terms of compression ratio for all cases except the *Slider* sequence where the Spike Coding strategy [16] performs slightly better.

We can observe from Figure 7 that the *Poster* and *Boxes* sequences yield the best compression gains. This is mainly because of the very high spike event rate. Such sequences result in a dense point cloud with millions of points in the 3D space. However, not all the high event rate sequences yield higher compression gains. The compression ratio is also dependent on the scene complexity. For instance, the compression ratio for the *Shapes* sequence is higher than the high event rate sequences of *Urban*, *Walking*, and *Running1*. Furthermore, the *Running2* and *Shapes* sequences yield approximately the same compression gains, with the *Running2* sequence having event rate five times higher (1229.4 Kev/s)

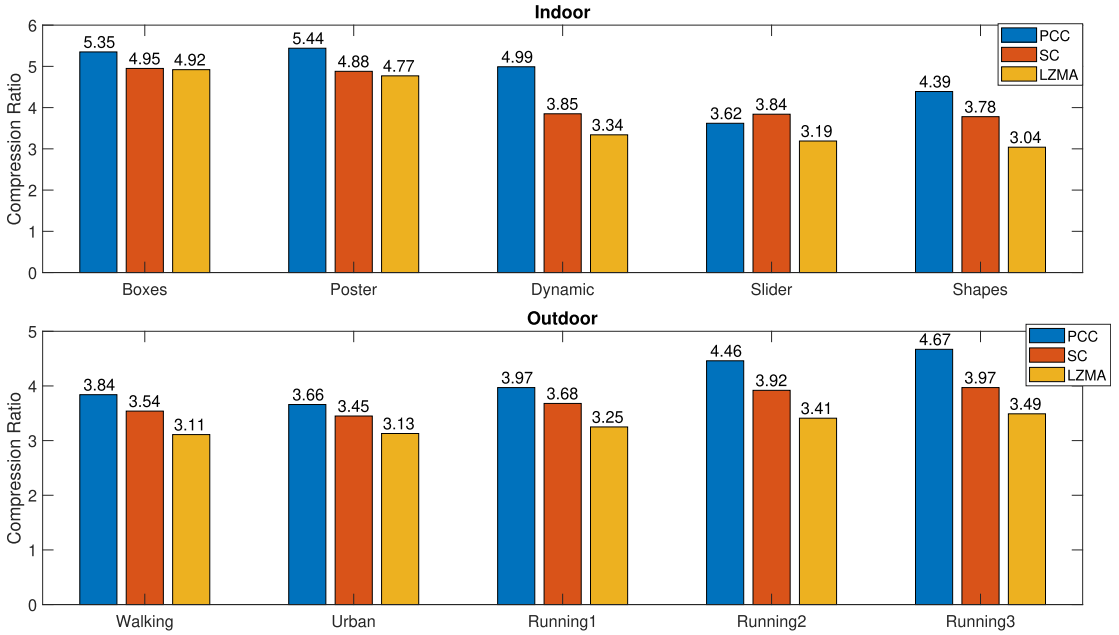


FIGURE 7. Compression ratio comparison of the benchmarks and the proposed strategy.

