Event cameras are emerging bio-inspired sensors that offer salient benefits over traditional cameras. With high speed, high dynamic range, and low power consumption, event cameras have been increasingly employed to solve existing as well as novel visual and robotics tasks. Despite rapid advancement in event-based vision, event data compression is facing growing demand, yet remains elusivey challenging and not effectively addressed. The major challenge is the unique data form, i.e., a stream of four-attribute events, encoding the spatial locations and the timestamp of each event, with a polarity representing the brightness increase/decrease. While events encode temporal variations at high speed, they omit rich spatial information, which is critical for image/video compression. In this paper, we perform lossy event compression (LEC) based on a quadtree (QT) segmentation map derived from an adjacent image. The QT structure provides a priority map for the 3D space-time volume, albeit in a 2D manner. LEC is performed by first quantizing the events over time, and then variably compressing the events within each QT block via Poisson Disk Sampling in 2D space for each quantized time. Our QT-LEC has flexibility in accordance with the bit-rate requirement. Experimentally, we show results with state-of-the-art coding performance. We further evaluate the performance in event-based applications such as image reconstruction and corner detection.

Keywords Event cameras · Image and video compression · Event-based vision

1 Introduction

Inspired by biological visual systems [1], event cameras are novel sensors designed to capture visual information with a data form drastically different from traditional images and videos [2, 3]. In an event camera, the pixels do not directly output the intensity signals as traditional cameras do. Instead, each event pixel compares the difference between the current log-intensity state and the previous state, and fires an event when the difference exceeds the firing positive or negative thresholds. An example is shown in Fig. 5 Column 1.

The novel event-based sensing has provided several benefits. First, the asynchronous mechanism enables very low latency (∼10µs) and therefore high speed imaging. Second, event cameras have high dynamic range (HDR, ∼120dB) compared to regular frame-based cameras (∼60dB). Third, the events reduce redundant captures of static signals. However, the events underrepresent the spatial structures, which have imposed challenges for event data compression. Last, event cameras consume lower power (10mW) than traditional cameras (∼1W). As such, event cameras have brought new solutions to many classical as well as novel problems in computer vision and robotics, including high
frame-rate video reconstruction \([4, 5, 6]\), with HDR \([7, 8]\) and high resolution \([9, 10, 11]\), and 3D reconstruction of human motion \([12]\) and scenes \([13, 14]\), as well as odometry \([15, 16]\) and tracking \([17, 18]\).

A challenging issue yet to be studied is the event compression. Prior art considers image/video compression and event compression as separate tracks of research, with event compression only focusing on lossless compression \([19, 20]\). The compression ratio is about 2.6536 against raw spike data and compared the results to conventional lossless coding algorithms such as LZ77 \([21]\) and LZMA \([22]\). This is far from satisfying compression demands. We approach event compression in the context of hybrid sensing (RGB + events), which have gained growing attention \([23, 24, 11, 16]\).

In this work we propose a novel lossy compression method of the events. This is the first attempt in the literature to do lossy compression of events. We perform priority-based compression driven by a quadtree (QT) structure. The QT is generated from the accompanying intensity (grayscale) image. The events are then variably compressed according to the block-size of the QT map by Poisson Disk Sampling (PDS). PDS randomly sample points in space with a distance \(r\) apart from each other producing a high frequency blue noise characteristics. Blue noise based sample set is widely applied in rendering, texturing, animation and related domains. Poisson samples have been generated in several ways \([25, 26, 27, 28, 29, 30]\) reducing the computational complexity and improving the blue noise characteristics. We leverage PDS as a method to eliminate events in space.

In particular, this paper makes the following contributions:

- We propose a novel framework for lossy compression of neuromorphic events based on the QT segmentation map from adjacent image, and PDS for spatial compression.
- We evaluate our approach qualitatively and quantitatively, on existing event datasets.
- We demonstrate the effectiveness of our compression by applying the compressed events for downstream tasks, including corner detection and video reconstruction.

Limitations: We approach the event compression via a QT segmentation map obtained from the intensity images. Hence we require both grayscale and events. On the other hand, with an event camera producing only events, we must generate a QT structure from events to apply lossy compression. The QT is obtained based on the available bit rate for the grayscale frames with optimal distortion. Using this QT, we compress the events which gives us a distortion for the events.

