This paper has been accepted for publication at the
IEEE Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, 2022. ©IEEE

Event-aided Direct Sparse Odometry

Javier Hidalgo-Carrió[1], Guillermo Gallego[2], Davide Scaramuzza[1]
[1]Dept. of Informatics, Univ. of Zurich and Dept. of Neuroinformatics, Univ. of Zurich and ETH Zurich.
[2]Technische Universität Berlin, Einstein Center Digital Future and SCIoI Excellence Cluster, Germany.

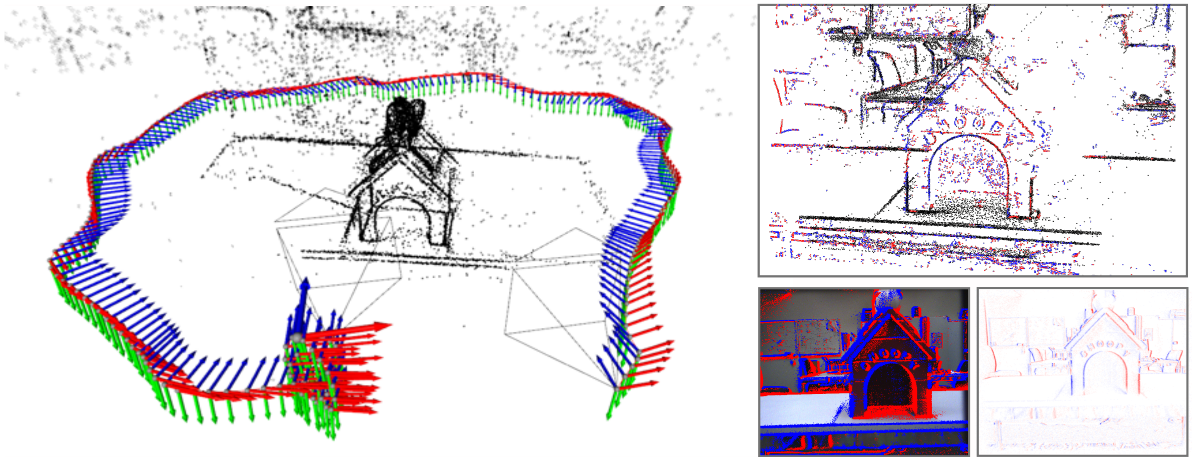

Figure 1. Camera trajectory and estimated 3D map (left). The top-right inset shows the sliding window map (grayscale points) with the current keyframe map (pseudo-colored blue-red points, according to event polarity). The bottom-right insets show the color image (frame) with the real events (left) and the image obtained by the event generative model (right).

Abstract

We introduce EDS, a direct monocular visual odometry using events and frames. Our algorithm leverages the event generation model to track the camera motion in the blind time between frames. The method formulates a direct probabilistic approach of observed brightness increments. Per-pixel brightness increments are predicted using a sparse number of selected 3D points and are compared to the events via the brightness increment error to estimate camera motion. The method recovers a semi-dense 3D map using photometric bundle adjustment. EDS is the first method to perform 6-DOF VO using events and frames with a direct approach. By design it overcomes the problem of changing appearance in indirect methods. Our results outperform all previous event-based odometry solutions. We also show that, for a target error performance, EDS can work at lower frame rates than state-of-the-art frame-based VO solutions. This opens the door to low-power motion-tracking applications where frames are sparingly triggered "on demand" and our method tracks the motion in between. We release code and datasets to the public.

Multimedia Material

Code and dataset can be found at: https://rpg.ifi.uzh.ch/eds

1. Introduction

Visual Odometry (VO) is a paramount tool in computer vision, robotics and, any application that requires spatial reasoning [1–3]. In recent years, considerable progress has been made on this topic [1, 4–6]. However, VO systems are limited by the capabilities of their physical devices (sensors, processors, and power). Some of these limitations (e.g., motion blur, dynamic range) can been tackled with novel and/or more robust sensors, such as event cameras.

Event cameras [7–9] are bio-inspired sensors that work radically different from traditional cameras. Instead of capturing brightness images at a fixed rate, they measure asynchronous, per-pixel brightness changes, called "events".[1] This principle of operation endows event cameras with outstanding properties, such as low latency, high temporal resolution (in the order of µs) and low power (milliwatts instead of watts). The large potential of event cameras to tackle VO and related problems in challenging scenarios has been investigated in [10–21]. We refer to a recent comprehensive survey paper for further details [22].

Event-based VO is a challenging problem that has been addressed step-by-step in scenarios with increasing complexity. Two fundamental *challenges* arise when working

[1] See an illustrative animation: https://youtu.be/LauQ6LWTkxM?t=30

with event cameras: noise (caused by timestamp jitter, pixel manufacturing mismatch or non-linear circuity effects) and data association [22, 23], i.e., establishing correspondences between events to identify which events are triggered by the same scene point. This is so because each event carries little information and the temporal edge patterns conveyed by the events depend on motion. These two issues make event-based keypoints used by indirect methods difficult to detect and track with sufficient stability; for this reason, grayscale frames [14] or motion compensation and inertial sensor fusion [18] have been used to mitigate their effect.

Event-based methods might be categorized according to whether they exploit the Event Generation Model (EGM) [13, 24] or not [14, 17, 25, 26]. The EGM states how events are created when a predefined contrast threshold is reached [7, 22]. It is a photometric relationship between brightness "changes" (i.e., events) and "absolute" brightness. Experiments on event-based feature tracking [23] have shown that methods exploiting the EGM achieve higher accuracy than those that do not. However such a comparison is yet to be performed for 6-DOF camera tracking (i.e., ego-motion estimation). Current solutions [13, 17] are prone to lose tracking, either because the convergence of the estimated 3D map is slow [13] or because the edge-patterns change quickly from one packet of events to the next one [17]. That is, there is no long-term appearance, like the grayscale frames in [23], to latch onto and improve tracking robustness, which we intend to do.

We propose to tackle the event-based VO problem by understanding and overcoming the shortcomings of previous methods. To the best of our knowledge, this work is the first monocular method to perform 6-DOF VO using events and frames with a direct approach. Our contribution lies in the front-end, fusing the information from events and frames tightly using the EGM (as opposed to previous works that loosely coupled them [18]). Estimated poses, points, and selected frames are fed into a sliding-window photometric bundle adjustment (PBA) back-end, which minimizes brightness errors. This is the first time that PBA is used in the context of event-camera VO. Internally, we adopt a key-frame-based approach, recovering inverse depth at pixels with high gradient as in DSO [5]. However, our method allows tracking the camera motion in the blind time between frames using only events. This opens the door for frames to be triggered *sparingly*, i.e., "on demand", thus potentially saving energy in the system, which is a desirable feature in AR/VR applications and in platforms with a limited power budget. Our contributions are summarized as follows:

- The first formulation of a monocular 6-DOF visual odometry combining events and grayscale frames in a direct approach with photometric bundle adjustment.
- Camera motion tracking using a sparse set of pixels, minimizing the normalized brightness change of projected (inverse depth) points.
- A compelling evaluation on publicly available datasets outperforming previous solutions, and a sensitivity study to gain insights about our method.
- A new dataset with high quality events, color frames, and IMU data to foster research in monocular VO.

	Events	**Frames**	**D/I**	**EGM**	**Remarks**
Kim et al. [13]	✓	✗	D	✓	Three parallel EKFs
Rebecq et al. [17]	✓	✗	D	✗	Parallel tracking and mapping
Kueng et al. [14]	✓	✓	I	✗	Tracking event features for VO
Rosinol et al. [18]	✓	✓	I	✗	Loosely coupled front-end
This work	✓	✓	D	✓	Tightly coupled front-end

Table 1. *Comparison of event-based monocular 6-DOF VO/SLAM methods*. The columns indicate the type of input (events and/or grayscale frames), the type of method (**D**irect or **I**ndirect), and whether the method exploits the event generation model (EGM).

2. Related Work

Event-based VO methods might be *direct* or *indirect* depending on whether they process raw pixel information *directly* or *indirectly* via some intermediate representation. Indirect methods [14, 15], like frame-based approaches, extract keypoints [27, 28] from the input (event) data in the front-end before passing them to the back-end. Direct methods [13, 17] attempt to directly process all events available. Since events correspond to per-pixel brightness changes, they naturally convey the information about the motion of the scene edges (assuming constant illumination). Early event-based VO works [11, 24, 29] tackled simple camera motions, such as 3-DOF (planar or rotational), and hence did not account for depth. The most general case of a freely moving camera (6-DOF) has been tackled only recently [13, 14, 17]. In terms of scene texture, high contrast and/or structured scenes have been addressed before focusing on more difficult, natural 3D scenes. Event-based VO is still in its infancy, with the majority of works addressing only camera tracking [12, 16, 20, 30–32] because of its simplicity. Table 1 summarizes the related work on event-based monocular 6-DOF tracking and mapping.

Why direct methods with events? Feature-based methods work well for standard cameras [33], where features are mature and easy to detect and track due to small intensity noise. However they are surpassed in accuracy by direct methods [5] (which use all data available, even pixels that do not comply with the definition of a feature). In event cameras, features are not easy to detect and track because the event "appearance" depends on motion (and texture) [22]. There are many event-based feature detectors and/or trackers [27, 28, 34, 35], but their application to enable VO is scarce [14] because they are not as accurate and stable as needed. On the other hand, (i) events are triggered by moving edges (which are semi-dense on the image plane), and (ii) semi-dense methods are state of the art for event-based 3-DOF VO [21,29,36,37]. All these ideas suggest that direct methods are the natural fit for event cameras and should also work well for 6-DOF motion estimation, as hinted by early works [13, 17].

Combining events and frames. Events and intensity frames are complementary sources of visual information [38,39]. Combining them has proven useful to improve accuracy and/or robustness in several applications, such as feature tracking [23], ego-motion estimation [18,32], depth prediction [40], video reconstruction [38, 41] and video frame interpolation [42]. We also seek to get the best of both visual sensors for VO-related tasks, as implied by [18,32]. In contrast to [18], which treats events and frames as unrelated visual sources (i.e., no effort is made in the front-end to fuse the same feature seen by both sensors), we fuse events and frames in a principled way in the VO front-end by exploiting the EGM [23, 43]. Hence, our approach is closer to [32], but the map is not externally given and errors are computed on a sparse set of pixels in the keyframes.

