Application of hierarchical clustering for object tracking with a Dynamic Vision Sensor*

Tobias Bolten[1], Regina Pohle-Fröhlich[1], and Klaus D. Tönnies[2]

[1] Hochschule Niederrhein - University of Applied Sciences, Institute of Pattern Recognition, Krefeld, Germany, {tobias.bolten,regina.pohle}@hs-niederrhein.de
[2] University of Magdeburg, Department of Simulation and Graphics, Magdeburg, Germany, klaus@isg.cs.uni-magdeburg.de

Abstract. Monitoring public space with imaging sensors to perform an object- or person-tracking is often associated with privacy concerns. We present a Dynamic Vision Sensor (DVS) based approach to achieve this tracking that does not require the creation of conventional grey- or color images. These Dynamic Vision Sensors produce an event-stream of information, which only includes the changes in the scene.
The presented approach for tracking considers the scenario of fixed mounted sensors. The method is based on clustering events and tracing the resulting cluster centers to accomplish the object tracking. We show the usability of this approach with a first proof-of-concept test.

Keywords: Object tracking · Dynamic Vision Sensor · Event clustering

1 Introduction

In the field of computer-vision automated object detection and tracking are challenging topics. Over the past decades, various approaches have been developed. In [1,2,3] methods under evaluation of the optical flow are considered, whereas in [4,5,6] variations of the Kalman filter and in [7,8,9] techniques of deep-learning based approaches are utilized.

However, these approaches are using conventional, frame-based (grey-value) images captured by classical CCD- or CMOS imagers [10]. Depending on the domain of application, this type of recording can quickly lead to problems with the privacy awareness of potential users (especially in in-home environments) or in the case of public places in complex legal issues [11].

The described use case of object tracking in this paper is part of a project whose goal is to improve the planning of public open space by including the specific user behavior in the basic urban design process. For this purpose, it is planned to construct a distributed, sensor-based system in order to automatically derive various parameters of the considered area. In the first step, we focus on the task of object detection and tracking to derive information about the number of users and their movements.

*This work is part of the project "plsm" which is founded by the European Regional Development Fund under the grant number EFRE-0801082.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13

To overcome privacy concerns and restrictions by laws, we suggest the utilization of an alternative image sensor, the so-called Dynamic Vision Sensor (DVS). This type of sensor is biological inspired and works not in a frame-based manner. Instead it transmits the changes within a scene in an asynchronous way when they happen.

The paper is structured as follows: In section 2 the DVS and its functionality are described. A filtering and clustering approach for object tracking based on a DVS is presented in the subsequent section. In section 4 a simple proof-of-concept comparison of the DVS solution to a classical image-processing solution is presented. Section 5 concludes with a short summary.

2 Dynamic Vision Sensor

CCD- or CMOS imagers typically operate at a fixed frame rate and produce a constant data stream independent of changes in the considered scene. This can lead to high redundancies in the individual captured frames. In contrast to this, the pixels of a Dynamic Vision Sensor operate independently and asynchronously, based on relative light intensity changes in the scene. This approach is, as a part of neuromorphic engineering, borrowed from biology. For this reason, DVSs are also called "silicon retinas".

Each pixel of a DVS only transmits an information (called an *event*) when a change in intensity greater than a pre-defined threshold is measured. As a consequence, a static scene generates no sensor output at all.

The output of this sensor is typically encoded as a sparse stream in an Address-Event Representation (AER). Each event in this stream includes [12]:

(x, y)-Coordinate:
The pixel coordinate in the sensor array that triggered the event.

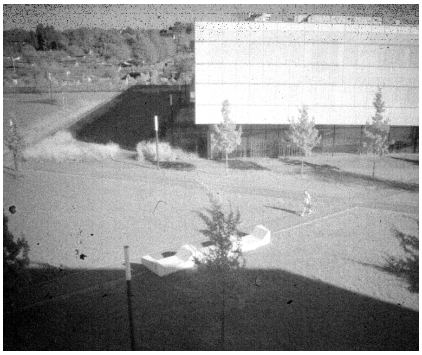
Timestamp:
The time of the occurrence of the event. Typically, in a resolution range of milliseconds.

