Object Tracking on Event Cameras with Offline–Online Learning

Preprint · July 2019

CITATIONS
0

READS
164

7 authors, including:

Rui Jiang
National University of Singapore
16 PUBLICATIONS **55** CITATIONS
SEE PROFILE

Yueyin Zhou
Huawei Technologies
17 PUBLICATIONS **38** CITATIONS
SEE PROFILE

Qinyi Wang
Nanyang Technological University
3 PUBLICATIONS **8** CITATIONS
SEE PROFILE

Object Tracking on Event Cameras with Offline-Online Learning

Rui Jiang[1], Xiaozheng Mou[1], Shunshun Shi[1],
Yueyin Zhou[1], Qinyi Wang[1], Meng Dong[1], and Shoushun Chen[1,2]

***Abstract*— In this work, an object tracking framework has been presented based on event count images from a Dynamic Vision Sensor (DVS). The framework contains an offline-trained detector and an online-trained tracker which complement each other: The detector benefits from pre-labeled data during training, but may have false or missing detections; The tracker provides persistent results for each initialized object but may suffer from drifting issues or even failures. Besides, process and measurement equations have been modeled, and a Kalman fusion scheme has been proposed to incorporate measurements from the detector and the tracker. Self-initialization and track maintenance in the fusion scheme ensure autonomous real-time tracking without user intervene. With self-collected DVS data in urban driving scenarios, experiments have been conducted to show the performance of the proposed framework and the fusion scheme.**

I. Introduction

Multiple object tracking has become an essential module in human-computer interaction [1], automated surveillance [2], and vehicle navigation systems [3], [4]. With the rapid development of sensor technology, event cameras (or Dynamic Vision Sensors, DVSs) have come into the market thanks to their superior performance with fast-moving objects and in high dynamic scenes. Although lots of research efforts have been made, the algorithms designed for event cameras are still far from mature. In particular, DVSs generate significantly different data compared to conventional image sensors (see Section III-A for details). There exist two technical routes for DVS data processing and interpretation. The first tries to establish an event-based processing framework, while the second makes use of the off-the-shelf image processing approaches to simplify the development. While the first route may have a better chance to unleash the full potential of the sensor, most existing methods belong to the second. Although data from DVSs can be approximately represented in frames, it is still not clear that if these data can be efficiently and effectively utilized by traditional and modern machine vision algorithms, such as feature extraction and Convolutional Neural Networks (CNNs). This work studies the visual tracking problem based on DVSs, with the main contributions:

1) An object tracking framework has been presented for event cameras, aiming at real-time applications for ground vehicles. The tracking framework contains a YOLO-based [5] offline-trained deep detector and a CF-based [6] online-trained tracker, which complements each other.
2) A fusion scheme, which incorporates the results from the detector and the tracker, has been proposed based on the Kalman filter. With automated initialization and track maintenance strategies in the fusion scheme, real-time intervene-free operations could be attained.
3) Features in event count images have been demonstrated well-learned by deep networks and feature extractors. Robustness and accuracy of the tracker have been presented using data in actual driving scenarios.

The paper is organized as follows: Section II presents related work in event-based object tracking. Section III introduces the event camera and its unique data format, and the essential modules in an online tracker. Then, the proposed tracking framework is elaborated in Section IV, followed by evaluation results in Section V. Lastly, Section VI concludes the paper.

II. Related Work

Object tracking has been extensively studied in the past decades. In [7], different implementations for the five building blocks in a tracking framework, namely the motion model, the feature extractor, the observation model, the model updater, and the ensemble post-processor have been evaluated to help design a tracker. Authors in [8] review multiple object tracking approaches thoroughly. A Multiple Object Tracker (MOT) is not a simple combination of single object trackers. In practice, manual initialization is often infeasible; thus an initialization scheme needs to be proposed for MOTs. Moreover, a maintenance mechanism is essential in MOTs to create new tracks, delete obsolete tracks and assign consistent labels, since the number of objects may vary at any time. Besides, data association is necessary for assigning detections to trackers such that the ensemble post-processor can update objects' position according to latest observations.

Recently, the concept of "tracking by detection" has become increasingly popular owing to its discriminative nature compared to conventional generative model-based trackers [9], [10], [11]. Many event-based detectors have been proposed: As an early attempt, a hierarchical spiking model for object recognition has been introduced in [12], where asynchronous data are fed into the spiking neural networks that take advantage of accurate timing from event cameras. An end-to-end object detector has been proposed for DVS

[1]Rui Jiang, Xiaozheng Mou, Shunshun Shi, Yueyin Zhou, Qinyi Wang, Meng Dong are with the CelePixel Technology Co. Ltd, 71 Nanyang Drive, Singapore 638075. *Email:* rui.jiang@celepixel.com

[1,2]Shoushun Chen is with the School of Electrical and Electronic Engineering, Nanyang Technological University, 50 Nanyang Ave, Singapore 639798, and with the CelePixel Technology Co. Ltd, 71 Nanyang Drive, Singapore 638075.

in [13], where the feature extraction network is designed by considering unique data structure from DVS; Moreover, the adaptive temporal pooling is applied to balance triggered events between rapid and slow motions. A comparison of object detection performance between traditional image frames and event frames has been presented in [14], which shows the advantages of event-based approaches in fast and low-light conditions. Authors in [15] proposes two neural networks architectures for object detection based on the YOLO [5] architecture.