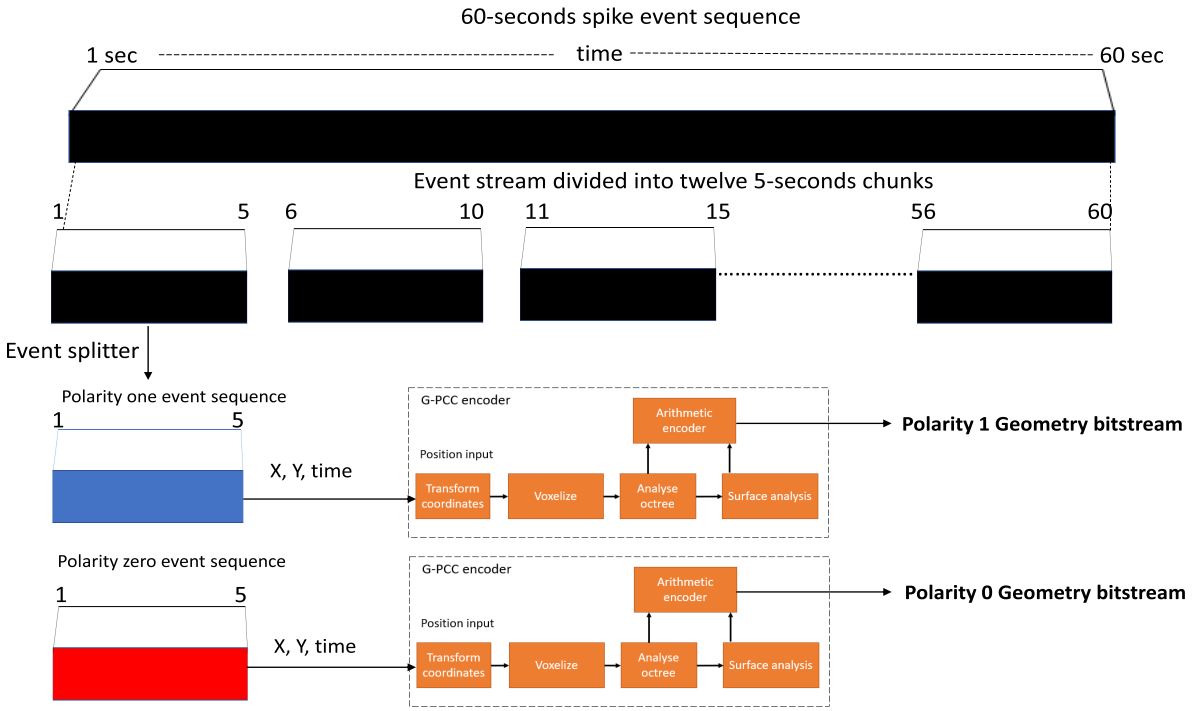


FIGURE 8. Setup for Compression Ratio (CR) evaluation with different point cloud sizes in experiment 2. The example reports the case of 5s chunks.

than the *Shapes* sequence (245 Kev/s). This is mainly because the scene complexity of *Shapes* is low which results in 3D geometric surfaces with high correlation which is exploited by the encoder.

It is important to note that with our approach the *Slider* sequence is the only one resulting in a lower compression gain as compared to the spike coding strategy. This is mainly because of the lower size of the 3D point cloud, i.e., the point cloud is only 3 seconds. In our second set of experiments, we observe that a point cloud size between 5 to 10 seconds yields the best compression gains.

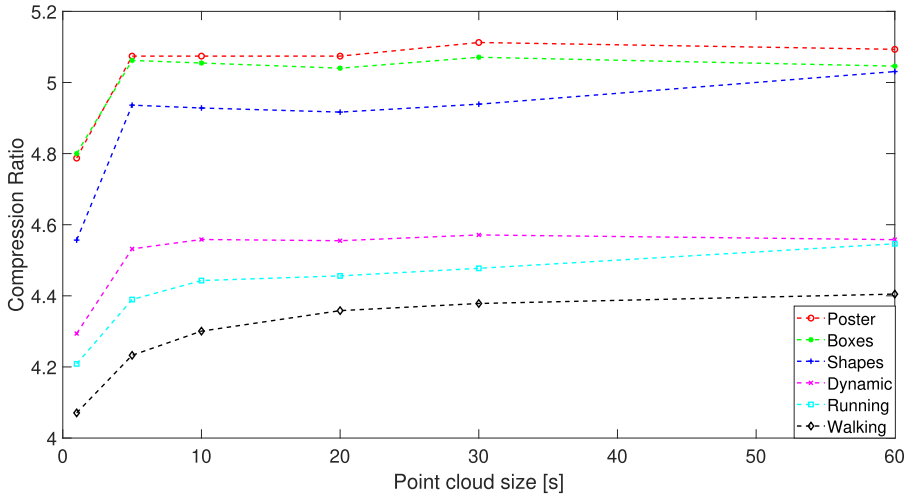


FIGURE 9. Compression ratio performance with different point cloud sizes for different sequences.

