Original!

Real-time Gesture Classification System Based on Dynamic Vision Sensor

English Scientific Paper Writing, Class 1

By

Wang Jian 20023101

College of Computer Science

18063344889

herewj@gmail.com

Changsha, Hunan

May，2021

Real-time Gesture Classification System Based on Dynamic Vision Sensor

Jian Wang
School of Computer
National University of Defense Technology
Changsha, China
wangjian.scrutiny@gmail.com

Xiaofan Chen
School of Computer
National University of Defense Technology
Changsha, China
chenxf_jeff@163.com

Lei Wang
School of Computer
National University of Defense Technology
Changsha, China
leiwang@nudt.edu.cn

***Abstract*—A biologically inspired event camera being able to produce more than 500 pictures per second [1], has been proposed in recent years. These event cameras have achieved profound efficiency in addressing many drawbacks of traditional cameras. In this paper, we explored the feasibility of implementing real-time intensity image reconstruction and classification, by combining traditional neural networks with Celex DVS data. We propose an elementary workflow with a LeNet-based network, for reconstructing DVS bin files into intensity images. Meanwhile the system will display those images with high frame rate, while updating the image labels in real time. We show that CNN applied with transformed DVS data can bring about relatively good real-time classification accuracy of around 94% on average, compatible to 96% in IBM's work in gesture recognition. Our proposed workflow can process and display the event data in real time with delay of less than 30ms per frame.**

***Index Terms*—DVS, LeNet, event**

I. Introduction

Convolutional neural networks (CNNs) are currently the state-of-art techniques for processing images in computer vision. Normally it takes a large dateset to train the networks, and this could be expensive in resources and time with traditional cameras. In recent years, event cameras naturally featured with high temporal resolution and high dynamic range [2] come into rapid development with boarder application range. These biologically inspired cameras only record the brightness change in the scene, if certain threshold is reached [3]. Therefore, much of the redundant background data in frame-based cameras will be avoided with event cameras. The sparse and highly temporal correlated event data make event cameras highly efficient on limited resources platforms.

There have been many studies applying event cameras with CNNs in recognition tasks. In [4], IBM implemented a system to recognize hand gestures in real-time from events streamed by a DVS128 camera. An adapted Point-net is proposed for real-time gesture recognition in [5], and the work [6] carried out a real-time hand gesture interface based on DVS and neuromorphic post-processing. LeNet-5 [7] is one of the most frequently studied neural network, and has been the most referred model for many advanced networks in the computer vision field. More recently, using a LeNet based network, [8] and [9] complete a hand symbol recognition system that can quickly be trained to incrementally learn new symbols recorded with an event-based camera.

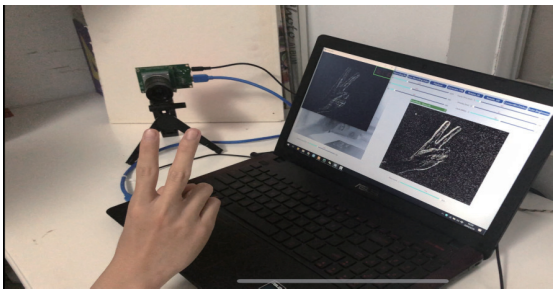

Fig. 1. Real-time image reconstruction and classification

As to our knowledge, there is few related work based on a new type of improved DVS cameras called Celex [1]. Many dvs gesture recognition work tend to focus on gestures of upper limbs or entire human body [4], [10], so low resolution event camera like DVS128 can provide sufficient events for the relatively coarse contour of these gestures. However, the high-resolution and even faster Celex camera [11] can capture more objects' information per frame, which means more features can be exploited for subtle recognition tasks like expression. And given the typicality and adaptability of the LeNet in the field of computer vision, it is worth exploring the network design of LeNet-5 with these event data, providing a reference to more complex network design for event data. We also find it of importance to exploit the real-time application of Celex camera for its incredible temporal resolution and high dynamic range.

In background, we introduce the Celex camera, its data format and reconstruction algorithms. In Related work, we analyze the related work with RGB and Event Cameras. In main work, we illustrate main work of our paper: a celex-based hand gesture dataset, proposed network as well as the real-time reconstructed flow in Fig 2. In experiment, we mainly set up four experiments: the input data exploration with three different data preprocessing methods, the exploration of reconstruction and key frame parameters, the network structure exploration, offline and real-time classification testing.