In short, as Table 1 shows, there is a gap in the characteristics of the approaches used to tackle the monocular 6-DOF VO problem, and this work fills that gap: fusing events and frames in the front-end via a direct method to exploit the EGM. Our method includes a back-end with a photometric bundle adjustment [5] adapted to the front-end, differing from the feature-based back-end in [18]. To the best of our knowledge, this path has not yet been explored.

3. Direct Odometry with Events and Frames

This section describes our method, which is summarized in the block diagram of Fig. 2. First, we review how an event camera works and the EGM (Sec. 3.1). Then we describe the system's front-end (Sec. 3.2), back-end (Sec. 3.3) and initialization (Sec. 3.4).

3.1. Event Generation Model (EGM)

How an Event Camera Works. Each pixel of an event camera produces an event $e_k \doteq (\mathbf{u}_k, t_k, p_k)$ whenever it detects that the logarithmic brightness L at that pixel, $\mathbf{u}_k = (x_k, y_k)^\top$, changes by a specified amount C (called contrast sensitivity) [7]:

$$\Delta L(\mathbf{u}_k, t_k) \doteq L(\mathbf{u}_k, t_k) - L(\mathbf{u}_k, t_k - \Delta t_k) = p_k\, C, \quad (1)$$

where the polarity $p_k \in \{+1, -1\}$ is the sign of the brightness change, and Δt_k is the time elapsed since the last event at the same pixel. Event timestamps t_k have µs resolution. A single pixel has its own sampling rate (which depends on the visual input) and produces events proportionally to the amount of scene motion. An event camera does not output images at a constant rate, but rather a stream of asynchronous events in space-time.

Linearized Event Generation Model. Collecting pixelwise the polarities of a set of events $\mathcal{E} \doteq \{e_k\}_{k=1}^{N_e}$ in the time interval $\mathcal{T} \doteq \{t_k\}_{k=1}^{N_e}$ produces a brightness increment image:

$$\Delta L(\mathbf{u}) = \sum_{t_k \in \mathcal{T}} p_k C\, \delta(\mathbf{u} - \mathbf{u}_k), \quad (2)$$

where the Kronecker δ selects the appropriate pixel. If the number of events N_e spans a small delta time $\Delta t = t_{N_e} - t_1$, the increment (1) can be approximated using Taylor's expansion. Further substituting the brightness constancy assumption gives that ΔL is caused by brightness gradients ∇L moving with velocity $\mathbf{v}$ on the image plane [23,43]:

$$\Delta L(\mathbf{u}) \approx -\nabla L(\mathbf{u}) \cdot \Delta \mathbf{u} = -\nabla L(\mathbf{u}) \cdot \mathbf{v}(\mathbf{u}) \Delta t. \quad (3)$$

3.2. Front-End

In the VO front-end (Fig. 2) we use (2) to create pseudo-measurements from the events (top branch of Fig. 2), and use the right hand side of (3) to predict those from the frame $\hat{L}$ and the current state of the VO system (middle branch of Fig. 2). Our goal is, roughly speaking, to estimate the VO state that best predicts the measurements (Fig. 3). This strategy is described in the upcoming subsections.

Event Weighting. As pointed out in [32], there is a trade-off in the selection of N_e to build (2): a small N_e does not yield sufficient SNR or evidence of existing edge motion, whereas a large N_e produces accumulation blur and breaks the assumption about events being triggered by the camera at a single location. To tackle this issue, we select N_e large to have enough SNR and we modify (2) to accumulate weighted polarities $w_k p_k \leftarrow p_k$. We use Gaussian weights w_k in time index k (top branch in Fig. 2). The weights emphasize the central part of the window of events $\mathcal{E}$, hence producing thinner edges (less accumulation blur) than in the unweighted case ($w_k = 1$).

Events Prediction using Frames. The middle branch of Fig. 2 computes the right hand side of (3) and selects only pixels at scene contours for event prediction. A keyframe in the middle branch of Fig. 2 comprises a brightness frame and a (semi-dense) inverse depth map (Sec. 3.2.2).

The spatial gradient (3) of the logarithmic normalized intensity of the keyframe $\hat{L}$ is computed using the Sobel operator. The image-point velocity $\mathbf{v}$ in (3) is purely geometric, given in terms of the camera pose T, the camera's linear and angular velocities $\dot{T} \doteq (\mathbf{V}^\top, \boldsymbol{\omega}^\top)^\top$, and the depth $d_\mathbf{u} \doteq Z(\mathbf{u})$ of the 3D point with respect to the camera [45]:

$$\mathbf{v}(\mathbf{u}) = J(\mathbf{u}, d_\mathbf{u})\, \dot{T}, \quad (4)$$

where $J(\mathbf{u}, d_\mathbf{u})$ is the 2×6 feature sensitivity matrix

$$J(\mathbf{u}, d_\mathbf{u}) = \begin{bmatrix} \frac{-1}{d_\mathbf{u}} & 0 & \frac{u_x}{d_\mathbf{u}} & u_x u_y & -(1 + u_x^2) & u_y \\ 0 & \frac{-1}{d_\mathbf{u}} & \frac{u_y}{d_\mathbf{u}} & 1 + u_y^2 & -u_x u_y & -u_x \end{bmatrix}. \quad (5)$$

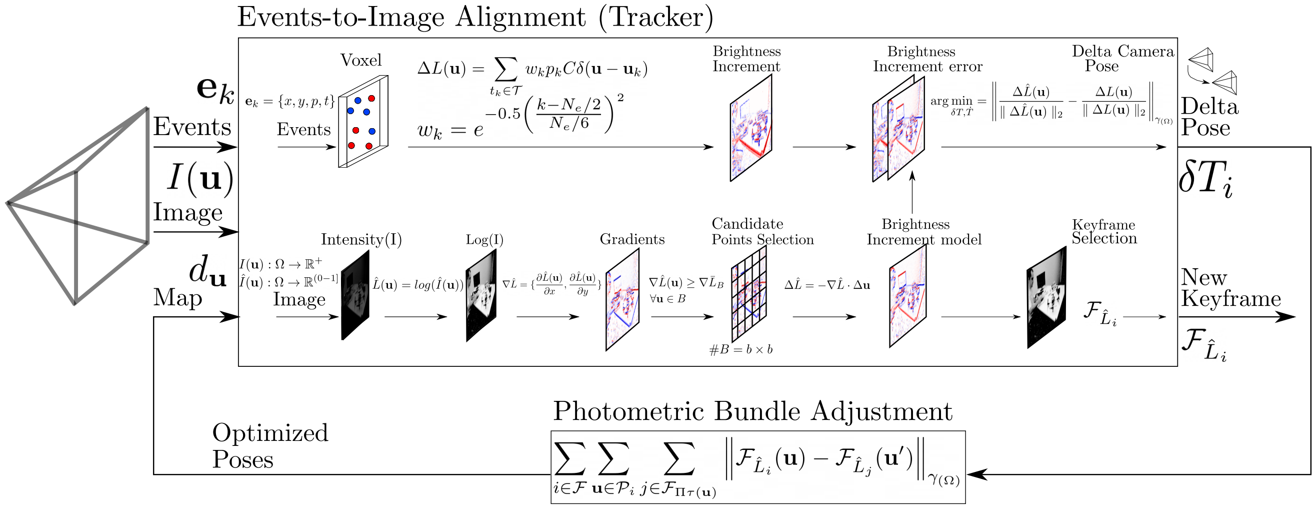
Events-to-Image Alignment (Tracker)
$\mathbf{e}_k$
Events
$I(\mathbf{u})$
Image
$d_{\mathbf{u}}$
Map
Voxel
$\mathbf{e}_k = \{x, y, p, t\}$
Events
$\Delta L(\mathbf{u}) = \sum_{t_k \in \mathcal{T}} w_k p_k C \delta(\mathbf{u} - \mathbf{u}_k)$
$w_k = e^{-0.5\left(\frac{k - N_e/2}{N_e/6}\right)^2}$
Brightness Increment
Brightness Increment error
Delta Camera Pose
$\arg\min_{\delta T, \dot{T}} = \left\| \frac{\Delta \hat{L}(\mathbf{u})}{\| \Delta \hat{L}(\mathbf{u}) \|_2} - \frac{\Delta L(\mathbf{u})}{\| \Delta L(\mathbf{u}) \|_2} \right\|_{\gamma(\Omega)}$
Delta Pose
δT_i
Intensity(I)
Log(I)
Gradients
Candidate Points Selection
Brightness Increment model
Keyframe Selection
$I(\mathbf{u}) : \Omega \to \mathbb{R}^+$
$\hat{I}(\mathbf{u}) : \Omega \to \mathbb{R}^{(0-1]}$
Image
$\hat{L}(\mathbf{u}) = log(\hat{I}(\mathbf{u}))$
$\nabla \hat{L} = \{\frac{\partial \hat{L}(\mathbf{u})}{\partial x}, \frac{\partial \hat{L}(\mathbf{u})}{\partial y}\}$
$\nabla \hat{L}(\mathbf{u}) \geq \nabla \bar{L}_B$
$\forall \mathbf{u} \in B$
$\#B = b \times b$
$\Delta \hat{L} = -\nabla \hat{L} \cdot \Delta \mathbf{u}$
$\mathcal{F}_{\hat{L}_i}$
New Keyframe
$\mathcal{F}_{\hat{L}_i}$
Photometric Bundle Adjustment
$\sum_{i \in \mathcal{F}} \sum_{\mathbf{u} \in \mathcal{P}_i} \sum_{j \in \mathcal{F}_{\Pi \tau(\mathbf{u})}} \left\| \mathcal{F}_{\hat{L}_i}(\mathbf{u}) - \mathcal{F}_{\hat{L}_j}(\mathbf{u}') \right\|_{\gamma(\Omega)}$
Optimized Poses

Figure 2. Block diagram of the proposed event-based direct odometry approach. Events and frames acquired by a camera, such as the DAVIS346 [44], are fed to the front-end, where they are fused using the event generation model (EGM). The front-end selects sparse points on scene edges (i.e., events) with respect to the keyframes. As the camera moves, it generates events, and the camera pose is estimated with respect to the last keyframe. Poses and keyframes are passed to the back-end, which performs non-linear refinement of the poses and depth estimates via photometric bundle adjustment. These are later fed back to the front-end to sustain the good performance of the VO system. Events are colored in blue/red according to polarity p_k, indicating positive/negative brightness increments.

