

Multi-Event-Camera Depth Estimation and Outlier Rejection by Refocused Events Fusion

Suman Ghosh and Guillermo Gallego*

Event cameras are bio-inspired sensors that offer advantages over traditional cameras. They operate asynchronously, sampling the scene at microsecond resolution and producing a stream of brightness changes. This unconventional output has sparked novel computer vision methods to unlock the camera's potential. Here, the problem of event-based stereo 3D reconstruction for SLAM is considered. Most event-based stereo methods attempt to exploit the high temporal resolution of the camera and the simultaneity of events across cameras to establish matches and estimate depth. By contrast, this work investigates how to estimate depth without explicit data association by fusing disparity space images (DSIs) originated in efficient monocular methods. Fusion theory is developed and applied to design multi-camera 3D reconstruction algorithms that produce state-of-the-art results, as confirmed by comparisons with four baseline methods and tests on a variety of available datasets.

1. Introduction

Intelligent navigation in our complex 3D world relies on robust and efficient visual perception, which is challenging for autonomous robots. However, humans use vision very efficiently to navigate 3D environments, even in novel scenarios. Inspired by human vision, neuromorphic spike-based sensing and processing has been recently investigated for robot vision^[1,2] and retinal implants.^[3]

Event cameras, such as the dynamic vision sensor^[4–6] (DVS), are neuromorphic sensors that acquire visual information very differently from traditional cameras. They sample the scene

asynchronously, producing a stream of spikes, called “events,” that encode the time, location, and sign of per-pixel brightness changes. Event cameras possess outstanding properties compared to traditional cameras: very high dynamic range (HDR), high temporal resolution ($\approx \mu\text{s}$), temporal redundancy suppression, and low power consumption. These properties offer the potential to tackle challenging scenarios for standard cameras (high speed and/or HDR).^[7–11] However, this calls for novel methods to process the unconventional output of event cameras to unlock their capabilities.^[2]

In this work, we tackle the problem of event-based stereo 3D reconstruction for visual odometry (VO) and Simultaneous

Localization And Mapping (SLAM). An efficient SLAM system is critical for the navigation of autonomous intelligent agents (like field robots) in challenging unstructured environments, especially in extreme ones like offshore drilling, nuclear power plants, etc.^[12] An event-based SLAM system has the potential to efficiently overcome difficult scenarios in these applications.^[13–15] For example, the HDR advantages of event cameras translate into high-fidelity depth maps in difficult lighting conditions, as demonstrated by a broad variety of works in VO/SLAM (Figure 8,^[14] 12,^[13] and 15^[15]).


Our work is inspired by EVO,^[13] which is the state of the art (SOTA) in event-based monocular VO. The effectiveness of EVO is largely due to its mapping module, event-based multi-view stereo (EMVS),^[10] which enables 3D reconstruction without the need to recover image intensity, without having to explicitly solve for data association between events, and without the need of a GPU (it is fast on a standard CPU—e.g., speed of $1.20 \text{ Mev s}^{-1} \text{ core}^{-1[10]}$). Additionally, EMVS admits an interpretation in terms of event refocusing or event alignment (contrast maximization),^[16] which is the state-of-the-art framework to tackle other vision problems.^[17–25] Our goal is to extend EMVS to the multi-camera setting (i.e., two or more event cameras in a multi-view configuration sharing a common clock), and in particular to the stereo setting, to benefit from these advantages and connections (Figure 1). In the process, we revisit the event simultaneity assumption used in stereo depth estimation and develop a theory of fusion of refocused events, which could be useful in other problems, such as feature or camera tracking.^[26]

In summary, our contributions are: 1) Simple, efficient and extensible solutions to the problem of event-based stereo 3D reconstruction for SLAM using a correspondence-free approach.

S. Ghosh, G. Gallego
Department of Electrical Engineering and Computer Science
Technische Universität Berlin
10623 Berlin, Germany
E-mail: guillermo.gallego@tu-berlin.de

G. Gallego
Einstein Center Digital Future
10117 Berlin, Germany

G. Gallego
Science of Intelligence Excellence Cluster
10587 Berlin, Germany

 The ORCID identification number(s) for the author(s) of this article can be found under <https://doi.org/10.1002/aisy.202200221>.

© 2022 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH. This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

DOI: 10.1002/aisy.202200221

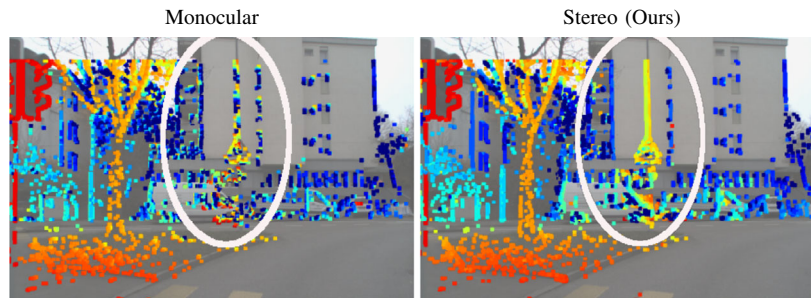


Figure 1. Semi-dense depth maps estimated by event-based monocular^[10] and stereo methods. Stereo is beneficial for more accurate estimation and outlier removal compared to monocular depth estimation (e.g., traffic sign in the center). Depth is pseudo-colored, from red (close) to blue (far). Color frames are only shown for visualization. Data from ref. [55].

We investigate early event data fusion strategies in two orthogonal directions: between cameras (“spatial stereo”, Section 3.3) and along time (“temporal stereo”, Section 3.5). 2) The investigation of several functions to fuse refocused events (using, e.g., generalized means (Section 3.4)), and its application to the two mentioned directions. 3) A comprehensive experimental evaluation of five publicly available datasets and comparing against several baseline methods, producing state-of-the-art results (Section 4). We also show how the method can naturally handle multi event-camera setups with linear complexity.

This research aims at developing robust multi-camera visual perception systems for the navigation of artificial intelligent systems in challenging environments, like stereo depth perception for SLAM and attention in robots.^[27,28]

2. Related Work

2.1. What

Stereo depth estimation using event cameras has been an interesting problem ever since the first event camera was invented by Mahowald and Mead in the 1990s.^[29] As such, they simultaneously designed a stereo chip^[30] to implement Marr and Poggio’s cooperative stereo algorithm.^[31] This approach has inspired a lot of literature that focuses on 3D reconstruction over short time intervals (“instantaneous stereo”).^[32–34] These methods work well with stationary cameras in uncluttered scenes (where events are caused only by a few moving objects), thus enabling the 3D reconstruction of sparse, dynamic scenes. For a detailed survey on these stereo methods, we refer to.^[35,36] In contrast, stereo event-based 3D reconstruction for VO/SLAM has been addressed recently.^[15,37] It assumes a static world and known camera motion (e.g., from a tracking method) to assimilate events over longer time intervals, to increase parallax and produce more accurate semi-dense depth maps. Some other works estimate depth by combining an event camera with other devices, such as light projectors^[8,38,39] or a motorized focal lens,^[40] which are different from our hardware setup and application.

2.2. How

Depth estimation with stereo event cameras is predominantly based on exploiting the epipolar constraint and the assumption

of temporal coincidence of events across retinas, namely that a moving object produces events of the same timestamps on both cameras.^[38,41,42] This aims at exploiting the high temporal resolution and redundancy suppression of event cameras to establish event matches across image planes and then triangulate. It is also known as event simultaneity or temporal consistency,^[37] and it is analogous to photometric consistency in traditional cameras. This assumption does not strictly hold,^[43,44] and so it is relaxed to account for temporal noise (jitter and delay). Essentially event simultaneity is exploited to solve the data association problem (establishing event matches), which is a well-known difficult problem due to the little information carried by each event and their dependency on motion direction (changing the “appearance” of events^[2,45]).

The aforementioned ideas are used in the mapping module of ref. [15] the state-of-the-art stereo 3D reconstruction method for VO/SLAM. In this method, temporal consistency is measured across space–time neighborhoods of events by first converting the events into time surfaces (TSs)^[46] and then comparing their spatial neighborhoods. Stereo point matches are established and provide depth estimates which are fused in a probabilistic way using multiple TSs to produce a more accurate semi-dense inverse depth map.

In contrast, we investigate a new way of doing stereo, without explicitly using event simultaneity and hence without establishing event matches. Therefore, to the best of our knowledge, we completely depart from previous event-based stereo methods. Correspondence-free approaches for depth estimation have been proposed for frame-based cameras.^[47–49] For example,^[49] proposes a method to estimate affine fundamental matrices without explicit correspondences, but it does not solve the full problem of depth estimation. Ref. [48] aims to solve the problem of stereo depth estimation, but for the special case of planar or quasi-planar scenes. It is an extension of ref. [50] for arbitrary camera positions. In contrast, our method solves the full-depth estimation problem and is not limited to the planar case, following the seminal idea of ref. [47] to sweep space and exploit the sparsity of scene edges. Inspired by refs. [10,47], we circumvent the data association task by leveraging the sparsity of events (event cameras naturally highlight edges, which are sparse, in hardware) and by exploiting the continuous set of camera viewpoints at which events are available. This provides a rich collection of back-projected rays through the events to estimate scene

structure. Our contributions pertain to the processing (e.g., fusion) of such back-projected rays or “refocused events,” which has not been considered before (since^[10] and newer approaches^[51] do not consider data fusion, e.g., across cameras).

3. Event-Based Stereo Depth Estimation

This section reviews how an event camera works (Section 3.1) and the monocular method EMVS (Section 3.2) before presenting our stereo depth estimation approach. Two main event fusion directions are presented: fusion of camera views (Section 3.3) using one of several functions (Section 3.4), and fusion of multiple time intervals (Section 3.5). Then, we revisit the event simultaneity assumption (Section 3.6) and analyze the computational complexity of the approach (Section 3.7).

3.1. How an Event Camera Works

Event cameras, such as the DVS,^[4] are bio-inspired sensors that capture pixel-wise brightness changes, called events, instead of brightness images. An event $e_k \doteq (x_k, t_k, p_k)$ is triggered when the logarithmic brightness L at a pixel exceeds a contrast sensitivity $\theta > 0$

$$L(x_k, t_k) - L(x_k, t_k - \Delta t_k) = p_k \theta \quad (1)$$

where $x_k \doteq (x_k, y_k)^T$, t_k (in) and $p_k \in \{+1, -1\}$ are the spatio-temporal coordinates and polarity of the brightness change, respectively, and $t_k - \Delta t_k$ is the time of the previous event at the same pixel x_k . Hence, each pixel has its own sampling rate, which depends on the visual input. Assuming constant illumination, pixels produce events proportionally to the amount of scene motion and texture.

3.2. EMVS: Monocular 3D Reconstruction

The problem of monocular depth estimation with an event camera consists of estimating the 3D structure of the scene given the events and the camera poses (i.e., position and orientation) as the sensor moves through the scene. The method in ref. [10] solves this problem, called EMVS, in two main steps: it builds a disparity space image (DSI) using a space sweeping approach^[47] and then detects the local maxima of the DSI. The key idea is that, as the camera moves, events are triggered at an almost continuous set of viewpoints, which are used to back-project events into space in the form of rays (a DSI). The local maxima of the ray density (where many rays intersect, as shown in Figure 2) are candidate locations for the 3D edges that produce the events. Specifically, the DSI is discretized on a projective voxel grid defined at a reference view, and local maxima are detected along viewing rays, thus producing a semi-dense depth map. Events are processed in packets of about 0.21 M events. The key benefits of EMVS are its simplicity, accuracy, efficiency (real-time, with $\approx 1.2 \text{ Mev s}^{-1}$ throughput per CPU core^[10]), and that it estimates depth without explicit data association.

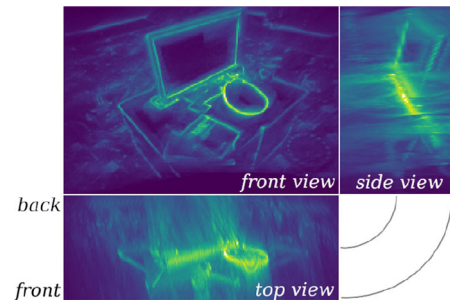


Figure 2. Disparity space image (DSI) projections of a non-planar scene (*rpg_monitor* data from ref. [37], also in Figure 5). DSI values (i.e., ray counts) are pseudo-colored, from blue (low) to yellow (high). The DSI has dimensions $w \times h \times N_z$, according to the resolution of the event camera DAVIS240 ($w = 240, h = 180$ pix) and the number of inverse-depth planes used, $N_z = 100$. The three DSI max-projections are: front view (top left, 240×180 pix), top view (bottom left, 240×100 pix) and side view (top right, 100×180 pix).