The main contribution of this paper includes:

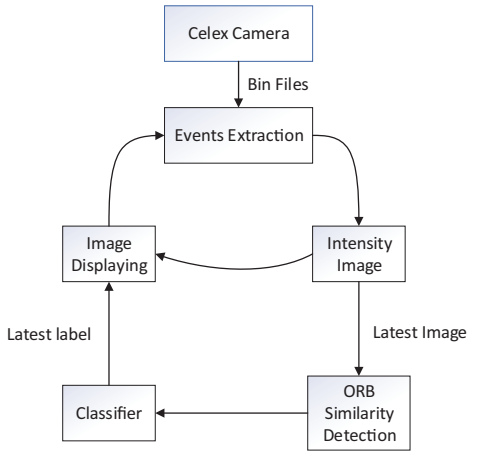

Fig. 2. The workflow of real-time reconstruction and classification.

TABLE I
CELEX EVENT DATA FORMAT [15]

Event Type	Size	Description	Comments
Row Event	4 bytes (32 bits)	4 bytes Byte 0: {1'b1, Y[6:0]} Byte 1: {1'b1, Y[9,7], T[3:0]} Byte 2: {1'b1, T[10:4]} Byte 3: {2'b10,T[16:11]}	Row Events carry row address(Y) and the activation time(T) for the pixels on that row
Column Event	4 bytes (32 bits)	4 bytes Byte 0: {1'b0, X[6:0]} Byte 1: {1'b0, C[0], X[8,7], A[3:0]} Byte 2: {3'b000, A[8:4]} Byte 3: {8'b00000000}	Column Events carry the column address(X) and follow right after a row event. They carry the sample value of absolute brightness when the pixel event is triggered(A). The timing information can be retrieved from the preceding Row Event.
Special Event	4 bytes (32 bits)	4 bytes Byte 0: {8'b11111111} Byte 1: {8'b11111111} Byte 2: {8'b11111111} Byte 3: {8'b11111111}	Special Events represent the end of a time block. After this event, timer count resets to 0. The duration of the time block is configurable.

1. A real-time reconstruction and classification system.
2. A four-class hand gesture DVS dataset.
3. A real-time keyframe detection algorithm based on ORB and Canny algorithm.

II. BACKGROUND

A. Celex Camera

Different from the early work of DVS, such as DVS128 [10], DAVIS [4] and ATIS [1], Celex pixel directly take the voltage on the logarithmic photo detector as its intensity information. So there is no need to match DVS pixel and intensity information, in both spatial and temporal domain [12]. Therefore, Celex pixels outperform the prior ones in both response speed and image content. In our case, the camera resolution should be within proper range for both image integrity and real-time needs. After comparing the specification [1], [13] of Celex Cameras, we adopted Celex VI (768 x 640) as our DVS camera.

The resolution of event cameras adopted by other classification work is relatively small, such as 128 x 128 for DVS128, and 346 x 260 for DAVIS346 [14], so when it comes to more detailed hand gesture, Celex cameras is able to provide better resolution.

B. Celex Data Format

The early form of event information is presented as (X, Y, P, T), where p is polarity or the event type of certain pixel. But Celex pixel records its sample value of absolute brightness. And the event is formulated as (X, Y, A ,T), where A is the pixel logarithmic gray level value [15]. The (X, Y, T) represents the spatio-temporal information of the events, X and Y for pixel locations/coordinates, and T for activation time.

For Celex pixels, the (X, Y, A ,T) is obtained by shifting and combining the information contained in three different kinds of raw events [15], namely the column, row and special events, as shown in the Table I. From the Row or Column events we can derive Y and T information or X and A respectively. Special events indicate the end of a time block for camera counter to update. All events mentioned are 4 bytes in length. Based on the byte information of raw events, ther are two algorithms developed to form intensity images and they are also mentioned in the Celex SDK reference [15]: one is event-accumulated [16], and the other is based on time slices within a sampling time window.

The event-accumulated algorithm accumulates a certain amount of real events to form one intensity image, while the slice-based algorithm assigns graylevel of 1 to 255 to active pixel elements, in consecutive 255 time window slices. And in order to lessen the reconstruction process in our workflow, we adopt the first method to obtain binary images, because only (X, Y) information should be calculated and binary images would be enough for contour feature extraction.

III. RELATED WORK

A. Gesture Recognition based on traditional Camera

There have been various approaches to handle traditional gesture recognition [17], ranging from mathematical models based on hidden Markov chains [18] to methods based on Machine learning. [17] proposed a method based on Hidden Markov Models (HMMs) for dynamic gesture trajectory modeling and recognition. A static hand gesture recognition system for Sign Language using Edge Oriented Histogram (EOH) features and multiclass SVM is proposed by [19]. Convolutional neural networks (CNNs) have been used in gesture recognition and have achieved good results [20].