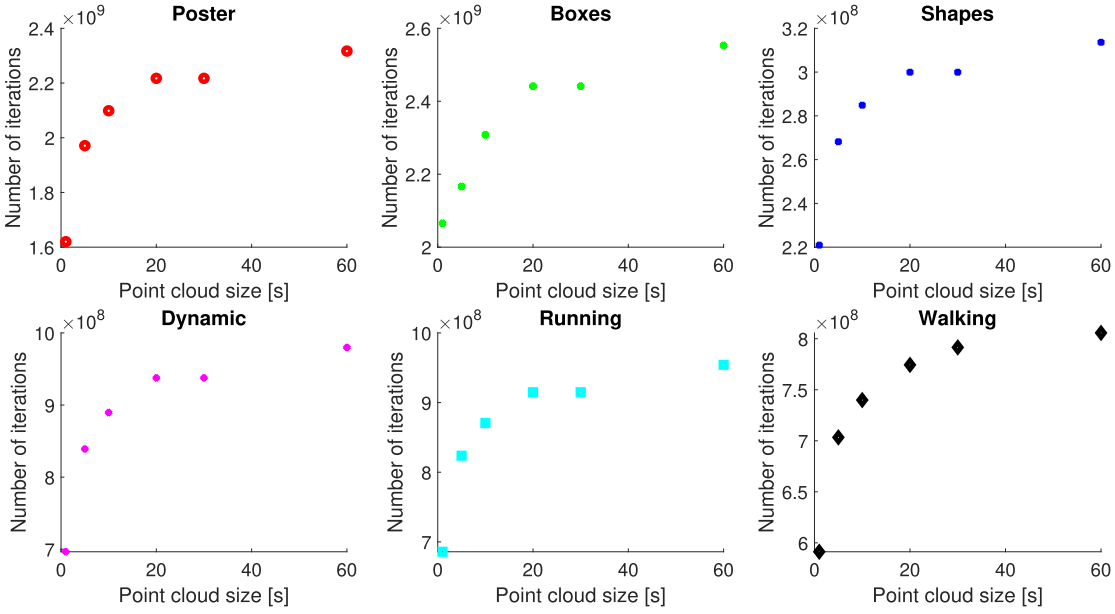


FIGURE 10. Number of iterations (to encode 60s) vs. point cloud size (see Figure 8).

B. EXPERIMENT 2: IMPACT OF CLOUD SIZE ON COMPRESSION PERFORMANCE

A single point cloud was considered for the whole (short) scenes for the results above. In a second experiment, we considered longer scenes of 60s duration split in point clouds of different sizes, in order to study the impact on compression performance of the size of the point cloud in the temporal domain. Figure 8 illustrates the setup for this second experiment.

Figure 9 shows the compression ratio performance of the proposed strategy for the scenes in Table 2. The horizontal axis reports the point cloud sizes considered for encoding (e.g., for point cloud size of 1 s, this means that in the considered 60 s we have 60 point clouds, while for point cloud size 60s only one point cloud was considered for encoding). We can observe that increasing the point cloud size increases the compression ratio, but saturation is reached at some point and increasing the point cloud size further leads to minimal benefits. For all sequences, increasing the point cloud size from 1s to 5s results in major benefits, while these decrease as we increase the point cloud size further. We note here that increasing the point cloud size increases the compression delay (we have to wait for more data to build the point cloud) and also increases the complexity of the encoding process (again resulting in further encoding delay). A suitable trade off should be then considered, based on the use case, between

