2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Daejeon Convention Center
October 9-14, 2016, Daejeon, Korea

Low-Latency Visual Odometry using Event-based Feature Tracks

Beat Kueng, Elias Mueggler, Guillermo Gallego and Davide Scaramuzza

***Abstract*— New vision sensors, such as the Dynamic and Active-pixel Vision sensor (DAVIS), incorporate a conventional camera and an event-based sensor in the same pixel array. These sensors have great potential for robotics because they allow us to combine the benefits of conventional cameras with those of event-based sensors: low latency, high temporal resolution, and high dynamic range. However, new algorithms are required to exploit the sensor characteristics and cope with its unconventional output, which consists of a stream of asynchronous brightness changes (called "events") and synchronous grayscale frames. In this paper, we present a low-latency visual odometry algorithm for the DAVIS sensor using event-based feature tracks. Features are first detected in the grayscale frames and then tracked asynchronously using the stream of events. The features are then fed to an event-based visual odometry algorithm that tightly interleaves robust pose optimization and probabilistic mapping. We show that our method successfully tracks the 6-DOF motion of the sensor in natural scenes. This is the first work on event-based visual odometry with the DAVIS sensor using feature tracks.**

Multimedia Material

A video attachment to this work is available on the authors' webpage.

I. Introduction

Vision systems for robotics are currently dominated by methods designed for conventional, frame-based cameras, which acquire entire images of the scene at fixed rates.

Recently, bio-inspired silicon retinas [1], [2] have been developed to overcome some of the limitations of frame-based cameras. These sensors constitute a paradigm shift since they operate asynchronously, transmitting only the information conveyed by brightness changes in the scene ("events"), at the time they occur with microsecond resolution. Event-driven algorithms have been developed to provide initial solutions to some robotics problems such as pose tracking [3], [4], visual odometry [5], Simultaneous Localization and Mapping (SLAM) [6], [7]. However, some of these approaches used additional sensors, such as depth sensors [5], [7], or were developed for high-contrast scenes [3], [4], [5].

The Dynamic and Active-pixel Vision Sensor (DAVIS) [8] has been introduced very recently (2014). It is an integrated sensor comprising a conventional frame-based camera and an asynchronous event sensor. This novel hybrid sensor calls for new methods that exploit the combined advantages of event and frame sensors to yield better solutions for robotics problems than those provided by each sensor individually. Such new methods must address the challenges that this sensor poses: it has a complicated analog circuitry, with non-linearities and multiple biases that can change the sensitivity of the pixels, and other dynamic properties, which unfortunately make the frames and events highly susceptible to noise.

The authors are with the Robotics and Perception Group, University of Zurich, Switzerland—http://rpg.ifi.uzh.ch. This research was supported by the DARPA FLA Program, the National Centre of Competence in Research Robotics (NCCR) and the UZH Forschungskredit.

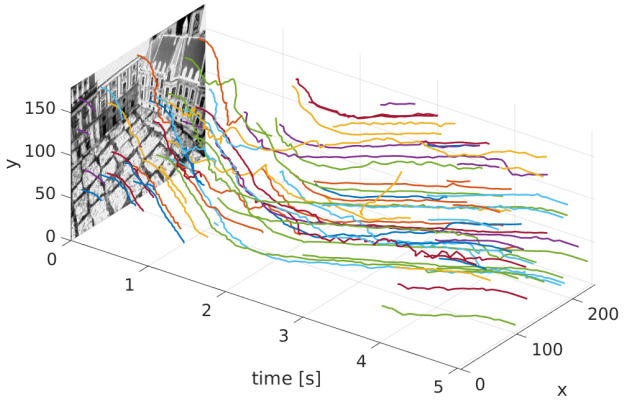


Fig. 1: Space-time view in the image plane of the tracked features' trajectories used for visual odometry. Features are tracked using the events produced by the DAVIS during arbitrary motion in a scene with natural textures. This example shows a back-and-forth dominant translation with a short pause in the middle. Events are not displayed to avoid cluttering. A sample of the events produced by the DAVIS in visual odometry applications is shown in Fig. 2

Contribution: In this paper, we present a low-latency visual odometry algorithm for the DAVIS sensor using event-based feature tracks. We make use of both the frames and the events provided by the sensor. More specifically, we achieve visual odometry using the geometric information conveyed by edge-like features that are adapted to the DAVIS characteristics and represent natural brightness patterns in the scene. Features are first detected using the frames and then tracked asynchronously using the events (see Fig. 1). Next, they are fed to an event-based visual odometry (VO) algorithm that computes a local probabilistic 3D map of the scene and tracks the 6 degree-of-freedom (DOF) pose of the sensor by robust reprojection error minimization. Pose updates are event-based, thus preserving the asynchronous and low-latency nature of the event data.

Outline: The remainder of the paper is organized as follows. Section II describes the sensor used. Section III reviews related literature on event-based feature tracking and event-driven motion estimation methods. Our approaches to feature tracking (2D) and visual odometry (3D) are described

978-1-5090-3762-9/16/$31.00 ©2016 IEEE

in Sections IV and V, respectively, and they are empirically evaluated in Section VI. Conclusions are highlighted in Section VII.

