Space-time Event Clouds for Gesture Recognition: from RGB Cameras to Event Cameras

Qinyi Wang[1] Yexin Zhang[1] Junsong Yuan[2] Yilong Lu[1]
[1]Nanyang Technological University, [2]State University of New York at Buffalo
{qwang014, yexin001, eylu}@e.ntu.edu.sg, jsyuan@buffalo.edu

Abstract

The recently developed event cameras can directly sense the motion by generating an asynchronous sequence of events, i.e., an event stream, where each individual event (x, y, t) corresponds to the space-time location when a pixel sensor captures an intensity change. Compared with RGB cameras, event cameras are frameless but can capture much faster motion, therefore have great potential for recognizing gestures of fast motions. To deal with the unique output of event cameras, previous methods often treat event streams as time sequences, thus do not fully explore the space-time sparsity and structure of the event stream data. In this work, we treat the event stream as a set of 3D points in space-time, i.e., space-time event clouds. To analyze event clouds and recognize gestures, we propose to leverage PointNet, a neural network architecture originally designed for matching and recognizing 3D point clouds. We adapt PointNet to cater to event clouds for real-time gesture recognition. On the benchmark dataset of event camera based gesture recognition, i.e., IBM DVS128 Gesture dataset, our proposed method achieves a high accuracy of 97.08% and performs the best among existing methods.

1. Introduction

Hand gestures are widely used in human-machine interaction [5, 13, 15, 24, 26], sign-language recognition [28] and gaming [4, 6]. Motion blur makes rapid gesture recognition a challenging task when using a 30 frames per second RGB or depth camera [10]. A common solution to capture high-speed motions is to increase frame rate. The successive frames contain enormous amounts of redundant information. Processing all the frames wastes memory access, computational power and time. Thus, real-time analysis of the big trunk of video data becomes another challenge.

Instead of capturing synchronized frames, event cameras directly sense the motions in the scene as an asychrounous sequence of events. As shown in figure 1(a), conventional camera captures all pixel intensities at a fixed frame rate. It captures a clear shape of stationary object (red ball) and a blurred shape of a fast moving object (blue ball). As shown in figure 1(b), event camera only captures an asychrounous sequence of intensity changes caused by the fast moving blue ball while information of stationary objects (red ball and background) will not be recorded. Each intensity change is referred as an event encoding the spatial coordinates (x, y) of a pixel reporting a change and a precise timestamp t indicating when that change happened. Recent explorations achieve amazing results on using event cameras for trajectory and motion estimation [8, 16], simultaneous localization and mapping (SLAM) [11] and steering prediction for self-driving cars [14]. Compared with conventional methods using RGB camera or depth camera [7, 22, 23, 25, 27] for gesture recognition, event cameras can easily capture movements which only cameras with more than 1000 frames per second can capture. On the other hand, only the local pixel-level changes are transmitted at the time they occur. Event cameras address limitations of conventional cameras easily by its output-by-demand na-

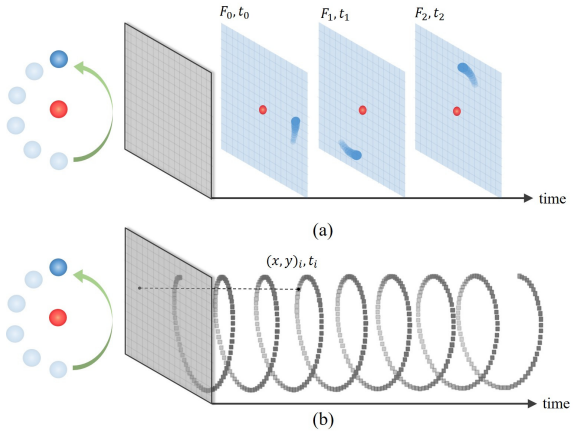

Figure 1. (a) Conventional RGB camera captures all pixel intensities at a fixed frame rate. (b) Event camera captures intensity changes caused by the moving objects asynchronously.

ture. These properties make event-based cameras a perfect complementary in real-time and embedded applications with limited computation resources and power budget.

In conventional event-camera based classification systems, event streams are usually treated as temporal sequences. Events are accumulated within fixed time intervals to form a virtual frame for feature learning. However, treating event streams as a sequence of virtual frames cannot fully utilize the spatio-temporal sparsity of event data. Meanwhile short-term spatio-temporal structure captured by the event camera within each time interval is lost when all events are accumulated over time. To address these limitations, we propose a novel representation to interpret an event sequence as a 3D point clouds in space and time. In our proposed method, each event becomes a point in a three-dimension continuum represented as (x, y, t). Each gesture generates a distinctive cloud of events in (x, y, t) coordinate system and we call it *space-time event clouds*. By interpreting event streams as space-time event clouds, spatial features and temporal features are fused in a 3D space-time continuum. Therefore, the recognition of a gesture becomes the recognition of geometric distribution of the event clouds generated by that gesture, which is in spirit similar to 3D object recognition.

To robustly differentiate point clouds and recognize corresponding gestures, we propose to leverage recent machine learning approaches that help recognize 3D objects. PointNet [20] is a neural network architecture originally designed to for 3D object classification and segmentation problems. We propose to adapt PointNet to analyze event-camera data, i.e., event clouds. The event cloud is hierarchically analyzed using a PointNet-based architecture to capture the essential spatio-temproal structure of the hand motion, then the learned feature is used for classification. The feature learning and classification are in an end-to-end way. To achieve real-time gesture recognition, we developed a rolling buffer framework. To achieve fast response in real-time, a sliding window is used to capture events in a small time interval to update events rolling buffer. The events rolling buffer storing most recent event clouds enables the network to make use of past information efficiently. The output rolling buffer is used to store prediction results and a decision filter is applied to remove unreliable predictions. We evaluate the proposed framework by comparing its end-to-end accuracy and latency with existing methods. The proposed framework achieves 97.08% accuracy in IBM DVS128 Gesture dataset and 118ms latency, which outperforms existing methods [1].

