A Voxel Graph CNN for Object Classification with Event Cameras - Supplementary Material -

Yongjian Deng[1] Hao Chen[2*] Hai Liu[3] Youfu Li[4*]

[1]College of Computer Science, Beijing University of Technology

[2]School of Computer Science and Engineering, Southeast University

[3]The National Engineering Research Center for E-Learning, Central China Normal University

[4]Department of Mechanical Engineering, City University of Hong Kong

This supplementary material includes connections between our proposed method with voxel-based approaches designed for traditional 3D vision in Sec. 1. Also, the detailed working principle of event cameras are provided in Sec. 2. In Sec. 3, we discuss the effects of different voxel sizes on the performance of our proposed model. In addition, we present comparisons of different vertex selection strategies in Sec. 4. Next, we include the experiment in Sec. 5 to evaluate the robustness of our model to the input vertex density. Finally, we conduct the experiment on the action recognition task in Sec. 6 to further validate the effectiveness of our method $w.r.t$ motion cues encoding.

1. Supplementary Related Work

Voxel-based approaches for 3D vision also hold connections with our method. Here, we briefly introduce several representative studies among previous voxel-based approaches for 3D vision and distinguish major differences between ours and their models. Inspired by the success of deep models on 2D CNNs, this branch of methods first formalizes point clouds into 3D voxels and applies 3D convolutions to encode features [4, 7, 9, 15]. Recently, PointPillars [8] and HVNet [14] alleviate the inefficient computational issue of volumetric convolution by representing point clouds as pseudo images which are compatible with 2D CNNs. These approaches incrementally refine the voxel-based methods in the field of 3D vision. However, the topic of adopting voxel-wise representation for event data remains unstudied. To exploit event data's sparsity and non-redundancy, we only select representative voxels for further processing instead of taking the whole voxels as input (dense grid-like representations). This sparse representation enables us to maintain the informative cues provided by voxel-wise input and realize low model and computational complexity (*EV-VGCNN*). Furthermore, the novel vertex selection strategy and the feature calculation approach are customized according to the natural properties of event data. Lastly, the newly introduced graph learning modules are able to encode spatial-temporal cues with flexible receptive fields.

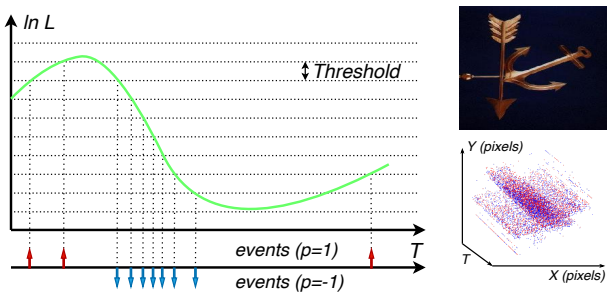

Figure 1. **Left:** A sketch of the working principle of event cameras. Events are produced asynchronously according to the lightness ($\ln L$) changes. Red and blue arrows represent positive and negative events, respectively. **Right:** The RGB image captured from the traditional RGB camera (top) and event signals in the original format produced from an event camera (bottom).

2. Event signals

As visualized in Fig. 1, event cameras produce events asynchronously when they detect changes in the log brightness ($\Delta \ln L(x, y, t)$) that exceed the contrast threshold C [5] as Eq. (1) described.

$$|\Delta \ln L| = |\ln L\,(x, y, t) - \ln L\,(x, y, t - \Delta t)| > C, \quad (1)$$

where Δt is the time between the new event and the last event generated at the same location. Each event $e_i = (x_i, y_i, t_i, p_i)$ is triggered at the pixel location of (x_i, y_i) at time t_i with polarity p_i $\left(p \in \{-1, 1\}\right)$. The polarity of an event shows the sign of brightness changes. Precisely, positive events ($p = +1$) represent the lightness increasing ($\Delta \ln L > C$) and negative events ($p = -1$) represent the lightness decreasing ($\Delta \ln L < -C$).

*: Corresponding author

3. Effects of Various Voxel Sizes

Voxel size	N-Cal	N-C	CIF10
3x3	0.712	0.942	0.623
5x5	0.748	0.953	0.651
7x7	0.751	0.937	0.670
9x9	0.736	0.933	0.648

Table 1. Comparisons of different voxel size choices.