Meta information:
Depending on a specific sensor model, e.g. the polarity of an event (ON: change from dark $\rightarrow$ bright, OFF: change from bright $\rightarrow$ dark) or the brightness value at the moment of the event generation (greyscale value).

Lichtsteiner et al. mentions in [12] that the first sensor of this type was developed in the mid-1990s. An overview of subsequent developments can be found in [13]. In the scope of this work, we used the "CeleX-IV" sensor, which is developed by Hillhouse Technology [14]. This sensor offers a 768×640 pixel array resolution, a high-speed AER output with 200Meps (events per second) and a high dynamic range of $\approx$ 120dB.

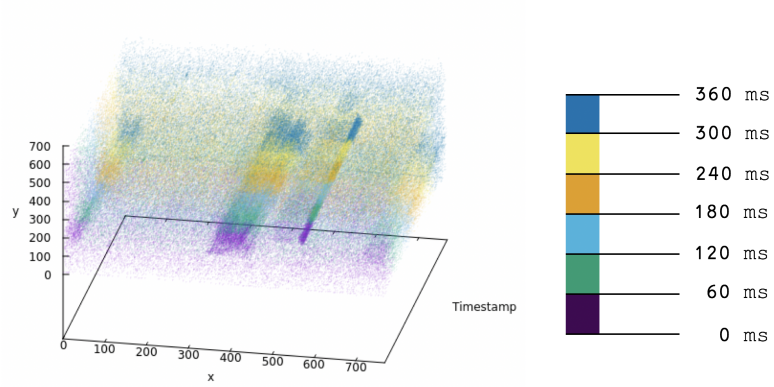
Figure 1 shows an example scene captured with this sensor. In Figure 1a the scene is displayed as a greyscale image, whereas Figure 1b shows the visualization of a 60ms time window of event data as a binary image. Each pixel, where at least one event occurred in the time window, is set to white. Figure 1c illustrates the

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13



(a) Greyscale reference

(b) Binary event visualization in a 60ms time window



(c) Spatiotemporal visualization of six continuous 60ms time windows (color-coded)

Fig. 1: Example visualizations of AER data captured with a CeleX IV-DVS

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13

spatiotemporal information within the stream of events. Each of the six colors in this figure represents a time window of 60ms (total 360ms) of events. The burst of events at the position of the moving human and the tree waving in the wind are clearly visible in these visualizations.