2. Related Work

In event-camera based classification system, how to extract useful information and features from the sparse and asynchronous event data is a key challenge. Intuitively, event streams can be cast back to frames to form grayscale or binary images and conventional feature extraction techniques can be applied. Chen *et al.* [3] proposed a bio-inspired hierarchical line segment extraction unit to perform size and position invariant human posture categorization on the binary event images. Inspired by the idea of using tempotron classifier [9] to recognize spatio-temporal neuron spiking patterns, Pérez-Carrasco *et al.* [19] proposed an event-driven convolutional neural network to consume event data. In this approach, sensor plane of event camera is viewed as an array of neurons in leaky integrate-and-fire (LIF) model, each event is treated as an input spike to fire corresponding neuron. O'Connor *et al.* [17] proposed a spiking deep belief network (SDBN) to perform feature extraction, information fusion, and classification at event level, which is robust to noise, scaling, translation and rotation. In [18], events are treated as LIF for motion detection. A probability-based method is proposed to combined specialty and popularity of events occurrence according to their addresses on the image plane. Lee *et al.* [12] were the first to develope an event-based gesture recognition system with an event camera and to show a postprocessing step with LIF. It achieved recognition rates well over 90% under a variety of variable conditions with static and dynamic backgrounds. In most recent work [1], they pointed out that the advantages of event cameras are diluted if their event streams must be cast back into synchronous frames for the benefit of conventional processors downstream. Conventional processors (e.g. CPUs and GPUs) are designed to efficiently process dense, synchronously delivered data structures, not sparse, asynchronous event streams. They solved this problem by combine an event camera with an event-based neuromorphic processor TrueNorth to perform real-time gesture recognition using CNN approach. It achieved an accuracy of 96.49% in IBM DVS128 Gesture Dataset.

3. Proposed Method

In conventional vision system, intensity information of all pixels are captured at each time. The temporal change captured is determined by the frame rate varying from tens of to thousands of frames per second. Thus, conventional video streams are dense in spatial domain but sparse in time domain. In event-camera based vision system, event camera outputs an event whenever the intensity change of a certain pixel exceeds the threshold. Each event carries only the spatial coordinates (x, y) of a pixel reporting a change and a timestamp t indicating when that change happened [1]. Event streams are asynchronous and have a much higher temporal resolution from microseconds to nanoseconds level. Thus event streams are sparse in spatial domain and dense in temporal domain. Although there are many different approaches to extract features from the unique type of asynchronous event streams, they all treat the event

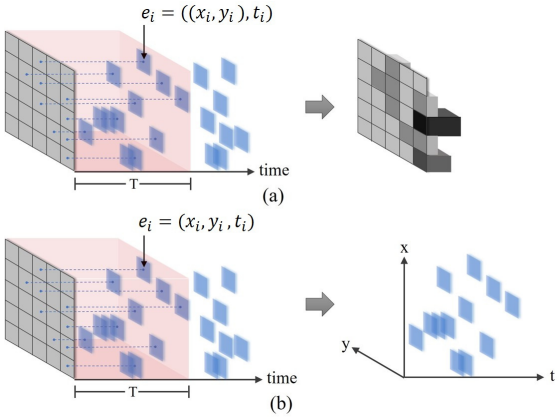

Figure 2. (a)Treat event streams as temporal sequence and cast events back to frames. (b)Treat events as space-time event clouds.

streams as temporal sequences. The concept of temporal sequences and matrix of pixels inherited from conventional video analysis of frames is applied to event streams in a similar way. However, treating event streams as temporal sequences does not fully utilize the spatial sparsity of event streams, meanwhile the dense temporal information captured is diluted. In this work, we think out of the conventional concept of temporal sequences and view the event streams in a three dimensional space continuum. In conventional 3D space, a point is denoted as (x, y, z) while in the world of event cameras, a point (event) in 3D space is denoted as (x, y, t). The continuous event stream forms a cloud of points (events) in 3D space thus we call it *space-time event clouds*.

3.1. From Temporal Sequences to Space-time Event Clouds

In human vision system, we see the world as three dimensions of space plus one dimension of time. Similarly, camera vision systems see the world as two dimensions of image space plus one dimension of time. In Newtonian view of space and time, time is a measurement separated from the space dimensions and it is an independent variable flowing on its own. In conventional video analysis, algorithms are intuitively developed under Newtonian idea of time, where time is considered as a measurement of duration, sequential order or frequency of motions. Thus, both frames from conventional cameras and event streams from event cameras are treated as temporal sequences. When treating event streams as temporal sequences, an event e_i is denoted as

$$e_i = ((x_i, y_i), t_i) \tag{1}$$

The event streams are time series data recording intensity changes in image space in a chronological order.

In modern physics, scientists fuse the three dimensions of space and the one dimension of time into a single four-dimensional continuum, where time is a dimension identical to the other three dimensions. Inspired by the modern physics' understanding of time, here, we fuse the two dimensions of image space and the one dimension of time into a three-dimension continuum in the event-based vision system. Time t is identical to x, y spaces and measured by a numerical number with physical meaning. In proposed approach, an event e_i is denoted as

$$e_i = (x_i, y_i, t_i) \tag{2}$$

An event becomes a point in a 3D space and event streams form 3D space-time event clouds.