2 Related Works

2.1 Image/Video coding techniques

Conventional image and video based coding techniques have evolved over last couple of decades which led to successful telecom revolution globally. Several standards came out as a result of the persistent effort, some of the latest video standards being HEVC, H.264, AVS2, VP9 \([31, 32, 33, 34]\). The most prevalent block-based hybrid-coding framework uses intra-/inter-prediction, transform coding, quantization and entropy coding. This exploited the spatial and temporal redundancy of the video data. Previous frames are taken into account in the inter-prediction while intra-prediction depends on the neighboring pixel space. The residual between the actual and the predicted blocks in a frame is also transformed and quantized. Run-length encoding followed by Huffman encoding of the coefficients are normally done in these standards.

In images and videos, the 2D and/or 3D spatio-temporal consistency of the data is exploited to have an efficient coding algorithm. However, events are essentially non-continuous in space and time. The events are strongly dependent on object and/or scene motion apart from luminance changes in the scene. Limited work \([20]\) has been done in predicting distribution of events in space, time and polarity. This lack of predictability of the events prevent us from applying conventional image/video based coding techniques directly on events.

2.2 Event communication protocols and compression

Currently, the events in the raw format are represented as per Address Event Representation (AER) protocol \([35]\). The AER is a communication protocol developed to communicate events between neuromorphic chips. DAVIS cameras (e.g. DVXplorer and DAVIS346) built by iniVation communicate uses the AEDAT protocol. AEDAT 4.0 released in July 2019 \([35]\) uses 96 bits representation for each event, while AEDAT 3 finalized in 2015 used 64 bit representation for each event. The timestamp uses maximum bits due to its resolution with 64 bits and 32 bits for AEDAT 4.0 and AEDAT3 respectively. Although AEDAT 4.0 has incorporated lossless encoding standards such as LZ4, LZ4_HIGH, ZSTD and ZSTD_HIGH, the lossy encoding of events has not been incorporated so far in the protocol.
Lossless event compression has been proposed in [20]. The authors divided the 3D event volume into smaller blocks which was then split into octree structure. The events in the octree structure was then coded in address-prior or time-prior mode. They further used entropy coding to reduce the bits. These works inspire us to develop the first ever lossy event compression algorithm reported in literature (to the best of authors’ knowledge), exploiting both spatial and temporal correlations as described in the next section.

3 Methods

In this section, we describe the lossy event compression pipeline. The first step is to generate a QT structure described in Sec. 3.1. The QT is a segmentation map which is applied on 3D events using Poisson Disk Sampling as covered in Sec. 3.2. Sec. 3.3 digs deep into the lossy event compression method while Sec. 3.4 discusses an event compression metric used for evaluating the compression performance.

3.1 Generation of Quadtree (QT) structure

The events are generated due to scene complexity and relative motion of the camera with respect to scene objects. This is highly dependent on the intensity frames. Khan et al. [36] showed that the event rate depends exponentially on the scene complexity and linearly on the sensor speed. The events are generated due to motion of objects with edges and textures. The event volume thus needs to be prioritised in order to identify regions which might be of importance or otherwise.

We use a QT structure in order to identify regions of importance. For an event volume $E_{t-1}$ between successive intensity frames $I_{t-1}$ and $I_t$, we leverage on the intensity frames to derive a relevant QT structure. We generate the QT based on the system described in Banerjee et al. [37]. The QT is generated by Viterbi optimization which provides a trade-off between the frame distortion and frame bit rate. This is done by minimizing the frame distortion $D$ over the leaves of the QT $x$ subject to a given maximum frame bit rate $R_{max}$. The constrained discrete optimization is solved using Lagrangian relaxation, leading to solutions in the convex hull of the rate-distortion curve. The Lagrangian cost function is of the form

$$J_\lambda(x) = D(x) + \lambda R(x),$$  \hspace{1cm} (1)

where $\lambda \geq 0$ is a Lagrangian multiplier.

The optimal QT segmentation map is derived for intensity frame $I_t$ based on the distorted frame $\hat{I}_{t-1}$ using Bezier curve as described in [38]. This $(x, y)$ segmentation map is applied to the event volume $E_t$ providing regions of priority for event compression - bigger blocks indicate less priority while smaller blocks are important regions. The events in the smaller blocks are of high fidelity while those in larger blocks are considered to be partly redundant. (For more details of QT generation please see supplementary material Sec. 1.)

3.2 Poisson Disk Sampling

The QT structure provides a segmentation map. However, removal of the events inside the QT blocks is challenging as the events may be indicate local structure. There may be aggregation of events at the edges which may be increased due to presence of noise in the sensor. Poisson disk sampling is applied on the events inside the $N \times N$ blocks with $N$ describing the size of QT boxes. Poisson disk sampling generates blue noise sample patterns, where all samples are at least $r$ distance apart from each other. The parameter $r$ is chosen by the user. Bridson [39] proposed a fast Poisson disk sampling in multidimensional space. We formulate the problem of eliminating events (samples) within the disk of radius $r$ instead of generating new events (samples). Thus, given $M$ original events, we obtain a $R$ subset of events.