3 Event-Clustering as a basic tracking approach

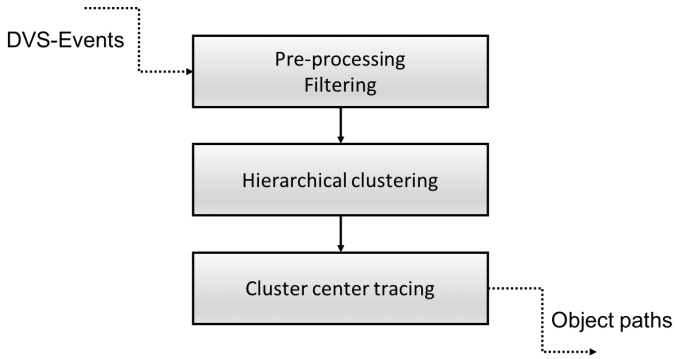


Fig. 2: Suggested processing-chain for object tracking

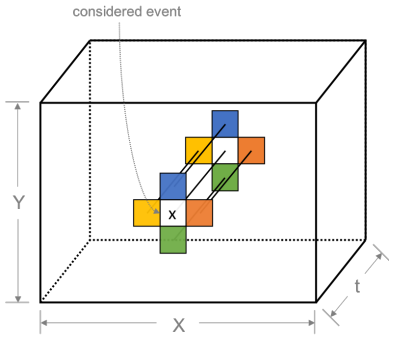
Based on the inherent properties of the event-based vision sensor, we propose the processing-chain in Figure 2 to achieve a tracking of moving objects. For this we use a neighborhood-based event filter as a pre-processing step, followed by a hierarchical clustering and a tracing of cluster centroids. These steps are explained in the following sub-sections. Our implementation is based on slicing the continuous event-stream in non-overlapping blocks of a fixed time length (following referred as sliding time window) and the processing of each of these blocks.

In addition to the privacy benefits (no grey- or color-value information of the scene is needed) offered by the sparsely populated event stream of a DVS, this approach offers the possibility to achieve a solution with little need of computational and power resources. An important point is, that the static background of the scene does not have to be considered. Especially in the context of sensor networks this can be a great advantage.

3.1 Event-Filtering

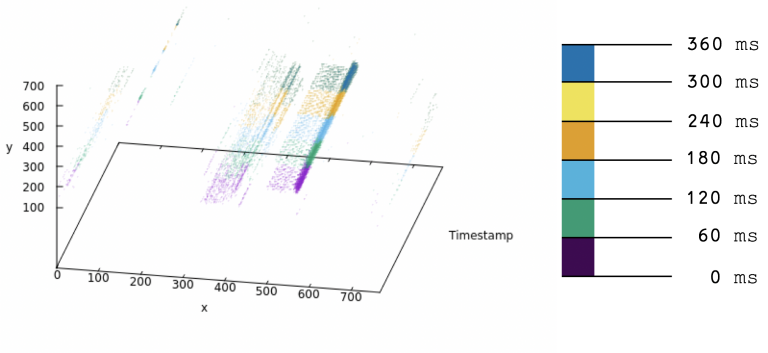
Figure 1b and 1c clearly shows that there is significant sensor noise in the recorded signal, which prevents a sensible use of clustering approaches. Therefore, we suggest a simple filtering step exploiting the spatial and temporal neighborhood for pre-processing. For each event, the number of other events in the

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13

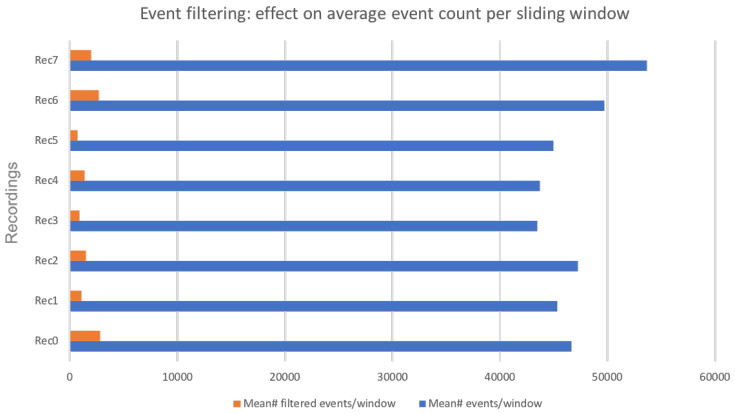


(a) Visualization of considered filter-neighborhood

(b) Filtered binary event visualization in a 60ms time window



(c) Result of the filter applied to data from Figure 1c



(d) Filter effect on the event count per sliding time window

Fig. 3: Visualization of the event filtering step

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13

von-Neumann neighborhood (4-neighborhood) within the current sliding time window is calculated as