In conventional event-camera based classification system, event streams are divided into multiple segments for feature extraction. Temporal segmentation divides events by fixed time intervals or fixed number of events, while soft segmentation adaptively obtain segments according to some certain predefined rules [18]. In a selected time interval T, assume a total number of n events are generated by an event camera with a $N \times M$ resolution. When treating event streams as temporal sequences, a set of events within time interval T is expressed as:

$$S^T_{temporal} = \{e_i = (x_i, y_i) | t_i \in T, i = 1, 2, ..., n)\} \tag{3}$$

When treating event streams as 3D space-time event clouds, a set of events within time interval T is expressed as:

$$S^T_{3D} = \{e_i = (x_i, y_i, t_i) | t_i \in T, i = 1, 2, ..., n)\} \tag{4}$$

In conventional time sequence approach, no matter modeling events as postsynaptic potentials for neural spiking pattern classification [19] or interpreting event streams as bag of events for joint probability distribution classification [18], feature extraction is conducted on set $S^T_{temporal}$ and events are accumulated over time on the image plane. Thus, an $N \times M$ array is required to store the spiking patterns or probability patterns as shown in Figure 2(a). Processing of event sequences becomes the processing of dense arrays as what we do in conventional video analysis. The event streams are extremely sparse in spatial domain. In DVS128 Gesture Dataset [1], only around 10% of the array are occupied by valid data. The advantage of spatial sparsity is largely diluted and memory cost is increased. The accumulation of events over a short time interval also dilutes the dense temporal information captured by event cameras. The temporal orders among events within that interval is lost during integration.

In proposed approach, event streams are treated as space-time event clouds, which are sets of points (events) in 3D space. Spatial features and temporal features are fused in

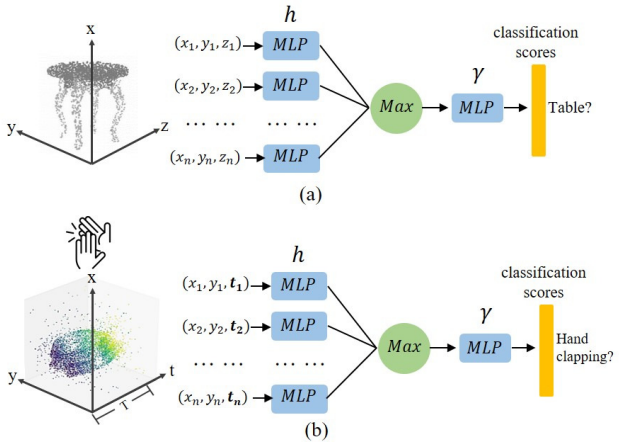

Figure 3. (a)PointNet consumes point clouds for object classification. (b)In this work, PointNet is adapted to consume space-time event clouds for gesture recognition in event-camera based classification system.

the 3D space-time continuum as shown in Figure 2(b). Processing of event streams becomes set operation over S^T_{3D}. It fully utilizes the spatial sparsity and achieves an effective usage of memories. As each event is a point in 3D space denoted with coordinates (x, y, t), there is no integration over time any more. Thus the dense time information is also well preserved and is transformed to geometric information. As conventional neural network models are no longer suitable to process sparse set data, a network architecture to learn geometric distribution features of event clouds in 3D space is needed. Finally, the recognition of a hand gesture becomes the recognition of geometric distribution of the 3D space-time event clouds generated by that gesture.

3.2. PointNet for Space-time Event Clouds Based Gesture Recognition

3.2.1 PointNet

PointNet [20] is a neural network architecture that directly takes point clouds as input and it learns to summarize the geometric features of the input point clouds. A point cloud is denoted as a set of 3D points $S = \{p_i | i = 1, ..., n\}$, where each point p_i is a vector of its coordinates (x, y, z). The architecture of PointNet is shown in Figure 3(a). Each input point p_i is processed by a shared multi-layer perceptron network and it is trained to capture different properties of the input set. As points in the set are unordered, the input sequence of points should not affect the output. A single symmetric function, max pooling, is applied. The network learns a set of optimization functions to select informative points of the input set of points. The fully connected layers aggregate these learnt optimal values into a global descriptor of the entire shape of point clouds for classification. Although simple, PointNet architecture demonstrates universal approximation ability of continuous set functions. A Hausdorff continuous symmetric function $f : 2^x \rightarrow \mathbb{R}$ can be arbitrarily approximated by PointNet as

$$|f(S) - \gamma(\underset{p_i \in S}{\text{MAX}}\{h(p_i)\})| < \epsilon \tag{5}$$

where $p_1, ..., p_n$ are elements in S ordered arbitrarily, h and γ are continuous functions approximated multi-layer perception networks, and MAX is an element-wise max pooling operator.

Considering that PointNet is highly efficient and demonstrates strong performance in point clouds classification, PointNet is adapted and trained for space-time event clouds classification in this work. As shown in Figure 3(b), our input is a set of event clouds within time interval T denoted as $S^T = \{e_i | i = 1, ..., n\}$, where each event e_i is a vector of its coordinates (x, y, t). The universal approximation becomes

$$|f(S^T) - \gamma(\underset{e_i \in S^T}{\text{MAX}}\{h(e_i)\})| < \epsilon \tag{6}$$

where $e_1, ..., e_n$ are elements in S^T ordered arbitrarily, h and γ are continuous functions approximated multi-layer perception networks, and MAX is an element-wise max pooling operator. PointNet architecture models set fuctions that directly take a set of events S^T as input and output a global feature of the input event cloud.