3.3. Fusion Across Cameras

We consider the problem of depth estimation from two synchronized and calibrated event cameras rigidly attached. Hence, the input consists of a stereo event stream and poses, and the desired output is a (semi-dense) depth map or, equivalently, a 3D point cloud with the scene structure (Figure 3).

3.3.1. Challenges and Proposed Architecture

A naive solution to the problem consists of running two instances of EMVS, one per camera, and fusing the resulting point clouds into a single one, including post-processing to mitigate redundant 3D points. This is a late-fusion approach, which greatly ignores the benefits that arise from having two cameras observing the same scene.

By contrast, we seek to perform fusion earlier in the processing pipeline: at the DSI stage. Hence, the first technical challenge is to define the DSI. EMVS defines a DSI per camera, located at a RV along the camera’s trajectory. However, the fusion of DSIs at two different RVs is prone to resampling errors. Thus it is key to define a common DSI location for both cameras.

The second challenge is to investigate sensible fusion strategies. Our approach includes, as a particular case, that of back-projecting the events from both cameras into a single DSI and simply counting rays. This is equivalent to the scenario of a single event camera that moves twice through the scene, with different motions, but uses the same DSI to aggregate rays. It doubles the ray count in the DSI, but summation discards valuable information for fusion, such as how many rays originate in each camera: given eight rays at a point, it is preferable to have four rays from each camera than an unbalanced situation (a 3D edge seen only by one camera).

To deal with the aforementioned challenges, we define two DSIs at a common reference view: having one DSI per camera allows us to preserve the origin of the event data, and having geometrically aligned DSIs avoids resampling errors during fusion. Without loss of generality, let the RV be a point along the

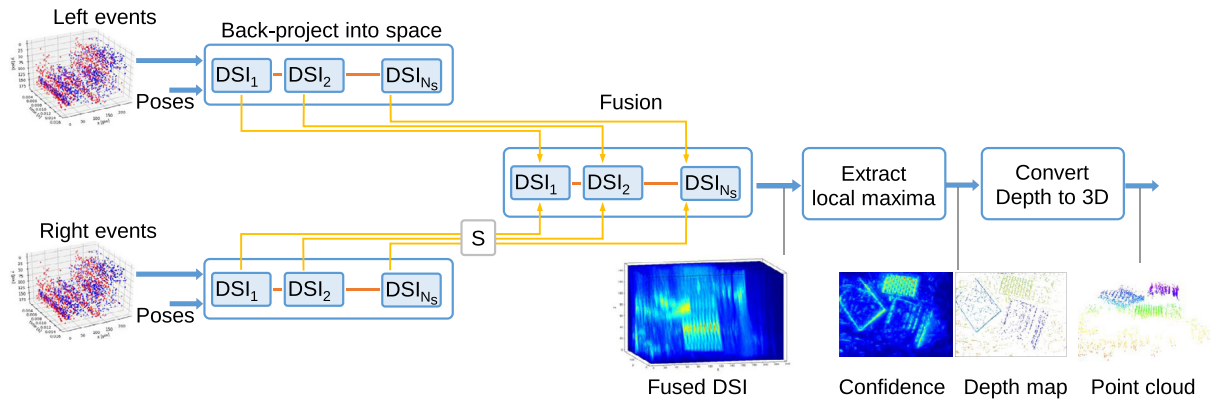


Figure 3. Our method takes as input the events from two or more synchronized, rigidly attached event cameras and their poses, and estimates the scene depth. Using space sweeping, it builds ray density DSIs from each camera data and fuses them into one DSI (Section 3.4), from which the 3D structure of the scene is extracted in the form of a semi-dense depth map, which may be cast into a point cloud. Fusion across cameras (Section 3.3) is represented with yellow lines, and temporal fusion (Section 3.5) with red lines. The optional shuffling block ("S") is presented in Section 3.6.

Algorithm 1. Stereo Event Fusion Across Cameras.

- 1: Input: stereo events in the interval $[0, T]$, camera trajectory, camera calibration (intrinsic and extrinsic).
- 2: Define a single reference view (RV) for both DSIs, coinciding with the left camera pose at say $t = T/2$.
- 3: Create 2 DSIs by back-projecting events from each camera.
- 4: Fusion: compute the pointwise harmonic mean of the DSIs.
- 5: Extract depth and confidence maps from the fused DSI f :

$$Z^*(x, y) \doteq \arg \max_f (X(x, y)), c^*(x, y) \doteq \max_f (X(x, y)).$$

trajectory of the left camera. We investigate how to compare and fuse the ray densities from each event camera. Figure 3 shows the block diagram of our stereo approach. For now, assume there is $N_s = 1$ DSI per camera ($N_s > 1$ is presented in Section 3.5). First, the aligned DSIs are populated with back-projected events from each camera, then they are fused (combined) into a single one (e.g., using a voxel-wise harmonic mean or other similarity scores (Section 3.4)), and finally, local maxima are extracted to produce a semi-dense depth map. The steps are specified in **Algorithm 1**.

3.3.2. Intuitive Example

To illustrate key differences between EMVS (monocular) and the stereo Algorithm 1, we use a sequence acquired with two event cameras^[52] performing a 1D motion (translation along the X-axis, using a linear slider). As input to EMVS, we use the data from the left camera. **Figure 4** shows the evolution of the DSIs as time progresses, i.e., as the camera rig moves and more events are acquired and back-projected onto the DSIs (and fused in the stereo case). We plot projections of the DSI along its three coordinate axes (like in Figure 2) at the reference viewpoint RV.

As Figure 4 shows, the stereo DSI (bottom row) converges faster to the 3D structure of the scene than the monocular DSI (top row). This is especially noticeable in the top views: only one set of nearly parallel rays, poor for triangulation, is visible in

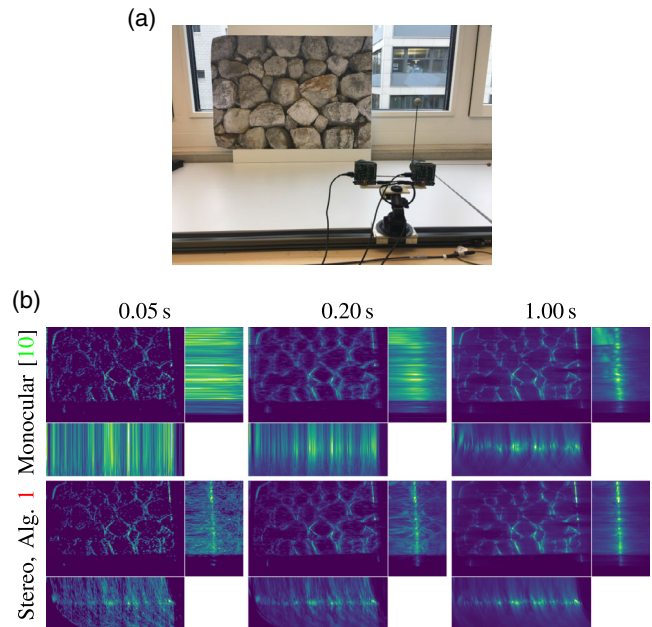


Figure 4. Intuitive Example: Monocular versus stereo method on planar rock scene and 1D motion along the camera's X axis. Plots of the evolution of the DSI projections (see text) for different methods (rows) as time increases (columns). The 3D edge patterns in the DSI (in yellow) are less localized in event-based multi-view stereo (EMVS) (top row) than in stereo Algorithm 1 (bottom).

the monocular case. By contrast, the top views of the stereo DSIs show two sets of rays, one from each camera: the rays from the left camera are nearly straight, whereas the rays from the right camera are curved due to inverse depth parametrization of the DSI grid and the fact that the DSI is projective. In both scenarios, the rays intersect at multiple voxels and as time progresses the true intersection locations dominate over others, i.e., the 3D structure emerges by event refocusing.^[10,17] Additionally, in the stereo case refocusing is combined with the proposed fusion functions (Section 3.4) to speed up the emergence and better

highlight 3D structure. In Algorithm 1, this is achieved by the harmonic mean, which deemphasizes the non-intersecting parts of the rays.

3.3.3. Output of the Stereo Method

Figure 5 shows the output of Algorithm 1 on two scenes. After DSI fusion, Algorithm 1 extracts a depth map by locating the DSI maxima along each viewing ray (through RV pixel $\mathbf{x} = (x, y)^T$). Letting $f: \mathbb{R}^3 \rightarrow \mathbb{R}_{\geq 0}$ be the fused DSI, its maxima provides the confidence or “contrast” map $c(\mathbf{x}) = f(\mathbf{x}, Z^*(\mathbf{x}))$ and the depth map $Z^*(\mathbf{x})$. Adaptive Gaussian thresholding (AGT) selects the pixels with the highest local value, thus making the depth maps semi-dense. A median filter is applied to remove isolated points. The front-view projection of the DSI in Figure 2 corresponds to the confidence map, which is called this way because it is used in AGT to “select the most confident pixels in the depth map”^[10] (since voxels with many ray intersections are more likely to capture true 3D points than voxels with few ray intersections).

3.4. DSI Fusion Functions

DSI fusion is the central part of our method (Figure 3). It takes two ray density DSIs on the same region of space as input (one per camera) and produces a merged DSI, which is then used to extract depth information (candidate locations of 3D edges). So, what are sensible ways to compare two DSIs? Ray density DSIs have very different statistics from natural images, hence standard similarity metrics for image patches may not be the most appropriate ones.^[53]

Formally, let $\mathcal{E}_l = \{e_k^l\}_{k=1}^{N_l^l}$ and $\mathcal{E}_r = \{e_k^r\}_{k=1}^{N_r^l}$ be stereo events over some time interval $[0, T]$, and $f_l, f_r: V \subset \mathbb{R}^3 \rightarrow \mathbb{R}_{\geq 0}$ be the ray densities (DSIs) defined over a volume V

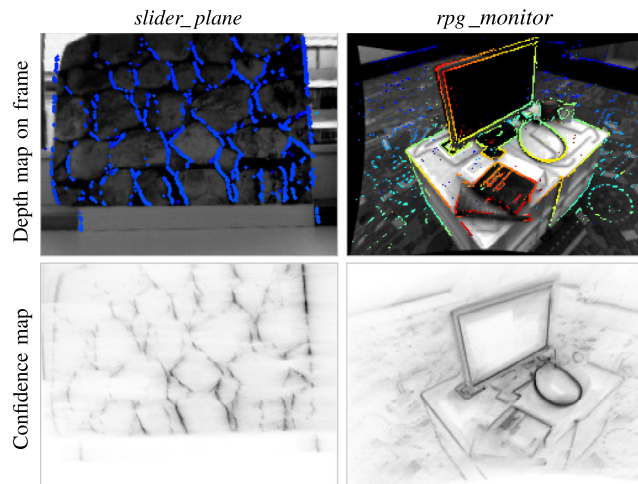


Figure 5. Output of Stereo Algorithm 1 on two scenes. Top: Our method produces a semi-dense depth map of the scene (color coded from red (close) to blue (far), overlaid on a grayscale frame of the dynamic and active pixel vision sensors (DAVIS)^[52], and a confidence map (bottom) with the maximum DSI value along each reference view pixel, in negated scale (bright = small; dark = large).

$$f_l(X) = \sum_{k=1}^{N_l^l} \delta(X - X_k^l(e_k^l)) \quad (2)$$

where $X_k^l(e_k^l) = (\mathbf{x}_k^{lT}, Z)^T$ is a 3D point on the back-projected ray through event e_k^l , at depth Z with respect to the RV. Events are transferred to RV using the continuous motion of the cameras and candidate depth values $Z \in [Z_{\min}, Z_{\max}]$

$$x_k^l = W(e_k^l, P^l(t_k), P_v, Z) \quad (3)$$

where $P^l(t)$ is the pose of the left event camera at time t and P_v is the pose of RV. The warp W corresponds to the planar homography induced by a plane parallel to the image plane of RV and at the given depth Z . In a coordinate system adapted to RV (i.e., $P_v = (I, 0)$), the planar homography is given by the 3×3 homogeneous matrix

$$H_W \sim \left(R + \frac{1}{Z} \mathbf{t} \mathbf{e}_3^T \right)^{-1} \quad (4)$$

where $P^l(\mathbf{t}_k) = (R, \mathbf{t})$ and $\mathbf{e}_3 = (0, 0, 1)^T$. A similar formula applies to compute f_r from \mathcal{E}_r and the corresponding camera poses. In practice, DSIs are discretized over a projective voxel grid with N_Z depth planes in $[Z_{\min}, Z_{\max}]$, and the Delta δ in Equation (2) is approximated by bilinear voting.^[10,54] Hence, each voxel counts the number of event rays that pass through it.