The physical definition of voxel size (v_h, v_w, v_a) in this paper can be categorized into two folds. The first two dimensions (v_a, v_w) correlates to the spatial resolution and the last dimension (v_a) correlates to the temporal resolution. In the following, we will detail the impact of different voxel size choices on the performance of our model. Initially, we conduct a comparison experiment on different choices of the first two dimensions of voxel size and list results in Table 1. Taking the dataset N-Cal as an example, when the voxel size increase from 3×3 to 5×5, the model performance increase dramatically. This is due to the larger voxel size can encode local appearance (texture or contours) better. Intuitively, if the voxel size is too small, then the feature of each vertex will be similar to a point and thus cannot achieve the target of encoding local coherent 2D semantics. Nevertheless, when the voxel size continuously increases to 9×9, the performance does not gain a large improvement and even drops significantly when the voxel size equals 9×9. We attribute this to the limitation brought from model capacity. Since the parameters in our model are shared for each input vertex, if the features contained in each vertex are too complex (depending on the size of the voxel and the complexity of objects in samples), then such a lightweight network will have difficulty in extracting the distinguishing characteristics. From the table, we can find that though the exact value of voxel sizes on accuracy improvement is different due to the variance across different datasets in recording approaches and object complexity, the statistical trend is invariant.

As for the last dimension of voxel size, we set its value manually depending on the discrepancy between various datasets in duration length and the complexity of motion conditions. For example, we set v_a as 3 for short-duration datasets ($\leq 300ms$) such as N-Cal, N-C, and ASL. And set v_a as 1 for datasets with long duration ($\geq 1000ms$) such as CIF10. Notice that even the duration length of N-M is less than $300ms$, we still set v_a as 1 to give its graph-based representation more capable of temporal cues embedding since we observe that each sample in N-M possesses high-speed motions and trajectories with large distance shift.

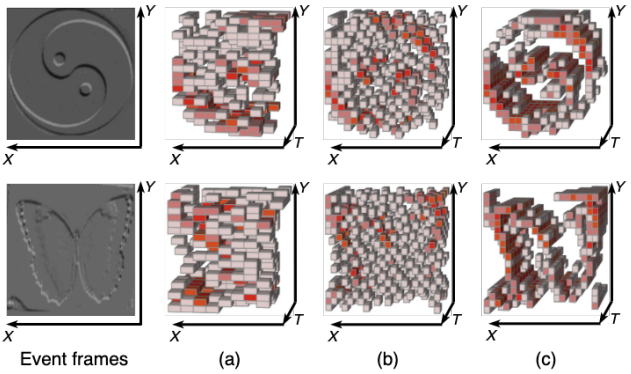

Figure 2. Visualization comparisons of different vertex selection strategies. (a), (b) and (c) represent visualizations of obtained vertex distributions by random selection, furthest selection, and proposed selection strategies. Different colors represent varying event point densities among voxels.

4. Comparisons on Various Vertex Selection Strategies

Selection method	N-Cal	N-C	CIF10
Random	0.736	0.935	0.652
Furthest	0.734	0.941	0.660
Ours	**0.748**	**0.953**	**0.670**

Table 2. Comparisons of different selection strategies.

This section presents comparisons of different vertex selection strategies. Besides our adopted vertex selection approach, we introduce other two strategies that are commonly used in the 3D point cloud field [10, 11], *i.e.*, random vertex selection and furthest vertex selection. The comparison results in Table 2 show that our adopted selection method achieves leading performance on three representative datasets. We attribute this improvement to the function of our selection approach in filtering noise vertex. To support our statement intuitively, we present visualizations of selected vertices (voxels) obtained by three different strategies. From Fig. 2, we can find that our adopted selection method is able to filter out the irrelevant vertices, and thereby can encode spatio-temporal semantics of event data effectively. On the contrary, the other two selection methods, though they can present the structure of objects to a certain extent, many noise vertices have also been kept. We believe that these remaining noise vertices will inevitably interfere with judgments of the model, resulting in the low performance of the model.

5. Robustness to Input Vertex Density

Practical applications in real life call for our graph-based classification network to be flexible to different input sizes.

It means that one can input our *EV-VGCNN* with an arbitrary number of vertices. This characteristic is significantly helpful to event-based models since real-world scenarios are naturally with different spatial scales or motion trajectories, which entail graphs with distinct numbers of vertices to embed their semantic features correspondingly. Therefore, we validate the robustness of our model (trained on graphs with 2048 vertices) to the input vertex density on the N-Cal dataset. Fig. 3 shows that although the classification accuracy drops dramatically when the number of vertices is fewer than 1536, our approach is capable of keeping a stable performance (performance drops within 1%) in a large range ([1536, 2816]) of input vertices, suggesting its robustness to the input vertex density and potential on real-world applications.

