Data Paper

The International Journal of
Robotics Research
2017, Vol. 36(2) 142–149

DOI: 10.1177/0278364917691115
journals.sagepub.com/home/ijr
SAGE

The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM

Elias Mueggler[1], Henri Rebecq[1], Guillermo Gallego[1], Tobi Delbruck[2,3,4] and Davide Scaramuzza[1]

Abstract

New vision sensors, such as the dynamic and active-pixel vision sensor (DAVIS), incorporate a conventional global-shutter camera and an event-based sensor in the same pixel array. These sensors have great potential for high-speed robotics and computer vision because they allow us to combine the benefits of conventional cameras with those of event-based sensors: low latency, high temporal resolution, and very 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. For this purpose, we present and release a collection of datasets captured with a DAVIS in a variety of synthetic and real environments, which we hope will motivate research on new algorithms for high-speed and high-dynamic-range robotics and computer-vision applications. In addition to global-shutter intensity images and asynchronous events, we provide inertial measurements and ground-truth camera poses from a motion-capture system. The latter allows comparing the pose accuracy of ego-motion estimation algorithms quantitatively. All the data are released both as standard text files and binary files (i.e. rosbag). This paper provides an overview of the available data and describes a simulator that we release open-source to create synthetic event-camera data.

Keywords
Event-based cameras, visual odometry, SLAM, simulation

1. Introduction

Over the past 50 years, computer-vision research has been devoted to standard, frame-based cameras (i.e. rolling or global shutter cameras) and only in the last few years have cameras been successfully used in commercial autonomous mobile robots, such as cars, drones, and vacuum cleaners, just to mention a few. Despite the recent progress, we believe that the advent of event-based cameras is about to revolutionize the robot sensing landscape. Indeed, the performance of a mobile robot in tasks such as navigation depends on the accuracy and latency of perception. The latency depends on the frequency of the sensor data plus the time it takes to process the data. It is typical in current robot-sensing pipelines to have latencies in the order of 50–200 ms or more, which puts a hard bound on the maximum agility of the platform. An event-based camera virtually eliminates the latency: data is transmitted using events, which have a latency in the order of *micro*-seconds. Another advantage of event-based cameras is their very high dynamic range (130 dB vs 60 dB of standard cameras), which makes them ideal in scenes characterized by large illumination changes. Other key properties of event-based cameras are low bandwidth, low storage, and low power requirements. All these properties enable the design of a new class of algorithms for high-speed and high-dynamic-range robotics, where standard cameras are typically not ideal because of motion blur, image saturation, and high latency. However, the way that event-based cameras convey the information is completely different from that of traditional sensors, so a paradigm shift is needed to deal with them.

1.1. Related datasets

Two recent datasets exist that also use the dynamic and active-pixel vision sensor (DAVIS): Rueckauer and Delbruck (2016) and Barranco et al. (2016).

[1]Robotics and Perception Group, University of Zurich, Switzerland
[2]Institute of Neuroinformatics, Zurich, Switzerland
[3]University of Zurich, Switzerland
[4]ETH Zurich, Switzerland

Corresponding author:
Elias Mueggler, Robotics and Perception Group, University of Zurich, Andreasstrasse 15, 8050 Zurich, Switzerland.
Email: mueggler@ifi.uzh.ch

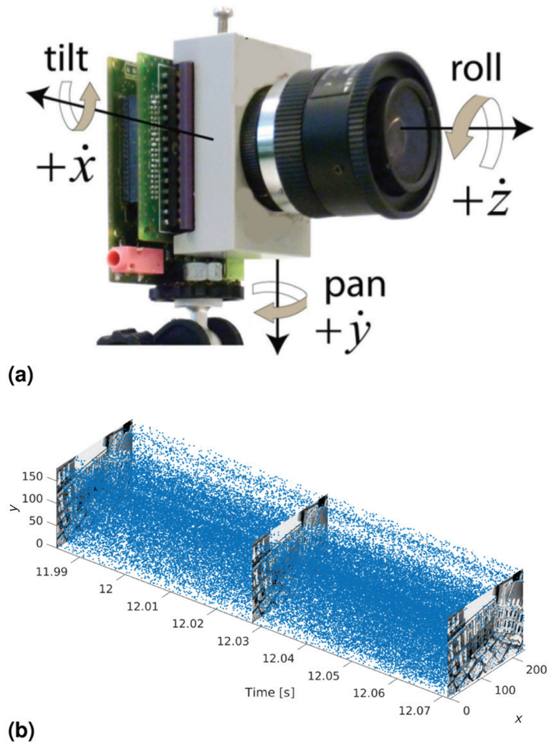

Fig. 1. The DAVIS camera and visualization of its output. (a) The DAVIS sensor and axes definitions. Figure adapted from Delbruck et al. (2014). (b) Visualization of the event output of a DAVIS in space-time. Blue dots mark individual asynchronous events. The polarity of the events is not shown.