Next, letting $u = f_l(X)$, $v = f_r(X)$ be the values of the DSIs at a 3D point X , we seek to define a fused value $g(X)$. For simplicity, we consider metrics operating in a point-wise (i.e., voxel-wise) manner, i.e., with a slight abuse of notation

$$g(X) \equiv g(f_l(X), f_r(X)) = g(u, v) \quad (5)$$

The metrics considered are the following

$$A(u, v) \doteq (u + v)/2 \quad \text{Arithmetic mean} \quad (6)$$

$$G(u, v) \doteq \sqrt{uv} \quad \text{Geometric mean} \quad (7)$$

$$H(u, v) \doteq 2/(u^{-1} + v^{-1}) \quad \text{Harmonic mean} \quad (8)$$

$$\text{RMS}(u, v) \doteq \sqrt{\frac{1}{2}(u^2 + v^2)} \quad \text{Quadratic mean} \quad (9)$$

$$\min(u, v) \quad \text{Minimum} \quad (10)$$

$$\max(u, v) \quad \text{Maximum} \quad (11)$$

They are special cases of the Generalized mean (power mean or Hölder mean) and satisfy an order (see Figure 6)

$$\min \leq H \leq G \leq A \leq \text{RMS} \leq \max \quad (12)$$

where the equal sign holds if and only if $u = v$. Equation (12) also establishes a qualitative order of the depth maps obtained after DSI fusion with the corresponding function. The arithmetic mean A (i.e., averaging ray densities) corresponds to the above-mentioned particular case of counting the back-projected stereo events on a single DSI. Hence, functions performing worse than this case are not pursued.

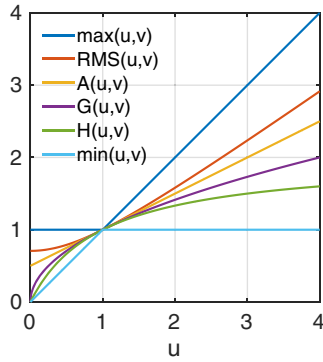


Figure 6. Fusion functions considered, for $u = [0, 4]$, $v = 1$.

3.4.1. What Are the Requirements for A Good Fusion Function?

Intuitively, given two ray densities defined on the same volume, a fusion function should emphasize the regions of high ray density on both DSIs and deemphasize the rest. It is not sufficient for one of the two densities to be large at a point X to signal the presence of a 3D edge; both densities have to be similar and large at X . The arithmetic mean A and its dominant functions (e.g., Root Mean Square (RMS) and max in Equation (12)) do not satisfy this “AND” logic, whereas the geometric mean, harmonic mean, and min functions do satisfy it (e.g., a large value $G(u, v)$ can only be achieved if both u and v are large).

Mathematically, this requirement is well described by the concavity properties of the function (plot in Figure 6). The arithmetic mean A and functions below it are concave (assuming non-negative inputs). Further, G , H , and \min are strictly concave. As the experiments will show, fusion using G still lets notorious outliers pass. Functions H and \min deemphasize considerably more than G . The \min function saturates strictly, treating values $u > v$ as $u = v$, hence clipping and discarding potentially beneficial information about ray density values. The harmonic mean H shows strong concavity and varies smoothly with both input arguments, without discarding information. H is dominated by the minimum of its arguments

$$\min(u, v) \leq H(u, v) \leq 2 \min(u, v) \quad (13)$$

(in terms of the plot in Figure 6 ($v = 1$), the green curve is bounded: $H(u, 1) \leq 2$). The goal of the present work is to introduce and study fusion functions rather than to select a single “best one.” To narrow the discussion, we often use a subset of the fusion functions.

3.4.2. Interpretation in Terms of Contrast Maximization

The proposed stereo fusion method is related to contrast/focus maximization.^[16,17] The depth slices of the DSIs count refocused events (warped by back-projection), i.e., they constitute so-called images of warped events (IWEs).^[16] The fused DSI can be interpreted as a similarity score between refocused events. Since fusion functions such as H try to emphasize DSI regions with large and similar values, stereo Algorithm 1 tries to maximize

the similarity score between refocused events (in-focus effect) on both cameras, jointly. The confidence map registers the maximum focus similarity score at each viewing ray of the fused DSI.

3.5. Temporal Fusion

The functions presented in Section 3.4 can be used to fuse any pair of aligned DSIs. Moreover, the functions can be extended to handle more than two inputs: they allow us to fuse an arbitrary number of registered DSIs, with a complexity that is linear in the number of DSIs. The DSIs may be populated by events from different cameras or, as we also investigate, from different time intervals. The main idea is to split an interval into multiple sub-intervals, build the DSI for each of them and fuse all DSIs into a single one (Figure 3). This strategy can be applied regardless of the number of cameras in the system, hence it represents an independent axis of variation. Moreover, the same technique enables camera- and time- fusion, which we collectively call **Algorithm 2**. The key lines of Algorithm 2 that change with respect to Algorithm 1 are lines 3 and 4.

Given N fusion functions ($N = 6$ in Figure 6), there are $2N^2$ possible fusion schemes considering the choice of temporal fusion function, across-camera fusion function, and the order of application (Algorithm 2). For brevity, we reduce the analysis to the comparison of $N = 2$ fusion functions: A and H , which yields 8 possible fusion schemes.

Let A_t denote the fusion operation along the time (t) axis using the arithmetic mean (A). Likewise, H_c is the fusion operation along the camera (c) axis using the harmonic mean (H). Then, $A_t \circ H_c$ first applies H_c (producing as many DSIs as sub-intervals) and then A_t . Out of the eight possibilities, there are only six distinct ones due to commutativity

$$A_c \circ A_t = A_t \circ A_c \quad \text{and} \quad H_c \circ H_t = H_t \circ H_c. \quad (14)$$

The four remaining fusion combinations are

$$A_t \circ H_c, \quad A_c \circ H_t, \quad H_t \circ A_c \quad \text{and} \quad H_c \circ A_t. \quad (15)$$

Clearly, $A_c \circ A_t$ is equivalent to the approach of summing all stereo events into a single DSI, and $H_t \circ H_c$ is very restrictive because only edges seen by all cameras in all subintervals will survive. Algorithm 1 is also a particular case of Algorithm 2 ($H_c \circ A_t$).

Algorithm 2. Stereo Event Fusion Across Cameras and Time.

- 1: Input: stereo events in the interval $[0, T]$, camera trajectory, camera calibration (intrinsic and extrinsic).
- 2: Define a single RV for all DSIs.
- 3: Divide the interval $[0, T]$ into N_s sub-intervals (of equal size or equal number of events). Create $2N_s$ DSIs by back-projecting events from each subinterval and camera.
- 4: $S_2 \circ S_1$: Two fusion axes (cameras and time). If $S_2 \equiv A_t$ and $S_1 \equiv H_c$, compute first the S_1 fusion (H -mean of two corresponding DSIs, on the same sub-interval); then compute the S_2 fusion (A -mean of all sub-interval DSIs).
- 5: Extract depth and confidence maps from the fused DSI.

3.6. Is Event Simultaneity Needed in Stereo?

Data association is a fundamental problem in event-based vision.^[2] In stereo, event simultaneity is a cornerstone assumption to resolve data association (i.e., find corresponding points) and subsequently infer depth. A thought-provoking discovery made while developing our method is that across-camera fusion does not need to be done at corresponding intervals (step 4 in Algorithm 2). We tested our method with the shuffling block in Figure 3 enabled and it still produced good results (see Section 4.4.3). Hence, the proposed stereo method foregoes the event simultaneity assumption. The explanation is that given the camera poses, events are transformed into a representation (i.e., the DSI), where event simultaneity is not as critical as in the instantaneous stereo problem (Section 2). The camera poses serve as a proxy allowing us to reproject stereo events to a common DSI and fuse them, even if the DSIs are well separated in time. The DSI representation is sufficient to produce 3D reconstructions. Stereo is not solved by matching events, but by comparing possibly non-simultaneous DSIs (each DSI spans several thousands of events).

3.7. Complexity Analysis

Let us analyze the complexity of the proposed stereo methods in comparison with the monocular case. The main steps of the methods are: DSI creation (event back-projection), DSI fusion, maxima detection along viewing rays of the DSI, and thresholding (AGT). If N_e is the number of events, N_p is the number of pixels in the reference view, N_z is the number of depth planes in the DSI, and N_k is the number of pixels in the AGT kernel (e.g., 5×5), then the complexity of ref. [10] is

$$O\left(\underbrace{N_e N_z}_{\text{DSI creation}} + \underbrace{N_z N_p}_{\text{argmax}} + \underbrace{N_p N_k}_{\text{AGT}}\right) \quad (16)$$

In the case of Algorithm 1 with N_c cameras, assuming that each camera produces N_e events, there are N_c DSIs to build and fuse. Hence, the complexity is

$$O\left(\underbrace{N_c N_e N_z}_{\text{DSI creation}} + \underbrace{N_c N_z N_p}_{\text{DSI fusion}} + \underbrace{N_z N_p}_{\text{argmax}} + \underbrace{N_p N_k}_{\text{AGT}}\right) \quad (17)$$

In the case of Algorithm 2 with N_s subintervals, there are N_s DSIs per camera, but each one has N_e/N_s events, and so the complexity of DSI creation does not change. Only the fusion step becomes more expensive

$$O\left(\underbrace{N_c N_e N_z}_{\text{DSI creation}} + \underbrace{N_s N_c N_z N_p}_{\text{DSI fusion}} + \underbrace{N_z N_p}_{\text{argmax}} + \underbrace{N_p N_k}_{\text{AGT}}\right) \quad (18)$$

4. Experiments

To assess the performance of our method, we test on a wide variety of real-world and synthetic sequences, which are introduced

in Section 4.1. Section 4.2 compares functions for fusion across cameras. Section 4.3 compares our method with three state-of-the-art methods on MVSEC and UZH data. Section 4.4 evaluates temporal fusion and sub-interval shuffling. Then, we evaluate on higher resolution data: driving dataset DSEC (Section 4.5), and 1Mpixel VIO dataset TUMVIE (Section 4.6). We also present trinocular examples (Section 4.7), and analyze runtime (Section 4.8). Finally, Section 4.9 summarizes the findings and Section 4.10 discusses the limitations of the method. The appendices analyze the sensitivity with respect to the camera's spatial resolution and the sparsity of the input events, respectively.

4.1. Datasets and Evaluation Metrics

4.1.1. Datasets

We evaluate our stereo methods on sequences from five publicly available datasets^[37,55–58] and a simulator. Sequences from refs. [37,57] were acquired with a handheld stereo or trinocular event camera in indoor environments. Sequences in the MVSEC dataset^[56] were acquired with a stereo event camera mounted on a drone while flying indoors. The sequences in the DSEC dataset^[55] were recorded with event cameras on a car that drove through Zurich's surroundings. The TUM-VIE dataset was recorded with the sensor rig mounted on a helmet, and its sequences contain indoor and outdoor scenes. The simulator^[59,60] provides synthetic sequences using an ideal event camera model and scenes built using CAD models.

Ground Truth: Some datasets contain ground truth poses from a motion-capture system, which we use as input to all tested methods. If camera poses are not available (e.g., TUM-VIE), we compute them using data from the sensor rig (e.g., a visual-inertial odometry algorithm). Some datasets, such as MVSEC and DSEC, contain ground truth depth for quantitative assessment of the 3D reconstruction methods. Depth is given by a LiDAR operating at 10–20 Hz. The event camera pixels corresponding to points outside the LiDAR's field of view (FOV) or points close to the sensor rig may not have a LiDAR depth value.

Rigs and Calibration: The main geometric parameters of the event cameras used in the aforementioned datasets are summarized in Table 1. The stereo rigs in refs. [37,56] consist of two dynamic and active pixel vision sensors (DAVIS).^[52] The DAVIS comprises a frame-based and an event-based sensor on the same pixel array, thus calibration (intrinsic and extrinsic) is achieved using the intensity frames, and then it is applied to the events. The datasets whose cameras output only events (EVIMO2, DSEC, and TUM-VIE), are calibrated by converting events to frames and calibrating the latter (e.g., using ref. [61]). All methods work on undistorted coordinates.