However, these approaches require the signer to wear color tape or restricted background color. It reveals that these systems can not cope with complex and changing situations [19].

B. Gesture Recognition based on DVS Camera

However, frame-based CNNs can be quite power-consuming due to the redundant data especially with high-speed cameras,

which can be naturally compensated by DVS cameras. Therefore, many reconstruction and feature extraction methods have been explored to match DVS data and the traditional CNNs. Simple intensity images can be formulated by accumulating the events along temporal axis, such as [21], [16] and [22]. And work like [23] and [15] saved the latest temporal or illumination information into pixel elements. These algorithms can provide very clear contour information using enough events. Optical flow methods [24] [25] [26] can help derive more features for CNN classification with low computation cost. Besides, [27] proposed delivery point based method to extract the hand region. And [28] proposed a four DOF method for fast and efficient motion estimation, which makes good use of the edge pattern in DVS data. [4] adopted a hardware temporal filter cascade and neurons with stochastic delay to generate spatio-temporal input for CNN.

[6] designed a real-time hand gesture GUI control system, in which SNN and various detection algorithms were applied to achieve robust recognition performance, with average classification delay around 120ms and competent accuracy at 96.49%. More recently, [8] combined the iCaRL incremental learning algorithm with CNN to boost the retraining efficiency. [11] implemented a CNN with 5 convolutional layers within jAER framework, achieving 99.3% accuracy with 70ms for one recognition, after training 40 epochs at 1.3M data. All the deep neural network approaches work well after reconstructing or deriving proper features suitable for traditional network. Besides, there are many other gesture related work based on DVS. [29] firstly proposed a n-HAR Model Architecture for human activity recognition analysis. [10] combined fusing motion and motion boundary histogram to recognize human activity.

IV. Main Work

A. "Rock-Scissor-Paper" Dataset

In order to keep our recorded data consistent to the real situation, we set up three illumination conditions namely the dark, lamp and normal lighting (adjusted with a desk lamp), with the camera event threshold set at 90 [15], and four different background settings. Four people were involved to make the dataset, each with three gestures and another background class recorded. Since the hand region recognition is our main focus, we try to involve as many types of position, scale, posture and direction as possible. The raw data was acquired by recording both hands with different sides facing the camera, while rotating and shifting the hand in the camera field. Given that the amount of raw event data is limited, we basically reconstruct the data with 8k, 16k and 24k per frame based on the event-accumulated method [16]. The total number of each gesture is around 50k, and we separate it into training and testing set randomly by the data ratio of training to testing at 3.

B. Proposed Network

LeNet-5 [7] is one of the most classical CNN structure, which contains a total of 7 layers, We first train a CNN

(a) 8k (b) 16k (c) 24k

Fig. 3. Reconstructing the data from Celex Camera with 8k, 16k and 24k per frame

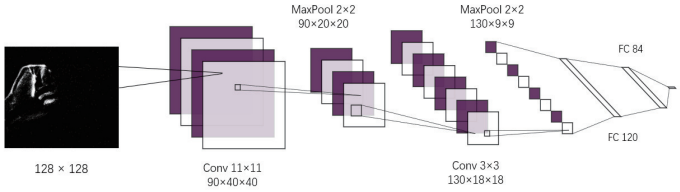

Fig. 4. The architecture of the adapted LeNet-5

on the preprocessed DVS data using the adapted LeNet-5 framework(Fig 4). The input image size is 128×128, which is 16 times the size of handwritten digital pictures. There are 4 types of gesture in this paper. The kernel size and stride of the first convolutional layer are respectively 11×11 and 3 × 3. And for the second convolutional layer, these two parameters are downscaled to 3×3 and 1 × 1 accordingly. The filter amount of the two convolutional layers are 90 and 130 respectively. Each convolutional layer is followed by a ReLU layer and a max pooling layer. The kernel size of each pooling layer is 2 × 2, with 2 × 2 stride.

The adapted network based on the LeNet-5 network does not meet our accuracy and real-time expectation (the accuracy is higher than 93%). We further improved this network as shown in Fig 5. There are 6 convolutional layers, 6 pooling layers and 1 fully-connected layer. The convolutional kernel size is 3 × 3 with stride 1 × 1. And the kernel size of all the pooling layers is 2 × 2. The training and test accuracy can reach 99%. "Valid-Padding" is performed during the convolution processed of CNN in our implementation. With the increase of convolutional-pooling layer combination, the features of original image input can actually be extracted in a fine-grained way. The original 11x11 convolutional operation can be break down into with several smaller convolution layers. Therefore, more compact features can be learned in this network compared to LeNet.