Given a neighborhood of $N \times N$ pixels, we find the centroid (geometric median) $x_m$, of the events $M$ events, with each $x_i \in R^m$ are the event locations

$$x_m = \arg\min_{x_m} \sum_{i=1}^M \|x_i - x_m\|_2,$$  \hspace{1cm} (2)

We consider the event at or nearest to the median and eliminate the events within disk radius of $r$. We then consider the nearest point outside radius $r$ and eliminate the events within disk radius $r$. This has effect of reducing the event density locally in neighborhood of $N \times N$ pixels, along with introducing a fixed spatial sampling pattern in the event volume. Fig. 2(a) shows a sample event frame with all the events aggregated over time. The uncompressed event frame had 12512 events. The events sampled with Poisson disk radius $r = 1$ and $r = 2$ in Fig. 2(b) and Fig. 2(c) respectively.

3
Figure 2: Original samples of an event frame along with Poisson disk sampled events. We encourage the reader to enlarge the view for better visualization.

The number of events sampled with $r = 1$ is 7108, while that sampled with $r = 2$ is 3882. Clearly, it can be seen that the density of events are reduced. However, the features and topology of the objects in the scene are retained.

### 3.3 Lossy Event Compression

The QT provides the segmentation map and the Poisson disk sampling provides efficient strategy to reduce the density of events. However, the QT has different sized blocks which indicate different priorities. We apply sampling based on different Poisson disk radius $r$ at different sized blocks. The larger blocks has larger $r$ while the smaller blocks has smaller $r$ or no removal of events at all. As mentioned in Sec 3.1, we obtain the QT at $t - 1$ from distorted intensity frame $\hat{I}_{t-2}$ and actual frame $I_{t-1}$, which is applied to event volume $E_{t-1}$. The event volume corresponds to events generated between intensity frames $I_{t-1}$ and $I_t$.

**Algorithm 1: Event Compression Pseudo Code**

```
set $\lambda_{\min}$, $\lambda_{\max}$, $R_{\max}$;
while intensity bit rate $R > R_{\max}$ do
    adjust $\lambda$ to obtain desired $R$ as in Sec. 3.1
end

Result: Optimal QT

set Tbins, $r_4$;
Quantize timestamps $t$;
while QT blocks = / 4 × 4, 8 × 8, 16 × 16, 32 × 32 / do
    compute the centroid $C$ in $(x, y)$;
    Shift $C$ to nearest event location $P$ (if $C$ does’n’t coincide with $P$);
    if all events not visited then
        Remove events within radius $r_i$ with respect to $P$;
        set $P$ as nearest event outside $r_i$;
    end
    Encode $(x, y)$ differentially followed by Huffman for each block;
    Encode $p$ as Run Length Encoding each block followed by Huffman;
    Encode $t$ differentially each block with respect to neighboring events;
end
while QT blocks = / 2 × 2 / do
    Encode $(x, y)$ differentially each block;
    Encode $p$ as Run Length Encoding each block followed by Huffman;
    Encode $t$ differentially each block with respect to neighboring events;
end
while QT blocks = / 1 pixel / do
    Consider all pixels in the frame ;
    Encode $p$ as Run Length Encoding followed by Huffman;
    Encode $t$ as differential with respect to neighboring events;
end

Result: Compressed Event Volume
```
The lossy events compression is performed in steps depending on the bandwidth available for communication between network-connected devices. The events are first quantized in time, then based on the spatial overlap of the event \((x, y)\) locations with the QT blocks, Poisson disk sampling is applied. The \((x, y)\) location is differentially encoded while the polarity is encoded by Run Length Encoding (RLE) followed by Huffman encoding (HE). The quantized timestamps for the events differentially encoded locally within the blocks. The algorithm is mentioned in details as pseudocode below. It must be clearly stated here that, we are developing a heuristics based rule for compressing the events. This algorithm offers flexibility in terms of event compression. Depending on the desired bitrate, the compression may be set by the user. In our experiments (Sec. 4), we show some of the experimental event compression results.

### 3.4 Event Compression Metric

The compressed events essentially represent the original 3D event volume in a quantized manner. The compression is both in space and time. To the best of the authors’ knowledge, there has been no application of any metric for computing event distortion. We make a first attempt in that direction of quantizing this distortion. Since the fidelity of the events in both spatio-temporal volume is important, we need to characterize the quality of the events from both these aspects.