Inserting (4) in (3) gives the predicted brightness change

$$\Delta \hat{L}(\mathbf{u}) \approx -\nabla \hat{L}(\mathbf{u}) \cdot J(\mathbf{u}, d_{\mathbf{u}}) \dot{T} \Delta t. \tag{6}$$

The pose T and velocity $\dot{T}$ are global quantities shared by all image pixels $\mathbf{u}$. The keyframe brightness $\hat{L}$ and the delta time Δt (given by the event timestamps) are known. As noted in the block diagram of Fig. 2, the depth $d_{\mathbf{u}}$ is an input to the front-end, given by the back-end. Initialization of the depth estimates is described in Sec. 3.4.

Candidate Points Selection. Only the most informative pixels of the keyframes are used. This focuses computational resources while maintaining accuracy. Specifically, we select pixels with a sufficiently strong gradient (i.e., contours). To have a good distribution of the selected pixels over the image plane, we follow a tiling approach, dividing the image into rectangular tiles (e.g., 11 × 11 tiles) and selecting a percentage of the pixels with the largest brightness gradient on each tile (typically 10-15% of the image pixels). Note that at a keyframe the pixels with strongest gradients overlap with the pixels where events are triggered (since events are due to moving edges). As the camera moves, these two sets of pixels begin to depart; however, it is through the estimation of the camera motion (Sec. 3.2.1) and the scene depth (Sec. 3.2.2) that we keep them aligned, thus maintaining a correspondence between them.

3.2.1 Camera Tracking

Camera tracking (illustrated on the top right of the front-end block in Fig. 2) is performed with respect to the last keyframe. We split the event stream into packets (i.e., temporal windows), and create event frames (2) using the Gaussian weighting mentioned above. We cast the camera tracking problem as a joint optimization over the camera motion parameters (6-DOF pose and its velocity):

$$(\delta T^*, \dot{T}^*) = \arg\min_{\delta T, \dot{T}} \left\| \frac{\Delta \hat{L}}{\|\Delta \hat{L}\|_2} - \frac{\Delta L}{\|\Delta L\|_2} \right\|_{\gamma}. \tag{7}$$

The error is the Huber norm γ of the difference between normalized brightness increments (ΔL from the events and $\Delta \hat{L}$ from the keyframe). Norms are computed over a sparse set: the above-mentioned selected pixels in the keyframe. Hence, the increments due to the events are transferred to the image plane of the keyframe: (i) first, we find the location $\mathbf{u}_e$ on the event image plane corresponding to a selected keyframe pixel $\mathbf{u}_f$:

$$\mathbf{u}_e = \pi\big(T_{e,f}\, \pi^{-1}\big(\mathbf{u}_f, d_{\mathbf{u}_f}\big)\big), \tag{8}$$

where $\pi^{-1} : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^3$ back-projects a keyframe pixel, $T_{e,f} \in SE(3)$ changes coordinates to the current event camera pose, and $\pi : \mathbb{R}^3 \to \mathbb{R}^2$ projects the point onto the event camera; and (ii) we compute $\Delta L(\mathbf{u}_e)$ by cubic interpolation of the brightness increment (2). The normalized values of $\Delta L(\mathbf{u}_e)$ and $\Delta \hat{L}(\mathbf{u}_f)$ are compared in (7).

The pose $T \in SE(3)$ is parametrized using a 3-vector and a quaternion. The velocity $\dot{T}$ is parametrized using a 6-vector. To minimize (7), we use the Ceres solver [46], which performs local Lie group parametrization of T.

Compared to the camera tracker [32], which projects a global photometric 3D map onto the current event camera location, we compute the error (7): (i) on the keyframe (local depth maps), rather than on the event frame (because we lack dense optical flow from a global 3D map to use in the EGM), (ii) only in a sparse group of pixels (as opposed to the entire image plane) conforming a local semi-dense map.

The output of the front-end is the current keyframe (brightness image and inverse depth map) and the estimated camera motion (relative to the keyframe) of each event packet. These are passed to the back-end for further non-linear refinement (Sec. 3.3).

Keyframe Selection. A keyframe is created when one of two conditions is met: (i) the number of selected points decreases by 20-30% (because they fall out of the field of view (FOV)). (ii) the relative rotation of the event camera with respect to the keyframe exceeds a given threshold.

3.2.2 Mapping (Depth Estimation)

As a new keyframe is created, the inverse depth estimates from past keyframes are used to populate those of the new keyframe. Likewise, the set of selected pixels is transferred to the new keyframe (akin to (8)). The inverse depth values at the remaining pixels are initialized using nearest neighbors with a k-d tree. This is simple and effective. In our VO system, inverse depth estimates are refined in the back-end.

3.3. Back-End (Non-linear Refinement)

The back-end (bottom branch in Fig. 2) performs non-linear refinement of the camera poses and the 3D structure via photometric bundle adjustment (PBA). It minimizes the objective function

$$\sum_{i\in\mathcal{F}} \sum_{\mathbf{u}\in\mathcal{P}_i} \sum_{j\in\mathcal{F}_{\mathit{\Pi}_{\boldsymbol{\tau}}(\mathbf{u})}} \left\| \mathcal{F}_{\hat{L}_i}(\mathbf{u}) - \mathcal{F}_{\hat{L}_j}(\mathbf{u}') \right\|_{\gamma(\mathit{\Omega})}, \qquad (9)$$

where $i \in \mathcal{F}$ runs over all keyframes $\mathcal{F}$, $\mathbf{u}$ runs over all selected pixels $\mathcal{P}_i$ in keyframe i, j runs over all keyframes in which point $\mathbf{u}$ is visible, and $\mathbf{u}'$ is the corresponding point to $\mathbf{u}$ on the j-th keyframe. We confer robustness by using the Huber norm γ, which deemphasizes bad correspondences, and by discarding outliers (based on obnoxiously large errors). Errors are measured around each image point using an 8-pixel patch and assuming the same depth estimate for all pixels in the patch (residual pattern #8 in [5]).

N_k keyframes are kept in a sliding window estimator (we use 7, as in [5], since it is a good accuracy-efficiency compromise). On average, around 2000–8000 points are used in the back-end. We have implemented a PBA using Ceres [46] with automatic differentiation. We have also combined our front-end with DSO's PBA [5], since our design is modular.

3.4. Bootstrapping

To initialize the system, we may try three methods on the frames: (i) classical multi-view geometry, (ii) learning-based monocular depth prediction or (ii) DSO's coarse initializer. The first approach uses the 8-point algorithm [47] for the first frames until the sliding window is full (we skip frames to increase parallax). Once initialization is successful, the frames become keyframes, we select points with large gradient, and perform one PBA step to refine the map. If the above fails (i.e.: planar scenes), we use single image depth prediction via MiDAS [48]. Ultimately, DSO's coarse initializer works the best to initialize. The magnitude of the predicted depth is arbitrary, but this is known in monocular VO since scale is unobservable (a Gauge freedom [49]).

4. Experiments

We now evaluate the designed monocular VO method. First, we present the datasets and metrics used for the evaluation (Sec. 4.1). Second, we introduce the baseline methods (Sec. 4.2). Third, we evaluate the performance of the method, comparing with the state of the art (Sec. 4.3). Finally, we analyze the sensitivity of the results to various perturbations and limitations in Sec. 4.4.

4.1. Datasets and Evaluation Metric

We test on sequences from the standard dataset[2] [50] to assess the performance of the proposed VO method. Data provided by [50] was collected with a hand-held stereo DAVIS240C in an indoor environment, and ground truth poses were given by a motion capture system with sub-millimeter accuracy. See the supplementary material for collected sequences with our custom-made beamsplitter. To assess the performance of the full VO method, we report ego-motion estimation results using standard metrics: absolute trajectory error (ATE) and rotation RMSE [51]. We use the toolbox from [52] to evaluate the poses given by different visual odometry solutions. Monocular methods are tested using data from the left camera only. The tracking and mapping parts of the VO methods are connected, so depth estimation errors are subsumed in the accuracy of the estimated camera trajectory, in a tightly coupled manner.

4.2. Baseline Methods

We compare with other methods in Tab. 1, with stereo event-based methods (for reference) and with event and frame-based methods [5, 33]. The event-based monocular methods used for comparison are EVO and USLAM.

• EVO [17] is a semi-dense VO approach that builds maps using event-based space sweeping [53] and tracks the camera motion by edge-map alignment, creating binary images of 1k-2k events and aligning them to the projected

[2] RPG Stereo DAVIS: https://rpg.ifi.uzh.ch/ECCV18_stereo_davis.html

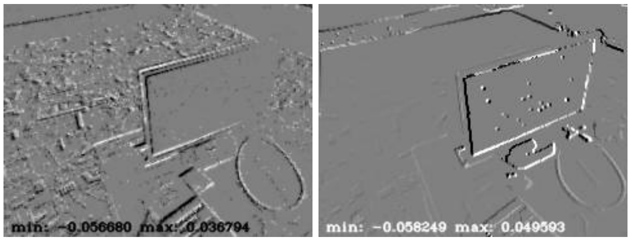

Figure 3. Event frames (2) (left) and EGM frames (6) (right) for the *monitor* sequence. Here, positive brightness changes are displayed in white, and negative ones in black. Gray color means there is no brightness change. This figure contains animations that can be viewed in *Acrobat Reader*.

point cloud map. EVO does not exploit the EGM (3) since it does not use frames nor recovers image brightness.

- USLAM [18] is an indirect monocular method that combines events, frames and IMU measurements. Its front-end converts events into frames by motion compensation using the IMU's gyroscope and the median scene depth. Then FAST corners [54] are extracted and tracked *separately* on the event frames and the grayscale frames, and are passed to a state-of-the-art geometric feature-based back-end [55]. USLAM is the only method that we consider with an IMU. The IMU is tightly used in the front-end for event frame creation, and so removing it is not possible without breaking the (robustness of the) method.
- ORB-SLAM [33] and DSO [5] are the state-of-the-art indirect and direct frame-based monocular visual odometry methods, respectively. We also compare with these methods, using different types of frames (from the DAVIS or reconstructed using E2VID [56]).
- EDS performs event-to-image alignment in the front-end using 20kevents with an overlap of 50%, generating a new event frame and tracking the camera motion every new 10kevents. A visualization is shown in Fig. 3. This gives an effective (and adaptive) event-frame rate of ≈ 60 FPS depending on camera motion while the standard frames are given at a fixed rate of 50 ms (i.e., 20 FPS).

4.3. Ego-motion Estimation Results

Tables 2–3 and Fig. 4 report quantitative and qualitative results of the comparison of our method with the state of the art on sequences from the dataset [50]. Fig. 4 shows sample estimated depth maps and corresponding point clouds (back-projected depth maps). We kept the original depth map visualization of the baseline methods (EVO and DSO).