The first work is tailored for comparison of event-based optical flow estimation algorithms (Rueckauer and Delbruck, 2016). It contains both synthetic and real datasets under pure rotational (three-degrees-of-freedom, or 3-DOF) motion on simple scenes with strong visual contrasts. Ground truth was acquired using the inertial measurement unit (IMU). In contrast, our datasets contain arbitrary, hand-held, 6-DOF motion in a variety of artificial and natural scenes with precise ground-truth camera poses from a motion-capture system.

A work more similar to ours is Barranco et al. (2016). Their focus is to create a dataset that facilitates comparison of event-based and frame-based methods for two-dimensional (2D) and three-dimensional (3D) visual navigation tasks. To this end, a ground robot was equipped with a DAVIS and a Microsoft Kinect RGB-D sensor. The DAVIS was mounted on a pan–tilt unit, and thus could be excited in five DOFs. The scene contains checkerboards, books, and a chair. Ground truth was acquired by the encoders of the pan–tilt unit and the ground robot's wheel odometry, and is therefore subject to drift. In contrast, our dataset contains hand-held, 6-DOF motion (slow- and high-speed) on a variety of scenes with precise ground-truth camera poses from a motion-capture system, which is not subject to drift.

2. The DAVIS sensor

The DAVIS (Brandli et al., 2014) (see Figure 1(a)) is an event camera that transmits *events* in addition to frames. Events are pixel-level, relative-brightness changes that are detected in continuous time by specially designed pixels.[1] The events are timestamped with *micro*-second resolution and transmitted asynchronously at the time they occur. Each event e is a tuple $\langle x, y, t, p \rangle$, where x, y are the pixel coordinates of the event, t is the timestamp of the event, and $p = \pm 1$ is the polarity of the event, which is the sign of the brightness change. This representation is sometimes also referred to as address-event representation. The DAVIS has a spatial resolution of 240×180 pixels. A visualization of the event output is shown in Figure 1(b). Both the events and frames are generated by the same physical pixels; hence, there is no spatial offset between the events and the frames.

Due to its low latency and high temporal resolution, both in the range of *micro*-seconds, event-based cameras are very promising sensors for high-speed mobile robot applications. Since event cameras are data-driven (only brightness *changes* are transmitted), no redundant data is transmitted. The required bandwidth thus depends on the motion speed and the type of scene. An additional advantage for robotic applications is the high dynamic range of 130 dB (compared to the 60 dB of expensive computer-vision cameras), which allows both indoor and outdoor operation without changing parameters. Since all pixels are independent, very large contrast changes can also take place within the same scene.

Over the course of the last seven years, several groups including ours have demonstrated the use of event-based sensors in a variety of tasks, such as SLAM in 2D (Weikersdorfer et al., 2013) and 3D (Kim et al., 2016; Kueng et al., 2016; Rebecq et al., 2016b), optical flow (Bardow et al., 2016; Benosman et al., 2014; Cook et al., 2011), visual odometry (Censi and Scaramuzza, 2014), 6-DOF localization for high-speed robotics (Mueggler et al., 2014), line detection and localization (Yuan and Ramalingam, 2016), 3D reconstruction (Rebecq et al., 2016a), image reconstruction and mosaicing (Kim et al., 2014; Reinbacher et al., 2016), orientation estimation (Gallego and Scaramuzza, 2016), and continuous-time trajectory estimation (Mueggler et al., 2015).

However, all these methods were evaluated on different, specific datasets and, therefore, cannot be compared with each other. The datasets we propose here are tailored to allow comparison of pose tracking, visual odometry, and SLAM algorithms. Since event-based cameras, such as the DAVIS, are currently still expensive (~US$5000), these data also allow researchers without equipment to use well-calibrated data for their research.

2.1. DAVIS IMU

In addition to the visual output (events and frames), the DAVIS includes an IMU that provides gyroscope and

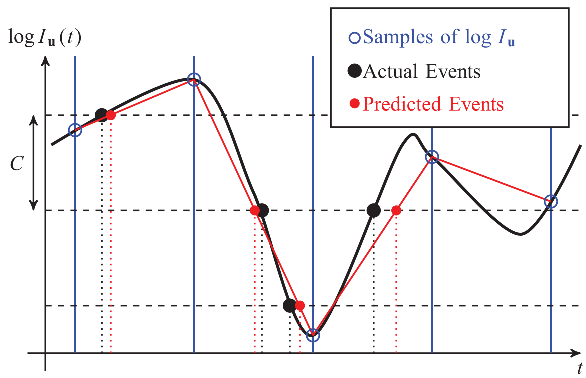

Fig. 2. DAVIS simulator. Per-pixel event generation using piecewise linear time interpolation of the intensities given by the rendered images. For simplicity, images were rendered at a fixed rate.