We separate the spatial and temporal fidelity of the event volume in order to have a complete understanding of the different parameters in the event compression algorithm. For computing the spatial distortion, we aggregate the events over the time bins into a \((x, y)\) event image. We compute the PSNR and SSIM based on the aggregated compressed and uncompressed event image.

For quantifying temporal distortion, we define a temporal error metric to quantify the quantization error in time

\[
T_{\text{error}} = \frac{1}{N_{\text{frames}}} \sum_{i=1}^{N_{\text{frames}}} \sqrt{\sum_{j}(T_{j,\text{org}} - T_{j,\text{quant}})^2},
\]

where, \(N_{\text{frames}}\) is the number of event volumes in the sequence, \(T_{j,\text{org}}\) is the timestamp of \(j\) th event in \(i\) th frame, \(T_{j,\text{quant}}\) is the timestamp of \(j\) th event in \(i\) th frame. This spatial and temporal metric are used in order to quantify the distortion and compression performance of the events in the next section.

### 4 Experiments and Results

The performance of the system is tested by simulating the proposed model with sequences from RGB-DAVIS dataset \([11, 40]\). The dataset has both calibrated rgb images and the corresponding events. The sequences Indoor6, Indoor9 and Outdoor3 are used for the experimental results.

#### 4.1 Performance with varying temporal binning only

In one of the experiments, the QT segmentation is not used. No events are removed based on Poisson disk sampling. However, the timestamps of the events are discretized temporally into \(N\) bins, with \(N \in \{8, 16, 24\}\). To quantify the benefits of the temporal compression, we compute the compression ratio of the original uncompressed events as well as the compressed events. Each uncompressed events are represented as 64 bits, as per AEDAT 3.1 format. Table 1 shows the compression ratio and the average \(T_{\text{error}}\) over the sequence.

| Sequence | Indoor6 | Indoor9 | Outdoor3 |
|----------|---------|---------|----------|
| \(T_{\text{bin}}\) | 8   | 16  | 24   | 8   | 16  | 24   | 8   | 16  | 24   |
| \(T_{\text{error}}\)  | 0.302 | 0.164 | 0.117 | 0.283 | 0.143 | 0.096 | 0.208 | 0.104 | 0.069 |
| CR       | 11.23  | 10.17  | 9.64  | 11.37 | 10.65 | 10.33 | 11.14 | 9.85  | 9.14  |

Table 1: Temporal binning only. CR: compression ratio.

Clearly, it can be seen that with the increase in the number of \(T_{\text{bin}}\), the compression ratio reduces alongwith the reduction of \(T_{\text{error}}\). Fig. 3(a) (row 1) shows the original events while Fig. 3(b), (c) and (d) (row 1) shows the variation of events in temporal space with 8, 16 and 24 quantized timestamps respectively.
4.2 Performance with varying Bit Rate

The QT is optimized for a particular operational bit rate for the intensities only. In these experiments, \( r_4 = 1, r_8 = 2r_4, r_{16} = 3r_4 \) and \( r_{32} = 4r_4 \). The temporal bins is \( T_{\text{bin}} = 16 \). The performance table is shown in Table 2. As the bit rate reduces, the PSNR and SSIM reduces, while \( T_{\text{error}} \) and CR increases. This indicates bigger blocks in the QT for lower bit rates with considerably low distortion. For the Indoor6 and Indoor9 sequences, the number of events are less than Outdoor3 sequence. Hence, we show the event compression corresponding to low bit rates for these sequences. Fig. 3 (row 2) shows the visual quality of the Outdoor3 (frame 150) for different intensity bit rates.

| Sequence | Bit Rate (Mbps) | PSNR (dB) | SSIM | \( T_{\text{error}} \) | CR |
|----------|----------------|-----------|------|----------------------|----|
| Outdoor3 | 1              | 36.06     | 0.9622 | 0.65                 | 11.56 |
|          | 0.5            | 34.57     | 0.9493 | 0.75                 | 11.98 |
|          | 0.3            | 33.64     | 0.9383 | 0.82                 | 11.30 |
| Indoor6  | 0.5            | 45.31     | 0.9972 | 0.24                 | 10.82 |
|          | 0.3            | 44.92     | 0.9959 | 0.26                 | 11.38 |
|          | 0.1            | 44.28     | 0.9936 | 0.30                 | 14.01 |
| Indoor9  | 0.5            | 43.20     | 0.9874 | 0.34                 | 12.25 |
|          | 0.3            | 42.78     | 0.9850 | 0.37                 | 13.22 |
|          | 0.1            | 41.62     | 0.9763 | 0.41                 | 16.55 |