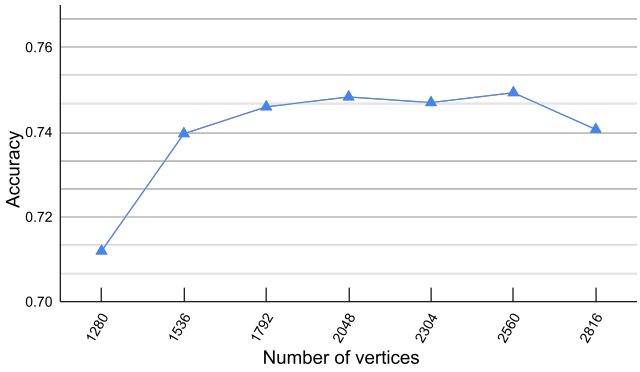

Figure 3. Results of the proposed model tested with different number of input vertices in the graph.

6. Action Recgnition

Motion encoding is helpful for event-based object classification since different objects and scenes hold various motion trajectories. Different from event-based object classification, event streams generated by human action normally convey more temporal (motion) patterns, meaning that evaluation on action recognition tasks can better illustrate ours' advantage in motion embedding. Thus, we follow the training strategy in [2] to conduct experiments on DVS128 Gesture dataset [1], which is recorded in the real world environment and contains eleven different classes of gestures. We set the voxel size as $(v_h, v_w, v_a) = (5, 5, 1)$ and vertex number N_p as 512. Each sample used for training and testing is with the duration of 0.25 seconds. The results in Table 3 show that our approach can achieve comparable performance $w.r.t$ SOTA method RGCNN+Res.3D with 13 times fewer parameters and 19 times fewer complexity, indicating the high efficiency of our model in motion cues encoding.

Method	**Acc. (0.25s)**	**GFLOPs**	**#Params**
EST [6]	0.941	4.28	21.38 M
MVF-Net [3]	0.950	5.62	33.62 M
EventNet [12]	0.885	0.91	2.81 M
PointNet++ [13]	0.940	4.03	1.74 M
RG-CNN (Res.3D) [2]	**0.961**	**13.72**	**12.43 M**
Ours (SFRL)	0.943	0.45	0.76 M
Ours (MFRL)	**0.956**	**0.46**	**0.84 M**

Table 3. Comparisons of models in terms of accuracy and complexity on action recognition dateset.

References

[1] Arnon Amir, Brian Taba, David Berg, Timothy Melano, Jeffrey McKinstry, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, et al. A low power, fully event-based gesture recognition system. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 7243–7252, 2017. 3

[2] Y. Bi, A. Chadha, A. Abbas, E. Bourtsoulatze, and Y. Andreopoulos. Graph-based spatio-temporal feature learning for neuromorphic vision sensing. *IEEE Trans. Image Process.*, pages 1–1, 2020. 3

[3] Yongjian Deng, Hao Chen, and Youfu Li. Mvf-net: A multi-view fusion network for event-based object classification. *IEEE Trans. Circuits Syst. Video Technol.*, pages 1–1, 2021. 3

[4] Liang Du, Xiaoqing Ye, Xiao Tan, Jianfeng Feng, Zhenbo Xu, Errui Ding, and Shilei Wen. Associate-3ddet: Perceptual-to-conceptual association for 3d point cloud object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 13329–13338, 2020. 1

[5] G. Gallego, J. E. A. Lund, E. Mueggler, H. Rebecq, T. Delbruck, and D. Scaramuzza. Event-based, 6-dof camera tracking from photometric depth maps. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(10):2402–2412, Oct 2018. 1

[6] Daniel Gehrig, Antonio Loquercio, Konstantinos G Derpanis, and Davide Scaramuzza. End-to-end learning of representations for asynchronous event-based data. In *IEEE/CVF Int. Conf. Comput. Vis.*, pages 5633–5643, 2019. 3

[7] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 11873–11882, 2020. 1

[8] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 12697–12705, 2019. 1

[9] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IEEE Int. Conf. Intell. Robot. Syst.*, pages 922–928. IEEE, 2015. 1

[10] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification

and segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 652–660, 2017. 2

[11] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Conf. Neural Inf. Process. Syst.*, pages 5099–5108, 2017. 2

[12] Yusuke Sekikawa, Kosuke Hara, and Hideo Saito. Eventnet: Asynchronous recursive event processing. In *IEEE Conf. Comput. Vis. Pattern Recog.*, June 2019. 3

[13] Qinyi Wang, Yexin Zhang, Junsong Yuan, and Yilong 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)*, pages 1826–1835, 2019. 3

[14] Maosheng Ye, Shuangjie Xu, and Tongyi Cao. Hvnet: Hybrid voxel network for lidar based 3d object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1631–1640, 2020. 1

[15] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4490–4499, 2018. 1