Comparison to event-based baselines. Qualitatively, Fig. 4 shows that EDS correctly estimates depth at most contour pixels, forming semi-dense structures in the colored depth maps and point clouds. The recovered 3D maps have a higher level of detail than those given by EVO. USLAM produced too sparse maps to convey any visual insight.

		ESVO [26]	USLAM [18]	EVO [17]	EDS (**Ours**)
	Input	E+E	E+F+I	E	E+F
Trans. [cm]	*bin*	2.8	7.7	13.2*	**1.1**
	boxes	5.8	9.5	14.2*	**2.1**
	desk	3.2	9.8	5.2	**1.5**
	monitor	3.3	6.5	7.8	**1.0**
Rot. [deg]	*bin*	7.61	7.18	50.26*	**0.99**
	boxes	9.46	8.84	170.36*	**1.83**
	desk	7.25	32.46	8.25	**1.87**
	monitor	2.74	7.01	7.77	**0.60**

Table 2. *Comparison with event-based 6-DOF VO methods* in terms of Absolute Trajectory Error (RMS) [cm] and Rotation Error (RMS) [deg]. Input data [50] may be: events (E), grayscale frames (F) or IMU (I). EVO entries marked with * indicate the method failed after completing at most 30% of the sequence.

		ORB-SLAM [33]	ORB-SLAM [33]	DSO [5]	DSO†	EDS (**Ours**)
	Input	F+F	F	F	F†	E+F
Trans. [cm]	*bin*	0.7	2.4	1.1	-	**1.1**
	boxes	1.6	3.9	**2.0**	-	2.1
	desk	1.8	3.8	10.0	1.6	**1.5**
	monitor	0.8	3.1	**0.9**	2.1	1.0
Rot. [deg]	*bin*	0.58	**0.84**	2.12	-	0.99
	boxes	4.26	2.39	2.14	-	**1.83**
	desk	2.81	2.52	63.5	**1.80**	1.87
	monitor	3.70	1.77	**0.33**	1.54	0.60

Table 3. *Comparison with frame-based 6-DOF VO methods* in terms of Absolute Trajectory Error (RMS) [cm] and Rotation Error (RMS) [deg]. Input data may be: events (E), grayscale frames (F) or reconstructed frames from events using [56] (F†). DSO† entries with a hyphen indicate failure after initialization, completing less than 10% of the sequence. Best monocular results are in bold.

Quantitatively (Tab. 2), EDS outperforms all other monocular baseline methods, even without using inertial measurements (which are known to improve robustness and increase accuracy in VO [57]). Our approach also outperforms the state-of-the-art event-only stereo method ESVO [26], despite the fact that our method is monocular and hence does not exploit the spatial parallax of stereo setups. The tight fusion in the front-end (between frames and events) and the PBA in the back-end compensate for the lack of stereo baseline in the event data.

Comparison to frame-based baselines. Qualitatively (Fig. 4), DSO produces dilated depth maps for visualization. Hence, they look more complete, but they have also more outliers than those produced by EDS.

Quantitatively (Tab. 3), in terms of translation errors our method is consistently better than monocular ORB-SLAM, and only slightly worse than stereo ORB-SLAM ("F+F") with bundle adjustment enabled [33], which is on average the best performing method. We obtain similar results as DSO, being more accurate (ATE 1.5cm) on the *desk* sequence due to a faster camera motion. We also run DSO on reconstructed frames at 60 FPS using E2VID [56], on the same 20kevents as EDS. The results show a lower ATE in the *desk* sequence (low texture scene) for DSO†. However

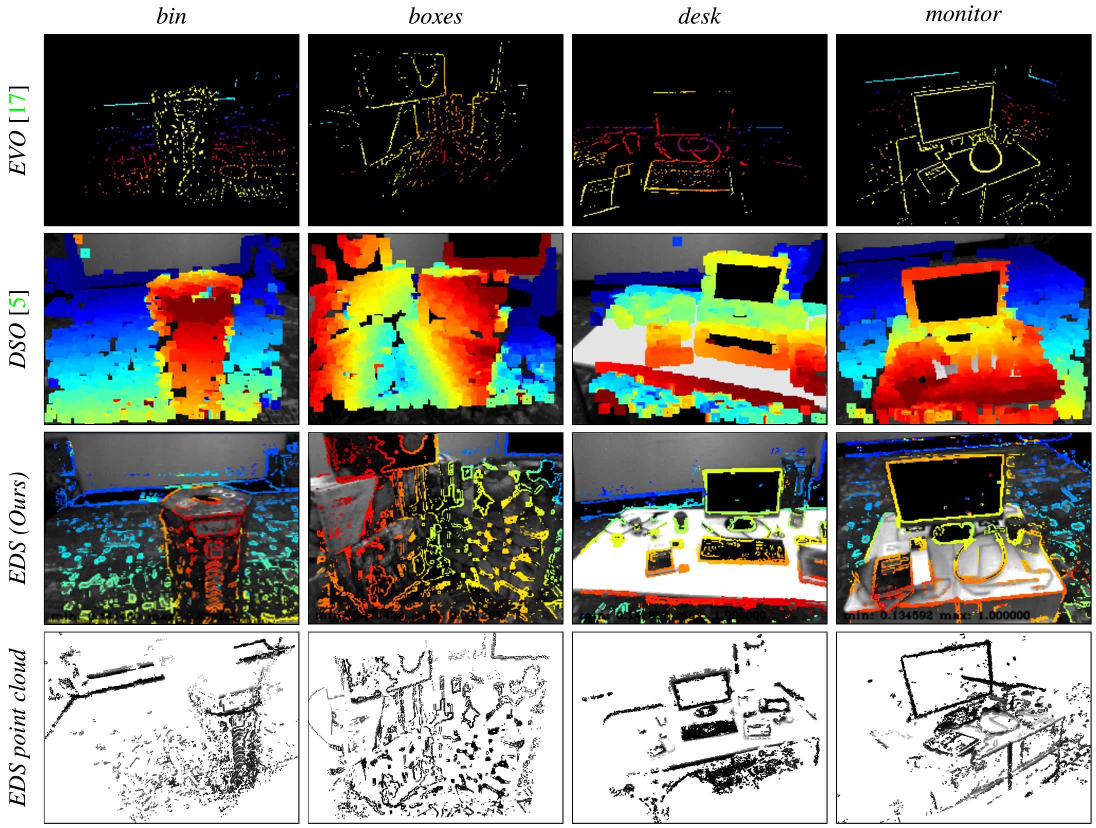

Figure 4. Qualitative comparison on four test sequences from [50]. The first three rows depict pseudo-colored inverse depth maps for each method. EVO's color code is yellow-near blue-far, while DSO and EDS colors are red-near blue-far. The depth range is 1–7 m in all sequences. The 3D point cloud reconstructed by EDS is shown in the last row, with grayscale values from the keyframe.

it fails in the sequences with high texture (*bin* and *boxes*), where E2VID has difficulties to correctly reconstruct the frames due to the limitations of the trained model. Our findings agree with those of EKLT (see [23, Tab. 7] for further details). In terms of rotation errors, EDS is in line with the baselines. Events help estimating camera rotation, especially during sudden motions. This is the reason why DSO† and EDS produce the most accurate results in the *desk* sequence. The large rotational error for DSO in *desk* is due to an insufficient frame rate for the given camera speed.

Low Frame Rate Experiments. Increasing motion speed makes frames further apart, hence it is effectively equivalent to reducing frame rate (except for motion blur), and event-based odometry excels at high-speed motion [16], where usually frame-based approaches fail. Therefore, next, we progressively reduce the frame rate and compare with the state-of-the-art monocular methods from Tab. 3.

We run DSO in two modes: (i) DSO with tracking recovery enabled using 27 different small rotations (see Sec. 3.1 in [5]) and (ii) DSO with tracking recovery disabled, indicated as DSO*. We also compare with ORB-SLAM (full SLAM system, for completeness) and ORB-SLAM* (a more fair baseline with loop closure disabled and the same number of nodes in the covisibility graph as our sliding window, i.e., 7 keyframes). A comprehensive visualization of the results is shown in Fig. 5, where the average ATE for all sequences is depicted at the different frame rates (including the original frame rate from Tab. 3). The plot shows that EDS is almost agnostic to a decrease of the frame rate, while the errors significantly grow for DSO and DSO* as the frame rate decreases. DSO with tracking recovery enabled succeeds until 10 FPS, but is not able to recover the camera pose at lower frame rates. Compared to DSO*, EDS does not require a tracking-recovery strategy since it is capable of continuously tracking at any frame rate using events, only limited by the computational cost. The ORB-SLAM variants produce very competitive results and degrade more gracefully than DSO for low frame rates. Still, our method also performs better than ORB-SLAM*. Therefore, the EDS tracker is more robust than frame-based state-of-the-art odometry in this scenario.

4.4. Sensitivity Study and Limitations

Depth inaccuracies and noise in C are the main subject of study. The brightness change in the EGM (6) highly depends on depth point estimation, which is reflected in (5).

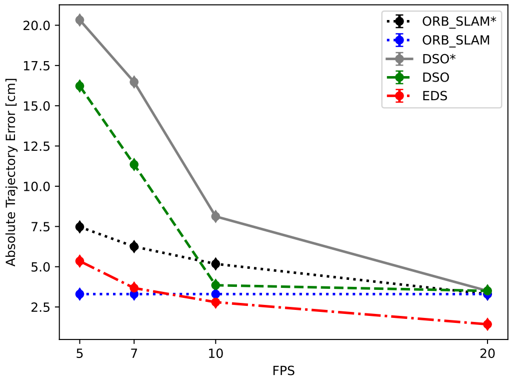

Figure 5. Average Absolute Trajectory Error (RMS) for ORB-SLAM [33], ORB-SLAM* (w/o loop closure), DSO [5], DSO* (w/o recovery-tracking) and EDS (ours) on dataset [50]. Errors are computed for EDS every time an event-frame is tracked (equal at any frame rate), whereas for the baseline frame-based methods they are computed only when a frame is received (i.e., according to the frame rate).

This makes camera tracking more sensitive to accurate point triangulation than direct image alignment methods. Events are also susceptible to noise in C due to manufacturing, which causes a mismatch in the optical flow constraint (3). We use the simulator in [58] to generate synthetic sequences with building-like scenes (e.g., atrium [32]) and to control the scene and event camera parameters, in order to understand the strengths and limitations of EDS.