accelerometer data, thus enabling the design of visual-inertial event-based algorithms. The DAVIS camera has the IMU mounted directly behind and centered under the image sensor pixel array center, at a distance of about 3 mm from it, so the IMU shares nearly the same position as the event sensor. The IMU axes are aligned with the camera axes (see Figure 1(a)). More specifically, the IMU is an InvenSense MPU-6150,[2] which integrates a three-axis gyroscope that can measure in the range $\pm 2000°/\mathrm{s}$ and a three-axis accelerometer for the range $\pm 16g$. It integrates six 16-bit analog-to-digital converters (ADCs) for digitizing the gyroscope and accelerometer outputs at a 1kHz sample rate.

3. DAVIS simulator

Simulation offers a good baseline when working with new sensors, such as the DAVIS. Based on the operation principle of an ideal DAVIS pixel, we created a simulator that, given a virtual 3D scene and the trajectory of a moving DAVIS within it, generates the corresponding stream of events, intensity frames, and depth maps. We used the computer graphics software Blender[3] to generate thousands of rendered images along the specified trajectory, ensuring that the motion between consecutive images was smaller than 1/3 pixel. For each pixel, we keep track of the time of the last event triggered at that location. This map of timestamps (also called surface of active events; Benosman et al., 2014), combined with time interpolation of the rendered image intensities, allows determining brightness changes of a predefined amount (given by the contrast threshold) in the time between images, thus effectively providing continuous timestamps, as if events were generated asynchronously. Time interpolation has an additional benefit: it solves the problem of having to generate millions of images for each second of a sequence, as would have been required to deliver microsecond-resolution timestamps in the absence of interpolation.

More specifically, Figure 2 illustrates the operation of the simulator for a single pixel $\mathbf{u} = (x, y)^\top$. The continuous intensity signal at pixel $\mathbf{u}$, $\log I_{\mathbf{u}}(t)$ (black), is sampled at the times of the rendered images (blue markers). These samples are used to determine the times of the events: the data is linearly interpolated between consecutive samples and the crossings of the resulting lines (in red) with the levels given by multiples of the contrast threshold C (i.e. horizontal lines) specify the timestamps of the events (red dots). As can be observed, this simple interpolation scheme allows for (i) higher resolution event timestamps than those of the rendered images, and (ii) the generation of multiple events between two samples if the corresponding intensity jump is larger than the contrast threshold.

The provided events are "perfect" measurements up to sampling and quantization; under this condition, an image $\hat{I}(\mathbf{u}; t)$ can be reconstructed from the event stream at any point in time t by accumulating events $e_k = \langle \mathbf{u}_k, t_k, p_k \rangle$ according to

$$\log \hat{I}(\mathbf{u}; t) = \log I(\mathbf{u}; 0) + \sum_{0 < t_k \le t} p_k \, C \, \delta(\mathbf{u} - \mathbf{u}_k) \, \delta(t - t_k)$$

where $I(\mathbf{u}; 0)$ is the rendered image at time $t = 0$ and δ selects the pixel to be updated on every event (pixel $\mathbf{u}_k$ of $\hat{I}$ is updated at time t_k). We used this scheme to check that the reconstructed image agreed with the rendered image at several points in time; specifically, the per-pixel intensity error was confined to the quantization interval $(-C, C)$.

Event generation operates on brightness pixels, which are computed from the rendered color images using the ITU-R Recommendation BT.601[4] for luma, that is, according to formula $Y = 0.299R + 0.587G + 0.114B$, with RGB channels in linear color space to better resemble the operation of the DAVIS.

Because realistic event noise is extremely difficult to model due to the complex behavior of event sensors with respect to their bias settings and other factors, the provided simulation datasets do not include event noise. Nevertheless, the simulator, and the datasets created with it, are a useful tool for prototyping new event-based algorithms. Our implementation is available as open-source software.[5]

4. Datasets

In this section, we describe the datasets that we provide. The datasets contain:

- the asynchronous event stream;
- intensity images at about 24 Hz;
- inertial measurements (three-axis gyroscope and three-axis accelerometer) at 1 kHz;
- ground-truth camera poses from a motion-capture system[6] with sub-millimeter precision at 200 Hz (for the indoor datasets);
- the intrinsic camera matrix.