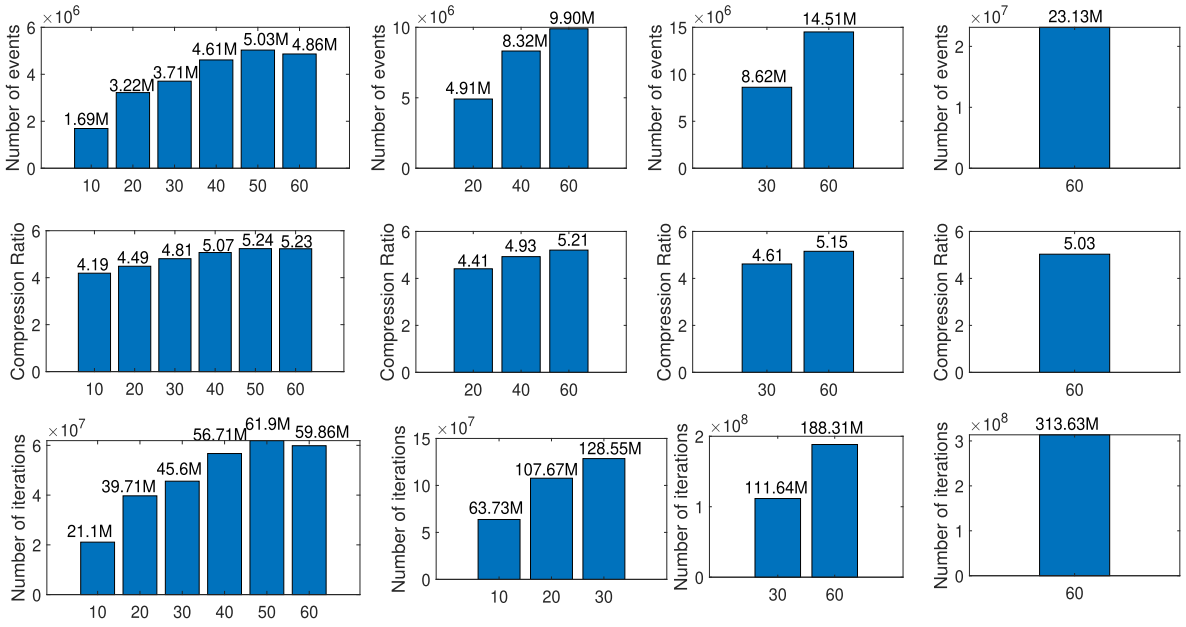


FIGURE 11. Performance for different sub-clouds of shapes sequence. The first row shows the number of events in each of the considered point cloud size. The second and third rows show the compression and number of iterations performance, respectively, of four different point cloud sizes (10s, 20s, 30s and 60s) of the shapes sequence.

compression ratio (higher for higher point cloud size) and compression delay.

Figure 10 shows number of iterations for different point cloud sizes. Each subfigure reports the results for a different scene. According to the figure, the increase in point cloud size increases the number of iterations required to compress the sequences. The larger point cloud size increases the encoding complexity as more iterations are required to perform further octree decomposition. The octree division continues until no voxels in the subcubes condition is met or the maximum octree depth is reached. The higher the number of iterations the higher the compression induced delay.

Figure 11 shows the compression performance (w.r.t number of events and iterations) of the *Shapes* sequence for four different point cloud sizes, *i.e.,* 4 columns of the figure represent point cloud chunk size of 10 (six point cloud chunks), 20 (three point cloud chunks), 30 (two point cloud chunks) and 60 (one point cloud chunk) seconds. In the first row, each bar shows the number of events (number of points in space) in each of the considered point cloud sizes. The second and third rows show the compression performance and number of iterations respectively. In the *Shapes* sequence, the rotation motion of the sensor in front of scene increases until the 50th second and then remains approximately constant (from 50 to 60 seconds). The scene complexity is approximately constant, therefore the number of events increases mainly due to the motion of the sensor. According to Figure 11, the compression ratio increases with the increase in the number of points in space (number of spike events) as shown by the point cloud chunks (number of bars within each subfigure with annotated compression ratios). However, as already observed, the increase in point cloud chunk size beyond a certain value (5 seconds) has minimal impact on the overall compression ratio as shown by the *Shapes* sequence compression performance in Figure 9. According to the experimental analysis, the point cloud size of 5 seconds achieves the best trade-off between compression and encoding complexity (number of iterations).