As for event-based object tracking, a DVS has been used to assist MOT in [16], demonstrating superior performance for fast-moving objects. In [17], an expectation maximization-based probabilistic multi-hypothesis tracker is presented to tackle MOT with false alarm observations. In [18], a moving object detector has been proposed for event cameras. After motion correction, those areas with different event timestamps are deemed as moving objects. An event-driven stereo vision system has been developed in [19], where cluster tracking and 3-D reconstruction are complementarily integrated such that ambiguity could be solved when occlusion occurred. By simple morphological operations, the moving objects are extracted and tracked using a Kalman filter with the constant acceleration model. Authors in [20] propose an online tracker-detector integration framework, where the local tracker is based on a Support Vector Machine (SVM) classifier, and the user-initialized global detector is responsible for failure recovery.

Compared to [20], this work considers multiple objects in the tracking framework. The offline-trained detector not only brings a priori information such that self-initialization is enabled but also improves final performance by providing drift-free, independent results for each frame. The online-trained tracker provides persistent results once initialized, and are free from drifting issues with the help of self-initialization.

III. Preliminaries

In this preliminary section, we first introduce DVSs and their data formats for those readers who are not familiar with event cameras. Then we present how to represent the appearance of a region of interest to generate "features", and how to utilize the features in single object tracking, and finally, how to generalize to MOT problems from a single object tracker.

A. Event Cameras and Event Frames

Event cameras detect intensity changes at each pixel independently. When the change is larger than a user-defined threshold, an "event" $\langle (x, y), I, t \rangle$ would be triggered, where x, y denote pixel coordinates, I is the intensity, and t represents the timestamp[1]. From these events, diverse processing techniques can be used to generate event-based data for easier implementation. By aggregating events in an interval as a tuple set $\mathcal{S} = \{\langle (x, y), I, t \rangle\}$, different types of event frames can be obtained. We write the fixed time interval for event frames as ΔT; t_1 denotes the timestamp of the first event in each interval; $I_k(x, y)$ represents the intensity at pixel (x, y) for the k-th frame. An event binary image is

$$I_k(x, y) = \begin{cases} 1, & \text{if } (x, y) \in \mathcal{S} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

An event intensity image, which utilizes the intensity information, can be generated as

$$I_k(x, y) = \begin{cases} I, & \text{if } (x, y) \in \mathcal{S} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

An event count image that accumulates event as intensity for each pixel is

$$I_k(x, y) = \sum_{(x,y)\in\mathcal{S}} 1 \tag{3}$$

An 8-bit event full frame picture, which reflects events' sequence, is obtained from

$$I_k(x, y) = \begin{cases} \lfloor \frac{t-[t_1+(k-1)\Delta T]}{\Delta T} \times 255 \rfloor, & \text{if } (x, y) \in \mathcal{S} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where $\lfloor \cdot \rfloor$ denotes the floor function. In this work, we use event count images as the only input of both detection and tracking.

B. Essentials of the Online Tracker

As this paper is not a comprehensive review, only those building blocks that have been applied to our framework are explained here for readers' convenience.

1) Appearance Representation: A natural idea to represent the region of interest on an image is to simply use the intensity values inside the region and write them as vector form. To include more information, intensities in color channels instead of grayscale have been used. However, those raw intensities are sometimes too specific to describe the substantial features of the interested region. The histogram of oriented gradients (HOG) [22], [23] has been demonstrated as a useful feature in object detection. In addition to hand-crafted features, it has been demonstrated that learned features [24], [25], [26] using data-driven techniques sometimes perform better. YOLO is one of the examples that shows excellent object detection results of deep convolutional features. In the proposed framework, deep features have been used in YOLO, while either intensity features or HOG could be applied to the CF.

2) The Correlation Filter (CF): The filter-based object trackers model the problem as regression. Given a set of features $\mathbf{y}_i$ with target points as training samples and their corresponding user-defined responses g_i, we aim to train a filter $f(\mathbf{y}_i)$ which gives the maximum correlation output on these points. Suppose the regression model is linear $f(\mathbf{y}_i) = \mathbf{u}^\top \mathbf{y}_i$, the problem becomes to find the model parameter $\mathbf{u}$ that leads to minimum squared regression error:

$$\min_{\mathbf{u}} \sum_i \left(f(\mathbf{y}_i) - g_i\right)^2 + \lambda \|\mathbf{u}\|^2 \tag{5}$$

[1]The explicit data format varies based on the sensor model. Events as $\langle (x, y), p, t \rangle$ where p is the polarity are common in literature [21].

where $\lambda > 0$ is the penalty term parameter. Correlation is computed in Fourier domain so that fast online training could be achieved because correlation operation becomes element-wise multiplication in Fourier domain. The analytical solution of (5) in Fourier domain has been derived as [27], [6]

$$\mathbf{u} = \left(X^H X + \lambda \mathbf{I}\right)^{-1} X^H \mathbf{g} \tag{6}$$

where X is the data matrix that contains one sample $\mathbf{x}_i$ for each row; $(\cdot)^H = (\cdot)^{*\top}$ denotes the Hermitian transpose, and $(\cdot)^*$ is the complex-conjugate notation; $\mathbf{I}$ is the identity matrix, and $\mathbf{g}$ contains regression targets g_i.