All information comes with precise timestamps. For datasets that were captured outside the motion-capture

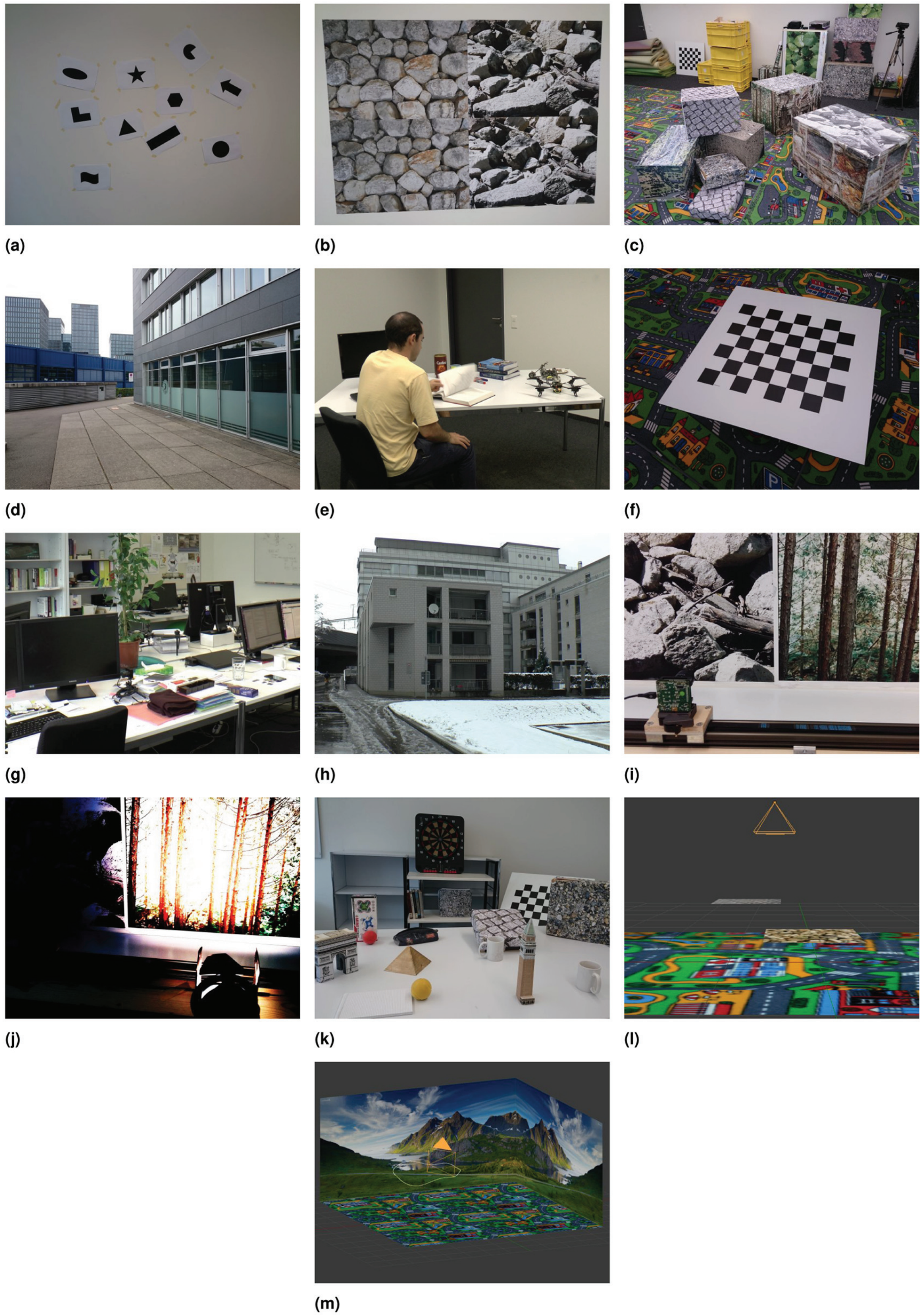

Fig. 3. Dataset scenes: (a) shapes; (b) wall poster; (c) boxes; (d) outdoors; (e) dynamic; (f) calibration; (g) office; (h) urban; (i) motorized linear slider; (j) motorized slider (HDR); (k) motorized slider with objects; (l) synthetic: three planes; (m) synthetic: three walls.

Table 1. Description of dataset format.

File	Description	Line Content
`events.txt`	One event per line	`timestamp x y polarity`
`images.txt`	One image reference per line	`timestamp filename`
`00000000.png`	Images referenced from `images.txt`	
`imu.txt`	One measurement per line	`timestamp ax ay az gx gy gz`
`groundtruth.txt`	One ground truth measurement per line	`timestamp px py pz qx qy qz qw`
`calib.txt`	Camera parameters	`fx fy cx cy k1 k2 p1 p2 k3`

system (e.g. in an office or outdoors), no ground truth is provided. Some datasets were collected using a motorized linear slider and ground truth was collected using the slider's position. Due to vibrations induced by the slider motor, the very noisy IMU data was not recorded.

4.1. Data format

The datasets are provided in the standard text form that is described here. For convenience, they can also be downloaded as binary rosbag files (the details are on the Dataset website). The format of the text files is described in Table 1.