C. DISCUSSION ON COMPARATIVE ENCODING COMPLEXITY VS. BENCHMARK STRATEGIES

We expect the encoding complexity of the Spike coding strategy to be higher than the point cloud encoder. AP and TP modes are computationally expensive as both employ the motion estimation concept of video encoding. It is important to note that both the modes are applied to every macro-cube. The mode yielding the best compression is chosen for each macro-cube. This results in high number of computations which increase the encoding delay. Furthermore, the computationally expensive CABAC is used as an entropy coding which has further impact on coding delay. LZMA utilizes a huge size dictionary along with a sophisticated dictionary data structure. This results in high encoding delay which is comparable to the spike coding algorithm.

The point cloud compressor has to perform separate encoding for each polarity. This results in higher encoding delay. We should observe, however, that when the data is split in multiple point clouds to perform compression, the latter can be performed in parallel (concurrent execution) for the different point clouds, lowering the time required for encoding.

VI. CONCLUSION

Acquiring visual data via neuromorphic sensors results in much lower data rates than conventional video, but in some scenarios there is a requirement to further compress such data. We proposed here a strategy to losslessly compress spike event data generated by NVSs via point cloud compression. According to the experimental analysis, the proposed strategy shows improved compression ratios, up to 30% higher than the benchmark strategies for 9 out of 10 short scenes considered in the first tests, while the performance is only slightly worse for the 10th scene (this can be explained with its shorter duration, not allowing the exploitation of the main benefits of our strategy). We have also shown how splitting the data in point clouds of different sizes in the temporal domain affects the compression performance and that increasing the point cloud size beyond a certain limit ($\sim 10\ s$ for the considered sequences) does not result in major benefits, while increasing compression complexity and delay. We expect these results will enable new use cases for NVSs in emerging Internet of Things scenarios.

REFERENCES

[1] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 × 128 120 dB 15 μs latency asynchronous temporal contrast vision sensor," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, Jan. 2008.

[2] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 × 128 120 db 30 mW asynchronous vision sensor that responds to relative intensity change," in *Proc. IEEE Int. Solid State Circuits Conf.*, San Francisco, CA, USA, Feb. 2006.

[3] 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," *Int. J. Robot. Res.*, vol. 36, no. 2, pp. 91–97, 2017.

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

[5] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake city, UT, USA, Jun. 2018.

[6] A. Rigi, F. B. 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.

[7] D. Tedaldi, G. Gallego, E. Mueggler, and D. Scaramuzza, "Feature detection and tracking with the dynamic and active-pixel vision sensor (DAVIS)," in *Proc. 2nd Int. Conf. Event-Based Control, Commun., Signal Process. (EBCCSP)*, Krakow, Poland, Jun. 2016.

[8] E. Mueggler, B. Huber, and D. Scaramuzza, "Event-based, 6-DOF pose tracking for high-speed maneuvers," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Chicago, IL, USA, Sep. 2014, pp. 2761–2768.

[9] M. Martini, N. Khan, Y. Bi, Y. Andreopoulos, H. Saki, and M. Shikh-Bahaei, "Challenges and perspectives in neuromorphic-based visual IoT systems and networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Barcelona, Spain, May 2020, pp. 8539–8543.

[10] N. Khan and M. G. Martini, "Bandwidth modeling of silicon retinas for next generation visual sensor networks," *Sensors*, vol. 19, no. 8, pp. 1–26, 2019.

[11] N. Khan and M. G. Martini, "Data rate estimation based on scene complexity for dynamic vision sensors on unmanned vehicles," in *Proc. IEEE 29th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, Bologna, Italy, Sep. 2018.