II. The Dynamic and Active-pixel Vision Sensor

The first event-based sensor, called the Dynamic Vision Sensor (DVS) [1], became commercially available in 2008. The DAVIS [8] is a novel vision sensor combining a conventional frame-based camera and a DVS in the same array of pixels. The global-shutter frames provide absolute illumination on demand, whereas the event sensor responds asynchronously to pixel-level brightness changes, independently for each pixel. If $I(t)$ is the illumination sensed at pixel (x, y), an event is triggered if the relative brightness change exceeds a global threshold. More specifically, an event is a tuple $e = (x, y, t, p)$ that conveys the spatio-temporal coordinates (x, y, t) and sign (i.e., polarity $p = \pm 1$) of the brightness change. Events are time-stamped with microsecond resolution and transmitted asynchronously when they occur and with very low latency (microseconds). The DAVIS has a very high dynamic range ($130\,\mathrm{dB}$) compared with the $70\,\mathrm{dB}$ of high-quality, traditional image sensors. The low latency, the high temporal resolution, and the very high dynamic range make the DAVIS extremely advantageous for future robotic applications.

A visualization of the DAVIS output is shown in Fig. 2. The spatial resolution of the DAVIS is 240×180 pixels. This is still small compared to the spatial resolution of state-of-the-art conventional cameras. Newer sensors, such as the color DAVIS [9] will have higher spatial resolution (640×480 pixels), thus overcoming current limitations.

Optically, the lenses mounted on the DAVIS are the same as those mounted on conventional cameras. Having the grayscale and DVS pixels perfectly aligned in the DAVIS simplifies camera calibration, an essential stage in robotics applications. State-of-the-art algorithms for conventional cameras can be applied on the frames alone to calibrate the sensor.

III. Related Work

We first review the related works on event-based feature tracking and then on event-based motion estimation.

A. Event-based Feature Detection and Tracking

Feature detection and tracking methods for frame-based cameras are well established. However, they cannot track in the blind time between consecutive frames, and are expensive because they process information from all pixels, even in the absence of motion in the scene. Conversely, event-based cameras acquire only relevant information for tracking and respond asynchronously, thus, filling the blind time between consecutive frames. Event-based cameras are particularly suitable for applications in motion analysis and high-speed control [10].

Early event-based feature trackers were very simple and focused on demonstrating the low-latency and low-processing requirements of event-driven systems, hence they

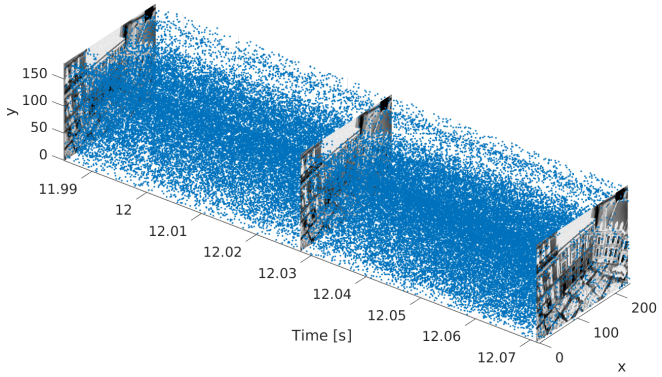


Fig. 2: Typical output of the DAVIS in a visual odometry scenario: as the sensor moves through the scene, it acquires both frames (at low frame rates) and events (asynchronously and fast). Thousands of events are triggered in the time between two frames since, due to the sensor's motion, intensity changes occur at all pixels.

tracked moving objects as clustered blob-like sources of events [11], [10]. The high-speed advantage of event cameras was also shown in [12] for a pencil-balancing robot. Tracking of the pencil was performed using a fast event-based Hough transform. Tracking of large contrast polygonal shapes was demonstrated in [13], where an event-based Iterative Closest Point (ICP) algorithm was able to track a black polygonal microgripper on a white background. The method allows for planar rigid-body transformations of the target shape and uses a nearest-neighbor strategy to match incoming events to the target shape. Tracking of complex shapes has been recently presented in [14] for the ATIS sensor [2]. The method continuously estimates the geometric transformation between the model and the events representing the object using a gradient descent update. It can handle isometries and mild affine distortions. Tracking the translations of arbitrary user-defined shapes ("kernels") has also been presented in [15]. Rotational and scaling distortions of a kernel are partially addressed by comparing the events against a collection of rotated and scaled kernels.

All previous methods require a priori knowledge or user input to determine the objects to track. The method in [16] does not detect and track user-defined objects but lower-level primitives, such as corner events defined by the intersection of two moving edges, which are obtained by fitting planes in the space-time stream of events.

The method of [17] uses both the events and frames to detect and track objects. The events are used to track clusters and generate regions of interest, while the frames serve for foreground-background separation using Convolutional Neural Networks (CNN). Since this classification is only applied to the regions of interest, a speedup factor of 70 is reported. The method requires training data and is only suitable for tracking few, large objects in the scene. For visual odometry, we are rather interested in tracking many local features whose position can be precisely determined.

Recently, we presented a hybrid method for feature de-

tection and tracking for the DAVIS [18]. The method first detects and extracts features in the frames and then tracks them using only the events. In the present paper, we improve [18] by (i) taking into account the observation that nearby pixels typically observe events at roughly the same time and (ii) introducing a tracking refinement step that works on a slower timescale to avoid drift. Furthermore, we improve the tracking speed and add dynamic reinitialization of new features (e.g., in new areas of the scene or when features are lost).

B. Event-based Motion Estimation

Event-based cameras have been used for robotics applications such as ego-motion estimation and visual odometry. One of the first works in this area is [6], where an event-based particle filter was used for robot self-localization. The VO system was limited to planar motion, 2D reference maps with very high contrast, and known scene depth. In the experiments, they used an upward-looking DVS mounted on a ground robot moving at low speed. The method was extended in [7] to a 3D SLAM system that requires an RGB-D sensor operating in parallel with the DVS.