Authors in [6] have noted that the regression problem could be further accelerated when the training data is composed of cyclic shifts. Moreover, Kernelized Correlation Filters (KCFs) have been derived to tackle kernel ridge regression problems. The proposed approach is generalized to multi-channel images, such that more features including HOG could be applied in appearance representation. Considering both speed and accuracy, the KCF is used for online learning in this work.

3) Data Association: One additional module for MOT compared to single object tracking is the data association, which assign detected objects to existing tracks. The problem is usually formulated as the minimization of total costs, which, could be solved by Hungarian algorithm [28], [29]. Particularly, suppose we have n detections pending assignment to m tracks, and a cost matrix $\mathbf{A}_{\text{cost}} \in \mathbb{R}^{m \times n}$ representing the cost of assigning the j-th detection to the i-th track. The algorithm outputs pairs of tracks and corresponding detections as the first and second columns in matrix $\mathbf{A}_{\text{assgnmt}} \in \mathbb{R}^{l \times 2}$ respectively, where l denotes the number of successfully paired assignments.

IV. Offline-Online Multiple Object Tracking

In this section, the proposed tracking framework, which contains two stages as illustrated in Fig. 1, is introduced. At stage one, offline training has been done based on YOLO detector [5] before tracking such that object detection results with objects' types could be obtained. At stage two, online training and tracking are in a simultaneous process. The real-time generated event count images are fed into the offline-trained detector and the CF. After data association, those assignments are sent to a Kalman ensemble as measurements. Meanwhile, the CF also outputs its tracked results to the Kalman ensemble as supplementary measurements. Reinitialization is triggered once inconsistencies are found between outputs from the ensemble and the CF, such that the ensemble is adaptive to various conditions with changes of appearance and scale.

A. Offline Learning

The offline learning follows the YOLOv3 framework [30] where the ground truth is manually labeled with bounding boxes on the objects in event count images. Although YOLO is able to detect multiple classes, only cars are considered in off-line training in this work due to lack of manually-labeled data. The features are then extracted by undergoing the convolutional neural network of Darknet-53. Finally, the model is learned or fine-tuned by minimizing the loss between the ground truth and the regression results.

B. Online Learning and Tracking

The online learning and tracking procedure, as the core contribution of this work, is summarized in Algorithm 1. The proposed tracking framework maintains a list `tracks` that keeps necessary properties (such as Kalman filter estimates, type, CF status and age variables) of all track candidates. A priori estimate and error covariance are computed from the function `predictKalman`, and `assignHungarian` tries to find the best match between `tracks` and `detections`. Then, estimated bounding boxes on current image I_k are used as positive samples for online learning in `correlationFilter`. Correction is done in `correctKalman` to incorporate online CF and offline detector together. The function `maintainTracks` removes obsolete tracks, creates new tracks and updates their properties according to assignment results. The conditions of track removal is detailed in the subsection IV-B.1.

1) System Modeling in Kalman Ensemble: As the Kalman filter has been widely used in object tracking [31], [32], the explicit equations have been omitted due to the page limit. In this work, notation $\hat{(\cdot)}$ denotes the estimated value; At discrete time k, subscript $\hat{(\cdot)}_{k|k-1}$ and $\hat{(\cdot)}_{k|k}$ represents the predicted (a priori) estimate and the updated (a posteriori) estimate, respectively.

We consider the following linear process model between k and $k+1$:

$$\mathbf{x}_{k+1} = \mathbf{F}_k \mathbf{x}_k + \mathbf{w}_k = \mathbf{F}_k \mathbf{x}_k + \mathbf{G}_k \breve{\mathbf{w}} \tag{7}$$

where the state vector $\mathbf{x} = [x, y, v_x, v_y, w, v_w, s]^\top \in \mathbb{R}^7$ contains properties of the bounding box being tracked, which includes central coordinates, velocities, the width, the change rate of width, and the ratio of width to height, respectively; By assuming that the process modeling error is caused by random impact in two-dimensional image plane, $\breve{\mathbf{w}} = [a_x, a_y, a_w, a_s]^\top \in \mathbb{R}^4$ denotes the zero-mean independent Gaussian noise with diagonal covariance matrix $\breve{\mathbf{Q}} \in \mathbb{R}^{4 \times 4}$, and the state transition matrix and the

$$\mathbf{F}_k = \begin{bmatrix} 1 & 0 & \Delta t_k & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \Delta t_k & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t_k & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{G}_k = \begin{bmatrix} \frac{1}{2}\Delta t_k^2 & 0 & 0 & 0 \\ 0 & \frac{1}{2}\Delta t_k^2 & 0 & 0 \\ \Delta t_k & 0 & 0 & 0 \\ 0 & \Delta t_k & 0 & 0 \\ 0 & 0 & \frac{1}{2}\Delta t_k^2 & 0 \\ 0 & 0 & \Delta t_k & 0 \\ 0 & 0 & 0 & \frac{1}{2}\Delta t_k^2 \end{bmatrix}$$

where Δt_k is the time interval between states at k and $k+1$. At time k, by denoting the process covariance matrix in