3.2.2 PointNet++: a PointNet-based Hierarchic Feature Extraction Architecture

PointNet learns to summarize a global feature of the input event clouds. It is not able to capture local structures induced by the metric space points live in, limiting its ability to recognize fine-grained patterns or nonuniform points distributions. The event clouds generated by movements are not uniform and are largely dependent on the speed of movements. The event clouds generated by hand gestures involving two hands are more complex. These two challenges make the global feature learned by PointNet less discriminative. PointNet++ [21] is an advanced version of PointNet that applies PointNet recursively on a nested partitioning of the input set. In this work, PointNet++ architecture is trained to aggregate local and global features of event clouds. A hierarchical feature extraction architecture is shown in Figure 4. Given the input event cloud of time interval T denoted as $S^T = \{e_1, e_2, ..., e_n\}$, farthest point sampling (FPS) is used to select N_1 number of events as the central point of N_1 sub-regions. Ball query finds out all the neighboring events within radius r_1 and a fixed K_1 events is sampled. Thus, the input event cloud is portioned into N_1 sub-event clouds and each sub event cloud contains K_1 events. A basic PointNet network is trained to learn

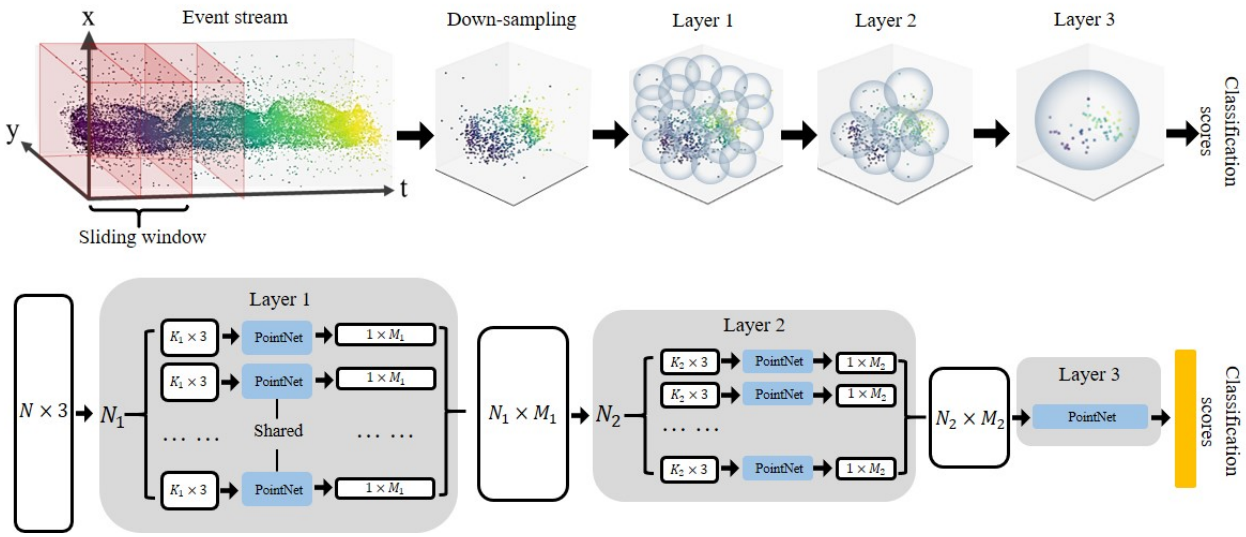

Figure 4. A PointNet-based hierarchic feature learning architecture, PointNet++, is trained to learn local and global features of event clouds generated by hand gestures

local feature of each sub event clouds. The learned feature summarizes the geometric distribution of events within each sub-region and each learned feature becomes a point in higher dimension M_1. They form a new event cloud in metric space which is again partitioned into smaller sub event clouds. A basic PointNet network learns the local feature of each newly selected sub-region. The local features are learned and aggregated layer by layer. The feature in the final layer contains both local and global features of the input event cloud and the fully connected layers are applied to classify entire distribution of the event clouds generated by movements.

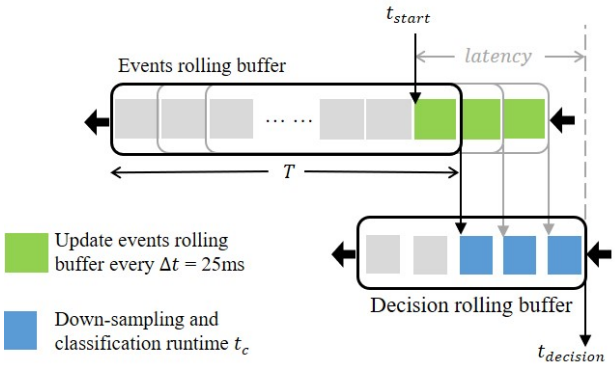

Figure 5. Rolling buffer framework for real time recognition

3.3. Online Real-time Gesture Recognition

Many real time interaction tasks desire very low latency. Some researches show that a good real-time gesture interaction application needs a response time within 100ms to 200ms [2]. This means that decision must be given within 200ms after starting of the gesture. As a 100-200ms duration including processing time only contains a very short snippet of the gesture, past information is combined with current input for a better classification of a gesture in this work. As shown in Figure 5, a rolling buffer mechanism is applied to store past events of a predefined time duration T. The events rolling buffer is updated every Δt, which means that the latest Δt events will be popped into the events rolling buffer each time while the earlier Δt events will be flopped. For each classification, the latest Δt input events are fused with past events stored in the events rolling buffer to form the input event clouds to the trained network for classification and down-sampling technique is applied. To improve the classification performance, a decision rolling buffer is used to store classification results for consecutive L windows and a majority decision filter is used to make final classification for the input window captured $\frac{(L+1)}{2} \times \Delta t$ before. This decision filter outputs the class with maximum frequency in decision buffer to remove unreliable classification results. The latency of this framework is $\frac{(L+1)}{2} \times \Delta t + t_c$, where t_c is the average processing time of down-sampling and classification. By this rolling buffer mechanism with a decision filter, a high classification accuracy can be achieved within a response time around 100ms. The proposed event-clouds based online processing framework with low memory requirement is suitable for many portable platforms with limited memory budget.