In our previous work [5], an RGB-D camera was attached to the DVS to estimate the relative displacement between the current event and the previous frame of the camera. However, the system was developed for planar 3-DOF motions and for scenes with very high contrast.

In another work, we presented an event-based algorithm to track the 6-DOF pose of the DVS during high-speed motions in a known environment [3]. However, the method was meant for artificial, B&W line-based maps; indeed, the system estimated the pose through minimizing the point-to-line reprojection error.

Tracking the 3D orientation of the DVS and simultaneously using the event stream to generate high-resolution panorama images of natural scenes was presented in [19], [20]. However, the system was restricted to rotational motions, and, thus, did not account for translation or depth.

None of the previous event-based motion estimation methods is based on tracking complex, natural features in the event stream. This is the approach that we develop in this work, as we show next.

IV. Feature Detection and Tracking with the DAVIS

The overall system workflow is shown in Fig. 3. The feature tracking module builds upon our previous work [18], which exploits the absolute brightness information provided by the frames to detect and extract features that are tracked using the event data, as illustrated in Fig. 4. To make this paper self-contained, we summarize the two main steps, feature detection and tracking, in the next two subsections and propose improvements in Section IV-C.

A. Feature Detection using the Frames

Since large contrast edges of moving parts of the scene trigger events more often than low-textured regions, we focus on this dominant information to devise suitable features to track. Moreover, we track only distinctive edge patterns that do not suffer from the aperture problem. Hence, we use the absolute brightness frames of the DAVIS to detect both edges (Canny's method [21]) and corners (Harris detector [22]). Around the most salient and best distributed corners in the frame (Fig. 4a), we use the edge pixels to define patches that mark the locations of the dominant sources of events (Fig. 4b). The patches are converted into binary masks that indicate the presence (1) or absence (0) of an edge. These patches resemble the customized kernels in [15]. The binary masks define the target shapes to be tracked as 2D point sets, called "model point sets" (Fig. 4c). These steps are summarized in the first part of Algorithm 1.

All patches are square and have the same size (Fig. 4b), which is an adjustable parameter, but it is straightforward to extend the method to consider different patch sizes.

Our method does not require frames to be provided at a constant rate since they are only used to initialize features. Frames can be acquired on demand to replace lost features.

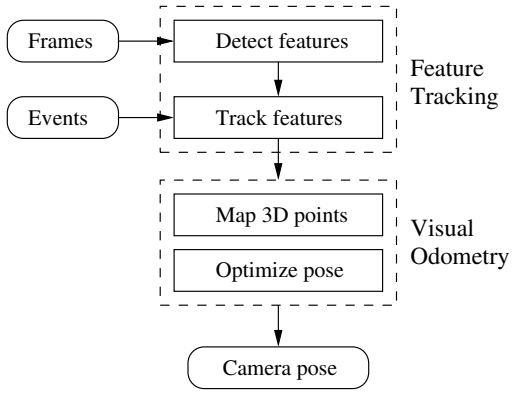


Fig. 3: Visual Odometry system: we track features using events and frames, and then recover the 3D structure of the scene and the DAVIS' pose.

Algorithm 1 High temporal resolution tracking

Feature detection:
- Detect corner points on the frame (Harris detector).
- Run Canny edge detector (returns a binary image, 1 if edge pixel, 0 otherwise).
- Extract local edge-map patches around corner points, and convert them into model point sets.

Feature tracking:
- Initialize the data point set and 2D histograms per patch.

for each incoming event **do**
- Update the corresponding data point set and histograms

  for each corresponding data point set **do**
  - Estimate the registration parameters between the data and the model point sets (weighted ICP).
  - Update registration parameters of the model points.

Every M_2 events, compare spatial histograms, compute and apply the best shift to mitigate drift.

(a) (b) (c)

Fig. 4: Feature detection and tracking. (a) Frame with centers of detected features (green crosses). (b) (1st zoom) edge map and square patches defining the features (in purple). (c) (2nd zoom) point sets used for feature tracking: model point set (in purple) and data point set (in blue).

B. Feature Tracking using the Events

Detected features are tracked using the event stream. The input to the event-based tracking algorithm consists of multiple, local model point sets. The tracking strategy is summarized in the second part of Algorithm 1.

1) Sets of Active Events: For every feature i, we define a data point set of the same size N_p^i as the model point set. Data point sets consist of subsets of the incoming events: an event is inserted in the i-th data point set if its coordinates are inside the i-th patch. A data point set defines the active set of events that are relevant for the registration of the corresponding feature (Fig. 4c). Data point sets are continuously updated: every incoming event replaces the oldest one in the set, and then the registration iteration proceeds. This strategy is event-based, which means that the registration parameters of the tracked feature are updated every time an incoming event is considered relevant for the feature under consideration. The algorithm is asynchronous by design and it can process multiple features simultaneously.