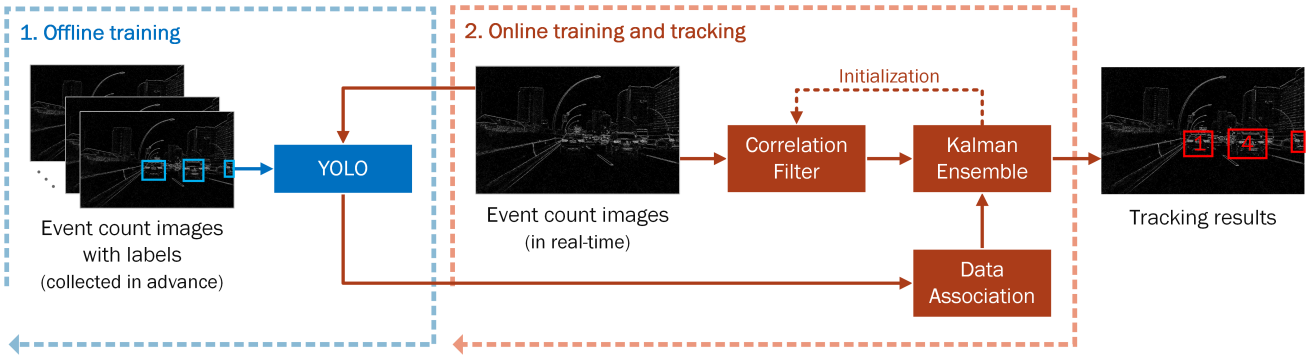

Fig. 1. The proposed object tracking framework.

standard form as $\mathbf{Q}_k$, we have

$$\begin{aligned}\mathbf{Q}_k &= \mathrm{Cov}[\mathbf{w}_k] = \mathrm{Cov}[\mathbf{G}_k\breve{\mathbf{w}}] = \mathrm{E}[(\mathbf{G}_k\breve{\mathbf{w}})(\mathbf{G}_k\breve{\mathbf{w}})^\top] \\ &= \mathbf{G}_k\mathrm{E}[\breve{\mathbf{w}}\breve{\mathbf{w}}^\top]\mathbf{G}_k^\top = \mathbf{G}_k\mathrm{Cov}[\breve{\mathbf{w}}]\mathbf{G}_k^\top \\ &= \mathbf{G}_k\breve{\mathbf{Q}}\mathbf{G}_k^\top \end{aligned} \tag{8}$$

where $\mathrm{Cov}[\cdot]$ and $\mathrm{E}[\cdot]$ denote the covariance matrix and the expectation of a random vector.

The measurement model at time k is written as

$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x}_k) + \mathbf{v}_k \tag{9}$$

By omitting the time index, we have

$$\mathbf{h}(\mathbf{x}) = \begin{bmatrix} x & y & w & \frac{w}{s} & x & y \end{bmatrix}^\top \tag{10}$$

The measurement vector $\mathbf{z} = [\mathbf{z}_\mathrm{d}^\top, \mathbf{z}_\mathrm{c}^\top]^\top$ contains two parts, where $\mathbf{z}_\mathrm{d}$, $\mathbf{z}_\mathrm{c}$ is from the offline-trained detector and the online-trained CF, respectively. Specifically, $\mathbf{z}_\mathrm{d} = [x_\mathrm{d}, y_\mathrm{d}, w_\mathrm{d}, h_\mathrm{d}]^\top$ and $\mathbf{z}_\mathrm{c} = [x_\mathrm{c}, y_\mathrm{c}]^\top$. Note that we have not considered width and height from the CF, since no scale change is allowed in CF for the sake of real-time performance. After linearization, the Jacobian $\mathbf{H}_k = \frac{\partial \mathbf{h}}{\partial \mathbf{x}}|_{\hat{\mathbf{x}}_{k|k-1}}$ is derived as

$$\mathbf{H}_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\hat{s}_{k|k-1}} & 0 & -\frac{\hat{w}_{k|k-1}}{\hat{s}^2_{k|k-1}} \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and $\mathbf{v}_k$ denotes the measurement noise that is assumed zero-mean Gaussian with covariance matrix $\mathbf{R}_k$.