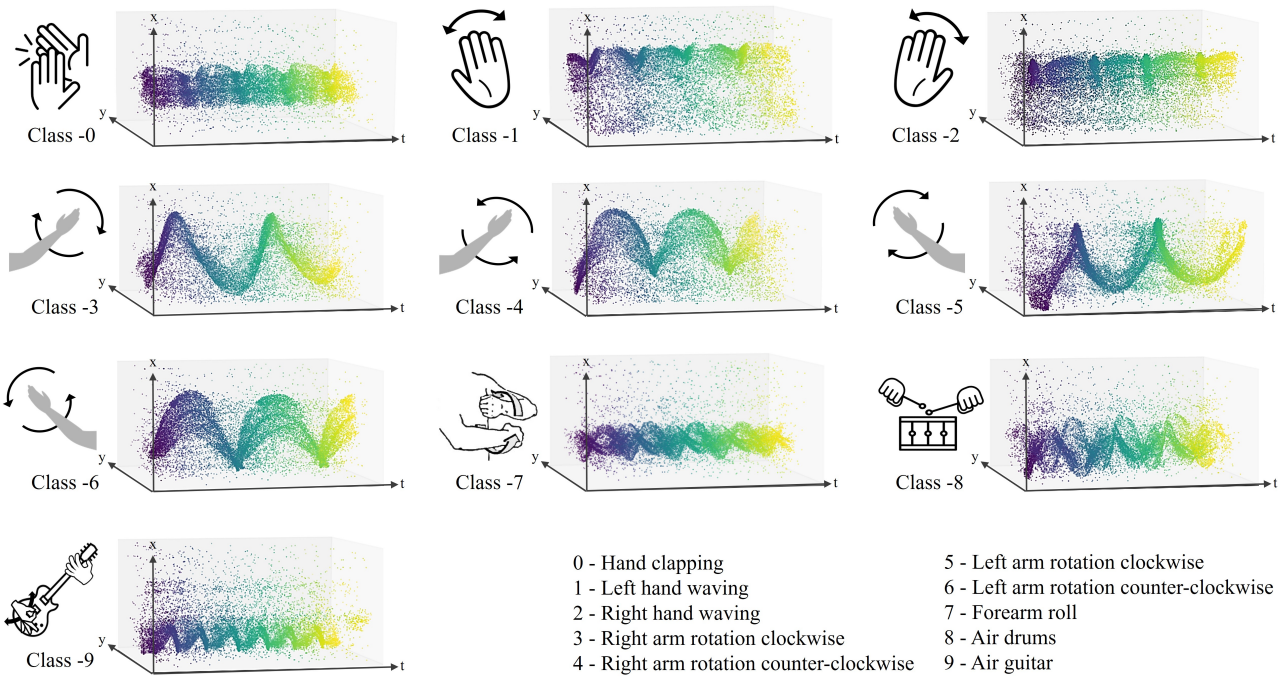

Figure 6. Two-second snippets of 10 classes in DVS128 Gesture Dataset.

4. Experiments

The experiments are divided into two parts. First, we evaluate the classification ability of proposed classifier against different classifiers for event streams in offline test. For comparison, a LSTM-based classifier is used to handle event streams as pure temporal sequences. Both PointNet-based and PointNet++-based classifiers are trained to demonstrate the contribution of the hierarchical feature extraction architecture. Second, we show the fast response real-time hand gesture recognition system, where a well-trained classifier is embedded in the rolling buffer framework. We compare the proposed system with state-of-the-art event-camera based hand gesture classification system.

4.1. Dataset

The DVS128 Gesture Dataset [1] from IBM is used in this paper. This dataset is captured by the iniLabs DVS128 camera with a 128×128 resolution. Thus, the spatial coordinates x, y are within the range of $[1, 128]$, where $x, y \in \mathbb{N}$. This dataset includes 1,342 instances of 11 classes of hand and arm gestures, grouped in 122 trials collected from 29 subjects under different lighting conditions including natural light, fluorescent light, and LED light. Snippets of first ten classes are shown in Figure 6. The 11th class is labeled as random gestures excluding first ten classes. As provided in the dataset [1], 23 subjects are used as the training set and 6 subjects are reserved for out-of-sample validation. The dataset is available at http://research.ibm.com/dvsgesture/.

4.2. Classifiers Training

Different from conventional 3D point clouds, event clouds are continuous video streams. To train the classifiers with the appropriate signals, we first preprocess the dataset by segmenting raw event streams through a fixed size sliding window that shapes the input to classifiers. The sliding window size determines input length of an event stream segment used for one classification, which equals to the total length of event streams stored in the event rolling buffer. In this experiment, the sliding window size is selected as a fixed time interval $T = 1s, 0.5s, 0.25s$ and each window is labeled accordingly. The step size is chosen to be the half of sliding window size when generating the training input.

As event cameras generate data on output-by-demand nature, event rate per second is not fixed and is largely determined by the range and frequency of movements. Given a sliding window of 1s, the number of events varies from 5k to 300k in the dataset [1] used in this experiment. In [20], it has been proved that PointNet learns to summarize a shape by a sparse set of critical points, which means that it is not necessary to input all the events (points) generated in order to obtain a reliable prediction. In our experiment, we randomly sampled a subset of events $S^T_{3D,n=256,512,1024}$ from the original event stream S^T_{raw} within each sliding window as critical points input to the classifiers.