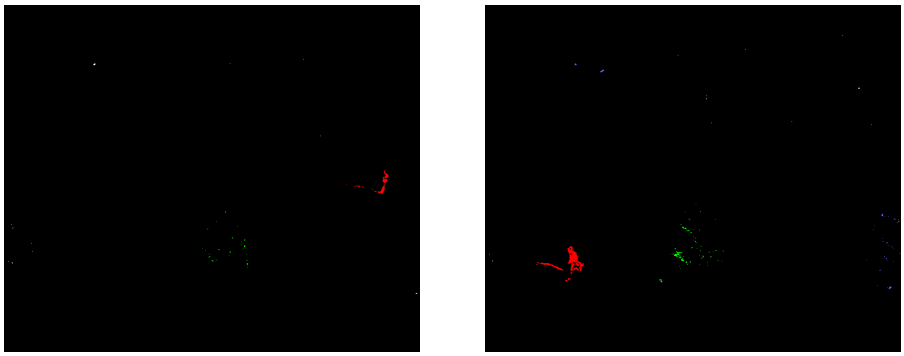
$$f(\text{event}_x, \text{event}_y) = \sum_{t \in \text{time window}} \text{count}(\text{event}_x \pm 1, \text{event}_y \pm 1, t) \qquad (1)$$

Figure 3a clarifies the considered spatio-temporal neighborhood for an event. An event is rejected when $f(\text{event}_x, \text{event}_y) < \text{threshold}$.

We suggest setting the threshold value depending on the width of the underlying sliding time window. We have chosen the value empirically and set it to 1/8 of the sliding window in ms. The filtering result is shown in Figures 3b and 3c (compare with the unfiltered version in Figure 1).

This filtering drastically reduces the number of events which must be processed in the next step, while preserving most of the events from the desired objects. The effect on the average number of events per sliding window is shown exemplarily in Figure 3d based on various recordings (compare with section 4). Within this pratical example a reduction of about 96% on the average event count per sliding window was achieved.

3.2 Hierarchical clustering



(a) Sliding window t_1

(b) Sliding window t_2

(c) Magnified cluster from 4a

(d) Magnified cluster from 4b

Fig. 4: Color-coded result of clustering at different sliding windows

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13

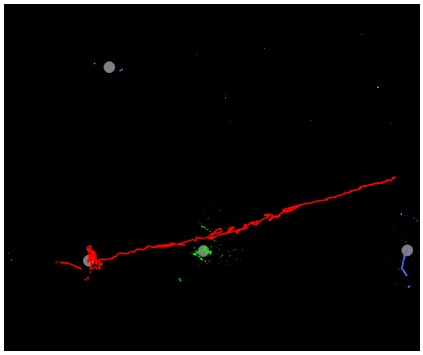


Fig. 5: Highlighted path of tracked center points between sliding window t_1 and t_2 (compare with Figure 4)

The next step in the processing chain consists of clustering the pre-filtered events to get semantic related groups of events. As the number of clusters (moving objects) in the scene is not known a priori, a hierarchical clustering approach is used.

We use a hierarchical *single-link* clustering (provided by the "fastcluster"-library [15]) based on the euclidean distance of the (x,y)-coordinates of the pre-filtered events. The clustering break point is controlled by a pre-defined cutoff-distance. Only clusters consisting of a minimum number of events are considered for further processing. Figure 4 illustrates the result of the clustering step at two different sliding windows in the form of color-coded clusters.

The Table 1 summarizes all parameters and their selected values of our presented approach.

3.3 Cluster path tracking

For each cluster resulting from the previous step, the center point is calculated. Based on this center point, the objects are traced over the time, i.e. over successive sliding time windows. Two center points from clusters in consecutive sliding time windows are considered as semantically linked when their euclidean distance is smaller than a pre-defined threshold (see Table 1). If there is no other point within this distance, the corresponding cluster is interpreted as a new object.