In one study, we perturb the ground truth 3D map with zero-mean Gaussian noise of standard deviation 1–50 % of the median scene depth, which is 9.7 m in the atrium sequence [32]. In another study, the contrast sensitivity is set to a mean value of $\mu_C = 0.5$ and is perturbed with Gaussian noise of standard deviation $\sigma_C \in [0.05, 0.25]$. Figures 6 and 7 show the depth and contrast errors resulting from these two sensitivity studies, respectively. As we observe in the box plots, the EDS tracker gracefully degrades as depth noise increases, while it has an abrupt degradation in terms of contrast noise, not being able to track the camera motion when the contrast noise has $\sigma_C > 0.15$. One limitation of this approach might be that events are HDR while frames are not necessarily, which makes the EGM estimation challenging (see Fig. 3). We refer the reader to the supplementary material for more details and experiments.

5. Conclusion

We have presented the first monocular direct 6-DOF VO method using events and frames. EDS has several innovations with respect to prior event-based methods, such as the tight coupling of the events and frames in the front-end and the photometric bundle adjustment in the back-end. We strove to compare with multiple event- and frame-based baselines; the results showed that EDS outperforms all event-based methods, and that it is in line with DSO at 10-20 FPS. However, EDS outperforms DSO and ORB-SLAM without loop closure in low frame rate scenarios, tracking with events accurately in between frames. The sensitivity study showed that EDS is robust to depth noise as well as contrast sensitivity event noise. We release the code and dataset to the public, and hope that this research will spark new ideas on low-power and robust VO by combining the best of events and frames.

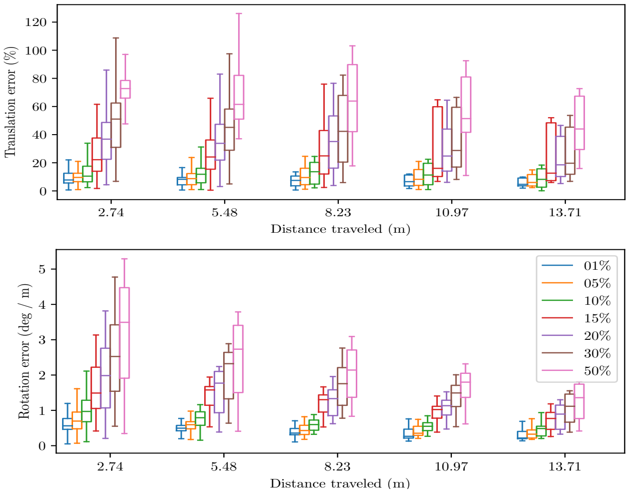

Figure 6. Effect of depth inaccuracies on VO (pose errors).

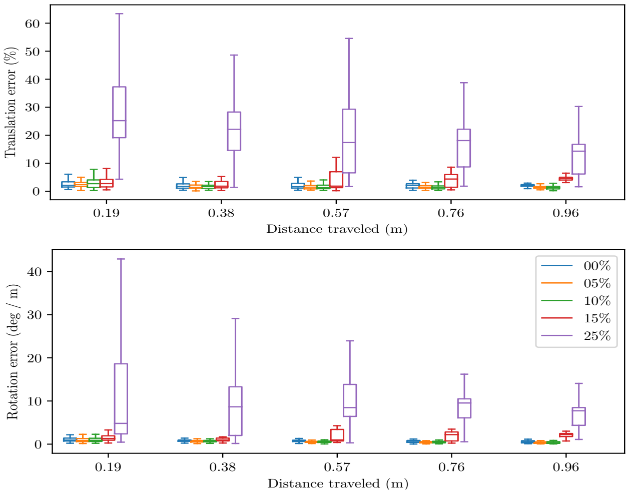

Figure 7. Effect of contrast C noise on VO (pose errors).

Acknowledgement

This work was supported by Huawei Zurich Research Center, the National Centre of Competence in Research (NCCR) Robotics through the Swiss National Science Foundation under grant agreement No. 51NF40_185543, and the European Research Council (ERC) under grant agreement No. 864042 (AGILEFLIGHT).

Event-aided Direct Sparse Odometry
—Supplementary Material—

Javier Hidalgo-Carrió[1], Guillermo Gallego[2], Davide Scaramuzza[1]
[1]Dept. of Informatics, Univ. of Zurich and Dept. of Neuroinformatics, Univ. of Zurich and ETH Zurich.
[2]Technische Universität Berlin, Einstein Center Digital Future and SCIoI Excellence Cluster, Germany.

Overview

In this supplementary material we present:

- Additional details on the low frame rate experiments (Sec. 6).
- Qualitative results about the sensitivity study in depth inaccuracies (Sec. 7).
- Our beamsplitter device and two additional experiments recorded with it (Sec. 8).
- A novel dataset with high quality events & RGB frames recorded with the beamsplitter device to foster research on the topic (Sec. 9).
- A more thorough discussion of limitations (Sec. 10).

6. Low Frame Rate Experiments

Table 4 shows the numerical values in the low frame rate experiment, used for plotting Fig. 5. The values shows that EDS performs the best when the number of frames decreases. Direct methods are less accurate than indirect methods at a lower frame rate (FPS). However at a higher frame rate direct methods use more information from the scene; achieving a higher accuracy.

Among all sequences in dataset [50], *desk* has the most challenging camera motions. In this scenario events excel and help standard frame-based methods. This is also shown in the results (see Tab. 4) where EDS outperforms previous frame-based methods at any given frame rate. Figure 8 shows the trajectories of DSO, ORB-SLAM and EDS for the *desk* sequence. It is qualitatively visible how events enhance standard direct methods guiding the camera pose and producing a more accurate trajectory.

7. Sensitivity with respect to Depth Noise

Depth estimation is the main limitation of our EDS method. This is because events are only used for camera tracking (front-end) and the EGM highly depends on optical flow, which is a function of depth. We previously presented how depth noise affects the camera tracker and entails camera pose inaccuracies. Figure 9 shows the Atrium sequences with the perturbed map. We might conclude that up to 10 % standard deviation of the median depth in the scene, events are still useful to track the camera motion. However, when the mapper wrongfully estimates depth points, events do not help the tracker, generating more inaccuracies than if those events were not used. Therefore, it is important to correctly select which points to incorporate in the EGM calculation.

8. Beamsplitter and Additional Experiments

Publicly available event-based Visual Odometry (VO) datasets are limited and/or contain low quality sensory data [50, 59], either because of noisy events recorded with pioneer neuromorphic devices (such as the DAVIS240B [60] or DAVIS346B) or due to low quality frames (i.e., no gamma correction, small fill factor). Recent publicly available datasets, such as [61, 62], are equipped with newer sensors, however they do not share the same optical axis, having frames and events in two image planes, standard and event-based camera separately. There is a clear need for up-to-date datasets in the event-based VO community with state-of-the-art sensors. We now evaluate our monocular VO method in an extended dataset with our custom co-capture device. First, we present our custom-made beamsplitter device and the calibration procedure in brief (Sec. 8.1). Second, we evaluate the performance of the method (Sec. 8.2).

8.1. Beamsplitter: Sensors and Calibration

We build a custom-made sensing device consisting of a Prophesee Gen3 event camera [63] (640 × 480 px resolution) and a color FLIR camera viewing the same scene through a beamsplitter. The Snoopy house scene in Fig. 1 is recorded with such a device (see Fig. 13 and Tab. 6 for details).

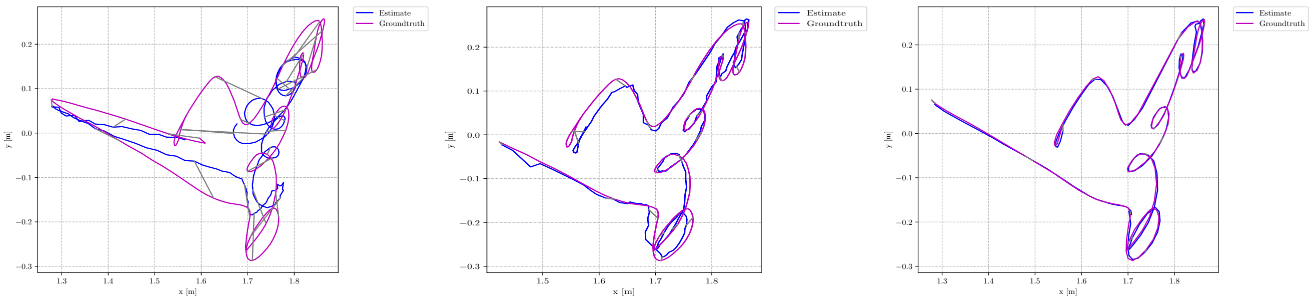

Figure 8. DSO (left), ORB-SLAM (center) and EDS (right) camera trajectory for sequence *desk* in the dataset from [50] at 20 FPS.

	20 FPS				10 FPS				7 FPS				5 FPS			
	DSO*	DSO	ORB_SLAM*	EDS (**ours**)	DSO*	DSO	ORB_SLAM*	EDS(**ours**)	DSO*	DSO	ORB_SLAM*	EDS(**ours**)	DSO*	DSO	ORB_SLAM*	EDS(**ours**)
Input	F	F	F	E+F	F	F	F	E+F	F	F	F	E+F	F	F	F	E+F
bin	1.1	1.2	2.4	**1.1**	1.8	1.8	2.4	**1.8**	3.5	2.5	**2.4**	2.5	16.9	4.8	**2.5**	2.6
boxes	**2.0**	2.0	3.9	2.1	13.5	**3.1**	3.9	3.8	14.8	14.6	**3.9**	5.0	14.8	14.6	7.0	**5.8**
desk	10.0	10.0	3.8	**1.5**	13.4	9.1	3.8	**3.4**	21.1	16.2	7.8	**4.7**	21.6	19.2	9.3	**5.0**
monitor	**0.9**	0.9	3.1	1.0	3.9	**1.5**	10.6	2.3	26.5	12.1	10.9	**2.5**	28.0	27.1	10.3	**8.0**

Table 4. Performance at different frame rates in terms of Absolute Trajectory Error (RMS) [t: cm]. Data from [50].