C. ORB-based keyframe detection

We mainly adopt ORB algorithm [30] to help us detect the key frame among the newly reconstructed binary images. Firstly, we use FAST algorithm [31] to extract feature points of the loaded frame, and further apply BRIEF algorithm [32] to obtain the feature description of these feature points. Then, we compare two frames detected continuously and consider the current frame as abnormal one if certain threshold (Th1) is met. Current frame will be set as reference frame until the

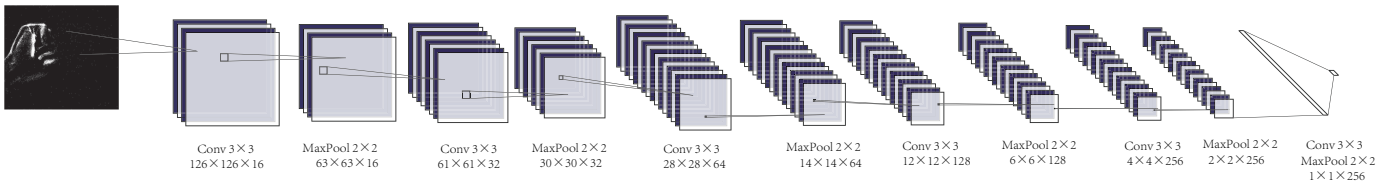

Fig. 5. The architecture of our present network which has 5 convolutional layers, 5 pooling layers and 1 fully-connected layer.

number of abnormal frames reaches another threshold (Th2), as illustrated in the following pseudo-code.

Algorithm 1 ORB-based key frame detection algorithm

```
orb // Oriented FAST and Rotated BRIEF package
kf_threshold, rec_threshold // threshold for determing key frame and triggering classification
INPUT Image_new, kf_threshold, rec_threshold
OUTPUT Label
while New Binary Images Generated do
  if Process Initilization Done then
    descriptor_cur = orb(Image_new)
    if similarity(descriptor_cur, descriptor_pre) > kf_threshold then
      Image_pre = Image_new
    else
      kf_num = kf_num + 1
      if kf_num >= rec_threshold then
        Label = Classifier(Image_new)
        kf_num = 0
      end if
    end if
  else
    Image_pre = Image_new
  end if
end while
```

D. Real-time reconstruction flow

Fig 4 depicts that when showing a certain gesture in front of a Celex camera, the reconstruction system is able to recognize it by updating label at the upper right corner of reconstructed image. Basically there are three processes running simultaneously as partly shown in Fig 6. And here are the latest and complete real-time workflow: 1) firstly monitor new .bin files from Celex cameras; 2) extract events from bin files, before reconstructing and displaying binary images; 3) keyframe detector selects keyframe before triggering the classifier when certain threshold is met; 4) keyboard detects whether there is a "stop" instruction input.

V. Experiment

The proposed network is implemented on the Tensorflow platform. Our experimental code is running in Inter(R) Core(TM) i7-9759H CPU without using GPU.

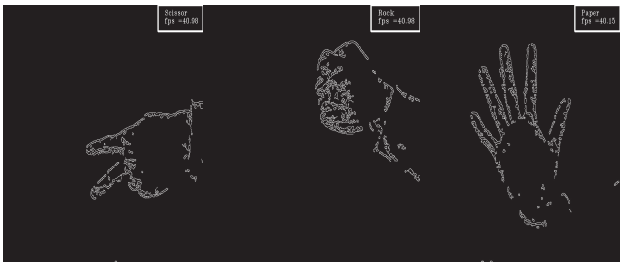

Fig. 6. The intensity images are displayed once a bin file detected in Celex Camera directory

TABLE II
The Impact of Noise on Classification Accuracy

dataset	training set	testing set
HeatMap	99.33%	97.14%
TimeFilter	99.20%	99.26%
GuasCanny	99.44%	99.68%

A. Dataset Exploration

We set up three datasets to measure the impact of noise on classification accuracy. The three datasets share the same raw bin files, but only consist of images reconstructed at 20k events per frame for simplicity. By conducting different denoising method for image reconstruction, we get three datasets. Firstly we applied time filter [33] to obtain the first dataset with most noise filtered (TimeFilter). Then we use the 9x9 Guassian kernal and Canny algorithm [34] to remove the noise in the dataset (GuasCanny); and finally a heatmap-based filtering method was used to get the third dataset (HeatMap) which comparably contains more noise points than the previous ones.