Table 1: Parameter setting overview

Parameter	Setting used in our experiments
DVS-Record: sliding time window	60 ms
Event-Filter: threshold	1/8 · sliding window length [ms]
Clustering: cutoff distance	50 px
Clustering: minimal cluster size	100 events
Cluster path tracing: maximum center distance	50 px

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13

Table 2: Scene description within recorded data

Recordname	Description of scene	# of sliding time windows
Rec 0	Pedestrian crossing, partially obscured by a tree	310
Rec 1	Pedestrian crossing, partially obscured by a tree	359
Rec 2	Pedestrian crossing, partially obscured by a tree	292
Rec 3	Pedestrian crossing	343
Rec 4	Cyclist crossing, partially obscured by a tree	259
Rec 5	Pedestrian crossing, partially obscured by a tree	303
Rec 6	Cyclist crossing, partially obscured by a tree	225
Rec 7	Riding lawn mower crossing	502

The result of tracking these cluster center points is exemplified in Figure 5, which shows the tracked path between the two sliding windows displayed in Figures 4a and 4b.

4 Proof of Concept

The presented cluster-based tracking approach on event-based vision information focuses currently on the special use case of a fixed mounted sensor and moving objects in the monitored scene. Due to the fact that the research area of event-based computer vision is fairly new, there is a lack of well-known standard databases covering various use cases and different DVS-sensor resolutions and characteristics.

Hu et al. [16] are describing an approach to convert "classical" frame-based vision datasets into synthetic event-based datasets. But the converted databases are not addressing the described use case of object tracking and the conversion tries to simulate a DVS sensor with a smaller sensor resolution than the one used in our practical experiments (see section 2). Hence, creating synthetic converted data for our specific sensor will produce artefacts. Thus, we decided to use a small, self-recorded database for the first *proof-of-concept* of the proposed approach. Table 2 briefly summarizes the considered scenes within this dataset.

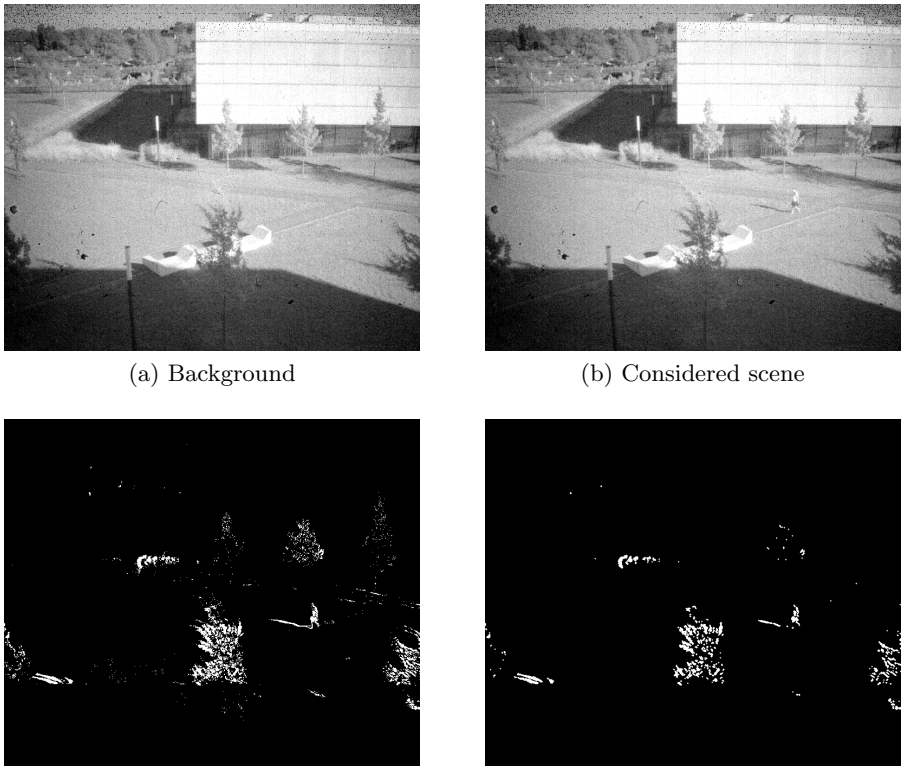
The following subsections present an alternative tracking approach using a frame-based imaging technique which is compared with the proposed DVS-clustering method.