It is noted that $\mathbf{z}_\mathrm{d}$ is not always available as 1) the detector may fail to recognize the object for each frame, or 2) the data association may not assign the detected bounding boxes to their corresponding tracks correctly. In this case, the ensemble still works in a short time by correcting objects' position using $\mathbf{z}_\mathrm{c}$, which is persistent once initialized. If neither $\mathbf{z}_\mathrm{d}$ nor $\mathbf{z}_\mathrm{c}$ is available, the ensemble predicts objects' position for a certain period of iterations τ, and removes those tracks if measurements cannot be recovered.

Algorithm 1 Online learning and tracking on event frames.

```
k = 1
tracks = initializeTracks()
{x̂_{0|0}, P_{0|0}} = initializeKalman()
while true do
    {x̂_{k|k-1}, P_{k|k-1}} = predictKalman(x̂_{k-1|k-1}, P_{k-1|k-1})
    z_d = assignHungarian(detections, tracks)
    z_c = correlationFilter(tracks, I_k)
    {x̂_{k|k}, P_{k|k}} = correctKalman(x̂_{k|k-1}, P_{k|k-1}, z_k)
    tracks = maintainTracks(tracks)
    k++
end while
```

2) Cost Function in Data Association: We use the assignment algorithm described in Section III-B.3. The values in H_cost play a central role in algorithm performance. In this work, we modified the cost function [33] at time k such that both position and type from the detector are considered in assignment:

$$\begin{aligned} A_\mathrm{cost}(i,j) &= (\mathbf{z}_\mathrm{d} - \mathbf{H}_\mathrm{d}\hat{\mathbf{x}}_{k|k-1})^\top \Sigma_\mathrm{d}^{-1}(\mathbf{z}_\mathrm{d} - \mathbf{H}_\mathrm{d}\hat{\mathbf{x}}_{k|k-1}) \\ &\quad + \ln|\Sigma_\mathrm{d}| + \eta_\mathrm{type}\psi_k(i,j) \end{aligned} \tag{11}$$

where $\mathbf{H}_\mathrm{d} \in \mathbb{R}^{4\times 7}$ contains the first four rows of $\mathbf{H}_k$; $A_\mathrm{cost}(i,j)$ denotes the element at row i, column j in matrix $\mathbf{A}_\mathrm{cost}$; $\Sigma_\mathrm{d} = \mathbf{H}_\mathrm{d}\mathbf{P}_{k|k-1}\mathbf{H}_\mathrm{d}^\top$ could be computed each time after Kalman prediction; $|\cdot|$ is the matrix determinant notation; parameter η_type is the weight of type detection error and

$$\psi_k(i,j) = \begin{cases} 1, & \text{if the track-detection pair } (i,j) \\ & \text{share the same type at time } k \\ 0, & \text{otherwise} \end{cases}$$

The discrete time index k is omitted in $A_\mathrm{cost}(i,j)$, $\mathbf{z}_\mathrm{d}$, $\mathbf{H}_\mathrm{d}$ and Σ_d for simplicity.

3) Initialization and Reinitialization of CF: Since the proposed tracking framework aims to work continuously without users' intervention, it is necessary to design an initialization scheme for the CF to deal with new objects. As the CF does not consider scale variation, reinitialization is required to update objects' bounding boxes so that the regions out of the objects could be excluded in the CF. Moreover, due to the change of object motion, appearance and illumination conditions, results from the CF may drift

unboundedly or even fail, which also need to be recovered by an reinitialization scheme. In this work, a Boolean flag `initializedCF` with the default status `false` would be assigned for each track. At time k, the initialization would be done to set `initializedCF` = `true` if *all* following conditions are satisfied:

- There is a reliable track from the Kalman ensemble;
- The track is with the status `initializedCF` = `false`;
- The measurement $\mathbf{z}_\mathrm{d}$ is available.

Each initialized track would be constantly checked at each iteration. Reinitialization of CF would be triggered to reset `initializedCF` = `false`, when *one of* the following conditions is satisfied:

- The difference of central coordinates between the CF and Kalman ensemble exceeds the user-defined threshold;
- The difference of bounding boxes' size between the CF and Kalman ensemble exceeds the user-defined threshold;
- The bounding box from the CF is too close to the edges of image.

The first and second condition deals with drifts and scale variations, respectively, while the third one is based on the fact that CF performs worse at image edges due to incomplete positive and negative learning samples.

V. Evaluation in Driving Scenarios

We have tested the approach on self-collected event data, a long sequence of 1663 images in urban areas, from CelePixel CeleX5 Sensor. The data show common behaviors of vehicles such as lane change and overtaking. Occlusion and scale variation are frequent due to the heavy traffic. Some typical scenarios in the dataset is shown in Fig. 2. The performance metrics used in this work include the Multiple Object Tracking Accuracy (MOTA) [34], [35] and the Maximum Label Number (MLN). MOTA is defined as

$$\mathrm{MOTA} = 1 - \frac{\sum_k (FN_k + FP_k + \Phi_k)}{\sum_k GT_k} \tag{12}$$

where FN_k, FP_k, Φ_k, and GT_k represent the number of missed detections (or false negatives), of false positives, of mismatched objects (or fragmentations), and of groundtruth objects at time k, respectively. MLN is defined as the maximum label number from the tracker. It reflects the unnecessary initialization of new tracks: the larger MLN is, the weaker label continuity is.