[12] N. Khan, K. Iqbal, and M. G. Martini, "Time-aggregation-based lossless video encoding for neuromorphic vision sensor data," *IEEE Internet Things J.*, vol. 8, no. 1, pp. 596–609, Jan. 2021.

[13] I. Schiopu and R. C. Bilcu, "Lossless compression of event camera frames," *IEEE Signal Process. Lett.*, vol. 29, pp. 1779–1783, 2022.

[14] 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. 103149–103163, 2020.

[15] K. Iqbal, N. Khan, and M. G. Martini, "Performance comparison of lossless compression strategies for dynamic vision sensor data," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Barcelona, Spain, May 2020, pp. 4427–4431.

[16] Z. Bi, S. Dong, Y. Tian, and T. Huang, "Spike coding for dynamic vision sensors," in *Proc. Data Compress. Conf.*, Snowbird, UT, USA, Mar. 2018, pp. 117–126.

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

[18] 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," 2020, *arXiv:1807.09480*.

[19] Y. Collet and E. M. Kucherawy. (Jul. 2018). *Zstandard—Real-Time Data Compression Algorithm*. [Online]. Available: http://facebook.github.io/zstd/

[20] P. Deutsch and J.-L. Gailly. *Zlib Compressed Data Format Specification Version 3.3*, document RFC 1950, May 1996.

[21] A. Lempel and J. Ziv, "Lempel—Ziv—Markov chain algorithm," Tech. Rep., 1996.

[22] J. Alakuijala and Z. Szabadka. *Brotli Compressed Data Format*, document RFC 7932, Internet Engineering Task Force, Fremont, CA, USA, 2016.

[23] D. Blalock, S. Madden, and J. Guttag, "Sprintz: Time series compression for the Internet of Things," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 2, no. 3, p. 93, 2018.

[24] D. Lemire and L. Boytsov, "Decoding billions of integers per second through vectorization," *Software, Pract. Exper.*, vol. 45, no. 1, pp. 1–29, Jan. 2015.

[25] S. H. Gunderson. (Apr. 2015). *Snappy: A Fast Compressor/Decompressor*. [Online]. Available: https://github.com/google/snappy

[26] R. Schnabel and R. Klein, "Octree-based point-cloud compression," in *Proc. Eurographics Symp. Point-Based Graphics*, 2006, pp. 111–120.

[27] A. Dricot and J. Ascenso, "Adaptive multi-level triangle soup for geometry-based point cloud coding," in *Proc. IEEE 21st Int. Workshop Multimedia Signal Process. (MMSP)*, Sep. 2019, pp. 1–6.

[28] S. Schwarz, M. Preda, V. Baroncini, M. Budagavi, P. Cesar, P. A. Chou, R. A. Cohen, M. Krivokuca, S. Lasserre, Z. Li, and J. Llach, "Emerging MPEG standards for point cloud compression," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 9, no. 1, pp. 133–148, Mar. 2019.

[29] C. Cao, M. Preda, and T. Zaharia, "3D point cloud compression: A survey," in *Proc. 24th Int. Conf. 3D Web Technol.*, 2019, pp. 1–9.

[30] H. Liu, H. Yuan, Q. Liu, J. Hou, and J. Liu, "A comprehensive study and comparison of core technologies for MPEG 3-D point cloud compression," *IEEE Trans. Broadcast.*, vol. 66, no. 3, pp. 701–717, Sep. 2020.

[31] D. Graziosi, O. Nakagami, S. Kuma, A. Zaghetto, T. Suzuki, and A. Tabatabai, "An overview of ongoing point cloud compression standardization activities: Video-based (V-PCC) and geometry-based (G-PCC)," *APSIPA Trans. Signal Inf. Process.*, vol. 9, no. 1, pp. 1–17, 2020.

[32] X. Lagorce, G. Orchard, F. Gallupi, B. E. Shi, and R. Benosman, "HOTS: A hierarchy of event-based time-surfaces for pattern recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1346–1359, Jan. 2017.