Experiment (model,classes,events)	Sliding window		
	T=0.25s	T=0.50s	T=1.00s
LSTM,10,256	80.96	85.58	84.10
LSTM,10,512	**88.17**	86.55	82.28
PointNet,10,256	87.85	89.63	88.54
PointNet,10,512	88.67	**90.20**	89.61
PointNet,10,1024	88.77	89.68	89.92
PointNet++,10,256	95.28	95.59	95.54
PointNet++,10,512	95.39	**96.34**	95.61
PointNet++,10,1024	94.93	95.89	95.97
PointNet++,11,256	91.92	93.38	93.61
PointNet++,11,512	92.23	**94.10**	93.83
PointNet++,11,1024	91.87	91.91	92.63

Table 1. Classifers test results. Accuracy is reported in percentage.

As the event streams can be treated as temporal sequences, a two-hidden-layer-stacked LSTM model with 256 hidden neurons each layer is trained to consume event streams as temporal sequences. For a fair comparison, the input event sequence $S^T_{temporal,n}$ to the LSTM-based model is the same set of events in $S^T_{3D,n}$ but treating them as a temporal sequence of n events. The detailed setting of PointNet classifier is $MLP(64; 64; 64; 128; 1024) \rightarrow FC(512) \rightarrow FC(256; 0.7) \rightarrow FC(K)$. The detailed setting of PointNet++ (SSG) classifier is $SA(256; 0.2; [64; 64; 128]) \rightarrow SA(64; 0.4; [128; 128; 256]) \rightarrow SA([256; 512; 1024]) \rightarrow FC(512; 0.5) \rightarrow FC(256; 0.5) \rightarrow FC(K)$. All the classifiers are trained on Linux system with single GPU.

4.3. Classification ability of different classifers

Classification ability of classifiers are evaluated with different sliding window length ($T = 0.25s, 0.5s, 1s$) and different number of down-sampled events ($n = 256, 512, 1024$). The out-of-sample test results of different classifiers are shown in Table 1. LSTM-based models consume event streams as pure temporal sequences and learn the temporal features from the events. Lacking sufficient spatial information makes LSTM-based models achieve the lowest accuracy. PointNet summarizing a global feature of the input event clouds achieves a accuracy $4\% - 5\%$ higher than the LSTM-based approach. PointNet++ with a hierarchical feature learning architecture demonstrates a classification accuracy $6\% - 7\%$ higher than PointNet. When including random gestures in PointNet++ classifier, the accuracy decreases $2\% - 3\%$.

The confusion matrices of different classifiers with the same experiment setup are shown in Figure 7. For LSTM classifier, the error rates among class 0/7/8/9 are high. PointNet classifier mainly reduces the misclassification rate of class 7. PointNet++ classifier reduces misclassification rate significantly. When adding the interference of random

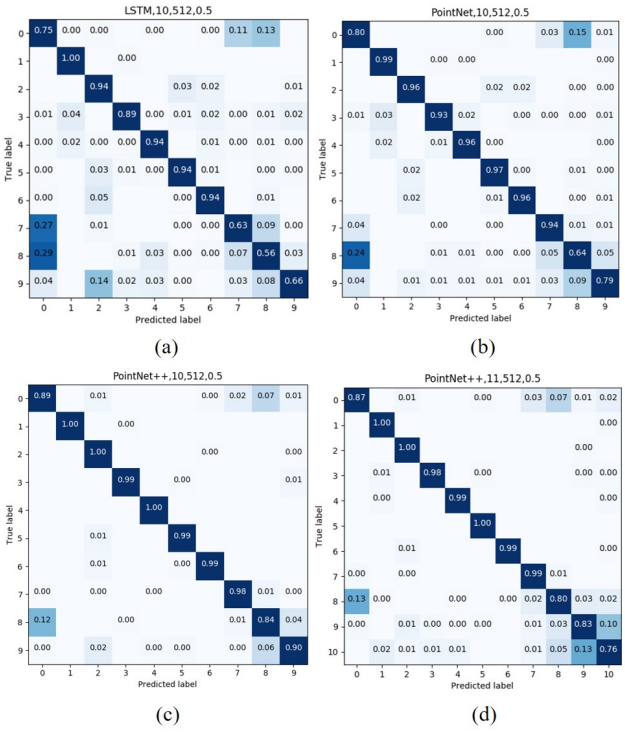

Figure 7. Confusion matrices of different classifiers on the same test setup, where 512 events are down-sampled from a 0.5s sliding window. Only non zero percentages are shown. (a)LSTM, 10 classes (b)PointNet, 10 classes (c)PointNet, 10 classes (d)PointNet++, 11 classes

gestures (class 10), it affects moslty the classification of class 8 and class 9. In summary, trained networks are more likely to be confused among these two-hands gestures (class 0/7/8/9). PointNet++ shows a significant increase on classification ability of two-hands gestures due to the hierarchical feature learning and aggregation process.