Although quantitative metrics related on tracking accuracy on bounding boxes (e.g., Multiple Object Tracking Precision, MOTP) is not available due to lack of ground truth, the improved performance can still be demonstrated by qualitative results. The full evaluation video have been made available online at `https://youtu.be/2T1730muwVM`.

A. General Results

We firstly run the YOLO detector alone as a baseline, which shows MOTA $= 0.62$ without considering Φ_k as there is no numbered label for YOLO detection results. With the proposed approach, it is observed that MOTA has been raised to 0.69 when using the linear kernel, HOG feature and track removal threshold $\tau = 10$. The metrics used for MOTA calculation could be found in Fig. 3. In particular, by implementing the proposed framework, the numbers of FN and TP become much smoother, as shown in Fig. 3 especially from $k = 600$ to 750. Moreover, there is no FP for the proposed approach after removing unreliable tracks, at the cost of a slower response to new objects. In the testing sequence, no sudden change of identification indexes for the same object has been found; thus, the fragmentation $\Phi_k = 0$.

TABLE I

Real-time performance of the proposed tracking framework.

Partial configuration	*Full configuration*		
	Feature	**Intensity**	**HOG**
459 FPS	Linear kernal	55.7 FPS	50.4 FPS
	Gaussian kernal	33.8 FPS	34.6 FPS

Fig. 4 shows the relation between MLN and the user-defined parameter τ. It is obvious that MLN becomes smaller with a larger τ, because a larger τ retains the track longer when the detection result is not available. However, a larger τ would also lead to higher drifting errors in case of no correction from the detector. In this work, by incorporating CF results, the tracking performance could be improved by setting a larger τ.

Without the online-trained tracker, the whole tracking framework still works by considering the output $\mathbf{z}_\mathrm{d}$ as the sole measurement vector. We have compared the real-time performance between two configurations: 1) the partial configuration without CF, and 2) the full configuration with offline-online learning, as shown in Table I. The time consumption covers core operations in the framework such as CF, Kalman filtering, and data association, and does not include YOLO detection as it is executed outside the loop. The FPS is computed on a mobile workstation running MATLAB R2018b, with an Intel Core i5-8250U CPU and 8GB memory. HOG orientation number and cell size is set to 6 and 4, respectively. Although kernelized CF could be used to improve the performance by bringing features into high-dimensional spaces [6], it lowers computational efficiency. It is noted that the performance gap between different kernels is small when a powerful feature is selected. These results are similar to [7].

B. Influence of Kalman Parameters

It is well known that parameters in Kalman ensemble, namely noise covariance matrices, affect final results. In all experiments, we set

$$\breve{\mathbf{Q}} = \mathrm{diag}(500, 1000, 20, 0.05) \tag{13}$$

$$\mathbf{R} = \mathrm{diag}(5, 5, 20, 20, 10, 10) \tag{14}$$

by experience and parameter tuning. The difference in x and y components in $\breve{\mathbf{Q}}$ is to deal with the uneven pavement or humps, where objects may jump in y direction. In practice,

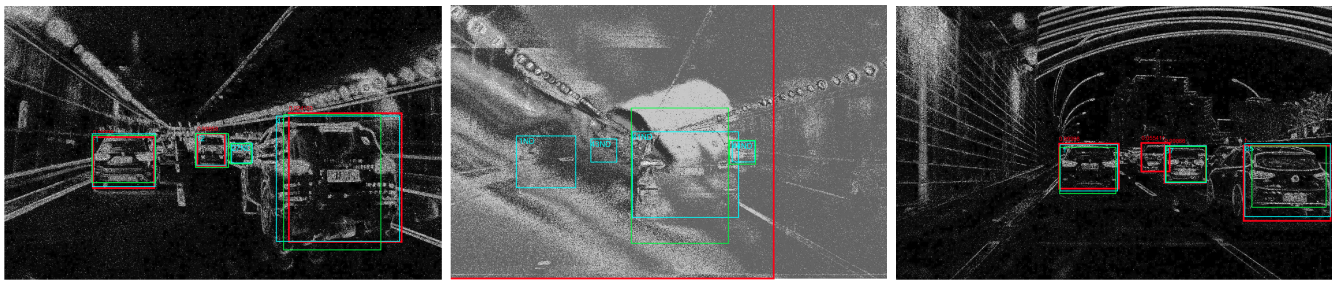

Fig. 2. Typical scenarios in the testing sequence. Left: lane change in a tunnel; Middle: a blurred event count image with no available detection; Right: Exiting the tunnel under high dynamic range. The red, green and cyan bounding boxes denote results from the detector, the CF, and the fusion scheme, respectively. Red numbers are confidence from the detector. Cyan numbers are object identification indexes. "ND" means no matched detection for current tracking result.