4.1.2. Metrics

The performance of the proposed method is quantitatively characterized using several standard metrics on the datasets with ground truth depth (i.e., MVSEC and DSEC). We provide mean and median errors between the estimated depth and the ground truth one (median errors are more robust to outliers than mean errors). We also report the number of reconstructed points, the number of outliers (bad-pix^[62]), the scale-invariant depth error

(SILog Err), the sum of the absolute value of relative differences in depth (AErrR), and δ -accuracy values on the percentage of points whose depth ratio with respect to ground truth is within some threshold (see ref. [63]). We also provide precision, recall, and F1-score curves.^[64] Precision is the percentage of estimations that are within a certain error from the ground truth. Recall (e.g., completeness or reconstruction density) is the percentage of ground truth points that are within a certain error from the estimations. The F1 score is the harmonic mean of precision and recall, which is dominated by the smallest of them. Since the depth maps obtained are semi-dense (while the ground truth is often more dense), recall often dominates.

The method in Section 3 is presented for the events in a time window. To apply the method to a whole sequence, we split the latter into non-overlapping time windows and apply the method to each of them. When thresholding to obtain semi-dense depth maps (AGT step), we normalize by a robust maximum DSI value obtained over the sequence, which makes the comparisons more stable.

4.2. Comparison of Across-Camera Fusion Functions

We first evaluate Algorithm 1 using the fusion functions in Figure 6. Figure 7 shows qualitatively the corresponding

Table 1. Parameters of stereo or trinocular event-camera rigs used in the experiments.

Dataset	Cameras	Resolution [pix]	Baseline [cm]	FOV [°]
ECCV18 ^[37]	DAVIS240C	240 × 180	14.7	62.9
MVSEC ^[56]	DAVIS346	346 × 260	10.0	74.8
EVIMO2 ^[57]	Samsung Gen3	640 × 480	trinocular	75
	2 × Prophesee Gen3	640 × 480	trinocular	70
DSEC ^[55]	Prophesee Gen3	640 × 480	60	60.1
TUM-VIE ^[58]	Prophesee Gen4	1280 × 720	11.84	90
ESIM ^[59]	Simulator	up to 1280 × 960	20	77.3

depth- and confidence maps for a sample sequence. The columns follow the order in Equation (12). The arithmetic mean (A) corresponds to counting the back-projected event rays from both cameras (left/right) on a single DSI. It has not been proposed before in the literature and is a particular case of our fusion methods. It provides moderate results by conveying that a 3D point is detected if enough rays are counted at a voxel, regardless of which camera the event ray originated from. However, it does not filter out spurious ray intersections (which do not correspond to actual 3D points) until there is sufficient evidence. Spurious ray intersections are more common in the stereo case than in the monocular one because rays are originated from two moving sources in stereo instead of just one. The columns in Figure 7 to the right of A (i.e., RMS and max) produce worse results than A. They convey that 3D points are detected if enough rays from at least one camera are counted at a voxel. This strategy might be good to mitigate occlusions (edges seen in only one camera), but it does not produce optimal results if the edge is visible from both event cameras. Finally, the columns to the left of A (i.e., G, H, min) produce better results than A. This is due to the fact that they implement a more conservative strategy: 3D points are detected only if enough rays from both cameras are counted at a voxel. The results are more noticeable in the confidence maps; these are more clear (sharper^[16]) in the first columns than in the last ones.

Table 2 quantitatively compares the six fusion functions on the three indoor flying sequences from ref. [56]. The experiment consists of running stereo Algorithm 1 on 200 s of data (110 million events), at 20 Hz. The estimated depth produces ≈ 28.4 million points on ≈ 4000 ground truth snapshots for each fusion function. The differences between fusion functions are most noticeable when less data is available, and so we use events packets of 0.1 s for this experiment. Table 2 reports mean and median depth errors, bad-pixel percentage, and number of reconstructed points. Errors and bad-pix follow a clear trend, decreasing toward the top rows. The number of reconstructed points also decreases, but non-monotonically in sequences 1 and 3. Median errors are

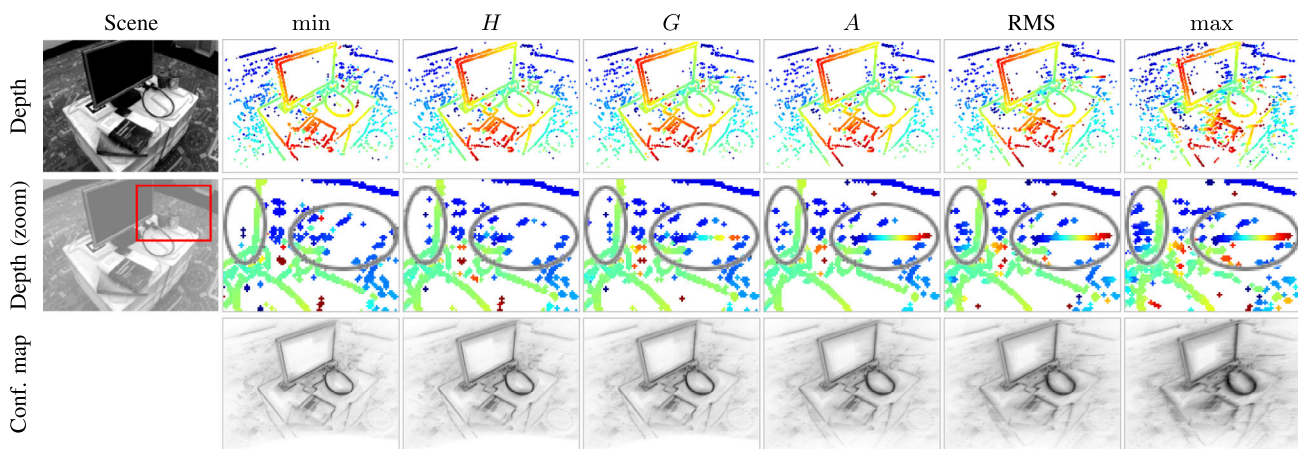


Figure 7. Fusion Functions. Semi-dense depth maps (top rows) and confidence maps (bottom row) produced by Algorithm 1 using the fusion functions in Figure 6 on data from Ref. [37]. The columns (fusion functions) follow the order in Equation (12). The differences are subtle: the depth maps on the left columns have fewer outliers than those on the right columns (zoomed-in insets). The differences are more noticeable in the confidence maps, whose sharpness increase from right to left. Depth is color coded, from red (close) to blue (far), in the range 0.55–6.25 m. Confidence maps are colored as in Figure 5.

Table 2. Depth errors for stereo DSI fusion across cameras (Section 4.2). Experiments on 200 s (110 million events) of the three indoor flying sequences from MVSEC.^[56] The maximum scene depth is 8.4 m.

Sequence ^[56]	Mean Abs Error [cm] ↓			Median Abs Error [cm] ↓			bad-pix [%] ↓			#Points [million] ↑		
	flying1	flying2	flying3	flying1	flying2	flying3	flying1	flying2	flying3	flying1	flying2	flying3
$\min_c \circ A_t$	58.13	68.79	49.96	24.07	39.20	20.35	17.40	38.82	12.30	6.13	12.72	5.83
Algorithm 1 ($H_c \circ A_t$)	60.16	68.83	51.94	25.45	39.63	21.15	18.58	38.91	13.70	6.61	13.54	6.27
$G_c \circ A_t$	63.57	70.48	55.22	28.17	41.51	22.97	20.30	39.68	15.30	6.90	14.18	6.35
$A_c \circ A_t$	78.78	79.95	60.35	38.15	47.63	25.03	27.20	44.64	17.56	5.97	14.53	4.50
$\text{RMS}_c \circ A_t$	99.15	88.46	88.21	61.73	55.71	47.45	36.84	48.89	29.86	7.65	17.48	5.41
$\max_c \circ A_t$	109.30	93.81	104.52	75.44	61.71	66.56	41.31	51.64	36.27	9.25	19.99	6.92

considerably smaller than mean errors, signaling the presence of outliers (points with large depth errors). We observe an accuracy-completion trade-off: the top rows are more accurate than the bottom rows, but the bottom rows provide more reconstructed points. Comparing the top two rows (highest accuracy), the differences in accuracy are small (mean: 2.41%, median: 3.43%) while the difference in the number of points is larger: 6.83%. Hence these results indicate, together with theoretical aspects (Section 3.4), that the H -mean is advantageous to fuse across cameras.

4.3. Comparison with Stereo SOTA

We assess the performance of our methods in comparison to several event-based stereo methods, in Figure 8 and Table 3.

4.3.1. Baseline Methods

The generalized time-based stereovision method (GTS)^[42] follows a classical two-step approach: stereo matching plus triangulation. Matching is based on a per-event time-based consistency score. the semi-global matching (SGM) method^[65] is adapted to event data by feeding time images^[46] and masking the produced depth map at recent event locations.^[37] We also compare against the mapping module of event-based stereo visual odometry (ESVO),^[15] which fuses multiple depth estimates using Student- t filters, with each estimate and its uncertainty obtained by maximizing spatiotemporal consistency between patches of stereo time images. GTS and SGM are also endowed with depth propagation-and-fusion filters, as

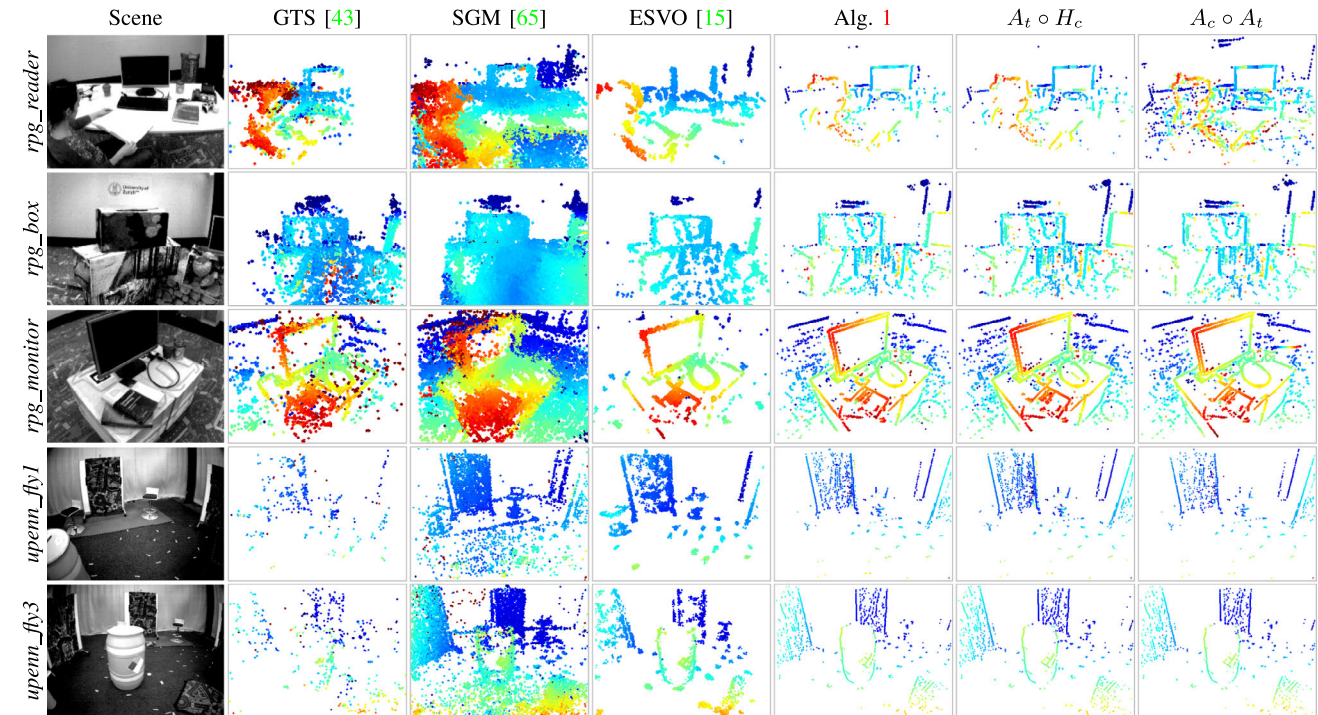


Figure 8. Event-based Stereo 3D Reconstruction. Comparison of depth estimation results on several sequences using various stereo methods. For visualization purposes, the first column depicts intensity frames from the DAVIS camera (not used by any method). Columns 2–7 show semi-dense inverse depth maps produced by generalized time-based stereovision method (GTS),^[42] semi-global matching (SGM),^[65] event-based stereo visual odometry (ESVO),^[15] our Algorithm 1 and 2 ($A_t \circ H_c$ and $A_c \circ A_t$, with $N_s = 2$), respectively. Depth maps are pseudo-colored, from red (close) to blue (far), in the range 0.55–6.25 m for the *rpg* sequences^[37] and in the range 1.0–6.5 m for the *upenn* MVSEC sequences.^[56]

Table 3. Quantitative evaluation and comparison of our proposed method with the state of the art (SOTA). All metrics are averaged over the three indoor MVSEC sequences (flying 1, 2, 3), where the maximum ground truth depth is 8.4 m. The methods are evaluated on 200 s of data (110 million events and 4000 ground truth depth maps). Each estimated depth map is computed using 1 s of event data (≈ 0.55 million events). MF: morphological filter (see text). Per-sequence results are in Table D1–D3 in the Appendix.