Both cameras are calibrated and their outputs are aligned with sub-pixel accuracy, giving the equivalent of a DAVIS camera with higher quality color frames at VGA (640×480 px) resolution. To calibrate the cameras, we reconstruct grayscale frames from events using e2calib [64] and input them into Kalibr [65], which computes the intrinsic and extrinsic camera parameters. We calibrate each camera separately to estimate the lens distortion parameters and focal lengths. We configure the FLIR camera to acquire frames at VGA resolution. The color image is warped into the event camera using extrinsic calibration with sub-millimeter accuracy. The intrinsic calibrations of both cameras have reprojection errors of approximately $0.20\,\mathrm{pixels}$. This gives a sufficiently good alignment into one single field of view image. Figure 10 depicts individual camera outputs during the calibration process, and Fig. 12 shows the rectified and undistorted image output with events aligned.

8.2. Ego-motion Estimation Results

We recorded two sequences with our beamsplitter device to prove generalization of our method to other (and newer) visual sensory data. The sequences were recorded in natural indoor scenes and we used COLMAP [66] to provide a ground truth trajectory, since external motion-capture system (i.e., Vicon or Optitrack) was not available. Figure 11 shows the reconstructed 3D map for EDS in the *kitchen* and *snoopy* sequences using events and frames. The colored points in the central part of the kitchen correspond to the 3D locations which are active (i.e., generating events using the event generation model - EGM) in the current keyframe. *snoopy* points cloud used RGB color for a more appealing visualization of the Snoopy reconstructed house.

Input	ORB-SLAM [33] F	EVO [17] E	DSO [5] F	EDS (**Ours**) E+F
kitchen	13.0±9.7	-	12.5±6.8	**9.6**±**5.5**
snoopy	35.1±28	-	**30.7**±**14**	30.9±13

Table 5. Comparison with state-of-the-art 6-DOF VO methods in terms of Absolute Trajectory Error (RMS) [t: cm] and its standard deviation on two beamsplitter sequences. Ground truth poses are computed using COLMAP. Input data may be: events (E) and/or grayscale frames (F). EVO entries marked with hyphen indicate the code did not manage to recover poses on these sequences.

Quantitatively (Tab. 5), our method outperforms all other monocular pure VO baseline methods in the *kitchen* sequence and achieves state-of-the-art accuracy in the *snoopy* sequence. We did not include comparison with USLAM [18] in Tab. 5 since it requires an IMU, which is not available in these two sequences. The beamsplitter device (see Sec. 8.2) has a better sensor quality than the DAVIS240C in [50]. It produces events with less noise, which makes the EGM more accurate (see multimedia material for details).

9. The EDS Dataset

We realized the existing gap in good quality monocular visual-inertial odometry (VIO) dataset with events. Therefore we built a beamsplitter device during the last part of our investigation. The aim of the beamsplitter was to record and release a new dataset to promote and facilitate research on the topic. The dataset targets VIO but it may be used to demonstrate other tasks, such as optical flow estimation, depth estimation, view synthesis (e.g., NeRFs), etc. Fig-

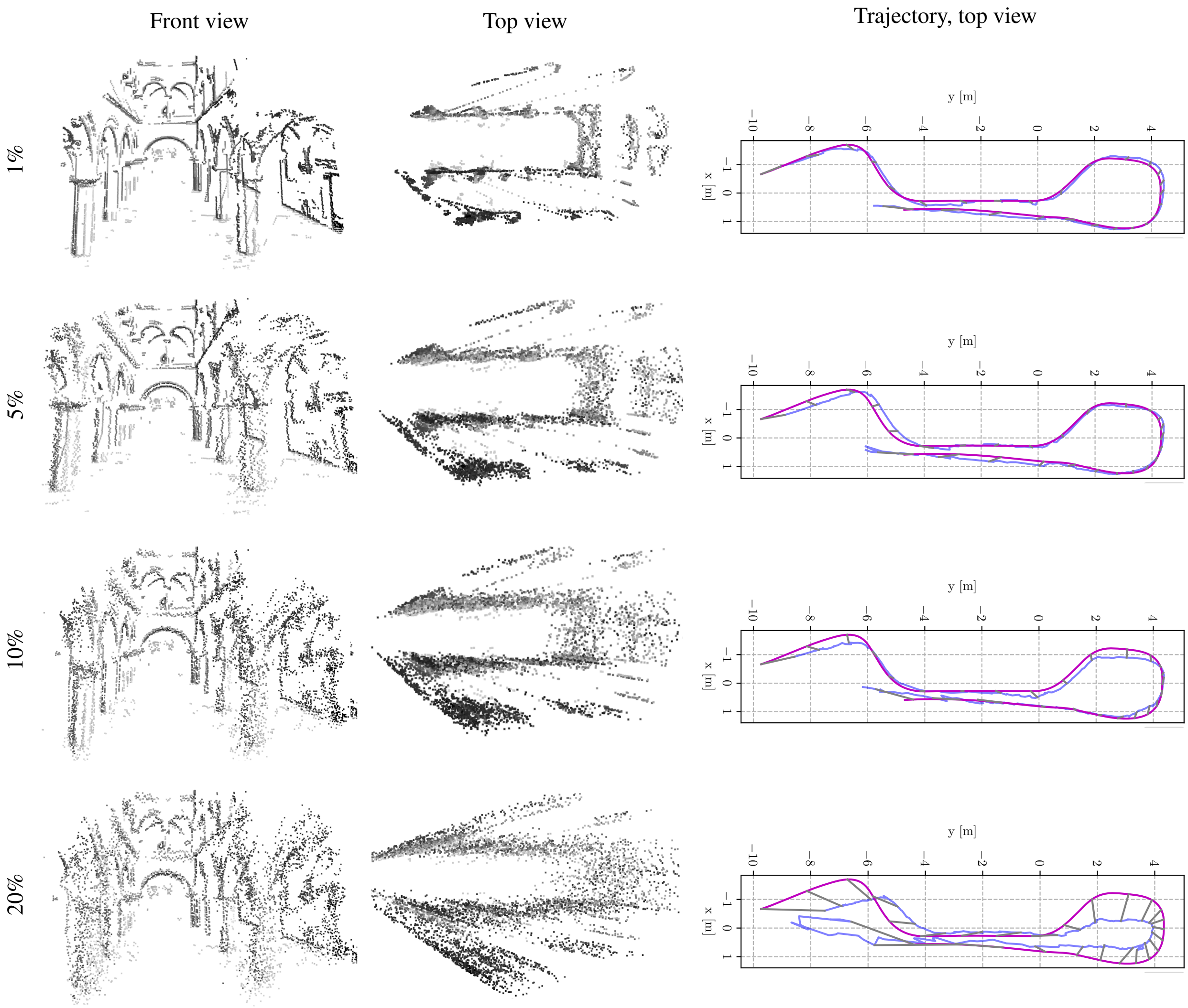

Figure 9. Sensitivity with respect to increasing noise in the depth of the 3D points, from 1% to 20% of the median scene depth. Atrium scene. The ground truth trajectory is in purple, while the estimated trajectory is in blue.

ure 14 shows snapshots of the recorded sequences with our beamsplitter device. When available, we also record ground truth poses from a motion-capture system.

In a nutshell, the EDS dataset provides synchronized events with RGB frames, IMU and ground truth data. The data is given in three different formats, pocolog[1], rosbag[2] and archive files with compressed events in HDF5 format. The images are timestamped, with exposure time and gain values, and the ground truth poses are at the camera frame (i.e., $T_{\text{marker\_cam}}$ already applied). The motion capture system is Vicon or Optitrack depending on the scene. Long sequences such as *04_floor_loop*, *05_rpg_building*, and *12_floor_eight_loop* provide ground truth at the start and end locations. *15_apartment_day* gives start and finish positions using an Apriltag[3] marker. The calibration results to align frames and events as well as the camera to IMU transformation are given.

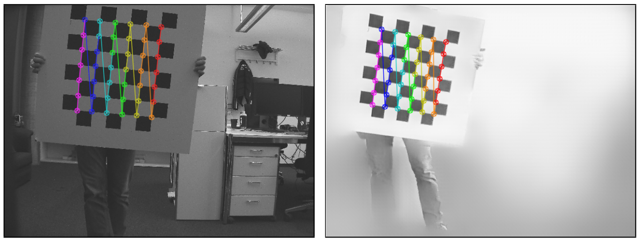

Figure 10. Calibration. Frames from the standard FLIR camera (top) and grayscale frames reconstructed from events (bottom), both with checkerboard corners used during calibration.

[1] https://www.rock-robotics.org
[2] http://wiki.ros.org/rosbag
[3] https://april.eecs.umich.edu/software/apriltag

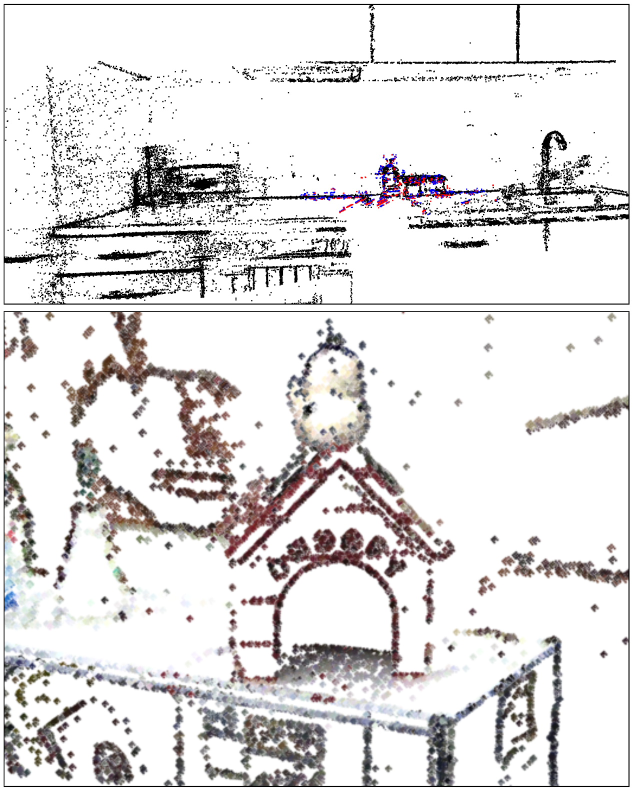

Figure 11. Point cloud maps of the kitchen (top) and snoopy (bottom) sequences.