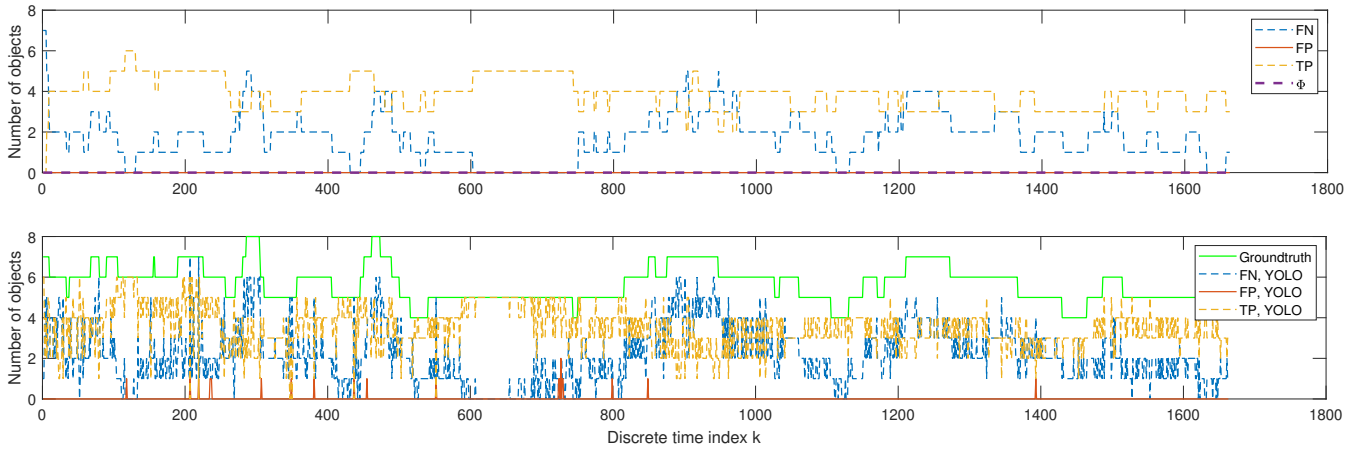

Fig. 3. Metrics used for MOTA calculation. "FN", "FP", "TP", and "Φ" denotes False Negative, False Positive, True Positive, and fragmentation, respectively.

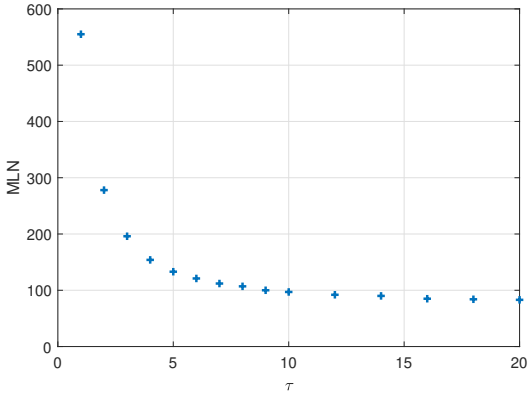

Fig. 4. The relation between MLN and parameter τ, which determines how long an invisible track would be removed if not detected consecutively.

we prefer to trust measurements from the offline-trained detector more, if the measurement is available, as the precision of CF bounding boxes' positions are less accurate since the image quality has been sacrificed for efficiency.

VI. Conclusion

This paper presents a real-time tracking framework based on event count images for event cameras. Compared to conventional tracking-by-detection, an offline-online training scheme has been presented to improve the tracking robustness and accuracy. Deep features, HOG or raw intensity features have been demonstrated effective in the detector and the tracker, respectively. A Kalman ensemble, together with the proposed self-initialization scheme, has been designed to incorporate measurements from the detector and the tracker such that the tracking consistency is enhanced with bounded position drifts.

Although some achievements have been made, the limitations of this work include: 1) Offline training is required, which makes objects being limited to pre-defined categories; 2) Experiments are still at the preliminary stage, where only cars in urban areas have been considered in the detector. In the future, the authors believe that the detector could be enhanced by introducing more data in a variety of environments. As for feature selection in CF, the possibilities of applying other features (such as LBP [36], Haar [37] and other deep features) that describe event count images better could be explored. The data association could be improved by adding appearance discrepancies between detected and predicted bounding boxes into the assignment cost function. It is promising to reduce the time consumption by exploiting data compression mechanisms.

REFERENCES

[1] S. Sridhar, F. Mueller, M. Zollhöfer, D. Casas, A. Oulasvirta, and C. Theobalt, "Real-time joint tracking of a hand manipulating an object from rgb-d input," in *European Conference on Computer Vision*. Springer, 2016, pp. 294–310.

[2] O. Javed and M. Shah, "Tracking and object classification for automated surveillance," in *European Conference on Computer Vision*. Springer, 2002, pp. 343–357.

[3] A. Mendes, L. C. Bento, and U. Nunes, "Multi-target detection and tracking with a laser scanner," in *IEEE Intelligent Vehicles Symposium, 2004*. IEEE, 2004, pp. 796–801.

[4] D. Zhao, H. Fu, L. Xiao, T. Wu, and B. Dai, "Multi-object tracking with correlation filter for autonomous vehicle," *Sensors*, vol. 18, no. 7, p. 2004, 2018.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[6] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 3, pp. 583–596, 2014.