Three different datasets are evalutaed on our proposed network, as illustrated in Table II. The results show that the dataset preprocessed by canny results in better recognition accuracy. This also demonstrates the importance of the denoised pictures for gesture recognition. We also used the noise detection algorithm in [35] to verify the noise amplitude. Since the images is binary, so we set the low threshold of 5. The average results of 5 randomly sampled images are as follows:

From the Table III we can see that TF method has least noise points while the processing delay is the largest. So taking our real-time requirement of no less than 30 fps into consideration, it is proper to use dataset of GC type. It is interesting to see that the GC data with more noise than TF while resulting in better accuracy. We assume the main reason could be that GC contains more refined contour features than the other two.

TABLE III
NOISE ALAYSIS ON THREE DIFFERENT DATASETS

dataset	HeatMap	TimeFilter	GuasCanny
Noise	8262	4195	6150
Signal	42313	20301	15195
Noise portion	1.68%	0.85%	1.25%
Time per frame	71.57ms	93.14 ms	57.68ms

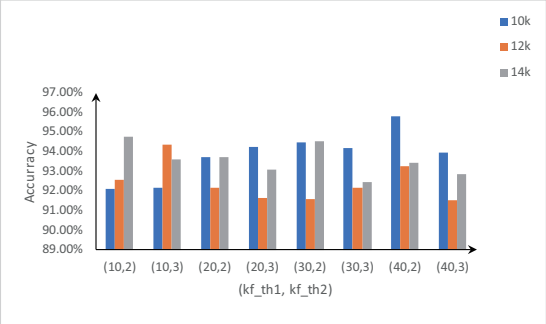

Fig. 7. The average Accuracy results

B. Parameter Exploration

1) Reconstruction and Keyframe Parameters: Since the requirement of real-time classification, it would be necessary to explore the proper threshold and reconstruction parameters. We split up four raw event bin files of different types with same recording duration (1 minute), and then randomly selected 4 to 6 from total 16 as sequential testing inputs. Meanwhile the corresponding label files were concatenated into one as reference label for real-time accuracy measurement.

We firstly observe the reconstruction integrity by deriving the percentage of incomplete images that are hard to distinguish. We find that the percentage is nearly 10% when using less than 7k per frame. And when it is above 14k per frame, the average delay is below 30fps. Therefore, we basically conducted the testing within the range of 8k to 14k. And by analyzing the number of similar feature description between two frames in a range of 8k to 2w events per frame, we find similarity number is almost below 15 between two images from different classes. But around 25 for the same type. When Th2 is more than 3, very little images will be

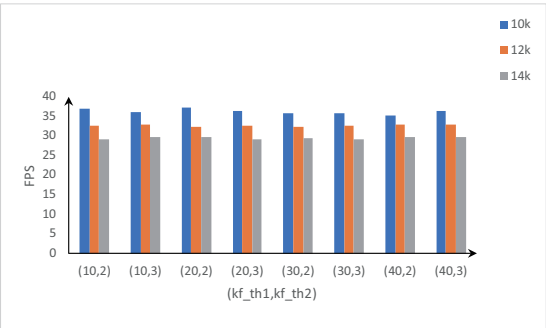

Fig. 8. The average Fps results

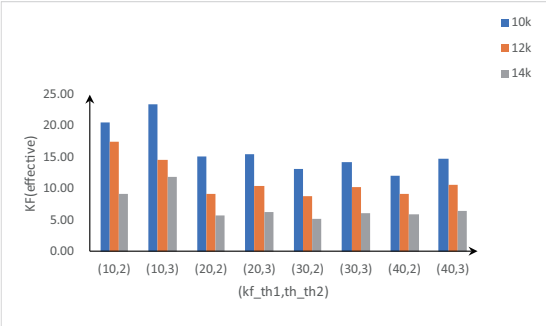

Fig. 9. The Effective KF results

detected as key frame(less than 3%). Therefore, considering both accuracy and real-time performance, two threshold values were set to be 10/20/30/40 and 2/3 respectively.

And from the above results, we can find that FPS is much more sensitive to the reconstruction parameter, while accuracy changes are apparently without clear discrimination for these three parameters. This is partly due to the changing data content along with different event amount per frame. And different keyframe setting results in quite different detected images for classification.