2) Registration: Registration is carried out by minimization of a weighted distance between the model (feature) and the data point sets (events), $\mathbf{m}_i$ and $\mathbf{p}_i$, respectively:

$$\arg\min_{\mathtt{R},\mathbf{t}} = \sum_{(\mathbf{p}_i,\mathbf{m}_i)\in\text{Matches}} b_i \left\|\mathtt{R}\mathbf{p}_i + \mathbf{t} - \mathbf{m}_i\right\|^2, \qquad (1)$$

where a Euclidean transformation $(\mathtt{R},\mathbf{t})$ is assumed (this is enough since the features do not significantly deform in the time between events), and the weights b_i take into account outlier rejection and simultaneous events due to edge structure (Section IV-C). The algorithm that minimizes (1) is a variation of the Iterative Closest Point (ICP) algorithm [23]. It yields the tracking update rule and is directly linked to our choice of feature representation. Each iteration of the algorithm has three stages: first, candidate matches are established, then the geometric transformation is estimated, and, finally, the transformation is applied to the model point set. The operation proceeds until the error difference between two consecutive iterations is below a certain threshold. Matches are established according to the minimum distance criterion, and those above a certain threshold are discarded; hence the method can handle outlier events, as those produced by noise.

Due to the high temporal resolution of the DAVIS, the transformation between consecutive events (in the same feature) is close to the identity and, therefore, our method yields good results even after a single iteration.

C. Tracking Improvements

1) Moving edges produce simultaneous events at neighboring locations: Tracking accuracy is improved by incorporating in the registration criterion the fact that our features are based on edges, which are not isolated points but form connected structures that normally trigger events in neighboring locations at roughly the same time. We do so by using weights b_i in (1) proportional to the number of events, out of the last $N_p^i/4$ of them, that fall in the 3×3 pixel neighborhood of the current event.

2) Tracking refinement based on local 2D histograms of events: We supplement weighted ICP with local 2D histograms designed to improve long-term tracking. Events are accumulated into patch-size histograms over longer times than those used in ICP, which makes them more robust to noise than the point sets. There are two histograms per feature: H_1 over the first M_1 events and a moving histogram H_2 over the last M_2 events. For well-tracked and rotation-compensated features, both histograms look almost identical, thus effectively filtering out noisy events. Otherwise, feature drift is detected as a shift of H_2 with respect to H_1. Every M_2 events, histograms are compared in search for shifts $\mathbf{s} = (o_x, o_y)^\top$ in the range ± 3 pixels in each dimension. The comparison metric is histogram intersection, given by the sum of the minimum of the two histogram values:

$$d(H_1, H_2, \mathbf{s}) = \sum_{\mathbf{x}} \min(H_1(\mathbf{x}), H_2(\mathbf{x}+\mathbf{s})), \qquad (2)$$

where $\mathbf{x} = (x, y)^\top$ iterates over the patch domain. The shift $\mathbf{s}$ with the largest intersection is applied to the feature, if it is larger than a given threshold. Histogram lengths pose a trade-off: the larger they are, the more event noise is filtered; however, this decreases the reaction speed of the algorithm and it can also yield blurred histograms for fast drifting features. The histogram lengths M_1 and M_2 are multiples of the patch size N; a good choice for M_2 is $5N$, while M_1 is set larger than M_2 to ensure a good initial histogram. Figure 5 shows a sequence of histograms with $M_2 = 5N$.

The benefit of this technique on tracking and visual odometry is demonstrated in Section VI-A.

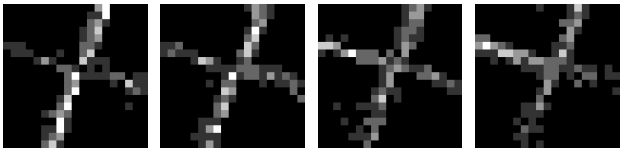


Fig. 5: From left to right: histogram H_1 (initial) and three instances of histogram H_2 as new events arrive.

V. Visual Odometry

The proposed visual odometry (VO) algorithm in Fig. 3 uses the 2D geometric information provided by the feature tracks to estimate the 3D structure of the scene and the location of the DAVIS. These two operations (localization and mapping) are performed in a tightly interleaved manner. We use depth-filters [24] to estimate the scene structure in a Bayesian way, and since we track in the order of one hundred features, the resulting probabilistic map is a sparse representation of the scene. The camera motion is tracked by minimization of a weighted reprojection error using the Gauss-Newton method, which is very fast since the motion between two events is almost zero. Both operations are adapted from the Semi-direct Visual Odometry (SVO) algorithm [24].

A. 3D Mapping using Depth Filters

During mapping, we estimate the depth of 2D features for which the corresponding 3D point is not yet known. The depth estimate of a feature is modeled with a probability distribution that is updated in a Bayesian framework (see Fig. 6) [24]. This is known as a depth-filter. When a depth-filter has converged, that is, when the variance of the distribution becomes small enough, a new 3D point is inserted in the map at the converged depth and it is immediately used for pose tracking.

More specifically, depth-filters are initialized with a high uncertainty and set to the mean depth of the scene. Feature tracks provide depth measurements that are processed by the filter. Each measurement $\tilde{d}_i^k$ (k-th observation of the i-th feature) is obtained by triangulation of the current feature location and its first detection, using the relative camera pose $\mathbf{T}_{r,k}$ (see Fig. 6). $\tilde{d}_i^k$ is modeled using a Gaussian + Uniform mixture [25]: good measurements are normally distributed around the true depth d_i, with variance τ_i^2, while outliers are uniformly distributed in the known range $[d_i^{\min}, d_i^{\max}]$,

$$p(\tilde{d}_i^k | d_i, \rho_i) = \rho_i \mathcal{N}(\tilde{d}_i^k | d_i, \tau_i^2) + (1-\rho_i)\mathcal{U}(\tilde{d}_i^k | d_i^{\min}, d_i^{\max}),$$

where ρ_i is the inlier probability. Well-tracked features are those with ρ_i close to 1. Further details on the filter update equations can be found in [25], however, note that we use inverse depth coordinates, as in [24].