[7] N. Wang, J. Shi, D.-Y. Yeung, and J. Jia, "Understanding and diagnosing visual tracking systems," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3101–3109.

[8] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, X. Zhao, and T.-K. Kim, "Multiple object tracking: A literature review," *arXiv preprint arXiv:1409.7618*, 2014.

[9] K. Okuma, A. Taleghani, N. De Freitas, J. J. Little, and D. G. Lowe, "A boosted particle filter: Multitarget detection and tracking," in *European conference on computer vision*. Springer, 2004, pp. 28–39.

[10] S. Avidan, "Ensemble tracking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 2, pp. 261–271, 2007.

[11] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, "Robust tracking-by-detection using a detector confidence particle filter," in *2009 IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 1515–1522.

[12] G. Orchard, C. Meyer, R. Etienne-Cummings, C. Posch, N. Thakor, and R. Benosman, "HFirst: a temporal approach to object recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 10, pp. 2028–2040, 2015.

[13] J. Li, F. Shi, W. Liu, D. Zou, Q. Wang, H. Lee, P. Park, and H. Ryu, "Adaptive temporal pooling for object detection using dynamic vision sensor," in *British Machine Vision Conf.(BMVC)*, 2017.

[14] M. Iacono, S. Weber, A. Glover, and C. Bartolozzi, "Towards event-driven object detection with off-the-shelf deep learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–9.

[15] M. Cannici, M. Ciccone, A. Romanoni, and M. Matteucci, "Event-based convolutional networks for object detection in neuromorphic cameras," *arXiv preprint arXiv:1805.07931*, 2018.

[16] D. Saner, O. Wang, S. Heinzle, Y. Pritch, A. Smolic, A. Sorkine-Hornung, and M. H. Gross, "High-speed object tracking using an asynchronous temporal contrast sensor." in *VMV*, 2014, pp. 87–94.

[17] B. Cheung, M. Rutten, S. Davey, and G. Cohen, "Probabilistic multi hypothesis tracker for an event based sensor," in *2018 21st International Conference on Information Fusion (FUSION)*. IEEE, 2018, pp. 1–8.

[18] A. Mitrokhin, C. Fermüller, C. Parameshwara, and Y. Aloimonos, "Event-based moving object detection and tracking," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–9.

[19] L. A. Camuñas-Mesa, T. Serrano-Gotarredona, S.-H. Ieng, R. Benosman, and B. Linares-Barranco, "Event-driven stereo visual tracking algorithm to solve object occlusion," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 9, pp. 4223–4237, 2018.

[20] B. Ramesh, S. Zhang, Z. W. Lee, Z. Gao, G. Orchard, and C. Xiang, "Long-term object tracking with a moving event camera," in *British Machine Vision Conference*, vol. 2, 2018.

[21] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza, "Event-based vision: A survey," *CoRR*, vol. abs/1904.08405, 2019. [Online]. Available: http://arxiv.org/abs/1904.08405

[22] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *international Conference on computer vision & Pattern Recognition (CVPR'05)*, vol. 1. IEEE Computer Society, 2005, pp. 886–893.

[23] C. Vondrick, A. Khosla, T. Malisiewicz, and A. Torralba, "Hoggles: Visualizing object detection features," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1–8.

[24] N. Wang and D.-Y. Yeung, "Learning a deep compact image representation for visual tracking," in *Advances in neural information processing systems*, 2013, pp. 809–817.

[25] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 58–66.

[26] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "Lift: Learned invariant feature transform," in *European Conference on Computer Vision*. Springer, 2016, pp. 467–483.

[27] R. Rifkin, G. Yeo, T. Poggio, *et al.*, "Regularized least-squares classification," *Nato Science Series Sub Series III Computer and Systems Sciences*, vol. 190, pp. 131–154, 2003.

[28] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the society for industrial and applied mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[29] M. L. Miller, H. S. Stone, and I. J. Cox, "Optimizing murty's ranked assignment method," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, no. 3, pp. 851–862, 1997.

[30] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[31] D. Y. Kim and M. Jeon, "Data fusion of radar and image measurements for multi-object tracking via Kalman filtering," *Information Sciences*, vol. 278, pp. 641–652, 2014.

[32] G. Y. Kulikov and M. V. Kulikova, "The accurate continuous-discrete extended Kalman filter for radar tracking," *IEEE Transactions on Signal Processing*, vol. 64, no. 4, pp. 948–958, 2015.

[33] R. Altendorfer and S. Wirkert, "Why the association log-likelihood distance should be used for measurement-to-track association," in *2016 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2016, pp. 258–265.

[34] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking," in *European Conference on Computer Vision*. Springer, 2016, pp. 17–35.

[35] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics," *Journal on Image and Video Processing*, vol. 2008, p. 1, 2008.

[36] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 7, pp. 971–987, 2002.

[37] P. Viola, M. Jones, *et al.*, "Rapid object detection using a boosted cascade of simple features," *CVPR (1)*, vol. 1, pp. 511–518, 2001.