Considering the number of keyframes are influenced by the above three parameters simultaneously, we defined the following equation to describe the effectiveness of classification.

$$\begin{aligned} KF_{effective} = & \frac{ACC_{avg}}{P(kf)} \\ & + ACC_{judge\_const} * \ (ACC_{avg} - ACC_{th}) \\ & + Fps_{judge\_const} * \ (Fps_{avg} - Fps_{th}) \end{aligned} \tag{1}$$

where, the avg for average result of the measured metrics, P(kf) for the percentage of key frames among the reconstructed images, and judge_const denoting the constant for evaluating absolute value of accuracy and fps according to preset thresholds.

The second and third terms are added to adjust the effectiveness of keyframe with global preset performance restrictions. And here we expect the Accuracy and Fps to be no less than 93% (average accuracy of LeNet) and 30 respectively. According to the Fig 9, we decided the proper parameter setting to be 10k and (10,3).

2) Network Setting: The gesture recognition needs to be both low-delay and accurate. So we firstly adjusted the number of LeNet-5 layers, and found the accuracy dropped dramatically to 70% - 80%. Then by adding one more layers to the network, and delay increased by 20%. So we decided to use the 8-layer stricture as the original network. And there could probably be problems while using original LeNet-5 in our gesture recognition, since 32 x 32 input size could be much less efficient in feature extraction for 768 x 640 images. Therefore, we enlarged the input size to 128 x 128. The kernel size is related to whether valid features can be extracted. And

the original convolution kernel with a size of 5 may be less efficient in extracting features for 128 x 128 inputs.

Therefore, we resized the convolution kernel. In this paper, the first convolution kernel with a size of 11 and the second with a size of 3 are used to improve the feature extraction. But for delay concerns, we meanwhile increased the stride for enlarged kernels. We also increased the number of convolution kernels, as more different features can be extracted. Considering the size and complexity of the input images, it is necessary to increase the number of convolution kernels to enhance the network's adaptability to distinguish between different models. We find that convolutional operation is the most time-consuming step. Therefore, two fully-connected layers were set up in our case. The first fully-connected layer has 120 units and the second has 84 units.

C. Network

1) Network Training: The network is trained from scratch with preprocessed Celex DVS gesture dataset. For the training, learning rate is set to 0.001, and 20,000 iterations are enough for the training on our dataset. Meanwhile, for all datasets, the batch size is 32, and the crop size for each image is 128×128 with which the proposed network performs best. The augmented data by accumulating binary images with different amount of events also contribute to the accuracy of training.

Two different convolutional neural networks (Network 1 and 2) are utilized to evalutate the proposed CNN when training on our preprocessed dataset.

Network 1: The original Lenet-5 network. The size of input image is 32 × 32, and the network structure is the same as the original LeNet-5.

Network 2: The kernel sizes of first and second convolutional layer in Network 1 are resized to 11 × 11 and 3 x 3 respectively (as displayed in Fig 4). In this case, the size of input image is 128 × 128, so the nerual network can get more feature information.

Network 3: The proposed network(as displayed in Fig 5) which has 6 convolutional layers, 6 pooling layers and 1 fully-connected layers. The kernel size of convolution layer is 3 × 3 with 1 × 1 stride. The kernel size of pooling layer is 2 × 2 with 2 × 2 stride.

2) Offline Accuracy Testing: The proposed dataset for training has three different settings for event amount per frame (8k/16k/24k) and is applied with Canny algorithm. Three different networks are trained on this dataset. Table V reveals our proposed network outperforms the original LeNet significantly. And our method obtains the accuracy of 99.13% on training set and 98.67% on testing set.

3) Real-time Accuracy Testing: We further sampled another 9 groups of combined bin files, and obtained the real-time accuracy results for fixed number of events per frame and two keyframe thresholds. As showed in Fig 10, the average accuracy remains over 92% and the FPS is sufficient to display the video smoothly. According to our observation, the accuracy loss mainly comes from the input initialization and gesture switch intervals. The overall displaying is satisfying though,

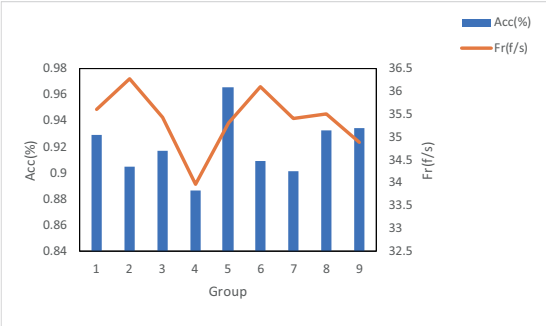

Fig. 10. The real-time accuracy results with fixed reconstruction and keyframe parameters

as our observers all reported that the label remains very stable and correct for one constant gesture, and the label transition interval between two different gestures is also within the neighborhood of human response time.