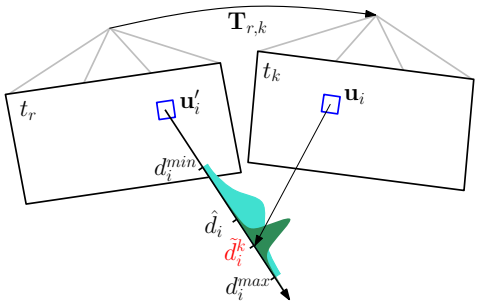


Fig. 6: Depth-filter update for a new measurement $\tilde{d}_i^k$, at current time t_k, of a feature that was extracted at reference time t_r. Over time the uncertain distribution (cyan) becomes narrower for an inlier (green). Image courtesy of [24].

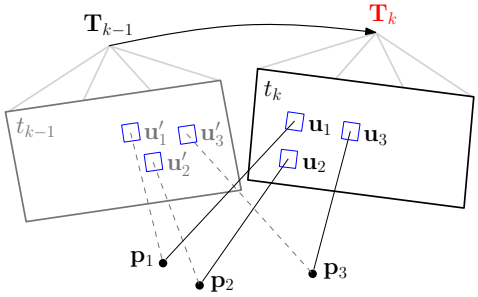


Fig. 7: The Gauss-Newton optimizer finds the new pose $\mathbf{T}_k$ at time t_k from 2D-3D feature correspondences $\mathbf{u}_i \leftrightarrow \mathbf{p}_i$ and the initial guess $\mathbf{T}_{k-1}$. Image courtesy of [24].

B. Pose Tracking by Reprojection Error Minimization

Given a sparse map of the scene, we obtain the current camera pose $\mathbf{T}_k$ by minimizing the reprojection error:

$$\mathbf{T}_k = \underset{\mathbf{T}}{\operatorname{argmin}} \tfrac{1}{2} \sum_i w_i \|\mathbf{u}_i - \pi(\mathbf{T}, \mathbf{p}_i)\|^2, \tag{3}$$

where $\mathbf{u}_i$ and $\mathbf{p}_i$ are the 2D and 3D positions of the i-th feature, as illustrated in Fig. 7, w_i are weights, and π projects 3D world points into the camera frame. This is solved iteratively with the Gauss-Newton method. For robustness against outliers, we use the bell-shaped Tukey weight function

$$w_i = \begin{cases} \left(1 - \frac{x^2}{b^2}\right)^2 & |x| \leq |b|, \\ 0 & \text{otherwise}, \end{cases} \tag{4}$$

with $x = \|\mathbf{u}_i - \pi(\mathbf{T}, \mathbf{p}_i)\|$ and $b = 5$ (pixels). Additionally we have found that by multiplying w_i by the inlier probability ρ_i and the normalized feature age (the number of events that fall into the patch), the results are significantly better. Features that are well-tracked over a long time are given the highest weight, while features that lose track are usually removed due to a large reprojection error.

C. Bootstrapping

Pose optimization relies on the availability of a map, and mapping relies on the availability of pose information. But neither of them are available at the beginning, hence we need to provide an initial map and pose estimate. We use two-view bootstrapping with a minimum mean disparity between

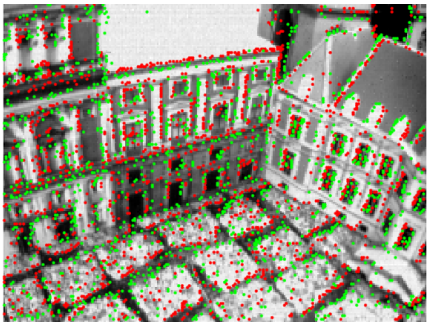


Fig. 8: Sample output of the DAVIS: frame with overlaid events, colored according to polarity (positive in green, negative in red), from a $0.5\,\mathrm{ms}$ interval.

the features. The relative camera pose is calculated from 2D point correspondences with the five-point algorithm for the essential matrix [26] and RANSAC [27]. This requires an initial camera motion with a translational component.

VI. Experiments

We evaluated the performance of the event-based VO system on several scenes with natural textures, rich in brightness changes of different magnitudes (e.g., Fig. 8). The resulting camera trajectory is compared against those acquired by a motion-capture system and a frame-based VO algorithm based on the state of the art [24]. No constraints were placed on the sensor's motion: the DAVIS was freely moved by hand through the scene.

A. Feature Tracking

A space-time visualization of the trajectories described by the tracked features in the image plane is displayed in Fig. 1. Qualitatively, it shows that the tracked features' trajectories have a coherent motion.

We evaluate our event-based feature tracker on two different scenes: a checkerboard-like scene (because it offers well-localized features) and a natural scene (with less localized features, Fig. 8). The results are reported in Fig. 9 and Fig. 10, respectively, in terms of tracking error vs. time. The tracking error is computed against ground truth. Ground truth was generated using a frame-based Lucas-Kanade tracker [28] and linearly interpolating the feature motion in the time interval between frames. Features were detected in the first frame ($t = 0$) and then tracked over the entire sequence using only events. In the checkerboard-like scene (Fig. 9), the mean tracking error is 1.5 pixels. In the natural scene (Fig. 8), the tracking accuracy gracefully degrades, yielding a mean tracking error of 2.5 pixels. This degradation results from two causes: (i) we do not model many of the non-linearities of the DAVIS, such as non-white noise and other dynamic properties; (ii) the detected features are based on a binary edge-map of the scene (resulted from the Canny detector), but such binary map is an exact representation of the underlying grayscale scene only if the contrast is very large. In natural scenes, edges can have all sort of different magnitudes, but our features still track the most dominant ones. We used patches of 19×19 pixels, which