Table 2: Performance with varying Bit Rate

4.3 Performance with varying Poisson Disk radius and temporal binning

In this experiment, the performance of the event compression system is evaluated at a particular bit rate, but by varying the Poisson disk radius \( r_4 \). As in section 4.2, \( r_8 = 2r_4, r_{16} = 3r_4 \) and \( r_{32} = 4r_4 \). The temporal binning of \( T_{\text{bin}} = 16 \) is used in this experiment. The PSNR and SSIM reduces as \( r_4 \) increases. The \( T_{\text{error}} \) increases with increase of \( r_4 \) indicating the increase of temporal distortion. The compression ratio also increases with the increase of \( r_4 \). A sample event images for the original and compressed events for Indoor6 sequence (frame 150) is shown in Fig. 3 (row 3) at 100 kbps. With the increase in \( r_4 \), the event image becomes less dense. It is noted that by setting different values of \( r_4 \), we can obtain a target bit rate for the events, but with a particular distortion value.
5 Applications

5.1 Corner Detection

We applied the uncompressed and compressed events for event based Harris corner detection. We used DAVIS dataset [24], Shapes Translation sequence which has moving objects of different shapes. We compress the events at Tbins = 16 based on QT generated from 200kbps bit rate for the intensity. The Poisson disk sampling radius is set at \( r_4 = 1 \) and \( r_4 = 2 \) respectively. Fig. 4(a) and Fig. 4(b) top row shows the original uncompressed event image and Harris corner detections (in red squares) respectively. The corner detections from compressed events with \( r_4 = 1 \) is shown in blue circles Fig. 4(c) top row, while green diamonds indicates the corners detected from compressed events with \( r_4 = 2 \), shown in Fig. 4(d) top row. It is evident that the red squares and blue circles overlap for a large candidate points, while the green diamonds overlap with the red squares are less.

![Corner detection](image)

Figure 4: Row 1: Corner detection on DAVIS [24]; Row 2: Image reconstruction on RGB-DAVIS [11]

5.2 Image Reconstruction

We use E2VID [8] for reconstruction of image from events. The results for reconstructed frame 6 of Outdoor3 sequence from RGB-DAVIS dataset is shown in Fig. 4 bottom row with temporal compression only. We used window duration of 33 ms in order to reconstruct these images. It is observed in Fig. 4(d) bottom row that with Tbins = 8, the reconstruction quality is poor. It improves with the increase in Tbins to 16 shown in Fig. 4(c) bottom row. With Tbins = 24, Fig. 4(b) bottom row, the reconstruction quality is close to reconstructed image with original events. Our preliminary results show the effectiveness of image reconstruction at much lesser event bit rate compared to the original uncompressed events. (For more results on Image Reconstruction please see supplementary material Sec. 2.)

6 Conclusion and Discussion

This paper proposes a novel method of lossy event compression method in spatio-temporal domain based on Poisson disk sampling which achieves state-of-the-art compression ratio. The algorithm relies on a QT segmentation map which for identifying the regions of interest. We show the effectiveness of the method in different experimental runs: with
different temporal binning, spatial binning and joint spatio-temporal binning. The compressed events are also applied to sample applications including corner detection and image reconstruction. As the performance deteriorates towards greater event compression, we observe that the downstream tasks are more sensitive to the temporal resolution. In the future, it will be interesting to evaluate our compression algorithm on other related tasks. However, note that the current event-based algorithms have not been optimized for uncompressed events. It is an interesting direction to adapt existing algorithms to compensate for the event distortions.

Appendix A: Quadtree Optimization

Quadtree (QT) may be generated in several ways depending on the system and application. In some prior works [37], where the system is developed as a Host-Chip communication problem in bandwidth-constrained environment, the QT is developed to compress the grayscale intensity frames. In this work, we use similar system setting in order to generate the QT based on the grayscale intensity frames. Note that the Host-Chip setting is not required. For a frame \( f_t \), we have a QT segmentation, skip or acquire modes for the leaves, and values for the leaves of acquire modes, denoted by \( S_t, Q_t, \) and \( V_t \), respectively. These are used to reconstruct the frame \( \hat{f}_t \) which is distorted compared to \( f_t \). The previously reconstructed frame \( \hat{f}_{t-1} \) is used to copy the values in the skip leaves of \( \hat{f}_t \). A combination of object detector and tracker determines the regions of interest (ROIs) for the next frame at \( t+1 \). These predicted ROIs for frame \( \hat{f}_{t+1} \) are used for generating \( S_{t+1}, Q_{t+1}, \) and \( V_{t+1} \).