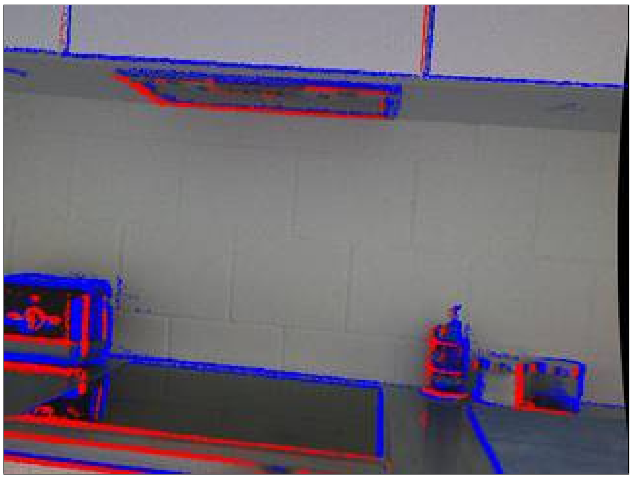

Figure 12. Beamsplitter output with aligned events and frames for the *kitchen* sequence. This figure contains animations that can be viewed in *Acrobat Reader*.

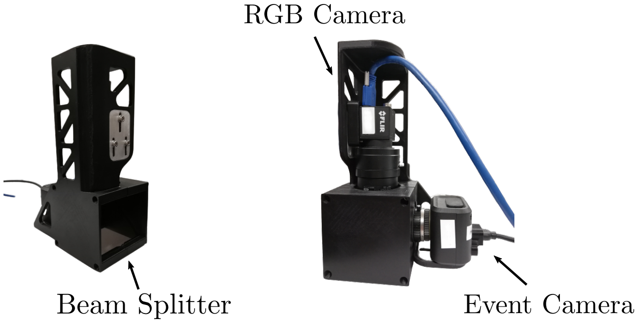

Figure 13. Beamsplitter with a Prophesee Gen3 event camera and a FLIR color camera. Mounted on a case with a 50R/50T beam splitter mirror that allows the sensors to share a spatially aligned field of view.

Sensor Type	**Description**
Prophesee Gen 3.1	Prophesee PPS3MVCD event camera 640 × 480 pixels, 3/4 CMOS Monochrome HFOV 59.4° VFOV 34.7° $\geq$120 dB dynamic range
FLIR Blackfly S USB3	FLIR BFS-U3-16S2C-CS 640 × 480 pixels, 1/2.9 CMOS Color HFOV 45.31° VFOV 34.7° 71.4 dB dynamic range up to 75 FPS (depending on the sequence)
Inversense MPU-9250	Inertial Measurement Unit MEMS, 16bits resolution 3 × Gyroscopes 3 × Accelerometers 3 × Magnetometers (not utilized) 1000Hz sampling rate
Motion-capture system	Optitrack: RPG Flying room, 11 cameras Vicon: RPG Drone Arena, 36 cameras Camera pose: position + rotation 150 Hz sampling rate

Table 6. Details of the hardware utilized for the dataset collection

10. Limitations

10.1. Grayscale Frames are not HDR

Our method works under the assumption of the availability of collocated grayscale frames and events. Current devices, such as the DAVIS camera [60], produce low-quality grayscale frames, with a dynamic range of $\approx 55\,\mathrm{dB}$, which is small compared to the high dynamic range (HDR) properties of event cameras ($>120\,\mathrm{dB}$). Hence, the frames from the DAVIS are not HDR, and one could point this as a limitation of the approach.

We tackled this problem in two ways: (i) by building our own sensing device with higher quality frames than the DAVIS (Fig. 13), and (ii) by testing alternatives, such as using grayscale frames reconstructed from the events (e.g., using state of the art [56]). The latter was tested on DSO and our approach. The results from DSO are reported in the main paper, indicated with DSO† in Tab. 3. We observed that it did not produce good results in the bin and boxes sequences. We observed the same failure effect when using reconstructed grayscale frames on EDS. Nevertheless, we expect that if using a higher-end device (e.g., (i)) is not an option, better grayscale frames for VO with our method

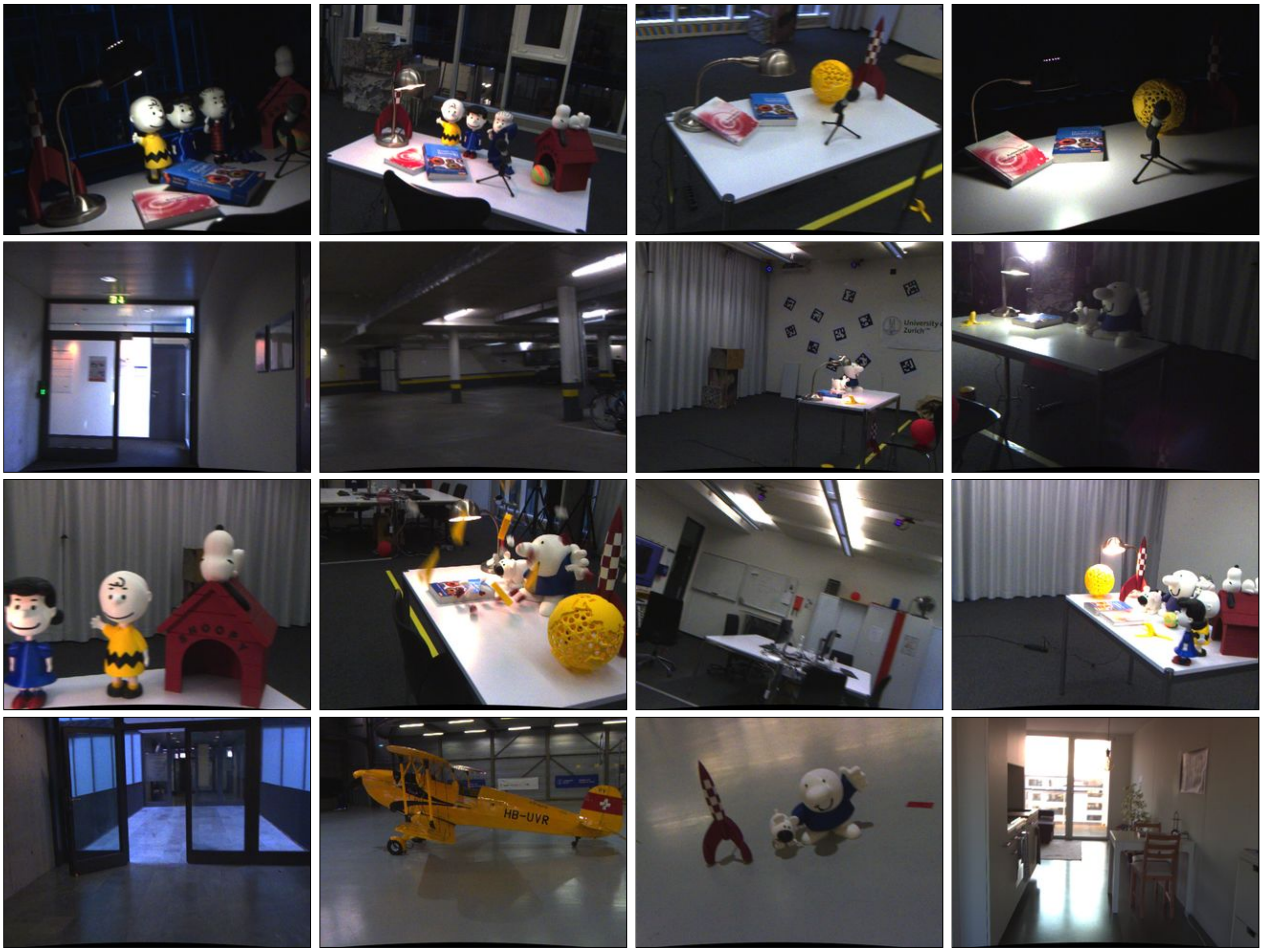

Figure 14. *Sequences from our EDS Dataset*: the dataset contains 16 sequences with events, color frames, IMU data and ground truth poses on a diverse set of environments.

would be obtained in the near future by means of improvements in image reconstruction methods and/or event cameras (i.e., lower event noise). Event cameras are evolving fast, and new prototypes, in combination with standard sensors may occur in the near future, for example to advance computational photography (e.g., [42]).

10.2. Computational Performance

The current implementation of our method is unoptimized and it is about 5× slower than real time. However, we think that there is large room for code improvement and engineering to make it real-time capable. Specifically, the back-end is un-optimized since automatic differentiation in Ceres [46] is not real time when having a large number of parameters (i.e., 4000 points in a 7-keyframe sliding window). It could be sped up by feeding fewer points to the back-end and better selecting a sparse set of points. This is the reason why we combined our front-end with DSO's [5] back-end, which is optimized and real-time capable with up to 2000 points in a similar sliding-window size. The main purpose of our work is to understand the limitations of previous event-based methods (e.g., why do many of the prior works lose track or have large errors?) and overcome them with a new design that has not been previously explored. Finally, if the method is used offline, e.g., for recovering a scene map and/or accurate camera trajectory, runtime is not an issue.

10.3. Setting an Arbitrary Scale for the World

Absolute scale is not observable in monocular VO without additional information or an IMU. Hence, in our method we need to provide values to set the depth range of the initial map, which will guide the rest of the visual odometry process. We did not consider using an IMU (accelerometer and gyroscope) to focus on the visual aspects of odometry. Sensor fusion with an IMU is left as future work.

References

[1] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian D. Reid, and John J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] Andrew J. Davison, "FutureMapping: The computational structure of spatial AI systems," *arXiv e-prints*, Mar. 2018.

[3] Antonio Loquercio, Elia Kaufmann, René Ranftl, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza, "Learning high-speed flight in the wild," *Science Robotics*, Oct. 2021.

[4] Raúl Mur-Artal, José M. M. Montiel, and Juan D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, 2015.

[5] Jakob Engel, Vladlen Koltun, and Daniel Cremers, "Direct Sparse Odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, pp. 611–625, Mar. 2018.

[6] Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, and Davide Scaramuzza, "SVO: Semidirect visual odometry for monocular and multicamera systems," *IEEE Trans. Robot.*, vol. 33, no. 2, pp. 249–265, 2017.

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

[8] Christoph Posch, Daniel Matolin, and Rainer Wohlgenannt, "A QVGA 143dB dynamic range asynchronous address-event PWM dynamic image sensor with lossless pixel-level video compression," in *IEEE Intl. Solid-State Circuits Conf. (ISSCC)*, pp. 400–401, 2010.

[9] Christoph Posch, Teresa Serrano-Gotarredona, Bernabe Linares-Barranco, and Tobi Delbruck, "Retinomorphic event-based vision sensors: Bioinspired cameras with spiking output," *Proc. IEEE*, vol. 102, pp. 1470–1484, Oct. 2014.