The ground-truth pose is with respect to the (arbitrary) motion-capture origin that has the z-axis gravity-aligned (pointing upwards). The orientation is provided as a unit quaternion $\mathbf{q} = (q_x, q_y, q_z, q_w)^\top$, where q_w and $\mathbf{q}_v = (q_x, q_y, q_z)^\top$ are the scalar and vector components, respectively. This convention was proposed as a standard by Jet Propulsion Laboratory (JPL) (Breckenridge, 1979).

All values are reported in SI units. While the timestamps were originally recorded as Portable Operating System Interface (POSIX), we subtracted the lowest timestamp as offset such that all datasets start at zero. This helps to avoid numerical difficulties when dealing with microsecond resolution timestamps of the events.

Images are provided as PNG files. The list of all images and their timestamps is provided in a separate file. The typical frame rate is 24 Hz, but it varies with the exposure time.

The IMU axes are aligned with the optical coordinate frame (i.e. the positive z-axis is identical to the optical axis and so are the x- and y-axes).

4.2. List of datasets

The provided datasets are summarized in Table 2 and Figure 3. All the datasets contain increasing speeds, different scenes, and varying DOFs:[7] for the `shapes`, `poster`, and `boxes` datasets, the motion first starts with excitation of each single DOF separately; then combined and faster excitations are performed. This leads to increasing difficulty and a higher event rate over time.

In the high-dynamic-range (HDR) sequences (`hdr_poster`, `hdr_boxes`, and `slider_hdr`), a spotlight was used to create large intrascene contrasts. For `hdr_poster`, we measured 80 lx and 2400 lx in the dark and bright areas, respectively.

The `outdoors` datasets were acquired in an urban environment both walking and running. While no ground truth is available, we returned precisely to the same location after a large loop.

The `dynamic` datasets were collected in a mock-up office environment viewed by the motion-capture system, with a moving person first sitting at a desk, then moving around.

A `calibration` dataset is also available, for instance in case the user wishes to use a different camera model or different methods for hand–eye calibration. The dimensions of the calibration pattern (a checkerboard) are 6×9 tiles of 40 mm. For the lower half of the table, different settings (lenses, focus, etc.) were used. Thus, while we provide the intrinsic calibration, no calibration datasets are available.

The `slider_close`, `slider_far`, `slider_hdr_close`, and `slider_hdr_far` datasets were recorded with a motorized linear slider parallel to a textured wall at 23.1 cm, 58.4 cm, 23.2 cm, and 58.4 cm, respectively.

For the datasets, we applied two different sets of biases (parameters) for the DAVIS, as listed in Table 3. The first set, labeled "indoors", was used in all datasets but `outdoors_walking`, `outdoors_running`, and `urban`, where the set "outdoors" was applied. For the simulated datasets, we used a contrast threshold of $\pm$ 15% and $\pm$ 20% for the `simulation_3planes` and `simulation_3walls`, respectively.

For the simulated scenes, we also provide the 3D world model in Blender (see Figure 3(l) and (m)). In addition to the intensity images and events, these datasets include a depth map for each image frame at 40 Hz, encoded as 32-bit floating-point values (in the OpenEXR data format).

5. Calibration

First, we calibrated the DAVIS intrinsically using a checkerboard pattern. Then, we computed the hand–eye calibration that we applied to the subsequent dataset recordings so that the ground-truth poses that we provide are those of the event camera (i.e. the "eye"), not those of the motion-capture trackable (i.e. the "hand") attached to the camera. We also included a calibration dataset in case a different camera model or improved hand–eye calibration method is required.

Table 2. List of datasets. Note that the calibration dataset only applies to the upper half of the table. The other datasets use different lenses and calibrations. GT: Ground truth. T: Duration. TS: Maximum translation speed. RS: Maximum rotational speed. NE: Number of events.