4.4. Online gesture recognition system

We embed the trained PointNet++ based event clouds classifier into the proposed rolling buffer framework. The performance of our online real-time gesture recognition framework is evaluated with different events rolling buffer length T and different decision buffer length L. The overall results are summarized in Table 2. Given the same dataset, Amir *et al* [1] using CNN approach achieved a system accuracy of 96.49% without interference of random gestures and a system accuracy of 94.59% when including random gestures. Our work using event clouds approach achieved a system accuracy of 97.08% without interference of random gestures and 95.32 %. In [1], the average latency from t_{start} to $t_{decision}$ are $104.6ms$ and $120.6ms$ respectively. In our approach, the average latency from t_{start} to $t_{decision}$ are $93ms$ (L=5), $118ms$ (L=7) and $143ms$ (L=9). For both tests of 10 classes and 11 classes, we achieved better than

This work							
Events rolling buffer length (= sliding window)	No.of events	System acc. (10 classes)			System acc. (11 classes)		
		$L = 5$	$L = 7$	$L = 9$	$L = 5$	$L = 7$	$L = 9$
T = 1.00s	n=1024	96.96	96.69	96.81	92.71	92.83	93.17
	n= 512	96.42	96.42	96.61	94.12	93.96	94.27
	n= 256	96.53	96.53	96.58	94.28	94.81	94.59
T = 0.50s	n=1024	96.84	96.89	96.84	92.45	92.58	92.47
	n= 512	96.93	96.81	96.80	94.33	94.47	94.73
	n= 256	96.91	**97.08**	96.97	94.68	95.08	**95.32**
T = 0.25s	n=1024	95.70	95.78	95.90	92.12	91.92	92.07
	n= 512	96.54	96.63	96.83	92.65	93.03	93.31
	n= 256	96.27	96.42	96.47	93.33	93.87	94.03
Benchmark accuracy [1]		96.49			94.59		

Table 2. System test results compared with state-of-the-art results. Accuracy is reported in percentage.

state-of-the-art accuracy with comparable system latency.

4.5. Robustness and efficiency of our method

As shown in Figure 6, it can be found that there are many outliers in the event clouds, which are not triggered by the target movements. It can be sensor noise or intensity changes caused by the background or light condition changes. In [20], it has been proved that PointNet itself is robust to extra noise points. Thus, in our approach, better than state-of-the-art result is achieved without using extra noise removal technique. Moreover, PointNet learns to summarize a shape by a sparse set of critical points without sacrificing the classification accuracy. It enables us to down sample a small subset of critical events from the large number of raw events as classifier input, which largely reduce the data to be processed by the classifier.

Compared with CNN-based approach, the data scale processed in proposed framework is particularly small. As shown in Figure 8, for a region of interest with 128×128 resolution, when casting the events back to virtual frames, each frame contains 16k pixels. The data processed by the CNN-based classifier per second is 16k×frame rate, where the frame rate is usually greater than 30fps. In our approach, the input to classifier is not more than 1k when down-sampling to 256 events per sliding window and the number of sliding window is set as 40. The data processed by the proposed classifier per second is around 1k×40, which is more than 10 times smaller than the CNN-based approach. It indicates that many real-time applications with strict timing and memory requirements are possible to be conquered by space-time event clouds concept with the proposed framework.

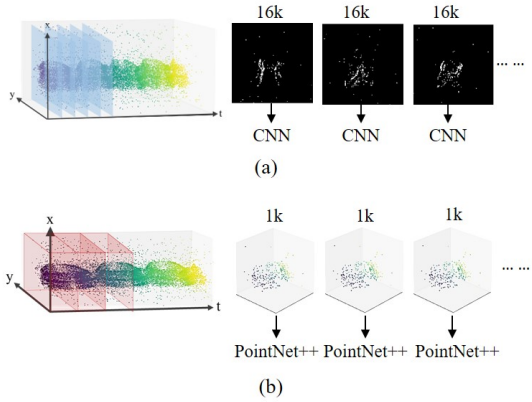

Figure 8. Efficiency of proposed framework (a)casting events back into virtual frames takes in 16k input (b)event clouds-based approach takes in 1k input

5. Conclusions

In real-time gesture recognition, event cameras successfully address the issues of motion blur and large scale redundant data inherent in conventional cameras by its output-by-demand nature. However, it is still challenging to think out of the conventional view of space and time when dealing with sparse and asynchronous event data. To the best of our knowledge, this is the first work to interpret event streams as space-time event clouds for gesture recognition problems. We leverage PointNet++ in 3D object recognition to analyze space-time event clouds. Our method achieves the best ever accuracy of 97.08% on IBM DVS128 Gesture dataset [1]. Our results show that the proposed event clouds concept is an effective representation to characterize the event streams from event cameras. It preserves both spatial and temporal information to analyze the event streams and is end-to-end learnable. As verified in our experiments, event clouds can be benefit from down-sampling too. We believe that space-time event clouds are promising to conquer other real-time multimedia and computer vision tasks with limited memory and computation power.

References

[1] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. D. Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza, J. Kusnitz, M. Debole, S. Esser, T. Delbruck, M. Flickner, and D. Modha. A low power, fully event-based gesture recognition system. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7388–7397, July 2017.

[2] S. K. Card, G. G. Robertson, and J. D. Mackinlay. The information visualizer, an information workspace. In *Proceedings of the SIGCHI Conference on Human factors in computing systems*, pages 181–186. ACM, 1991.

[3] S. Chen, P. Akselrod, B. Zhao, J. A. P. Carrasco, B. Linares-Barranco, and E. Culurciello. Efficient feedforward categorization of objects and human postures with address-event image sensors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(2):302–314, Feb 2012.

[4] H. Cheng, L. Yang, and Z. Liu. Survey on 3d hand gesture recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(9):1659–1673, Sept 2016.

[5] M. V. den Bergh and L. V. Gool. Combining rgb and tof cameras for real-time 3d hand gesture interaction. In *2011 IEEE Workshop on Applications of Computer Vision (WACV)*, pages 66–72, Jan 2011.