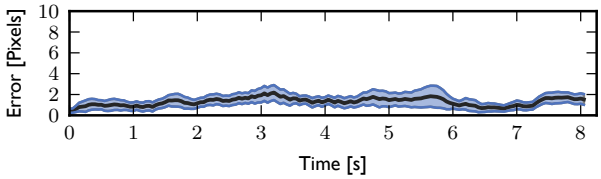


Fig. 9: Feature tracking error of our event-based algorithm on the checkerboard-like scene. The mean tracking error of all features is marked in black. The blue bands around the mean indicate the ± 1 standard-deviation confidence interval. The overall mean error (in black) is $1.5\,\mathrm{pixels}$.

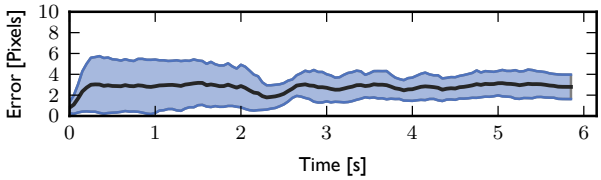


Fig. 10: Feature tracking error of our event-based algorithm on the natural scene in Fig. 8. The mean tracking error of all features is marked in black. The blue bands around the mean indicate the ± 1 standard-deviation confidence interval. The overall mean error (in black) is $2.5\,\mathrm{pixels}$.

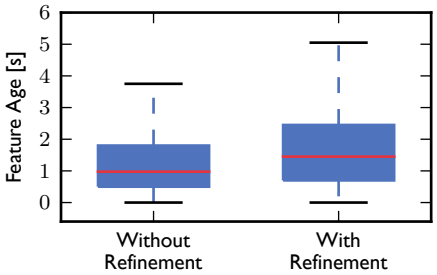


Fig. 11: Long-term tracking improves by using local spatial histograms of events. With them, the "feature age" distribution shifts toward higher values; e.g., the median increases from $1.0\,\mathrm{s}$ to $1.5\,\mathrm{s}$.

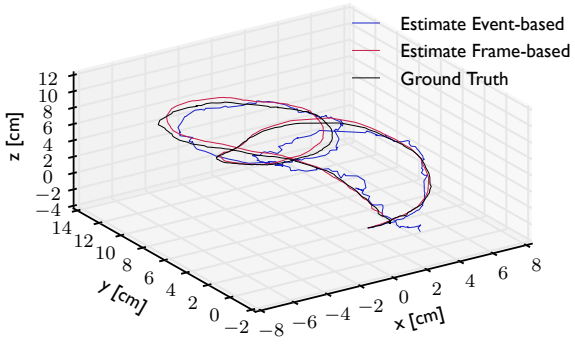
were experimentally proven to be best for a broad class of scenes.

The positive effect of the feature refinement technique described in Section IV-C is shown in Fig. 11. Feature refinement increases the duration of the feature tracks (called "feature age"). This has also a positive effect on the performance of the VO algorithm since points that are tracked for longer times imply higher VO accuracy.

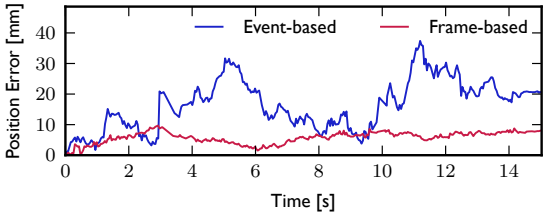
B. Visual Odometry

The VO system tracks a constant number of features (120) that are chosen to be well distributed in the image plane by means of a grid/binning strategy. Using fewer than 100 features does not yield good results. New features are initialized when features are lost or leave the field of view (FOV) of the DAVIS.

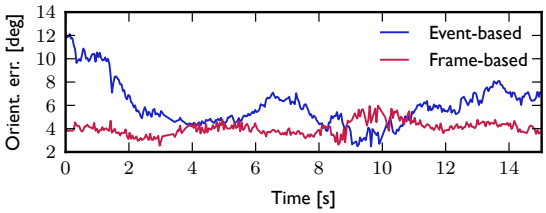
Figs. 12 and 13 show the VO results on two sequences. The figures show the trajectory produced by our VO algorithm, the ground truth (motion-capture system), and the trajectory produced by a frame-based solution, as well as



(a) 3D view of camera trajectories

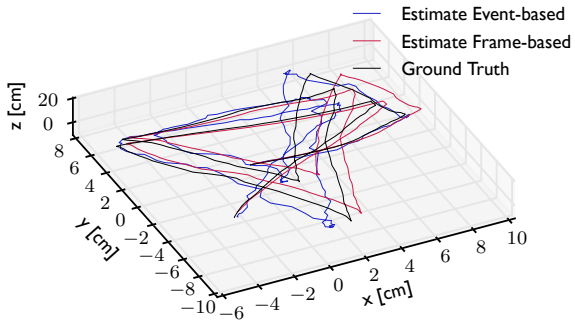


(b) Position error of the estimated trajectories (event-based and frame-based) with respect to ground truth. The mean scene depth is 40 cm.