Name	Motion	Scene	GT	T [s]	TS [m/s]	RS [°/s]	NE [-]
shapes_rotation	Rotation, incr. speed	Figure 3(a)	yes	59,787844	0.83	730.17	23126288
shapes_translation	Translation, incr. speed	Figure 3(a)	yes	59,734202	2.6	270.78	17363976
shapes_6dof	Six DOFs, incr. speed	Figure 3(a)	yes	59,72497	2.35	714.78	17962477
poster_rotation	Rotation, incr. speed	Figure 3(b)	yes	59,791966	0.84	884.15	169350136
poster_translation	Translation, incr. speed	Figure 3(b)	yes	59,790562	2.58	240.46	100033286
poster_6dof	Six DOFs, incr. speed	Figure 3(b)	yes	59,832581	2.51	937.22	133464530
boxes_rotation	Rotation, incr. speed	Figure 3(c)	yes	59,818996	0.85	669.12	185688947
boxes_translation	Translation, incr. speed	Figure 3(c)	yes	59,776029	3.43	318.59	112388307
boxes_6dof	Six DOFs, incr. speed	Figure 3(c)	yes	59,772908	3.84	508.51	133085511
hdr_poster	Six DOFs, incr. speed	Figure 3(b)	yes	59,77225	2.28	597.09	102910720
hdr_boxes	Six DOFs, incr. speed	Figure 3(c)	yes	59,780373	2.94	591.71	118499744
outdoors_walking	Six DOFs, walking	Figure 3(d)	no†	133.434313	n/a	n/a	64517638
outdoors_running	Six DOFs, running	Figure 3(d)	no†	87.619327	n/a	n/a	98572164
dynamic_rotation	Rotation, incr. speed	Figure 3(e)	yes	59,759053	0.45	542.29	71324510
dynamic_translation	Translation, incr. speed	Figure 3(e)	yes	59.798989	1.86	227.2	35809924
dynamic_6dof	Six DOFs, incr. speed	Figure 3(e)	yes	59,730322	2.91	627.12	57174637
calibration	Six DOFs, slow	Figure 3(f)	yes	59,791377	0.32	67.21	21340629
office_zigzag	Six DOFs, zigzag, slow	Figure 3(g)	no	10,914726	n/a	n/a	7735308
office_spiral	Six DOFs, spiral, slow	Figure 3(g)	no	11,180662	n/a	n/a	6254774
urban	Linear, slow	Figure 3(h)	no	10,730572	n/a	n/a	5359539
slider_close	Linear, const. speed	Figure 3(i)	yes*	6.459701	0.16	0	4032668
slider_far	Linear, const. speed	Figure 3(i)	yes*	6.396429	0.16	0	3442683
slider_hdr_close	Linear, const. speed	Figure 3(j)	yes*	6.53924	0.16	0	3337787
slider_hdr_far	Linear, const. speed	Figure 3(j)	yes*	6.467819	0.16	0	2509582
slider_depth	Linear, const. speed	Figure 3(k)	yes*	3.386478	0.32	0	1078541
simulation_3planes	Translation, circle	Figure 3(l)	yes#	2.0	0.63	0	6870278
simulation_3walls	Six DOFs	Figure 3(m)	yes#	2.0	5.31	109.02	4104833

†Same start and end pose after a large loop.
*Ground truth from motorized linear slider. No IMU data due to vibrations.
#Simulated DAVIS using Blender. No IMU data included.

Table 3. List of biases applied to the DAVIS. The DAVIS uses two stages of biases, coarse and fine, which we report here.

Bias	Indoors		Outdoors	
	Coarse	Fine	Coarse	Fine
DiffBn	2	39	4	39
OFFBn	1	62	4	0
ONBn	4	200	6	200
PrBp	3	72	2	58
PrSFBp	3	96	1	33
RefrBp	3	52	4	25

5.1. Intrinsic camera calibration

We used the standard pinhole camera model with radial–tangential distortion using the implementation of Robot Operating System (commonly known as ROS) and OpenCV.[8] We used three radial distortion coefficients (k_1, k_2, and $k_3 = 0$) and two for tangential distortion (p_1 and p_2). The distortion coefficients are listed in `calib.txt` in the same order as in OpenCV. We provide a dataset for post-calibration in case another method is preferred.

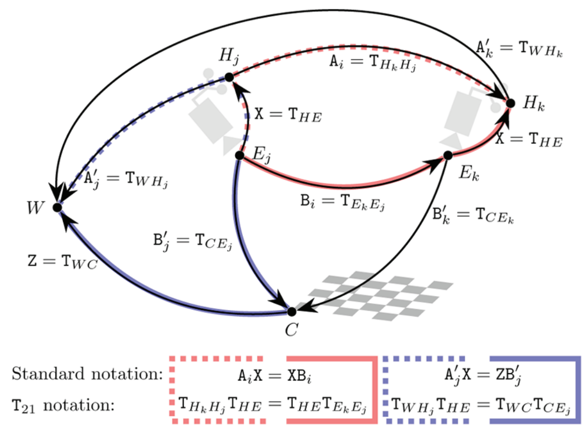

Fig. 4. Hand–eye calibration. Coordinate frames and transformations involved in the case of the hand–eye device at two different positions (j and k). The red loop between two stations of the hand–eye device is used in the first type of hand–eye calibration problem, of the form $\mathtt{A}_i\mathtt{X} = \mathtt{X}\mathtt{B}_i$, and the blue loop is used in the second type of hand–eye calibration problem, of the form $\mathtt{A}'_j\mathtt{X} = \mathtt{Z}\mathtt{B}'_j$. We use a combination of both approaches to solve for the constant, hand–eye calibration transform $\mathtt{X}$.