VI. Discussion and Future Work

In this paper, we investigated proper way of preprocessing celex event data. We also preliminarily explored the proper threshold setting for keyframe detection, as well as the amount of events per frame, by taking into account the impact of these parameters on classification accuracy and delay.

According to the recognition results of three models, by setting the optimal parameters explored with basic LeNet, our proposed network can achieve an average accuracy of 98.67% and 92.03% for offline and real-time testing. The reasons for the accuracy gap could be following: 1) The proposed method does not separate region of hands from the whole scene, so gestures appearing at edges of the field are difficult to recognize; and 2) there are blank frames (with little events within) during reconstruction and it is easy to trigger label updated wrongly; 3) there is inevitable and uncertain label update delay betdween gesture shifting.

From the real-time testing results, we find the main bottleneck for our workflow performance is attributed by classification (0.16s/frame). By deploying keyframe with classifier in the same process, this bottleneck is actually alleviated much with minus accuracy loss. From the keyframe percentage, we can tell that key frame detection decreases the classification by nearly 50% (0.053 : 0.109) with parameters set at 10k and (10,3). Although we adopted a multi-processing approach, the run time performance of our workflow is well satisfied (average 35fps for displaying images formed by 10k events).

VII. Conclusion

In this paper, we present a real-time reconstruction and classification system. The evaluation results demonstrate that our proposed network is more suitable than the original LeNet-5 network for DVS gesture recognition. The output of our real-time workflow is very robust to complex backgrounds behind gestures. However, there are still many options for improvement, such as better keyframe detection algorithms,

TABLE IV
CLASSIFIER PARAMETERS EXPLORATION. ACCURACY IS REPORTED IN PERCENTAGE.

Experiment	Input W × H	kernel size(1st layer)	kernel size(2nd layer)	test acc	train acc
E1	128×128	11×11(stride=3)	3×3(stride=1)	94.87%	97.90%
E2	128×128	5×5(stride=1)	5×5(stride=1)	93.16%	95.78%
E3	32×32	5×5(stride=1)	5×5(stride=1)	85.49%	80.31%
E4	227×227	11×11(stride=3)	3×3(stride=1)	93.21%	99.40%
E5	227×227	5×5(stride=1)	5×5(stride=1)	92.21%	94.21%

TABLE V
THE OFFLINE ACCURACY FOR THREE NETWORKS WITH SAME GC INPUTS

Network	Training set	Testing set
original (Lenet-5 network)	80.59%	82.45%
improved (Adapted Lenet)	94.17%	96.27%
final (6conv-6pooling-1fc)	99.13%	98.67%

better real-time data feature extraction for CNNs, and better reconstruction algorithm for event images of high quality.

The accumulation of events at fixed amount may cause inconsistency between the reconstructed video stream and those by traditional cameras, since events density can be various due to different movement intensity and lighting condition. Therefore, in order to obtain reconstructed real-time video stream that is close to frame-based scenario, it would probably be necessary to combine spatio-temperal information and the amount of event per frame in a dynamic way.

REFERENCES

[1] S. Chen and M. Guo, "Live demonstration: Celex-v: a 1m pixel multi-mode event-based sensor," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[2] G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis *et al.*, "Event-based vision: A survey," *arXiv preprint arXiv:1904.08405*, 2019.

[3] J. Huang, M. Guo, and S. Chen, "A dynamic vision sensor with direct logarithmic output and full-frame picture-on-demand," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2017, pp. 1–4.

[4] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza *et al.*, "A low power, fully event-based gesture recognition system," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7243–7252.

[5] Q. Wang, Y. Zhang, J. Yuan, and Y. Lu, "Space-time event clouds for gesture recognition: From rgb cameras to event cameras," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 1826–1835.

[6] J. H. Lee, T. Delbruck, M. Pfeiffer, P. K. 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*, vol. 25, no. 12, pp. 2250–2263, 2014.

[7] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[8] I. A. Lungu, S.-C. Liu, and T. Delbruck, "Fast event-driven incremental learning of hand symbols," in *2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*. IEEE, 2019, pp. 25–28.

[9] I. A. Lungu, S.-C. Liu, and Delbruck, "Incremental learning of hand symbols using event-based cameras," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 4, pp. 690–696, 2019.