[33] T. Dong, K. Kim, and E. S. Jang, "Performance evaluation of the codec agnostic approach in MPEG-I video-based point cloud compression," *IEEE Access*, vol. 9, pp. 167990–168003, 2021.

[34] K. Mammou, P. A. Chou, D. Flynn, M. Krivokuća, O. Nakagami, and T. Sugio. *G-PCC Codec Description v2*, Standard ISO/IEC JTC1/SC29/WG11 N18189, 2019.

MARIA MARTINI (Senior Member, IEEE) received the Laurea degree in electronic engineering *(summa cum laude)* from the University of Perugia, Italy, in 1998, and the Ph.D. degree in electronics and computer science from the University of Bologna, Italy, in 2002. She is currently a Professor with the Faculty of Engineering, Computing, and the Environment, Kingston University London, where she also leads the Wireless Multimedia Networking Research Group. She has led the KU Team in a number of national and international research projects, funded by the European Commission (e.g., OPTIMIX, CONCERTO, QoE-NET, and Qualinet), U.K. Research Council, U.K. Technology Strategy Board/InnovateU.K., and international industries. She has authored more than 200 scientific articles, contributions to standardization groups (IEEE and ITU), and several patents on wireless video. Her research interests include QoE-driven wireless multimedia communications, decision theory, video quality assessment, the Internet of Things, neuromorphic vision, immersive environments, and medical applications. She is an Expert Evaluator of the European Commission and EPSRC. She chaired/organized a number of conferences and workshops. She is part of international committees and expert groups, including the NetWorld Europe (European Technology Platform Expert Advisory Group), the Video Quality Expert Group (VQEG), and the IEEE Multimedia Communications Technical Committee, where she has served as the Vice-Chair (2014–2016), the Chair (2012–2014) of the 3-D Rendering, Processing, and Communications Interest Group, and Key Member for the QoE and Multimedia Streaming IG. She is currently chairing the IEEE Standards Association Working Group on the quality assessment of light field imaging. She was an Associate Editor of *IEEE Signal Processing Magazine* (2018–2021) and IEEE TRANSACTIONS ON MULTIMEDIA (2014–2018). She has been a Lead Guest Editor of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS Special Issue on ''QoE-aware wireless multimedia systems'' and a Guest Editor of the IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS, IEEE MULTIMEDIA, and the *International Journal of Telemedicine and Applications*.

JAYASINGAM ADHURAN (Member, IEEE) received the B.S.Eng. (Hons.) degree in electronics engineering and the M.Eng. degree in microelectronics and embedded systems from the Asian Institute of Technology (AIT), Thailand, in 2015 and 2017, respectively. He is currently pursuing the Ph.D. degree with the Centre for Vision, Speech and Signal Processing (CVSSP), University of Surrey. He worked as a Lecturer (Probationary) at the University of Jaffna, from July 2018 to September 2018. He is affiliated with the Wireless Multimedia Networking Research Group, Kingston University London. He is also working at the BBC Research and Development Team, U.K., as a Research and Development Engineer. His research interests include video coding and adaptive streaming technologies.

NABEEL KHAN (Member, IEEE) received the B.S. degree in electronics engineering from the Sir Syed University of Engineering and Technology (SSUET), Karachi, Pakistan, in 2007, and the M.Sc. degree in data communication and the Ph.D. degree in video optimization over LTE networks from Kingston University London, U.K., in 2009 and 2014, respectively. He has been a Lecturer with the Faculty of Science, Engineering and Computing, Kingston University London, and he is still a Honorary Member of Staff at Kingston University London. He is currently a Senior Lecturer and a Course Leader at the University of Chester, U.K. During his postdoctoral research, he has contributed on various EU and U.K. projects, e.g., EU FP7 Concerto and EPSRC IoSiRe. His research interests include data communication, cyber security, computer vision, data compression and modeling, and the Internet of Things.