The full resolution frame \( f_{t+1} \) is acquired at time \( t+1 \) from the imager. The ROIs, \( f_{t+1} \) and \( \hat{f}_t \) are inputs to the Viterbi Optimization Algorithm which provides the optimal QT structure \( S_{t+1} \) and skip-acquire modes \( Q_{t+1} \) subject to the bandwidth constraint \( B \). The Viterbi optimization provides a trade-off between the frame distortion and frame bit rate. This is done by minimizing the frame distortion \( D \) over the leaves of the QT \( x \) subject to a given maximum frame bit rate \( R_{max} \). The reconstructed frame \( \hat{f}_t \) along with frame \( f_{t+1} \) is used to compute the distortion.

The optimization is formulated as follows

\[
\text{argmin} \ D(x), \quad s. \ t. \ R(x) \leq R_{max}
\]

The distortion for each node of the QT is based on the skip-acquire acquisition mode \( Q_t \) of that node. If a particular node \( \hat{x}_t \) of a reconstructed frame at time \( t \) is skip, the distortion with respect to the new node at time \( t+1, x_{t+1} \), is given by

\[
D_s = |x_{t+1} - \hat{x}_t|,
\]

On the contrary, if the node is an acquire, the distortion is proportional to the standard deviation \( \sigma \) as shown in Eq. [6] where \( N \) is the maximum depth of the QT and \( n \) is the level of the QT where distortion is computed. The root and the most subdivided level is defined to be in level 0 and \( N \) respectively:

\[
D_a = \sigma \times 4^{N-n},
\]

The total distortion is therefore defined as

\[
D = D_s + D_a,
\]

The constrained discrete optimization of Eq. [4] is solved using Lagrangian relaxation, leading to solutions in the convex hull of the rate-distortion curve [41]. The Lagrangian cost function is of the form

\[
J_\lambda(x) = D(x) + \lambda R(x),
\]

where \( \lambda \geq 0 \) is a Lagrangian multiplier. It has been shown that if there is a \( \lambda^* \) such that

\[
x^* = \text{argmin} J_\lambda^*(x),
\]

which leads to \( R(x^*) = R_{max} \), then \( x^* \) is the optimal solution to Eq. [4]. This is solved using the Viterbi algorithm, shown in detail in [42].

In the distortion term, we want to prioritize the regions based on the bounding boxes, which are the ROIs of region \( i \). This is introduced by the weight factors \( w_i \) in each region \( i \). However, in case where region \( i \) occupies a large area within the frame, the amount of distortion may heavily outweigh other smaller regions. We want to have a weighted distortion independently of the area of ROI \( i \). This is done by dividing the weighted distortion by the area of the ROI of region \( i \), thus modifying Eq. [5] as
Figure 5: Rate Control using Bezier Curve [41]

\[ J_\lambda(x) = \sum_{i \in \Omega} \frac{w_i D_i(x_i)}{A_i} + \lambda R(x), \]  

(10)

where, \( \Omega \) is the set of differently weighted regions, \( D_i \) the distortion of region \( i \), \( w_i \) the weight of region \( i \), \( A_i \) the area of region \( i \) and \( x_i \) the leaves in the QT of region \( i \).

In order to compress the grayscale intensity frames at desired bit-rate in an optimal manner, \( \lambda \) value in the Lagrangian multiplier is adjusted at each frame to achieve the desired bit rate. The optimal \( \lambda^* \) is computed by a convex search in the Bezier curve [41]. A sample curve is shown in Fig. 1.

In this work, for compressing the events, we consider the events in both the skip-acquire regions of the QT and compress the events depending on the QT block size. However, depending on the application, we may just compress events in the acquire regions of the QT.

Appendix B: Additional Image Reconstruction Results

We perform additional experiments on image reconstruction from events. The results for reconstructed frame 16 of Outdoor3 sequence from RGB-DAVIS dataset [40] is shown in Fig. 1. We generated the QT structure with 0.3 Mbps. The Poisson sampling disk radius was set to \( r_4 = 1 \). The reconstruction at Tbins = 8, 16 and 24 are shown in Fig. 2(d), Fig. 2(c) and Fig. 2(b) respectively. We used window duration of 33 ms in order to reconstruct these images. It is observed that with Tbins = 8, the reconstruction quality is poor, but improves with the increase in Tbins to 16 and further With Tbins = 24.

The image reconstruction of frame 6 of Office spiral sequence of DAVIS dataset [24] is shown in Fig. 3 using Tbins = 24. Fig. 3(a) shows the image reconstructed from original events while Fig. 3(b) and Fig. 3(c) shows the image reconstruction for Poisson disk radius \( r_4 = 1 \) and \( r_4 = 2 \) respectively. We also reconstruct image using Tbins = 16 along with Poisson disk radius \( r_4 = 1 \) and \( r_4 = 2 \) shown in Fig. 4(b) and Fig. 4(c) respectively. The reconstructions are close to the original image reconstruction. In both these experiments (shown in Fig. 3 and Fig. 4) we used window duration of 60 ms to reconstruct these images.
Figure 6: Image reconstruction at 0.3 Mbps bit rate at different temporal binning. We encourage the reader to enlarge the view for better visualization.