[10] S. A. Baby, B. Vinod, C. Chinni, and K. Mitra, "Dynamic vision sensors for human activity recognition," in *2017 4th IAPR Asian Conference on Pattern Recognition (ACPR)*. IEEE, 2017, pp. 316–321.

[11] I.-A. Lungu, F. Corradi, and T. Delbrück, "Live demonstration: Convolutional neural network driven by dynamic vision sensor playing roshambo," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2017, pp. 1–1.

[12] Y. Zhang, "Celexmatlabtoolbox," [EB/OL], https://github.com/yucicheung/CelexMatlabToolbox Accessed April 15, 2020.

[13] M. Guo, J. Huang, and S. Chen, "Live demonstration: A 768× 640 pixels 200meps dynamic vision sensor," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2017, pp. 1–1.

[14] www.inivation.com, "Neuromorphic vision systems," [EB/OL], http://inivation.com Accessed April 15, 2020.

[15] CelePixel, "Celepixel technology," [EB/OL], https://github.com/CelePixel/CeleX4-OpalKelly/tree/master/Documentation Accessed April 15, 2020.

[16] D. P. Moeys, F. Corradi, E. Kerr, P. Vance, G. Das, D. Neil, D. Kerr, and T. Delbrück, "Steering a predator robot using a mixed frame/event-driven convolutional neural network," in *2016 Second International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP)*. IEEE, 2016, pp. 1–8.

[17] V. I. Pavlovic, R. Sharma, and T. S. Huang, "Visual interpretation of hand gestures for human-computer interaction: A review," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 19, no. 7, pp. 677–695, 1997.

[18] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[19] S. Nagarajan and T. Subashini, "Static hand gesture recognition for sign language alphabets using edge oriented histogram and multi class svm," *International Journal of Computer Applications*, vol. 82, no. 4, 2013.

[20] W. Cheng, Y. Sun, G. Li, G. Jiang, and H. Liu, "Jointly network: a network based on cnn and rbm for gesture recognition," *Neural Computing and Applications*, vol. 31, no. 1, pp. 309–323, 2019.

[21] C. Ye, "Learning of dense optical flow, motion and depth, from sparse event cameras," Ph.D. dissertation, 2019.

[22] A. I. Maqueda, A. Loquercio, G. Gallego, N. García, and D. Scaramuzza, "Event-based vision meets deep learning on steering prediction for self-driving cars," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5419–5427.

[23] A. Zihao Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "Ev-flownet: Self-supervised optical flow estimation for event-based cameras," *arXiv preprint arXiv:1802.06898*, 2018.

[24] P. Bardow, A. J. Davison, and S. Leutenegger, "Simultaneous optical flow and intensity estimation from an event camera," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 884–892.

[25] R. Benosman, S.-H. Ieng, C. Clercq, C. Bartolozzi, and M. Srinivasan, "Asynchronous frameless event-based optical flow," *Neural Networks*, vol. 27, pp. 32–37, 2012.

[26] T. Brosch, S. Tschechne, and H. Neumann, "On event-based optical flow detection," *Frontiers in neuroscience*, vol. 9, p. 137, 2015.

[27] E. Y. Ahn, J. H. Lee, T. Mullen, and J. Yen, "Dynamic vision sensor camera based bare hand gesture recognition," in *2011 IEEE Symposium On Computational Intelligence For Multimedia, Signal And Vision Processing*. IEEE, 2011, pp. 52–59.

[28] J. H. Lee, K. Lee, H. Ryu, P. K. Park, C.-W. Shin, J. Woo, and J.-S. Kim, "Real-time motion estimation based on event-based vision sensor," in *2014 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2014, pp. 204–208.

[29] B. Pradhan, Y. Bethi, S. Narayanan, A. Chakraborty, and C. Thakur, "N-har: A neuromorphic event-based human activity recognition system using memory surfaces," 05 2019, pp. 1–5.
[30] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.
[31] D. G. Viswanathan, "Features from accelerated segment test (fast)," *Homepages. Inf. Ed. Ac. Uk*, 2009.
[32] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *European conference on computer vision*. Springer, 2010, pp. 778–792.
[33] S. Guo, Z. Kang, L. Wang, L. Zhang, X. Chen, S. Li, and W. Xu, "A noise filter for dynamic vision sensors based on global space and time information," *arXiv preprint arXiv:2004.04079*, 2020.
[34] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.
[35] P. K. Ms. Swapna M. Patil, "Advance method of detection and removal of noise from digital image," *International Journal of Advanced Research in Computer Engineering & Technology(IJARCET)*, vol. 2, pp. 3172–3176, 2013.