[10] Andrea Censi and Davide Scaramuzza, "Low-latency event-based visual odometry," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 703–710, 2014.

[11] Hanme Kim, Ankur Handa, Ryad Benosman, Sio-Hoi Ieng, and Andrew J. Davison, "Simultaneous mosaicing and tracking with an event camera," in *British Mach. Vis. Conf. (BMVC)*, 2014.

[12] Elias Mueggler, Basil Huber, and Davide Scaramuzza, "Event-based, 6-DOF pose tracking for high-speed maneuvers," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, pp. 2761–2768, 2014.

[13] Hanme Kim, Stefan Leutenegger, and Andrew J. Davison, "Real-time 3D reconstruction and 6-DoF tracking with an event camera," in *Eur. Conf. Comput. Vis. (ECCV)*, pp. 349–364, 2016.

[14] Beat Kueng, Elias Mueggler, Guillermo Gallego, and Davide Scaramuzza, "Low-latency visual odometry using event-based feature tracks," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, pp. 16–23, 2016.

[15] Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis, "Event-based visual inertial odometry," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pp. 5816–5824, 2017.

[16] Guillermo Gallego, Jon E. A. Lund, Elias Mueggler, Henri Rebecq, Tobi Delbruck, and Davide Scaramuzza, "Event-based, 6-DOF camera tracking from photometric depth maps," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, pp. 2402–2412, Oct. 2018.

[17] Henri Rebecq, Timo Horstschäfer, Guillermo Gallego, and Davide Scaramuzza, "EVO: A geometric approach to event-based 6-DOF parallel tracking and mapping in real-time," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 593–600, 2017.

[18] Antoni Rosinol Vidal, Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza, "Ultimate SLAM? combining events, images, and IMU for robust visual SLAM in HDR and high speed scenarios," *IEEE Robot. Autom. Lett.*, vol. 3, pp. 994–1001, Apr. 2018.

[19] Guillermo Gallego, Henri Rebecq, and Davide Scaramuzza, "A unifying contrast maximization framework for event cameras, with applications to motion, depth, and optical flow estimation," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pp. 3867–3876, 2018.

[20] William Oswaldo Chamorro Hernandez, Juan Andrade-Cetto, and Joan Solá Ortega, "High-speed event camera tracking," in *British Mach. Vis. Conf. (BMVC)*, 2020.

[21] Haram Kim and H. Jin Kim, "Real-time rotational motion estimation with contrast maximization over globally aligned events," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 6016–6023, 2021.

[22] Guillermo Gallego, Tobi Delbruck, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew Davison, Jörg Conradt, Kostas Daniilidis, and Davide Scaramuzza, "Event-based vision: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

[23] Daniel Gehrig, Henri Rebecq, Guillermo Gallego, and Davide Scaramuzza, "EKLT: Asynchronous photometric feature tracking using events and frames," *Int. J. Comput. Vis.*, 2019.

[24] Matthew Cook, Luca Gugelmann, Florian Jug, Christoph Krautz, and Angelika Steger, "Interacting maps for fast visual interpretation," in *Int. Joint Conf. Neural Netw. (IJCNN)*, pp. 770–776, 2011.

[25] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis, "Unsupervised event-based learning of optical flow, depth, and egomotion," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.

[26] Yi Zhou, Guillermo Gallego, and Shaojie Shen, "Event-based stereo visual odometry," *IEEE Trans. Robot.*, vol. 37, no. 5, pp. 1433–1450, 2021.

[27] Elias Mueggler, Chiara Bartolozzi, and Davide Scaramuzza, "Fast event-based corner detection," in *British Mach. Vis. Conf. (BMVC)*, 2017.

[28] Ignacio Alzugaray and Margarita Chli, "Asynchronous corner detection and tracking for event cameras in real time," *IEEE Robot. Autom. Lett.*, vol. 3, pp. 3177–3184, Oct. 2018.

[29] David Weikersdorfer, Raoul Hoffmann, and Jörg Conradt, "Simultaneous localization and mapping for event-based vision systems," in *Int. Conf. Comput. Vis. Syst. (ICVS)*, pp. 133–142, 2013.

[30] David Weikersdorfer and Jörg Conradt, "Event-based particle filtering for robot self-localization," in *IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, pp. 866–870, 2012.

[31] Elias Mueggler, Guillermo Gallego, and Davide Scaramuzza, "Continuous-time trajectory estimation for event-based vision sensors," in *Robotics: Science and Systems (RSS)*, 2015.

[32] Samuel Bryner, Guillermo Gallego, Henri Rebecq, and Davide Scaramuzza, "Event-based, direct camera tracking from a photometric 3D map using nonlinear optimization," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2019.

[33] Raúl Mur-Artal and Juan D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, pp. 1255–1262, Oct. 2017.

[34] Valentina Vasco, Arren Glover, and Chiara Bartolozzi, "Fast event-based Harris corner detection exploiting the advantages of event-driven cameras," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2016.

[35] Philippe Chiberre, Etienne Perot, Amos Sironi, and Vincent Lepetit, "Detecting stable keypoints from events through image gradient prediction," in *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, 2021.

[36] Guillermo Gallego and Davide Scaramuzza, "Accurate angular velocity estimation with an event camera," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 632–639, 2017.

[37] Christian Reinbacher, Gottfried Munda, and Thomas Pock, "Real-time panoramic tracking for event cameras," in *IEEE Int. Conf. Comput. Photography (ICCP)*, pp. 1–9, 2017.

[38] Cedric Scheerlinck, Nick Barnes, and Robert Mahony, "Continuous-time intensity estimation using event cameras," in *Asian Conf. Comput. Vis. (ACCV)*, 2018.

[39] Zhenjiang Ni, Aude Bolopion, Joel Agnus, Ryad Benosman, and Stéphane Régnier, "Asynchronous event-based visual shape tracking for stable haptic feedback in microrobotics," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1081–1089, 2012.

[40] Daniel Gehrig, Michelle Regg, Mathias Gehrig, Javier Hidalgo-Carrio, and Davide Scaramuzza, "Combining events and frames using recurrent asynchronous multimodal networks for monocular depth prediction," *IEEE Robot. Autom. Lett.*, 2021.

[41] Liyuan Pan, Richard I. Hartley, Cedric Scheerlinck, Miaomiao Liu, Xin Yu, and Yuchao Dai, "High frame rate video reconstruction based on an event camera," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

[42] Stepan Tulyakov, Daniel Gehrig, Stamatios Georgoulis, Julius Erbach, Mathias Gehrig, Yuanyou Li, and Davide Scaramuzza, "TimeLens: Event-based video frame interpolation," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2021.

[43] Guillermo Gallego, Christian Forster, Elias Mueggler, and Davide Scaramuzza, "Event-based camera pose tracking using a generative event model." arXiv:1510.01972, 2015.

[44] Gemma Taverni, Diederik Paul Moeys, Chenghan Li, Celso Cavaco, Vasyl Motsnyi, David San Segundo Bello, and Tobi Delbruck, "Front and back illuminated Dynamic and Active Pixel Vision Sensors comparison," *IEEE Trans. Circuits Syst. II*, vol. 65, no. 5, pp. 677–681, 2018.

[45] Peter Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*. Springer Tracts in Advanced Robotics, Springer, 2017.

[46] A. Agarwal, K. Mierle, and Others, "Ceres solver." http://ceres-solver.org.

[47] Richard Hartley and Andrew Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003. 2nd Edition.

[48] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

[49] Zichao Zhang, Guillermo Gallego, and Davide Scaramuzza, "On the comparison of gauge freedom handling in optimization-based visual-inertial state estimation," *IEEE Robot. Autom. Lett.*, vol. 3, pp. 2710–2717, July 2018.

[50] Yi Zhou, Guillermo Gallego, Henri Rebecq, Laurent Kneip, Hongdong Li, and Davide Scaramuzza, "Semi-dense 3D reconstruction with a stereo event camera," in *Eur. Conf. Comput. Vis. (ECCV)*, pp. 242–258, 2018.

[51] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, Oct. 2012.

[52] Zichao Zhang and Davide Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2018.

[53] Henri Rebecq, Guillermo Gallego, Elias Mueggler, and Davide Scaramuzza, "EMVS: Event-based multi-view stereo—3D reconstruction with an event camera in real-time," *Int. J. Comput. Vis.*, vol. 126, pp. 1394–1414, Dec. 2018.

[54] Edward Rosten and Tom Drummond, "Machine learning for high-speed corner detection," in *Eur. Conf. Comput. Vis. (ECCV)*, pp. 430–443, 2006.

[55] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial SLAM using nonlinear optimization," *Int. J. Robot. Research*, 2015.

[56] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza, "High speed and high dynamic range video with an event camera," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019.

[57] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza, "On-manifold preintegration for real-time visual-inertial odometry," *IEEE Trans. Robot.*, vol. 33, no. 1, pp. 1–21, 2017.

[58] Henri Rebecq, Daniel Gehrig, and Davide Scaramuzza, "ESIM: an open event camera simulator," in *Conf. on Robotics Learning (CoRL)*, 2018.

[59] Elias Mueggler, Henri Rebecq, Guillermo Gallego, Tobi Delbruck, and Davide Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM," *Int. J. Robot. Research*, vol. 36, no. 2, pp. 142–149, 2017.

[60] Christian Brandli, Raphael Berner, Minhao Yang, Shih-Chii Liu, and Tobi Delbruck, "A 240x180 130dB 3μs latency global shutter spatiotemporal vision sensor," *IEEE J. Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, 2014.

[61] Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza, "DSEC: A stereo event camera dataset for driving scenarios," *IEEE Robot. Autom. Lett.*, 2021.

[62] Simon Klenk, Jason Chui, Nikolaus Demmel, and Daniel Cremers, "TUM-VIE: The TUM stereo visual-inertial event dataset," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2021.

[63] Prophesee Evaluation Kits. https://www.prophesee.ai/event-based-evk/, 2020.

[64] Manasi Muglikar, Mathias Gehrig, Daniel Gehrig, and Davide Scaramuzza, "How to calibrate your event camera," in *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, 2021.

[65] Joern Rehder, Janosch Nikolic, Thomas Schneider, Timo Hinzmann, and Roland Siegwart, "Extending kalibr: Calibrating the extrinsics of multiple imus and of individual axes," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 4304–4311, 2016.

[66] Johannes Lutz Schönberger and Jan-Michael Frahm, "Structure-from-motion revisited," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2016.