	Algorithm	Mean Err [cm] ↓	Median Err [cm] ↓	bad-pix [%] ↓	SILog Err $\times 100$ ↓	AErrR [%] ↓	log RMSE $\times 100$ ↓	$\delta < 1.25$ [%] ↑	$\delta < 1.25^2$ [%] ↑	$\delta < 1.25^3$ [%] ↑	#Points [million] ↑
SOTA	EMVS ^[10] (monocular)	33.78	14.35	3.84	4.20	12.74	20.72	84.75	94.87	97.99	1.27
	ESVO ^[15]	25.00	10.59	3.35	3.48	10.19	18.83	90.44	95.76	97.98	2.04
	ESVO indep. 1s	22.70	9.83	2.83	3.03	9.59	17.53	91.82	96.50	98.38	1.56
	SGM indep. 1s	35.42	12.35	6.39	8.45	16.17	29.49	85.34	93.05	96.03	14.46
	GTS indep. 1s	389.00	45.43	38.45	74.47	102.92	89.08	49.56	62.19	69.36	0.06
Ours	$H_c \circ A_t$ (Alg 1)	20.07	9.53	1.35	1.72	7.80	13.24	95.04	98.08	99.21	0.81
	$H_c \circ A_t$ (Alg 1) + MF	20.64	9.72	1.43	1.80	7.94	13.54	94.74	97.95	99.17	3.00
	$H_c \circ H_t$	23.45	10.98	1.89	2.18	8.86	14.93	93.00	97.49	99.03	1.47
	$H_t \circ A_c$	22.61	10.68	1.67	2.03	8.61	14.44	93.49	97.76	99.12	1.25
	$A_c \circ H_t$	23.35	10.94	1.84	2.15	8.83	14.85	93.07	97.53	99.04	1.42
	$A_c \circ A_t$	20.38	9.60	1.51	1.80	7.93	13.55	94.67	98.01	99.20	0.99
	$A_t \circ H_c$	20.92	9.76	1.66	1.86	8.05	13.81	94.39	97.80	99.14	1.14
	$A_t \circ H_c$ + shuffling	22.60	10.71	1.68	2.11	8.58	14.28	93.49	97.76	99.13	1.24

implemented in ref. [15]. Finally, we also compare stereo against the monocular method EMVS,^[10] which has not been carried out before. All baseline methods produce depth maps at the LiDAR rate (20 Hz) and use ground truth poses to propagate depth estimates in time, if needed. ESVO on MVSEC data works by fusing 20 depth maps generated at 20 Hz, i.e., 1 s of data. EMVS works on event packets of 1 s, shifted by 50 ms (20 Hz). For a sensible comparison with EMVS, baseline stereo methods are also run on event packets of 1 s, shifted by 50 ms; this is highlighted as “indep 1s” in Table 3.

4.3.2. Results

Figure 8 compares qualitatively the inverse depth maps produced by the aforementioned stereo methods. To illustrate the appearance of the scenes, the first column shows grayscale frames from the DAVIS.^[66] The remaining columns show the output of GTS, SGM, ESVO, and our methods. Because event cameras naturally respond to the apparent motion of edges, which occupy only a small portion of the image plane, most stereo methods produce semi-dense depth maps representing 3D scene edges. GTS produces modest results, albeit with many outliers. SGM has the most dense results because its regularizer fills in depth estimates in regions where the data fidelity term (time-image consistency) is not dominant. ESVO gives remarkable results in terms of accuracy and completeness, thus showing the effectiveness of its probabilistic inverse depth filters. Finally, our methods produce the best results: visually similar to ESVO but with finer details, thus being able to resolve more and finer edges in the scene.

Table 3 summarizes the quantitative performance with the metrics defined in Section 4.1 and on the same MVSEC sequences as^[15,37,67,68] (with ground truth depth). The best result per column is highlighted in bold, and the second best is underlined.

Detailed, per-sequence tables are provided in the Supporting Information. Contrary to previous works, we test on the entire sequences, consisting of 200 s (110 M events). The top part of Table 3 reports the results of the baseline methods, where ESVO is a top performer. EMVS is slightly worse than ESVO, consistently in most metrics, which shows that sensible stereo depth estimation (ESVO’s Student-*t* filters) is beneficial to gain accuracy and reduce the number of outliers with respect to the monocular case. The bottom part of Table 3 reveals the results of several variations of Algorithm 2 ((14)–(15)) with $N_s = 2$ sub-intervals. All accuracy and outlier metrics of Algorithm 1 are significantly better than those of ESVO and EMVS, demonstrating the effectiveness of our fusion approaches: outperforming the SOTA and quantifying the gap between monocular and stereo methods.

Regarding completion, Algorithm 1 recovers fewer points than ESVO. This has a natural explanation: ESVO generates several depth estimates per second that are propagated and fused. Even using ground truth poses, the estimates are noisy, and so they transfer to the fused image plane producing thick edges (Figure 8). By contrast, our stereo method generates depth estimates at the AGT thresholding step, which is called just once, and therefore generates thinner edges than ESVO (finer details and well distributed at scene edges, as shown in Figure 8). To justify that lower completion values are not an issue, we applied a 4-neighbor morphological filter (MF) to dilate the mask of the depth map produced by AGT, and filled in the depth values using the center pixels. This almost quadrupled the number of reconstructed points (from 0.81 to 3 M) while having a minimal effect on accuracy (row “MF” in Table 3), and therefore reduces the importance of comparing completion values given by very different algorithms. This also aligns with ideas in semi-dense and sparse SLAM, where fewer but more accurate points are

preferred for several reasons: to have better distributed points on the image plane and to reduce the computational load (i.e., increase efficiency and speed).^[69]

Variants $A_t \circ H_c$ and $A_c \circ A_t$ are also top performing. Despite the abundance of events accumulated in the DSI, Table 3 quantifies a gap between Algorithm 1 and $A_c \circ A_t$ (the gap between the arithmetic mean and more concave fusion functions like H would be most noticeable if less data was used, as in Table 2). The last row of Table 3 is discussed in Section 4.4.3.

Driving Sequences: The tests on the outdoor MVSEC sequences did not give good results because the camera baseline is very small compared to the scene depth, so geometrically the data is poor for 3D reconstruction. As noticed in,^[67] most points in those sequences are beyond the depth resolved by a disparity of 1 pix. Instead, we show the results of our method on driving sequences from the DSEC dataset (Section 4.5), which has a larger baseline of 60 cm.

4.4. Temporal Fusion Experiments

4.4.1. Stereo

Figure 8 and Table 4 show the effect of temporal fusion. Using as few as $N_s = 2$ sub-intervals in Algorithm 2 already delivers gains compared to the SOTA. While the optimal choice of the number of subintervals N_s depends on many factors, such as the number of events processed (duration of the intervals), the camera motion, etc., we found that using $N_s \in [2, 8]$ gives satisfactory results. This is reported in Table 4, which is an ablation study of Algorithm 2 $A_t \circ H_c$ with respect to N_s . There is a clear trend: accuracy and completion values increase with N_s . However, memory and complexity also increase (linearly with N_s , see ref. [18]). Comparing the values in Tables 3 and 4, we notice that the improvement due to temporal fusion is not as pronounced as that due to stereo parallax (for example, the median error improves 32% from EMVS to stereo $A_t \circ H_c$ ($N_s = 2$), and 5.02% from $A_t \circ H_c$ $N_s = 2$ to $N_s = 8$).

4.4.2. Monocular

As a by-product, temporal fusion using the H -mean is a simple modification that can be applied to lightly improve the monocular method,^[10] especially when little data is available. While this is not the focus of the article, we provide an example: Figure 9 shows the effect of monocular temporal fusion on the same slider rock-plane sequence as Figure 4. It compares the evolution of DSIs with and without temporal fusion. As expected, with H -mean fusion the DSI converges faster to the 3D structure and

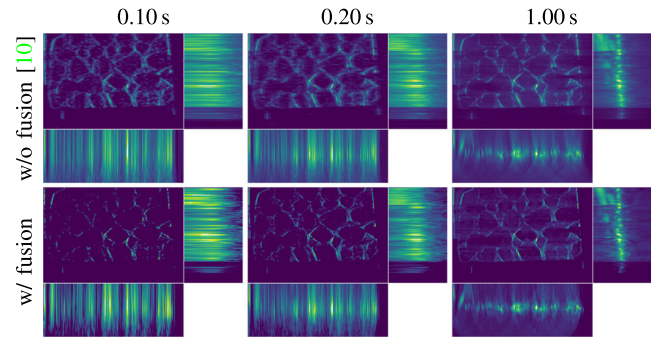


Figure 9. Effect of temporal fusion ($N_s = 4$ sub-intervals) on monocular 3D reconstruction. Rocks scene in Figure 4 and 5.

deemphasizes the locations of spurious ray intersections visible in the unfused DSI.

The effect of temporal fusion is dramatic if we compare the depth maps obtained from each subinterval with the depth map obtained after fusion. This is illustrated in Figure 10, where an interval of the *upenn_flying3* sequence^[56] is divided into $N_s = 4$ sub-intervals. The first four columns depict depth maps produced by EMVS (row 1) and Algorithm 1 (row 2) applied to individual sub-intervals (each with ≈ 56 k events per camera). These depth maps are noisy; however, when the DSIs are temporally fused and depth is extracted (Algorithm 2), the final depth maps are remarkably cleaner (last column). Also, the fused stereo depth map has fewer outliers than the monocular one.

4.4.3. Stereo Fusion from Shuffled Sub-Intervals

Arranging events in time subintervals allows us to question the event simultaneity assumption for depth estimation. Figure 11 shows the results of applying the system in Figure 3 with the shuffling block enabled to the MVSEC and UZH sequences. The shuffling block modifies line 4 in Algorithm 2 to use different subintervals for the S_1 fusion. The results are surprising: despite using non-corresponding subintervals (i.e., non-simultaneous events) for DSI fusion across cameras, the obtained depth- and confidence maps are very similar to those obtained with corresponding subintervals, with some added noise. Hence, event simultaneity is not needed for stereo depth estimation with our system (Figure 3). Only the similarity between the DSIs to be fused (intermediate ray density representations built by combining events and camera poses) is required.

Table 4. Sensitivity of $A_t \circ H_c$ (Algorithm 2) with respect to the number of subintervals N_s . Continuation of Table 3.

N_s	Mean Err [cm] ↓	Median Err [cm] ↓	bad-pix [%] ↓	SILog Err ×100 ↓	AErrR [%] ↓	log RMSE ×100 ↓	$\delta < 1.25$ [%] ↑	$\delta < 1.25^2$ [%] ↑	$\delta < 1.25^3$ [%] ↑	#Points [million] ↑
2	20.92	9.76	1.66	1.86	8.05	13.81	94.39	97.80	99.14	1.14
4	20.49	9.58	1.63	1.94	7.88	13.65	94.64	97.82	99.15	1.19
8	19.80	9.27	1.56	1.84	7.63	13.31	94.99	97.90	99.19	1.20

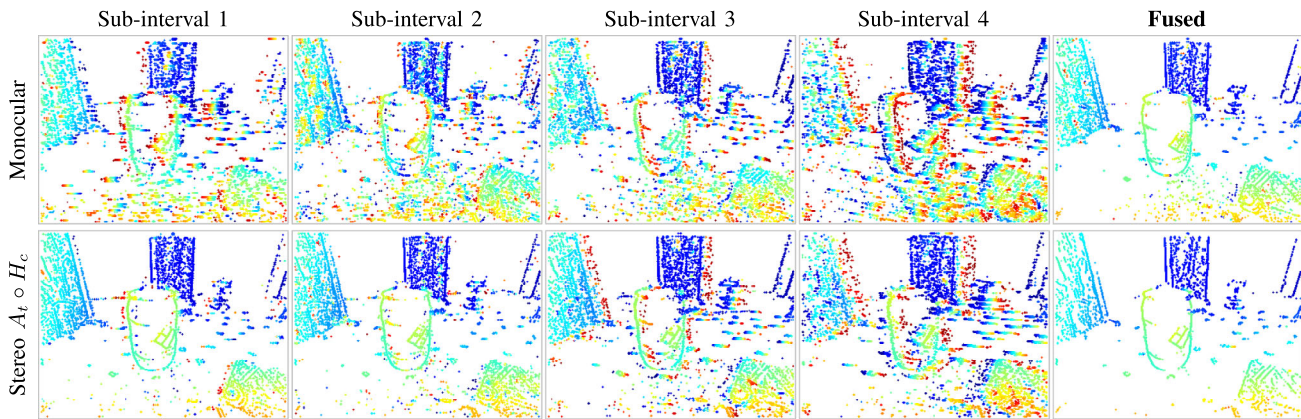


Figure 10. Effect of temporal fusion on the obtained depth maps. Depth is color coded as in Figure 8.