5.2. *Hand–eye calibration*

For the indoor datasets, we provide accurate and high-frequency (200 Hz) pose data from a motion-capture system. However, the coordinate frame used by the motion-capture system is different from the optical coordinate frame of the DAVIS. Therefore, we performed a hand–eye calibration before acquiring the datasets. Figure 4 shows the coordinate frames and transformations used to solve the hand–eye calibration problem. The frames are those of the world W, the hand H, the camera E (Figure 1(a)), and the checkerboard C. For the transformations, Figure 4 shows both the compact standard notation of hand–eye calibration problems and a more explicit one: the Euclidean transformation $\mathtt{T}_{ba}$ (4×4 homogeneous matrix representation) maps points from frame a to frame b according to $\mathbf{P}_b = \mathtt{T}_{ba}\mathbf{P}_a$.

More specifically, we first use a linear algorithm (Tsai and Lenz, 1989) to provide an initial solution of the hand–eye calibration problem $\{\mathtt{A}_i\mathtt{X} = \mathtt{X}\mathtt{B}_i\}_{i=1}^{N}$, where $\mathtt{A}_i \leftrightarrow \mathtt{B}_i$ are N correspondences of relative hand–hand ($\mathtt{A}_i := \mathtt{T}_{H_kH_j}$) and eye–eye ($\mathtt{B}_i := \mathtt{T}_{E_kE_j}$) poses at different times (j and k), respectively, and $\mathtt{X} := \mathtt{T}_{HE}$ is the unknown eye-to-hand transformation. Then, using the second formulation of hand–eye calibration problems, of the form $\{\mathtt{A}'_j\mathtt{X} = \mathtt{Z}\mathtt{B}'_j\}_{j=1}^{N+1}$, where $\mathtt{A}'_j := \mathtt{T}_{WH_j}$ and $\mathtt{B}'_j := \mathtt{T}_{CE_j}$ are the hand-to-motion-capture and eye-to-checkerboard transformations for the jth pose, respectively, we refined $\mathtt{T}_{HE}$ by jointly estimating the hand–eye $\mathtt{X}$ and robot–world $\mathtt{Z} := \mathtt{T}_{WC}$ (i.e. motion-capture–checkerboard) transformations that minimize the reprojection error in the image plane:

$$\min_{\mathtt{X},\mathtt{Z}} \sum_{mn} d^2\big(\mathbf{x}_{mn}, \hat{\mathbf{x}}_{mn}(\mathtt{X}, \mathtt{Z}; \mathtt{A}'_m, \mathbf{P}_n\mathtt{K})\big)$$

where $d^2(\mathbf{x}_{mn}, \hat{\mathbf{x}}_{mn})$ is the squared Euclidean distance between the measured projection $\mathbf{x}_{mn}$ of the nth checkerboard corner $\mathbf{P}_n$ on the mth camera and the predicted corner $\hat{\mathbf{x}}_{mn} = \mathbf{f}(\hat{\mathtt{B}}'_m; \mathbf{P}_n, \mathtt{K})$, which is a function of the intrinsic camera parameters $\mathtt{K}$ and the extrinsic parameters $\hat{\mathtt{B}}'_m := \mathtt{Z}^{-1}\mathtt{A}'_m\mathtt{X}$ predicted using the motion-capture data. This non-linear least-squares problem is solved iteratively using the Gauss–Newton method. The initial value of $\mathtt{Z}$ is given by $\mathtt{Z} = \mathtt{A}'_1\mathtt{X}\mathtt{B}'^{-1}_1$, with $\mathtt{X}$ provided by the above-mentioned linear algorithm. We included a dataset for post-calibration in case another method is preferred.

The ground-truth pose gives the position and orientation of the event camera with respect to the world (i.e. the motion-capture system). Hence, it has already incorporated the computed hand–eye transformation. That is, while the motion-capture system outputs $\mathtt{T}_{WH_j}$, we apply the hand–eye calibration $\mathtt{T}_{HE} \equiv \mathtt{T}_{H_jE_j}\ \forall j$ and directly report $\mathtt{T}_{WE_j} = \mathtt{T}_{WH_j}\mathtt{T}_{H_jE_j}$ as the ground-truth pose.

6. Known issues

6.1. *Clock drift and offset*

The clocks from the motion-capture system and the DAVIS are not hardware-synchronized. We observed clock drift of about 2 ms/min. To counteract the clock drift, we reset the clocks before each dataset recording. Since all datasets are rather short (in the order of 1 min), the effect of drift is negligible. A small, dataset-dependent offset between the DAVIS and motion-capture timestamps is present since the timestamps were reset in software.

Dataset website

All datasets and the simulator can be found on the web: `http://rpg.ifi.uzh.ch/davis_data.html` A video containing visualizations of the datasets: `https://youtu.be/bVVBTQ7l36I`

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by SNSF-ERC Starting Grant, the DARPA FLA Program, by the Google Faculty Research Award, by the Qualcomm Innovation Fellowship, by the National Centre of Competence in Research Robotics (NCCR), by the Swiss National Science Foundation, and by the UZH Forschungskredit (Grant No: K-34142-03-01).