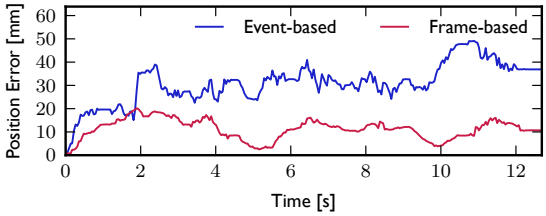


(c) Orientation error of the estimated trajectories (event-based and frame-based) with respect to ground truth.

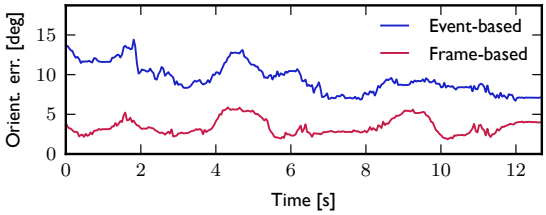
Fig. 12: *VO Experiment 1*. Comparison of DAVIS trajectories estimated by our event-based VO algorithm, a frame-based solution, and ground truth (motion-capture system).



(a) 3D view of camera trajectories



(b) Position error of the estimated trajectories (event-based and frame-based) with respect to ground truth. The mean scene depth is 40 cm.



(c) Orientation error of the estimated trajectories (event-based and frame-based) with respect to ground truth.

Fig. 13: *VO Experiment 2*. Comparison of DAVIS trajectories estimated by our event-based VO algorithm, a frame-based solution, and ground truth (motion-capture system).

the corresponding errors in position and orientation (with respect to ground truth). The position error is given by the Euclidean distance between camera locations. The orientation error is given by the angle of the relative rotation between camera reference frames, which is the geodesic distance in SO(3) [29]. In Experiment 1 (Fig. 12), the average position errors of our method and the frame-based one are 16 and 6 mm, respectively. Since the mean scene depth is 40 cm, this corresponds to relative errors of 4.0 % and 1.5 %, respectively. In Experiment 2 (Fig. 13), these errors are 30 and 11 mm (7.5 % and 2.8 % of the mean scene depth), respectively. Overall, the quality of the estimate produced by our event-based algorithm is comparable to that of the frame-based solution, but provides low-latency pose updates since it preserves the event-based nature of the data. These results are very promising and represent a first step towards a fully integrated frame-plus-events feature tracking and VO solution with this novel sensor in natural scenes and in 6-DOF motions, which are challenging conditions that have not been addressed in previous work.

C. Runtime Analysis

The runtime of event-based algorithms depends on the event rate, which itself depends on several factors: the apparent speed of moving objects in the scene, the amount of texture and edges, the sensor parameters (bias configuration), etc. Roughly speaking, the DAVIS generates 10^5–10^6 events/s for normal motions. For faster motion, it is in the order of a few millions. We tested the implementation on a laptop with Intel Core i7-4710MQ CPU @ 2.50GHz with 8GB RAM. The C++ code runs completely single threaded. Our algorithm is able to process 160 kevents/s on average. The performance analysis is shown in Fig. 14. Running ICP on every incoming event is excessive since an event does not have a large influence. We experimentally found that running ICP every $N/3$ events (N being the size of the point sets) speeds up the VO algorithm by a factor of 6 while its accuracy is preserved. Running ICP only every N events increases the error. In spite of this speed-up, a significant portion of the computational time (58.5%, see Fig. 14) is spent in an off-the-shelf ICP library [30] that can match

point sets in arbitrary dimensions. By using a customized implementation, runtime could be improved.

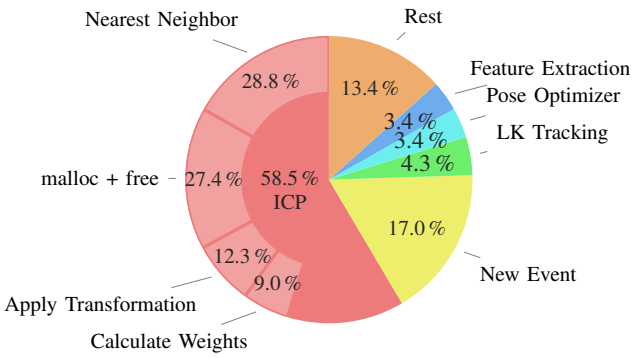


Fig. 14: Runtime analysis of the VO system in Fig. 3, calling ICP every $N/3$ incoming events. "New Event" checks whether a new event is within a patch and updates the target point set accordingly. "LK Tracking" is not part of our algorithm and is only used for comparison. Feature extraction and pose tracking are very efficient. Percentages within ICP are relative to the aggregated cost (58.5%).

VII. Conclusion

We have developed a low-latency, event-based visual odometry algorithm adapted to the characteristic of the DAVIS, a prototype sensor with great potential for robotics, which combines a conventional camera and an event-based sensor in the same pixel array. Our method extracts visual features in the frames and tracks them asynchronously using the events. Features are designed to be trackable using only the events and are sufficiently generic to be applicable to non-structured environments. We used two cooperative techniques to achieve event-based tracking: a weighted point-set minimization for short-term tracking and comparison of spatial histograms of events for long-term tracking. Feature tracks were used to infer 3D quantities, using probabilistic depth-filters for mapping and robust reprojection error minimization for pose tracking. We demonstrated successful tracking and VO performance of a moving DAVIS in 6-DOF and in scenes with natural textures, which are challenging conditions that have not been previously addressed in the literature.

References

[1] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128×128 120 dB 15 μs latency asynchronous temporal contrast vision sensor," *IEEE J. of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[2] C. Posch, D. Matolin, and R. Wohlgenannt, "A QVGA 143 dB Dynamic Range Frame-Free PWM Image Sensor With Lossless Pixel-Level Video Compression and Time-Domain CDS," *IEEE J. of Solid-State Circuits*, vol. 46, no. 1, pp. 259–275, Jan 2011.