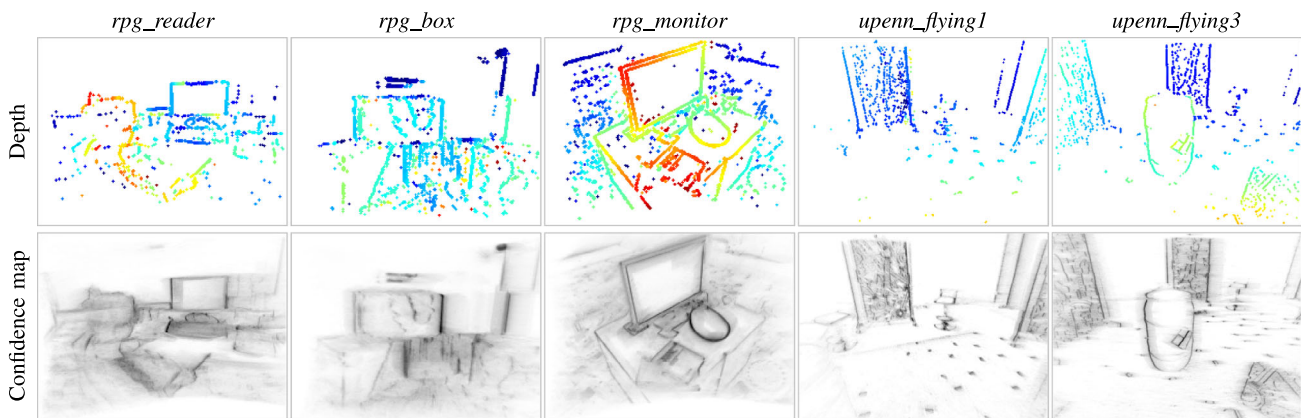


Figure 11. Event Simultaneity? Fusion across cameras and time with shuffled time intervals ($N_s = 2$). Same scenes as in Figure 8.

Quantitatively, the last row of Table 3 informs about the depth errors and outliers incurred by shuffling. The differences with respect to the unshuffled case are small (slightly higher errors and outliers), which is remarkable given the quite diverse input events.

4.5. Experiments on DSEC Driving Dataset

We also give results on sequences from the driving dataset DSEC.^[55] Figure 12 shows qualitative results. Driving scenarios are challenging for event-based sensors because forward motions

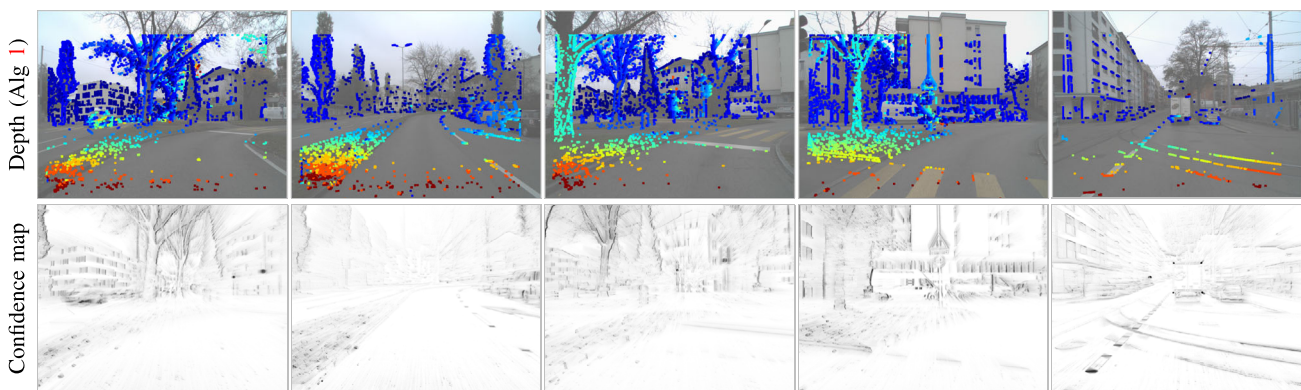


Figure 12. Results on DSEC data.^[55] Top row: Semi-dense depth maps (overlaid on color frames) estimated by Algorithm 1 on event packets of 200 ms. Depth is color-coded from red (close) to blue (far), in the range 4–200 m. Bottom row: confidence maps.

Table 5. Quantitative evaluation on the driving dataset DSEC (zurich04a sequence) with maximum ground truth depth 50 m. The methods are evaluated on 35s of stereo data, consisting of 635 million events and containing 350 ground truth depth maps. Each depth map is computed using 0.2 s of event data (≈ 3.5 million events). ESVO is executed fusing two depth maps generated at 10 Hz (LiDAR rate), i.e., 0.2s of event data. MF: morphological filter.

Algorithm	Mean Err [m] ↓	Median Err [m] ↓	bad-pix [%] ↓	SILog Err $\times 100$ ↓	AErrR [%] ↓	log RMSE $\times 100$ ↓	$\delta < 1.25$ [%] ↑	$\delta < 1.25^2$ [%] ↑	$\delta < 1.25^3$ [%] ↑	#Points [million] ↑
EMVS [10] (monocular)	5.64	2.52	13.68	13.23	25.52	36.49	72.56	87.12	93.56	1.31
ESVO[15]	3.88	1.56	12.08	9.23	18.89	30.80	84.53	92.57	95.63	3.40
$H_c \circ A_t$ (Alg 1)	3.27	0.90	10.75	8.19	17.48	28.73	83.30	91.56	95.62	1.25
$H_c \circ A_t$ (Alg 1) + MF	3.51	0.96	11.81	8.89	18.84	29.99	81.72	90.68	95.07	3.83

typically produce considerably fewer events in the center of the image (where apparent motion is small) than in the periphery. Forward motion is also not particularly amenable for 3D reconstruction methods compared to sideways motions. Nevertheless, our stereo method shows notable results in this dataset. As expected, more 3D points are recovered in the periphery than in the center of the image, except when the car is turning.

Quantitative results are summarized in **Table 5**. Because the amount of events recorded by the VGA-resolution Prophesee Gen3 cameras is exorbitant, the experiments are carried out on a subset of the dataset. We test Algorithm 1 and the baselines on 635 million events. Similarly to the results on MVSEC, Table 5 indicates that Algorithm 1 has higher accuracy than ESVO and EMVS (at least 18.65% better in mean absolute error, and 42.3% in median absolute error). ESVO has marginally higher inlier values (1.45% for $\delta < 1.25$). The monocular method produces the largest errors (the mean and median absolute errors are $1.7\text{--}2.8\times$ larger than those of stereo Algorithm 1), and also a larger number of bad pixels and outliers. We also apply a morphological dilation filter (MF), with similar conclusions as in Table 3: the number of reconstructed points triples while the accuracy remains better than ESVO's. Finally, we notice that the ground truth depth of DSEC is sparser than that of MVSEC. This is due to the increased pixel resolution (VGA size) and the fact that LiDAR points do not fill as many camera pixels (percentage-wise) as in lower resolution cameras. Please see the accompanying

video for a visual comparison between our method, ESVO, EMVS, and GT.

4.6. Experiments on TUM-VIE Dataset

We present depth estimation results using Algorithm 1 on the TUM-VIE dataset,^[58] the first public visual-inertial dataset with 1 megapixel stereo event cameras (Prophesee Gen4).^[5] To the best of our knowledge, our work is the first to provide results on this new event-based dataset (the original article presented the data but did not evaluate it on any event-based algorithm). Our experiments have served as a means to debug and fix the dataset. Since the dataset has no ground truth depth, we only present qualitative results.

Figure 13 presents results in indoor and outdoor sequences. Indoor sequences recorded in a room have ground truth poses given by a motion capture (mocap) system (columns 1 and 2). The indoor scene depth is small relative to the camera baseline (11.84 cm). Nevertheless, our method does a notable job in recovering 3D structure, with a reduced number of outliers and cleaning the depth maps. Having set the reference view of the fused DSI on one camera trajectory, the large baseline makes the event rays back-projected from the other camera appear nearly parallel, which does not favor fusion.

For sequences recorded outside the mocap room, we computed ground truth poses using Basalt's VIO^[70] on the stereo

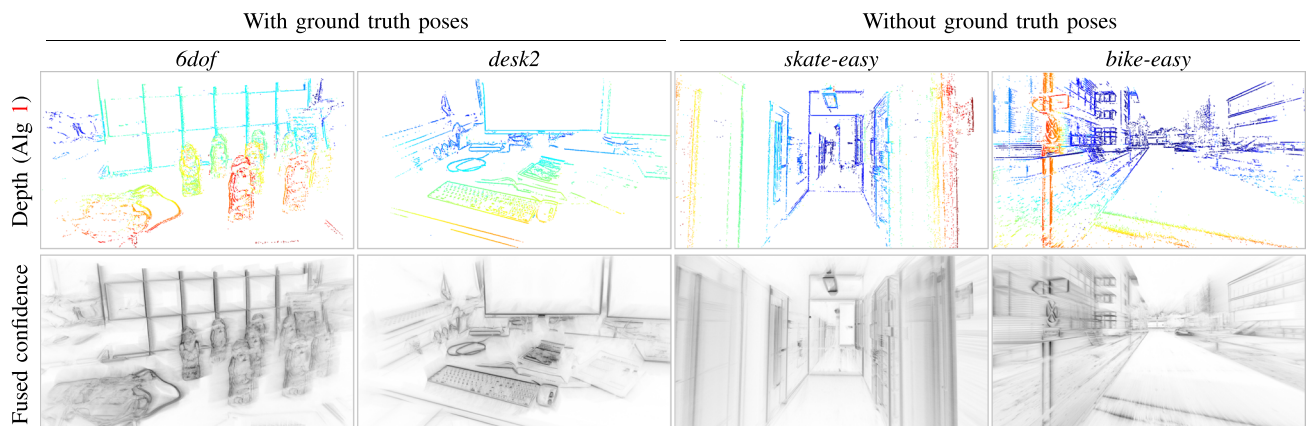


Figure 13. Depth estimation using 1Mpix event cameras. Depth estimated using Algorithm 1 on 0.5 s intervals of data from TUM-VIE.^[58] Depth maps are color-coded from red (near) to blue (far). The range is 0.45–4.00 m for indoor sequences in the motion-capture room (*6dof*, *desk2*), 1–20 m for the corridor sequence *skate-easy* and 3–200 m for the outdoor sequence *bike-easy*.

frames and IMU. The last two columns of Figure 13 depict the performance of our stereo method on such sequences. The space-sweeping method to build the DSIs works best with sideways translations that produce the parallax necessary for the convergence of the back-projected rays. However, the majority of the non-mocap sequences comprise forward camera motions, which contribute little parallax and also produce fewer events; hence they are not amenable for 3D reconstruction. The forward motion and the lower quality of camera pose lead to an overall poorer reconstruction quality compared to the sequences in the mocap room. In contrast, the stereo setup is able to exploit the camera baseline as additional parallax for 3D reconstruction. In the accompanying video, we provide a visual comparison between our method and ESVO on the TUM-VIE dataset, showing that our method performs better at higher resolutions than ESVO.

We noticed that the calibrationA sequences (skate-easy, desk2) produced better results than the calibrationB sequences, which leads us to believe that the calibration errors were significant in the latter. We also compared the results of our method with two different sources of ground truth poses (the mocap system and Basalt), and observed no significant differences in the depth- and confidence maps. Hence, we concluded that (i) the poses from Basalt may be considered as accurate as the mocap for short time intervals (e.g., 0.5 s, with ≈ 10 M events), and (ii) there is robustness to noise: our methods provide reasonable depth estimates using poses from a VIO (non-mocap) algorithm.

Event cameras offer advantages over frame-based cameras to handle HDR scenes, as demonstrated in Figure 14. In the first row, the grayscale frame is overexposed in the outdoor area, whereas the events capture the floor tiles and garden scene well. In the second row, the end of the corridor is underexposed in the frame, whereas the events capture the whole scene. Our stereo method correctly estimates depth in all regions due to the HDR capabilities of the events.