[6] F. Destelle, A. Ahmadi, K. Moran, N. E. O'Connor, N. Zioulis, A. Chatzitofis, D. Zarpalas, P. Daras, L. Unzueta, J. Goenetxea, M. Rodriguez, M. T. Linaza, Y. Tisserand, and N. M. Thalmann. A multi-modal 3d capturing platform for learning and preservation of traditional sports and games. In *Proceedings of the 23rd ACM International Conference on Multimedia*, MM '15, pages 747–748, New York, NY, USA, 2015. ACM.

[7] F. Dominio, M. Donadeo, G. Marin, P. Zanuttigh, and G. M. Cortelazzo. Hand gesture recognition with depth data. In *Proceedings of the 4th ACM/IEEE International Workshop on Analysis and Retrieval of Tracked Events and Motion in Imagery Stream*, ARTEMIS '13, pages 9–16, New York, NY, USA, 2013. ACM.

[8] G. Gallego, H. Rebecq, and D. Scaramuzza. A unifying contrast maximization framework for event cameras, with applications to motion, depth, and optical flow estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[9] R. Gütig and H. Sompolinsky. The tempotron: a neuron that learns spike timingbased decisions. *Nature Neuroscience*, 9:420428, Feb 2006.

[10] N. Kardaris, I. Rodomagoulakis, V. Pitsikalis, A. Arvanitakis, and P. Maragos. A platform for building new human-computer interface systems that support online automatic recognition of audio-gestural commands. In *Proceedings of the 2016 ACM on Multimedia Conference*, MM '16, pages 1169–1173, New York, NY, USA, 2016. ACM.

[11] H. Kim, S. Leutenegger, and A. J. Davison. Real-time 3d reconstruction and 6-dof tracking with an event camera. In B. Leibe, J. Matas, N. Sebe, and M. Welling, editors, *Computer Vision – ECCV 2016*, pages 349–364, Cham, 2016. Springer International Publishing.

[12] J. H. Lee, T. Delbruck, M. Pfeiffer, P. K. J. Park, C.-W. Shin, H. Ryu, and B. C. Kang. Real-time gesture interface based on event-driven processing from stereo silicon retinas. *IEEE Transactions on Neural Networks and Learning Systems*, 25(12):2250–2263, Dec 2014.

[13] H. Liang, J. Yuan, D. Thalmann, and N. M. Thalmann. Ar in hand: Egocentric palm pose tracking and gesture recognition for augmented reality applications. In *Proceedings of the 23rd ACM International Conference on Multimedia*, MM '15, pages 743–744, New York, NY, USA, 2015. ACM.

[14] A. I. Maqueda, A. Loquercio, G. Gallego, N. Garca, and D. Scaramuzza. Event-based vision meets deep learning on steering prediction for self-driving cars. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[15] S. Mohatta, R. Perla, G. Gupta, E. Hassan, and R. Hebbalaguppe. Robust hand gestural interaction for smartphone based ar/vr applications. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 330–335, March 2017.

[16] E. Mueggler, G. Gallego, H. Rebecq, and D. Scaramuzza. Continuous-time visual-inertial trajectory estimation with event cameras. *CoRR*, abs/1702.07389, 2017.

[17] P. O'Connor, D. Neil, S.-C. Liu, T. Delbruck, and M. Pfeiffer. Real-time classification and sensor fusion with a spiking deep belief network. *Front Neurosci.*, 7, Oct 2013.

[18] X. Peng, B. Zhao, R. Yan, H. Tang, and Z. Yi. Bag of events: An efficient probability-based feature extraction method for aer image sensors. *IEEE Transactions on Neural Networks and Learning Systems*, 28(4):791–803, April 2017.

[19] J. A. Pérez-Carrasco, B. Zhao, C. Serrano, B. na Acha, T. Serrano-Gotarredona, S. Chen, and B. Linares-Barranco. Mapping from frame-driven to frame-free event-driven vision systems by low-rate rate coding and coincidence processing–application to feedforward convnets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2706–2719, Nov 2013.

[20] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, July 2017.

[21] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[22] Z. Ren, J. Meng, J. Yuan, and Z. Zhang. Robust hand gesture recognition with kinect sensor. In *Proceedings of the 19th ACM International Conference on Multimedia*, MM '11, pages 759–760, New York, NY, USA, 2011. ACM.

[23] Z. Ren, J. Yuan, and Z. Zhang. Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera. In *Proceedings of the 19th ACM International Conference on Multimedia*, MM '11, pages 1093–1096, New York, NY, USA, 2011. ACM.

[24] A. Sanin, C. Sanderson, M. T. Harandi, and B. C. Lovell. Spatio-temporal covariance descriptors for action and gesture recognition. In *2013 IEEE Workshop on Applications of Computer Vision (WACV)*, pages 103–110, Jan 2013.

[25] P. Wang, Z. Li, Y. Hou, and W. Li. Action recognition based on joint trajectory maps using convolutional neural networks. In *Proceedings of the 2016 ACM on Multimedia Conference*, MM '16, pages 102–106, New York, NY, USA, 2016. ACM.

[26] H. Zhang, V. M. Patel, M. Fathy, and R. Chellappa. Touch gesture-based active user authentication using dictionaries. In *2015 IEEE Winter Conference on Applications of Computer Vision*, pages 207–214, Jan 2015.

[27] X. Zhao, X. Li, C. Pang, X. Zhu, and Q. Z. Sheng. Online human gesture recognition from motion data streams. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 23–32, New York, NY, USA, 2013. ACM.

[28] X. Zhu, W. Liu, X. Jia, and K. Y. K. Wong. A two-stage detector for hand detection in ego-centric videos. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–8, March 2016.