[3] E. Mueggler, B. Huber, and D. Scaramuzza, "Event-based, 6-DOF pose tracking for high-speed maneuvers," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2014.

[4] E. Mueggler, G. Gallego, and D. Scaramuzza, "Continuous-time trajectory estimation for event-based vision sensors," in *Robotics: Science and Systems (RSS)*, 2015.

[5] A. Censi and D. Scaramuzza, "Low-latency event-based visual odometry," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014.

[6] D. Weikersdorfer and J. Conradt, "Event-based particle filtering for robot self-localization," in *IEEE Int. Conf. on Robotics and Biomimetics (ROBIO)*, 2012.

[7] D. Weikersdorfer, D. B. Adrian, D. Cremers, and J. Conradt, "Event-based 3D SLAM with a depth-augmented dynamic vision sensor," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Jun. 2014.

[8] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240x180 130dB 3us latency global shutter spatiotemporal vision sensor," *IEEE J. of Solid-State Circuits*, vol. 49, no. 10, 2014.

[9] C. Li, C. Brandli, R. Berner, H. Liu, M. Yang, S. Liu, and T. Delbruck, "An RGBW Color VGA Rolling and Global Shutter Dynamic and Active-Pixel Vision Sensor," in *Int. Image Sensor Workshop (IISW)*, Vaals, Netherlands, June 2015.

[10] T. Delbruck and P. Lichtsteiner, "Fast sensory motor control based on event-based hybrid neuromorphic-procedural system," in *Int. Conf. on Circuits and Systems (ISCAS)*, May 2007, pp. 845–848.

[11] M. Litzenberger, C. Posch, D. Bauer, A. Belbachir, P. Schon, B. Kohn, and H. Garn, "Embedded vision system for real-time object tracking using an asynchronous transient vision sensor," in *IEEE 12th Digital Signal Proc. Workshop, 4th IEEE Signal Proc. Education Workshop*, Sept 2006, pp. 173–178.

[12] J. Conradt, M. Cook, R. Berner, P. Lichtsteiner, R. Douglas, and T. Delbruck, "A pencil balancing robot using a pair of AER dynamic vision sensors," in *Int. Conf. on Circuits and Systems (ISCAS)*, 2009.

[13] Z. Ni, A. Bolopion, J. Agnus, R. Benosman, and S. Regnier, "Asynchronous event-based visual shape tracking for stable haptic feedback in microrobotics," *IEEE Trans. Robotics*, vol. 28, 2012.

[14] Z. Ni, S.-H. Ieng, C. Posch, S. Regnier, and R. Benosman, "Visual Tracking Using Neuromorphic Asynchronous Event-Based Cameras," *Neural Computation*, vol. 27, pp. 925–953, 2015.

[15] X. Lagorce, C. Meyer, S.-H. Ieng, D. Filliat, and R. Benosman, "Asynchronous event-based multikernel algorithm for high-speed visual features tracking," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, pp. 1710–1720, 2015.

[16] X. Clady, S.-H. Ieng, and R. Benosman, "Asynchronous event-based corner detection and matching," *Neural Networks*, vol. 66, 2015.

[17] H. Liu, D. P. Moeys, G. Das, D. Neil, S.-C. Liu, and T. Delbruck, "Combined frame- and event-based detection and tracking," in *Int. Conf. on Circuits and Systems (ISCAS)*, 2016.

[18] D. Tedaldi, G. Gallego, E. Mueggler, and D. Scaramuzza, "Feature detection and tracking with the dynamic and active-pixel vision sensor (DAVIS)," in *Int. Conf. on Event-Based Control, Comm. and Signal Proc. (EBCCSP)*, Krakow, Poland, Jun. 2016.

[19] H. Kim, A. Handa, R. Benosman, S.-H. Ieng, and A. J. Davison, "Simultaneous mosaicing and tracking with an event camera," in *British Machine Vision Conf. (BMVC)*, 2014.

[20] M. Cook, L. Gugelmann, F. Jug, C. Krautz, and A. Steger, "Interacting maps for fast visual interpretation," in *Int. Joint Conf. on Neural Networks (IJCNN)*, 2011, pp. 770–776.

[21] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Machine Intell*, vol. 8, no. 6, pp. 679–698, Nov 1986.

[22] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of The Fourth Alvey Vision Conference*, vol. 15. Manchester, UK, 1988, pp. 147–151.

[23] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Machine Intell*, vol. 14, pp. 239–256, 1992.

[24] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014, pp. 15–22.

[25] G. Vogiatzis and C. Hernández, "Video-based, real-time multi view stereo," *Image Vision Comput.*, vol. 29, no. 7, pp. 434–441, 2011.

[26] D. Nister, "An efficient solution to the five-point relative pose problem," *IEEE Trans. Pattern Anal. Machine Intell*, vol. 26, no. 6, 2004.

[27] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, pp. 381–395, 1981.

[28] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Int. Joint Conf. on Artificial Intelligence*, 1981, pp. 121–130.

[29] D. Q. Huynh, "Metrics for 3D rotations: Comparison and analysis," *J. of Math. Imaging Vis.*, vol. 35, no. 2, pp. 155–164, 2009.

[30] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets," *IEEE Aerospace Conf.*, vol. 34, no. 3, pp. 133–148, Feb. 2013.