4.1 Comparative approach: difference image

Due to privacy concerns that need to be considered (compare with the project description in the introduction), it is not possible to use "classical" greyscale or color images to monitor the desired space. One possible option from the field of image processing is the approach of using difference images and binarization.

For this purpose, a recording of the background (scene without any objects, see Figure 6a) is taken. Each frame of the actual recording (see Figure 6b) is

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13



(a) Background

(b) Considered scene

(c) Calculated binary difference image

(d) Morphological (Opening, 3x3 cross) filter result

Fig. 6: Visualization of the described privacy aware 'classical' imaging approach

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13

compared with this background in that the difference between these two images is calculated. To ensure the privacy concerns this difference image can be binarized (see Figure 6c), so that no restoration of color- or greyscale values is possible.

Similar to the described filtering of DVS-events this approach allows also the use of a filtering as an additional step. In this case, the use of morphological operations is one possible way. Figure 6d shows the filtered result which arises when using a morphological opening operation with a 3x3 cross-structure kernel element.

Based on these images the use of well-known computer vision object tracker is possible. For comparison we used the implementations[‡] in the openCV-library [17].

4.2 Comparison: Event-Clustering & openCV-tracker

Compared to the presented clustering procedure on the DVS event data, the implementations of the openCV trackers require a bounding box, which includes the object to be tracked, as input parameter. Due to this fact, we decided to compare the two approaches on the basis of the tracked path of this selected object. This means, that in terms of this proof of concept comparison a *single object tracking* is performed, although the DVS clustering-based approach could track multiple objects in a scene.

For the two approaches the algorithmically determined object center is compared to a manually defined ground-truth position. In case of the DVS-clustering this object center is the cluster center point and for the openCV-tracker the center point from the points within the returned object bounding box is used. By estimating this center point over continuous sliding time windows (or the generated and filtered binary images), an object path is determined. An example is given by the red line in Figure 5.

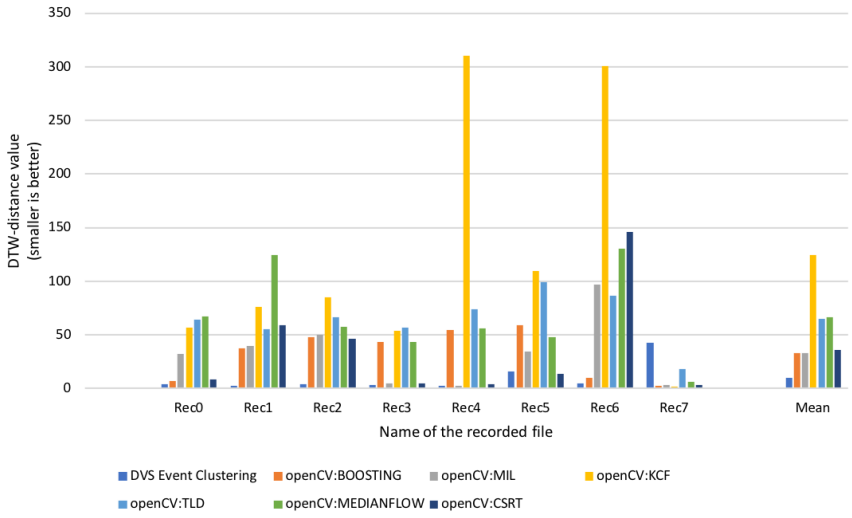
For the quantitative comparison of these paths in comparison with the ground-truth path the dynamic time warp distance is used [18]. This distance measure allows the determination of similarity even for different lengths of results.

In Figure 7 the calculated distances for the DVS-clustering approach, and for six different in openCV implemented object tackers (for details please compare with openCV documentation) are shown. The distance values for each of the openCV trackers is averaged over 15 different executions with identical initial parameters to compensate stochastic effects within some of the trackers. In the considered footage (compare with Table 2) our event-based approach, with one major exception, is comparable or better than the openCV-trackers.