4.7. Fusing More than Two Event Cameras

We also test our method on sequences from the EVIMO2 dataset,^[71] recorded with a trinocular event-camera rig consisting of a

Samsung DVS Gen3^[72] and two Prophesee CD Gen3 event cameras,^[73] all with 640×480 pix. The field of views (FOVs) of the cameras have a narrow overlap due to the way they are arranged in the sensor rig (Prophesee event cameras are in portrait mode, whereas the Samsung DVS, in the middle, is in landscape mode). We set the central camera (Samsung DVS) as the reference one.

Figure 15 shows the depth- and confidence maps from each camera separately and for the fused DSI during two motions (two pairs of columns): a normal one and a fast motion, with retinal speeds between 1900 pix s^{-1} for objects in the far end and 3500 pix s^{-1} for objects close to the camera. The resulting semi-dense depth map obtained from the fused DSI inherits the above-mentioned narrow FOV overlap of the cameras. Overall, the fused depth map suppresses noise that would otherwise appear as very prominent outliers in the individual depth maps. This experiment shows that our method naturally fuses multiple cameras with linear complexity, i.e., without handling them in pairs, as prior works do.

4.8. Runtime

Complementing the complexity analysis in Section 3.7, Table 6 presents the average time taken in each step (DSI creation, DSI fusion, argmax, and AGT thresholding) over 100 sample runs, on a laptop with an Intel i7-10510U 8-core CPU. We consider typical numbers from a stereo DAVIS346 configuration.^[74] For comparison, we also report the numbers obtained with only one of the cameras (monocular setup^[10]). DSI creation takes the longest time as it is a complex transformation and depends on the number of input events. We did not find major runtime differences in our implementation of the six fusion functions tested (Equation (6)–(11)).

The numbers in Table 6 agree with complexity formulas (16)–(18). The DSI creation runtime of the stereo methods is roughly twice ($N_c = 2$) that of the monocular method (520 vs 234 ms). The temporal fusion step with $N_s = 2$ sub-intervals is twice as expensive as that with one interval (98 vs 51 ms). The arg max and AGT steps are the same for all methods, thus no major runtime differences are observed.

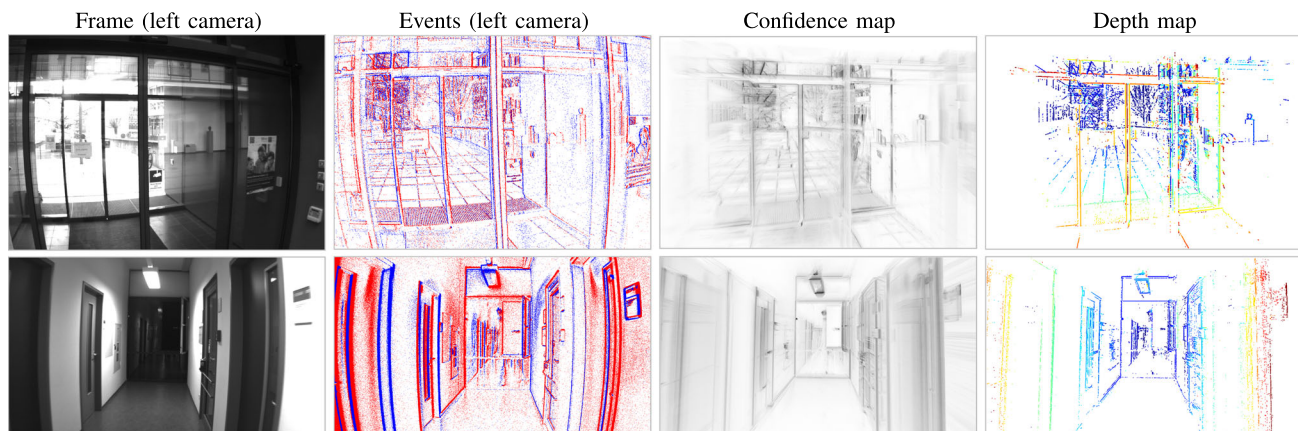


Figure 14. High dynamic range (HDR) scenes. Output of Stereo Algorithm 1 on HDR scenes from the TUM-VIE dataset. Unlike the frame-based camera, event cameras can perceive both under and over-exposed regions of the scene well, leading to good depth estimation throughout.

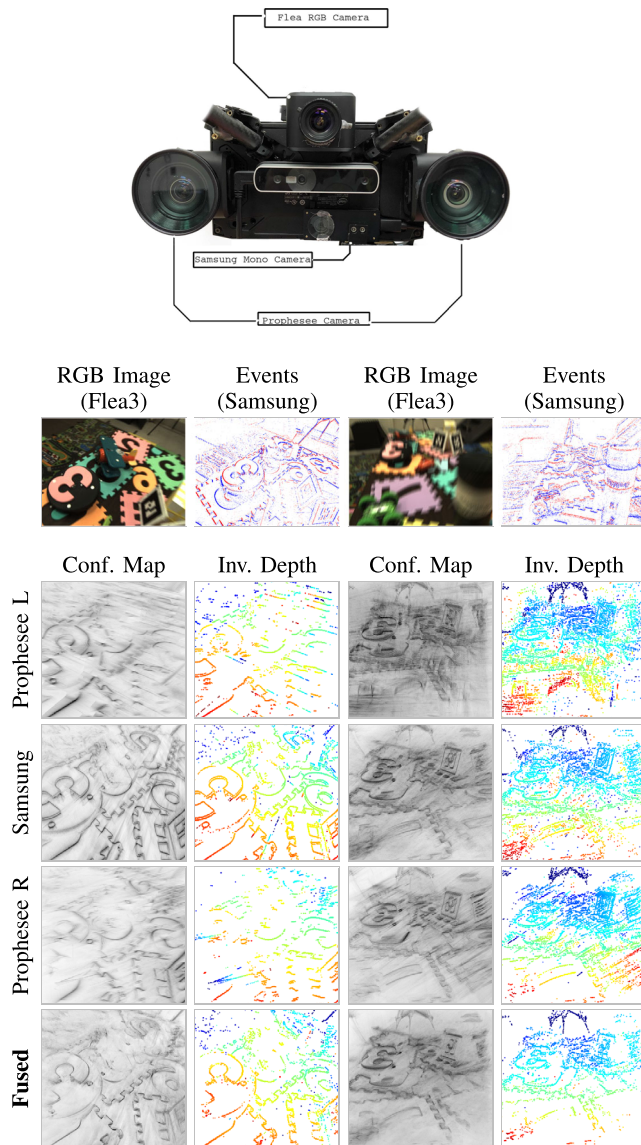


Figure 15. Trinocular event-camera fusion. Output of Stereo Algorithm 1 on the Structure from Motion sequence 03_00 from the EVIMO2 dataset.^[71] The first row shows an RGB image of the scene and the events (from the Samsung DVS^[72]) over a short time duration, displayed in red/blue according to polarity. The following rows show the output of our stereo method using the three event cameras in the dataset.

Table 6. Runtime comparison of the different steps of three algorithms. Parameters: $N_e \approx 235$ k events, $N_c = 2$, $N_p \approx 90$ k (DAVIS346), $N_z = 100$, $N_k = 25$ (5×5) pix, $N_s = 2$ for Algorithm 2 (here, $A_i \circ H_c$). Time is given in ms.

	EMVS ^[10]	Algorithm 1	Algorithm 2
DSI creation	233.96	520.01	
DSI fusion	–	51.09	98.01
arg max	36.34	35.62	
AGT	0.28	0.28	

4.9. Discussion

Let us summarize some of our findings. In the UZH dataset (Figure 8, top three rows), our method achieves the best results compared to the SOTA. Figure 8 also shows that our stereo method, in its different variations, recovers depth at more fine-detailed structures than the SOTA method ESVO.

In the indoor MVSEC dataset, Table 3 shows that our stereo method outperforms SOTA quantitatively across multiple standard metrics. We also show that the effect of time fusion, while being significant because it speeds up structure convergence and cleans up depth maps, has a smaller effect than that of fusion induced by parallax from an additional sensor (i.e., switching from monocular to stereo) (Tables 3 and 4). Remarkably, both strategies are unified in the same theory of fusion of refocused events that we propose.

The outdoor driving MVSEC sequences have a very small stereo baseline, which makes them poor for 3D reconstruction,^[75] hence we test on the recent DSEC dataset. Here, our method also outperforms the SOTA method ESVO (Table 5).

The TUM-VIE dataset allowed us to demonstrate experiments on high event camera resolution (1 Mpix) and robustness to errors in the camera poses. The EVIMO2 dataset allowed us to establish multi-camera (trinocular) depth estimation and during high-speed motion (which blurs regular frames).

Throughout the experiments (MVSEC, DSEC, TUM-VIE, time fusion, etc.), we have shown the gains with respect to the monocular method; the main advantages of stereo are: higher accuracy, outlier rejection, and faster convergence (due to the additional parallax).

Additionally, we have analyzed the sensitivity with respect to the camera's spatial resolution and contrast threshold (see Appendices): the higher the resolution or the lower the threshold, the more accurate our method becomes, at the expense of computational burden due to the larger number of input events. We also analyzed the computational performance, showing the agreement between complexity (theory) and runtime (practice).

Most interestingly, we have analyzed the effect of shuffling events: our method does not need event simultaneity. It can fuse DSIs even if they are built from events well separated in time. The best results are obtained by fusing identical intervals.

4.10. Limitations

Our method assumes, like multi-view 3D reconstruction methods for standard cameras, accurate calibration and camera poses. These assumptions allowed us to concentrate our efforts on investigating fusion functions for event-based 3D reconstruction under good multi-view alignment conditions. Calibration is today of good quality using the DAVIS frames or image reconstruction^[61] if frames are not available. In a full SLAM system, noise in the poses can propagate to the 3D reconstruction module. The TUM-VIE experiments, with different pose sources (mocap vs VIO algorithm), showed the robustness of our method to realistic poses (i.e., noisy as produced by a VIO algorithm). This is encouraging, as future research on event-based camera tracking could achieve such accuracy and be combined with our stereo method. Recent results on the monocular case^[76] suggest that

such accuracy and robustness are achievable by combining events and frames.

Another assumption of our method is that events are dominantly triggered by moving edges (brightness constancy). Hence, it may fail to estimate depth accurately from events that are not due to motion, such as those caused by flickering lights. Such events may be removed during pre-processing.^[77]

5. Conclusion

We presented simple and effective SOTA algorithms for event-based multi-camera 3D reconstruction in SLAM, combining across-camera and temporal fusion. Thanks to the availability of accurate camera poses, matching within and across event cameras happens implicitly in DSI space, which removes the need for event simultaneity (explicit data association). We investigated DSI space fusion methods, and showed how the same technique unifies temporal and camera fusion. We proposed a spectrum of fusion functions, including summing event rays in multi-camera settings, and objectively analyzed the results they produced. Our theoretical design was supported by a comprehensive set of experiments: testing on five established datasets and a simulator on a variety of scenarios, on millions of input events, while outperforming SOTA methods. Fusion functions like the H -mean are beneficial for fusion because they have strong concavity, and are bounded and smooth. The effect of the parallax given by an additional camera is stronger than the effect of temporal fusion. Our method works well regardless of the spatial resolution, which is interesting given the increasingly high spatial resolution of event cameras (see the comparisons with ESVO on DSEC and TUM-VIE datasets in the accompanying video).

Future research may look into combining the proposed method with an appropriately designed camera tracking algorithm to yield a full event-based stereo visual SLAM pipeline. The results on the driving datasets open the door to applying the proposed technique for creating HDR edge maps of a vehicle's surroundings and using it for later processing stages of Spatial AI and mobile intelligent systems,^[78] such as spatial awareness or extraction of semantic information.

Appendix A. Additional Theoretical Remarks

A1. More Fusion Functions

Additional means exist beyond those in Figure 6, such as the contraharmonic mean, logarithmic mean, quasi-arithmetic mean, arithmetic-geometric mean, Heronian mean, and weighted generalized means. However, they are not covered for the sake of brevity and to avoid clutter. In some cases, the order relation (Equation (12)) can be extended to justify their limited practical interest. Seeking more fusion functions, one could combine functions in Figure 6 with non-linear transformations of the input DSIs, in a homomorphic filtering fashion. For example, the A -mean of the log-DSIs is related to the G -mean of the DSIs, which has a stronger concavity than the A -mean of the DSIs. The same idea can be applied to other functions

to increase concavity: the G -mean of the log-DSIs, $G(\log(1+u), \log(1+v))$, has stronger concavity than the G -mean of the original DSIs. The logarithm plays down large DSI values, thus deemphasizing differences between corresponding DSIs before fusion. For simplicity, we restrict the study to the functions in Figure 6.