Figure 7: Image reconstruction at 0.3 Mbps bit rate at Tbins=24 with different Poisson Radius $r_4$.

References

[1] Giacomo Indiveri and Rodney Douglas. Neuromorphic vision sensors. *Science*, 288(5469):1189–1190, 2000.

[2] Bongki Son, Yunjae Suh, Sungwo Kim, Heejae Jung, Jun-Seok Kim, Changwoo Shin, Keunju Park, Kyoobin Lee, Jinman Park, Jooyeon Woo, et al. A 640 × 480 dynamic vision sensor with a 9µm pixel and 300Meps address-event representation. In *IEEE International Solid-State Circuits Conference (ISSCC)*, pages 66–67, 2017.

[3] Patrick Lichtsteiner, Christoph Posch, and Tobi Delbruck. A 128 × 128 120 db 15 µs latency asynchronous temporal contrast vision sensor. *IEEE journal of solid-state circuits*, 43(2):566–576, 2008.

[4] Zihao Winston Wang, Weixin Jiang, Kuan He, Boxin Shi, Aggelos Katsaggelos, and Oliver Cossairt. Event-driven video frame synthesis. In *Proc. of the IEEE International Conference on Computer Vision (ICCV) Workshops*, 2019.

[5] Cedric Scheerlinck, Henri Rebecq, Daniel Gehrig, Nick Barnes, Robert Mahony, and Davide Scaramuzza. Fast image reconstruction with an event camera. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 156–163, 2020.

[6] Prasan Shedligeri and Kaushik Mitra. Photorealistic image reconstruction from hybrid intensity and event-based sensor. *Journal of Electronic Imaging*, 28(6):063012, 2019.

[7] Lin Wang, Yo-Sung Ho, Kuk-Jin Yoon, et al. Event-based high dynamic range image and very high frame rate video generation using conditional generative adversarial networks. In *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10081–10090, 2019.

[8] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. High speed and high dynamic range video with an event camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[9] Lin Wang, Tae-Kyun Kim, and Kuk-Jin Yoon. Eventsr: From asynchronous events to image reconstruction, restoration, and super-resolution via end-to-end adversarial learning. *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
Figure 8: Image reconstruction at 0.3 Mbps bit rate at Tbins=16 with different Poisson Radius $r_4$.

[10] IS Mostafavi, Jonghyun Choi, and Kuk-Jin Yoon. Learning to super resolve intensity images from events. *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[11] Zihao Winston Wang, Peiqi Duan, Oliver Cossairt, Aggelos Katsaggelos, Tiejun Huang, and Boxin Shi. Joint filtering of intensity images and neuromorphic events for high-resolution noise-robust imaging. In *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[12] Lan Xu, Weipeng Xu, Vladislav Golyanik, Marc Habermann, Lu Fang, and Christian Theobalt. Eventcap: Monocular 3d capture of high-speed human motions using an event camera. *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[13] Henri Rebecq, Guillermo Gallego, Elias Mueggler, and Davide Scaramuzza. Emvs: Event-based multi-view stereo—3d reconstruction with an event camera in real-time. *International Journal of Computer Vision*, 126(12):1394–1414, 2018.

[14] Hanme Kim, Stefan Leutenegger, and Andrew J Davison. Real-time 3d reconstruction and 6-dof tracking with an event camera. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 349–364. Springer, 2016.

[15] Andrea Censi and Davide Scaramuzza. Low-latency event-based visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 703–710. IEEE, 2014.

[16] Antoni Rosinol Vidal, Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza. Ultimate slam? combining events, images, and imu for robust visual slam in hdr and high-speed scenarios. *IEEE Robotics and Automation Letters*, 3(2):994–1001, 2018.

[17] Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis. Event-based feature tracking with probabilistic data association. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4465–4470. IEEE, 2017.

[18] Xavier Lagorce, Cédric Meyer, Sio-Hoi Ieng, David Filliat, and Ryad Benosman. Asynchronous event-based multikernel algorithm for high-speed visual features tracking. *IEEE transactions on neural networks and learning systems*, 26(8):1710–1720, 2014.

[19] Zhichao Bi, Siwei Dong, Yonghong Tian, and Tiejun Huang. Spike coding for dynamic vision sensors. In *2018 Data Compression Conference*, pages 117–126. IEEE, 2018.