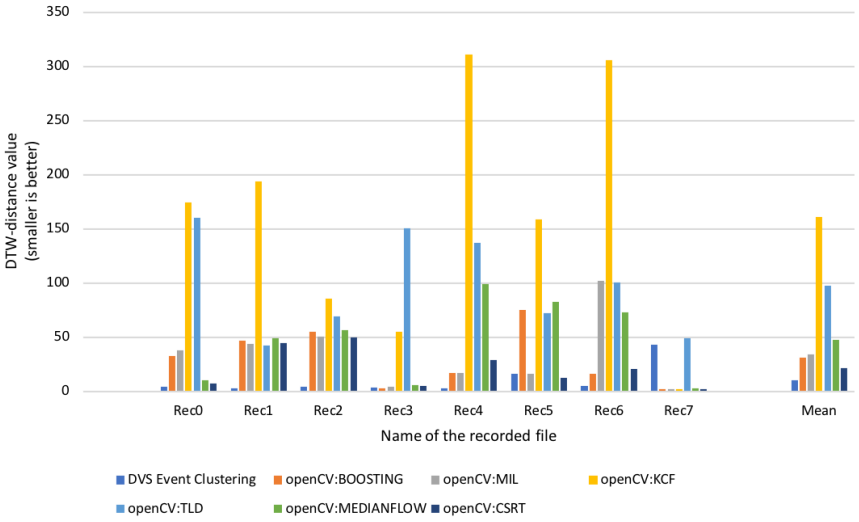
The presented approach fails in *Rec7* due to the performed event filtering step. The Figure 8a shows the unfiltered events which are generated by the object and Figure 8b contains the corresponding filtered events, whereas Figures 8c and 8d show that too many events are removed in the further course of the recording. As a result, the minimal cluster size condition (compare with Section 3.2) is not reached. Therefore, the continuous tracking of the object is lost.

[‡]see https://docs.opencv.org/3.4.2/d0/d0a/classcv_1_1Tracker.html

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13



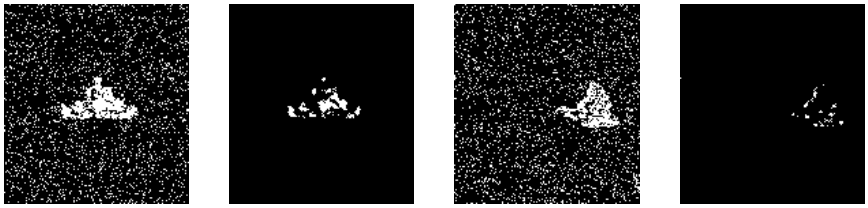
(a) No morphological post-processing on binary difference images



(b) With morphological opening as post-processing on binary difference images

Fig. 7: Calculated path distances for DVS-clustering and six different object trackers implemented by openCV

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13



(a) Plain t_x (b) Filtered t_x (c) Plain t_y (d) Filtered t_y

Fig. 8: Magnified selection of unfiltered and filtered events in *Rec7*

5 Conclusion

We presented an initial approach to track moving objects by clustering and tracing their center points based on event data from a Dynamic Vision Sensor. Our method is simple and fast, while respecting possible privacy concerns of depicted persons.

The method is currently implemented on standard PC hardware. Since filtering of the event data significantly reduces the number of events, outsourcing the filter stage to a FPGA would enable implementation on less powerful microprocessors.

Another important aspect for further research is the creation and publication of a larger event-based database with corresponding ground truth annotations to allow a systematic evaluation that goes beyond the presented proof-of-concept test.

Also, the improvement of the presented approach itself is an aspect for further research. The time information for each event (which has a resolution of milliseconds) is mostly unused in the current approach, which represents potential for additional improvement.