Notes

1. Video illustration: https://youtu.be/LauQ6LWTkxM
2. IMU data sheet: https://store.invensense.com/ProductDetail/MPU6150-invensense/470090/
3. https://www.blender.org/
4. https://www.itu.int/rec/R-REC-BT.601
5. https://github.com/uzh-rpg/rpg_davis_simulator

6. We use an OptiTrack system from NaturalPoint.
7. The DAVIS was moved by hand: the dominant motion is described.
8. http://wiki.ros.org/camera_calibration/Tutorials/Monocular Calibration

References

Bardow P, Davison AJ and Leutenegger S (2016) Simultaneous optical flow and intensity estimation from an event camera. In: *IEEE international conference on computer vision and pattern recognition (CVPR)*. pp. 884–892.

Barranco F, Fermuller C, Aloimonos Y, et al. (2016) A dataset for visual navigation with neuromorphic methods. *Frontiers in Neuroscience* 10(49).

Benosman R, Clercq C, Lagorce X, et al. (2014) Event-based visual flow. *IEEE Transactions on Neural Networks and Learning Systems* 25(2): 407–417.

Brandli C, Berner R, Yang M, et al. (2014) A 240x180 130dB 3us latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits* 49(10): 2333–2341.

Breckenridge WG (1979) Quaternions – proposed standard conventions. Technical report, NASA Jet Propulsion Laboratory, La Canada Flintridge, CA.

Censi A and Scaramuzza D (2014) Low-latency event-based visual odometry. In: *IEEE international conference on robotics and automation (ICRA)*. pp. 703–710.

Cook M, Gugelmann L, Jug F, et al. (2011) Interacting maps for fast visual interpretation. In: *International joint conference on neural networks (IJCNN)*, pp.770–776.

Delbruck T, Villanueva V and Longinotti L (2014) Integration of dynamic vision sensor with inertial measurement unit for electronically stabilized event-based vision. In: *International conference on circuits and systems (ISCAS)*, pp.2636–2639.

Gallego G and Scaramuzza D (2016) Accurate angular velocity estimation with an event camera. *IEEE Robotics and Automation Letters*. 2(2): 632–639. DOI: 10.1109/LRA.2016.2647639.

Kim H, Handa A, Benosman R, et al. (2014) Simultaneous mosaicing and tracking with an event camera. In: *British machine vision conference (BMVC)*. pp. 1–12.

Kim H, Leutenegger S and Davison A (2016) Real-TIME 3D reconstruction and 6-DoF tracking with an event camera. In: *European conference on computer vision (ECCV)*. pp. 349–364. Available at: http://link.springer.com/chapter/10.1007/978-3-319-46466-4_21

Kueng B, Mueggler E, Gallego G, et al. (2016) Low-latency visual odometry using event-based feature tracks. In: *IEEE/RSJ international conference on intelligent robots and systems (IROS)*. pp. 16–23.

Mueggler E, Gallego G and Scaramuzza D (2015) Continuous-time trajectory estimation for event-based vision sensors. In: *Robotics: Science and systems (RSS)*. pp. 1–9.

Mueggler E, Huber B and Scaramuzza D (2014) Event-based, 6-DOF pose tracking for high-speed maneuvers. In: *IEEE/RSJ international conference on intelligent robots and systems (IROS)*. pp. 2761–2768.

Rebecq H, Gallego G and Scaramuzza D (2016a) EMVS: Event-based multi-view stereo. In: *British machine vision conference (BMVC)*. pp. 1–11.

Rebecq H, Horstschäfer T, Gallego G, et al. (2016b) EVO: A geometric approach to event-based 6-DOF parallel tracking and mapping in real-time. *IEEE Robotics and Automation Letters*. 2(2): 593–600. DOI: 10.1109/LRA.2016.2645143.

Reinbacher C, Graber G and Pock T (2016) Real-time intensity-image reconstruction for event cameras using manifold regularisation. In: *British machine vision conference (BMVC)*. pp. 1–12.

Rueckauer B and Delbruck T (2016) Evaluation of event-based algorithms for optical flow with ground-truth from inertial measurement sensor. *Frontiers in Neuroscience* 10(176). pp. 1–17.

Tsai RY and Lenz RK (1989) A new technique for fully autonomous and efficient 3D robotics hand/eye calibration. *IEEE Transactions on Robotics* 5(3): 345–358.

Weikersdorfer D, Hoffmann R and Conradt J (2013) Simultaneous localization and mapping for event-based vision systems. In: *International conference on computer vision systems (ICVS)*. pp. 133–142.

Yuan W and Ramalingam S (2016) Fast localization and tracking using event sensors. In: *IEEE international conference on robotics and automation (ICRA)*. pp. 4564–4571.