[20] Siwei Dong, Zhichao Bi, Yonghong Tian, and Tiejun Huang. Spike coding for dynamic vision sensor in intelligent driving. *IEEE Internet of Things Journal*, 6(1):60–71, 2018.

[21] Jacob Ziv and Abraham Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on information theory*, 23(3):337–343, 1977.

[22] https://www.7-zip.org/sdk.html Last accessed on April 30, 2020.

[23] Alex Zihao Zhu, Dinesh Thakur, Tolga Ozaslan, Bernd Pfrommer, Vijay Kumar, and Kostas Daniilidis. The multivehicle stereo event camera dataset: An event camera dataset for 3d perception. *IEEE Robotics and Automation Letters*, 3(3):2032–2039, 2018.

[24] Elias Mueggler, Henri Rebecq, Guillermo Gallego, Tobi Delbruck, and Davide Scaramuzza. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *The International Journal of Robotics Research*, 36(2):142–149, 2017.
[25] Dong-Ming Yan, Jian-Wei Guo, Bin Wang, Xiao-Peng Zhang, and Peter Wonka. A survey of blue-noise sampling and its applications. *Journal of Computer Science and Technology*, 30(3):439–452, 2015.

[26] Robert L. Cook. Stochastic sampling in computer graphics. *ACM Transactions on Graphics (TOG)*, 5(1):51–72, 1986.

[27] Li-Yi Wei. Parallel poisson disk sampling. *ACM Transactions on Graphics (TOG)*, 27(3):1–9, 2008.

[28] Florent Wachtel, Adrien Pilleboeuf, David Coeurjolly, Katherine Breeden, Gurprit Singh, Gaël Cathelin, Fernando De Goes, Mathieu Desbrun, and Victor Ostromoukhov. Fast tile-based adaptive sampling with user-specified fourier spectra. *ACM Transactions on Graphics (TOG)*, 33(4):1–11, 2014.

[29] Cem Yuksel. Sample elimination for generating poisson disk sample sets. In *Computer Graphics Forum*, volume 34, pages 25–32. Wiley Online Library, 2015.

[30] Alexander Dieckmann and Reinhard Klein. Hierarchical additive poisson disk sampling. In *Proceedings of the Conference on Vision, Modeling, and Visualization*, pages 79–87. Eurographics Association, 2018.

[31] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on circuits and systems for video technology*, 22(12):1649–1668, 2012.

[32] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the h.264/avc video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, 2003.

[33] Wen Gao and Siwei Ma. An overview of avs2 standard. In *Advanced Video Coding Systems*, pages 35–49. Springer, 2014.

[34] Debargha Mukherjee, Jim Bankoski, Adrian Grange, Jingning Han, John Koleszar, Paul Wilkins, Yaowu Xu, and Ronald Bultje. The latest open-source video codec vp9—an overview and preliminary results. In *2013 Picture Coding Symposium (PCS)*, pages 390–393. IEEE, 2013.

[35] [https://inivation.com/support/software/fileformat/](https://inivation.com/support/software/fileformat/) Last accessed on April 30, 2020.

[36] Nabeel Khan and Maria G. Martini. Bandwidth modeling of silicon retinas for next generation visual sensor networks. *Sensors (Basel)*, 19(8):1751–1777, 2019.

[37] Srutarshi Banerjee, Juan G. Serra, Henry H. Chopp, Oliver Cossairt, and Aggelos K. Katsaggelos. An adaptive video acquisition scheme for object tracking. In 2019 27th European Signal Processing Conference (EUSIPCO), pages 1–5. IEEE, 2019.

[38] Guido M Schuster and Aggelos K Katsaggelos. An optimal quadtree-based motion estimation and motion-compensated interpolation scheme for video compression. *IEEE Transactions on Image Processing*, 7(11):1505–1523, 1998.

[39] Robert Bridson. Fast poisson disk sampling in arbitrary dimensions. *SIGGRAPH sketches*, 10:1278780–1278807, 2007.

[40] [https://sites.google.com/view/guided-event-filtering](https://sites.google.com/view/guided-event-filtering) Last accessed on April 30, 2020.

[41] Guido M Schuster and Aggelos K Katsaggelos. An optimal quadtree-based motion estimation and motion-compensated interpolation scheme for video compression. *IEEE Transactions on Image Processing*, 7(11):1505–1523, 1998.

[42] Guido M Schuster and Aggelos K Katsaggelos. A video compression scheme with optimal bit allocation among segmentation, motion, and residual error. *IEEE Transactions on Image Processing*, 6(11):1487–1502, 1997.