References

1. A. Ranftl, F. Alonso-Fernandez and S. Karlsson, “Face Tracking Using Optical Flow,” 2015 International Conference of the Biometrics Special Interest Group (BIOSIG), Darmstadt, 2015, pp. 1-5.
2. Y. Liu, Y. Lu, Q. Shi and J. Ding, “Optical Flow Based Urban Road Vehicle Tracking,” 2013 Ninth International Conference on Computational Intelligence and Security, Leshan, 2013, pp. 391-395.
3. L. Dan, J. Dai-Hong, B. Rong, S. Jin-Ping, Z. Wen-Jing and W. Chao, “Moving object tracking method based on improved lucas-kanade sparse optical flow algorithm,” 2017 International Smart Cities Conference (ISC2), Wuxi, 2017, pp. 1-5.
4. C. M. Bukey, S. V. Kulkarni and R. A. Chavan, “Multi-object tracking using Kalman filter and particle filter,” 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), Chennai, 2017, pp. 1688-1692.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13

5. X. Mu, J. Che, T. Hu and Z. Wang, "A video object tracking algorithm combined Kalman filter and adaptive least squares under occlusion," 2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), Datong, 2016, pp. 6-10.
6. N. Najafzadeh, M. Fotouhi and S. Kasaei, "Object tracking using Kalman filter with adaptive sampled histogram," 2015 23rd Iranian Conference on Electrical Engineering, Tehran, Iran, 2015, pp. 781-786.
7. J. Redmon, A. Farhadi, "YOLOv3: An Incremental Improvement", Technical Report, arXiv:1804.02767, 2018
8. B. Mocanu, R. Tapu and T. Zaharia, "Single object tracking using offline trained deep regression networks," 2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA), Montreal, QC, 2017, pp. 1-6.
9. K. Behrendt, L. Novak and R. Botros, "A deep learning approach to traffic lights: Detection, tracking, and classification," 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 2017, pp. 1370-1377.
10. S. Mehta, A. Patel and J. Mehta, "CCD or CMOS Image sensor for photography," 2015 International Conference on Communications and Signal Processing (ICCSP), Melmaruvathur, 2015, pp. 0291-0294.
11. Q. Mahmood Rajpoot, C. Jensen, "Video Surveillance: Privacy Issues and Legal Compliance", Promoting Social Change and Democracy through Information Technology, IGI global, 2015, ISBN 9781466685024
12. P. Lichtsteiner, C. Posch and T. Delbruck, 'A 128×128 120 dB 15μs Latency Asynchronous Temporal Contrast Vision Sensor," in IEEE Journal of Solid-State Circuits, vol. 43, no. 2, Feb. 2008, pp. 566-576
13. T. Delbrück, B. Linares-Barranco, E. Culurciello and C. Posch, "Activity-driven, event-based vision sensors," Proceedings of 2010 IEEE International Symposium on Circuits and Systems, Paris, 2010, pp. 2426-2429.
14. M. Guo, J. Huang and S. Chen, "Live demonstration: A 768×640 pixels 200Meps dynamic vision sensor," 2017 IEEE International Symposium on Circuits and Systems (ISCAS), Baltimore, MD, 2017, pp. 1-1.
15. D. Müllner, "fastcluster: Fast Hierarchical, Agglomerative Clustering Routines for R and Python," in Journal of Statistical Software, 53(9), 2013, pp. 1 - 18.
16. Y. Hu, H. Liu, M. Pfeiffer, T. Delbruck, "DVS Benchmark Datasets for Object Tracking, Action Recognition, and Object Recognition," in Journal Frontiers in Neuroscience, 10, 2016, pp. 405-410.
17. G. Bradski, "The OpenCV Library," in Dr. Dobb's Journal of Software Tools, 2000
18. M. Müller, "Information Retrieval for Music and Motion", Springer Berlin Heidelberg, 2007, pp. 69-84, ISBN 9783540740483

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_13