A2. Loose Connection with Prior Fusion Work

A method for fusing 3D representations called “temporally synchronized event disparity volumes” was proposed in ref. [67], where two binary data volumes I_L, I_R were fused using an intersection-over-union (IoU) cost. It resembles the H mean

$$\text{IoU} = \frac{\sum_{\mathbf{x} \in W} I_L(\mathbf{x}, d) \cap I_R(\mathbf{x}, d)}{\sum_{\mathbf{x} \in W} I_L(\mathbf{x}, d) \cup I_R(\mathbf{x}, d)} \quad \text{vs} \quad H = 2 \frac{uv}{u+v} \quad (\text{A1})$$

where the product in the numerator (“intersection”) acts as an “AND” condition and the sum in the denominator (“union”) acts as a normalization factor. However, note that the IoU in ref. [67] is computed by aggregating binary data I_L, I_R over spatial windows W (of 32×32 pixels), whereas H (Figure 6) is computed voxel-wise, without spatial aggregation (i.e., it has higher spatial resolution), on continuous ray densities.

Appendix B. Effect of Varying the Sensor's Spatial Resolution

So far, results on datasets from three different resolutions have been presented: DAVIS346 (MVSEC), Prophesee Gen3 (DSEC), and Prophesee Gen4 (TUM-VIE). However, the scenes are all different, some do not have ground truth depth, and calibration errors may influence the results. To analyze the response of our stereo Algorithm 1 to varying pixel resolutions under controlled conditions (same scene, same FOV, etc.), we generate events using a simulator (ESIM^[59]). We use the textured scene *flying_room*, with a camera baseline of 20 cm and the OpenGL renderer. The scene is visualized in Figure B1 (bottom right). Inspired by MVSEC, DSEC, and TUM-VIE datasets, three sensor resolutions are tested: 320×240 , 640×480 , and 1280×960 pix, respectively.

We run Algorithm 1 on stereo events generated within the same time duration (increasing the sensor resolution increases the number of events generated within the same time duration: roughly 0.25, 2.25, and 14.3 Mev for the three aforementioned resolutions, respectively). Figure B1 compares qualitatively the semi-dense depth maps and confidence maps recovered by our method on the three sensor resolution inputs. The observations from these images are complemented by the quantitative analysis in Figure B2. Figure B2a demonstrates quantitatively that depth errors decrease as the sensor resolution increases, for both monocular and stereo cases. Stereo fusion reduces the error almost by half with respect to the monocular case. Next, we also analyze the density of the reconstructions. Figure B2 also reports the precision, recall, and F1-score plots for the depth maps in Figure B1. A clear trend is observed: the precision increases with increasing camera resolution (lower errors, like in Figure B2a). For the highest resolution tested (1280×960 pix),

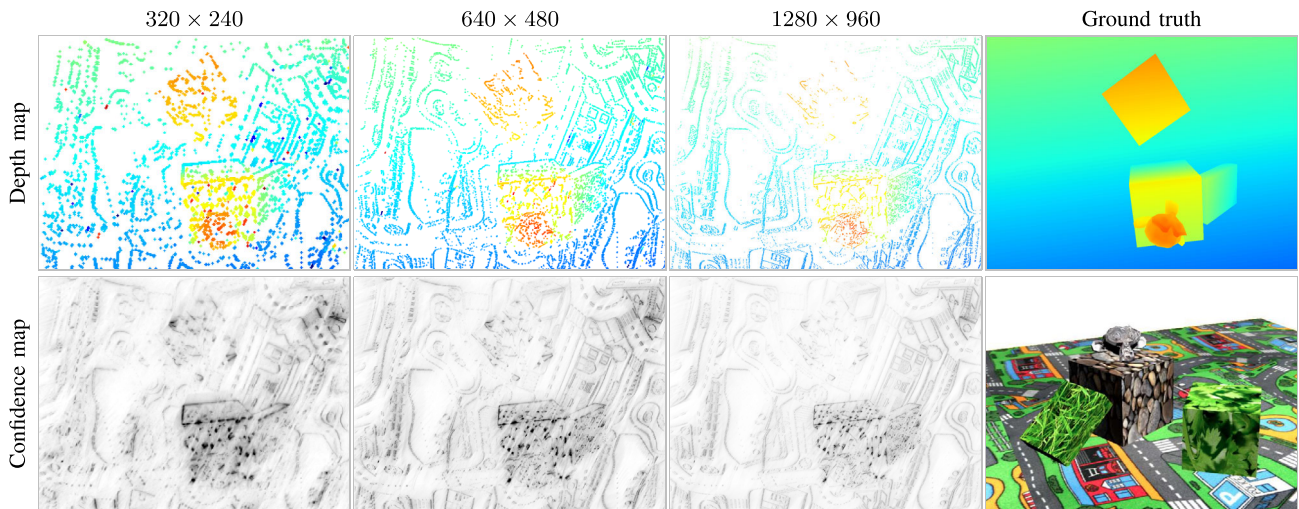


Figure B1. Effect of the sensor's spatial resolution. Output of Algorithm 1 on events simulated with different camera resolutions. The scene *flying_room* is shown on the top right. Depth maps are colored from red (near) to blue (far) in the range 1.3–4.0 m.

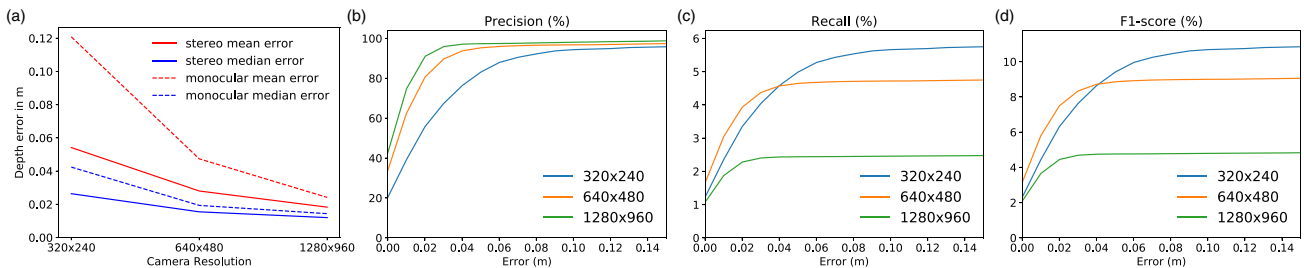


Figure B2. a) Depth errors across various camera resolutions for monocular^[10] and stereo Algorithm 1 methods. Maximum scene depth is 2.7 m. b–d) Precision, recall and F1-score for depth estimated using Algorithm 1 on scenes of different resolutions.

the precision reached $\approx 97\%$ within 4% of depth error tolerance. However, the recall curves state that a lower resolution may allow us to recover more 3D points in the scene (6% vs 2%), at the expense of bigger depth errors. Since the depth outputs are highly sparse due to the nature of event data, the F1-score is heavily skewed by the low values of recall.

Appendix C: Effect of Varying the Camera's Contrast Sensitivity

Due to the sparse nature of event data, which is largely controlled in the camera by the contrast sensitivity threshold, it is interesting to analyze the performance of our 3D reconstruction method for various sparsity levels. To this end, we ran Algorithm 1 on stereo events simulated using ESIM^[59] with five levels of event generation contrast threshold $\theta = \{0.05, 0.1, 0.2, 0.4, 0.8\}$. A small contrast threshold implies that a small change in brightness is sufficient to trigger events, and thus leads to the generation of many events from edges and textures in the scene (i.e., high sensor sensitivity). **Figure C1** illustrates the 3D reconstruction quality across varying contrast thresholds for the flying room sequence. In general, we observe stable reconstructions as the contrast threshold increases, except for the fact that fewer points

are recovered (e.g., on the floor) and more noisy points (i.e., outliers) appear. This is also observed quantitatively in Figure C2a, where the mean error increases for high contrast thresholds (the larger number of outliers distorts the mean error) while the median error remains fairly constant (a sign of “stability” if outliers are removed). Figure C2b–d depicts the precision, recall, and F1-score curves for depth predicted using data from different contrast thresholds. We observe that low contrast thresholds provide better precision and recall since they have fewer noisy outliers and recover more 3D points, respectively. The increase in precision as well as recall comes at the cost of increased computational overhead as more events need to be processed at lower contrast thresholds.

In summary, the synthetic experiments suggest that increasing the event count either by decreasing the contrast threshold or increasing the spatial resolution of the camera improves 3D reconstruction accuracy. This comes at the cost of increased computational effort needed to process more input data.

While event cameras are dominated by those that compute temporal contrast^[4] (DVS), an “event” could have a broader interpretation, such as any meaningful information that decreases demands on bandwidth, memory, and power for data transmission, storage, and processing. This work has analyzed the advantages that event cameras offer for stereo depth estimation.

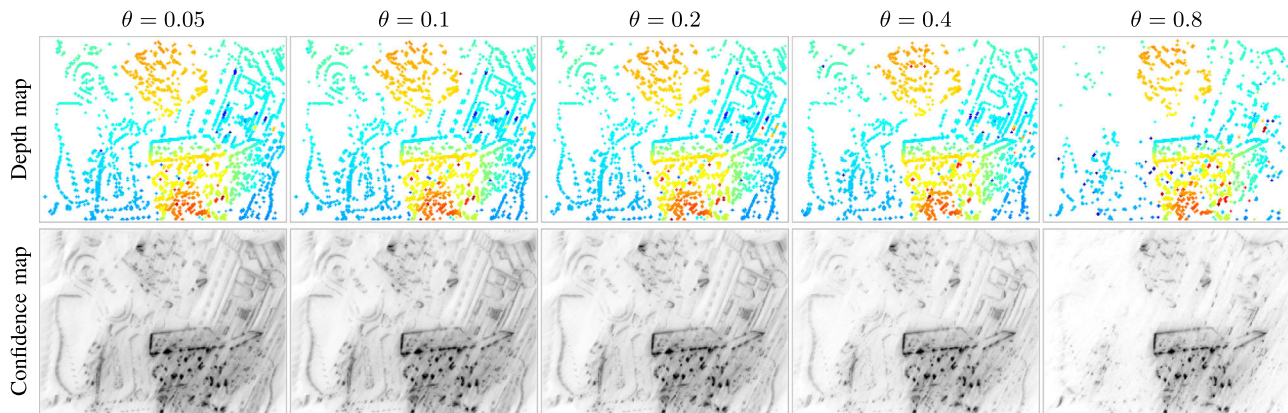


Figure C1. Effect of varying the camera's contrast threshold. Output of stereo Algorithm 1 on simulated events generated with varying contrast threshold θ . As θ increases, the sparsity of recovered points increases and the quality of depth errors decreases. Depth maps are colored from red (near) to blue (far) in the range 1.3–4.0 m.

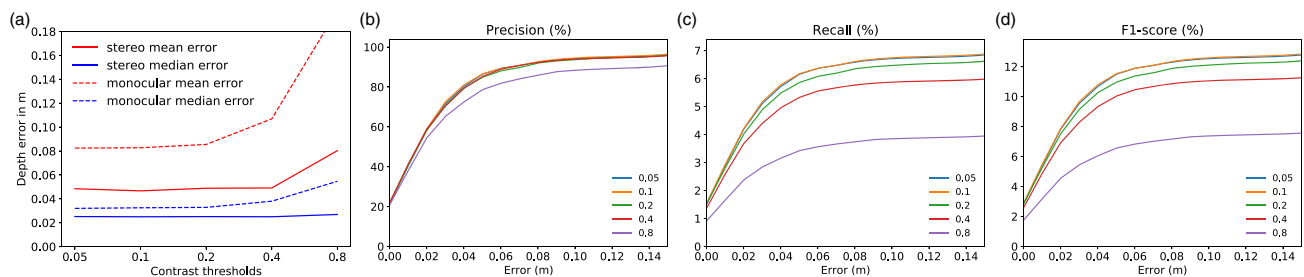


Figure C2. a) Sensitivity of depth errors with respect to the sensor's contrast threshold. Maximum scene depth is 2.7 m. b–d) Precision, recall and F1-score for depth estimated using Algorithm 1 using events simulated with different contrast thresholds.

Varying more than the temporal contrast is possible with different prototype vision sensors, such as parallel processor arrays (PPAs). These sensors embed a processor within each pixel and are thus programmable, enabling a richer family of operations (at the expense of larger pixel sizes or more transistor layers).^[79]

Appendix D. Per-Sequence Quantitative Results

Tables D1, D2, and D3 report depth estimation metrics on individual sequences of the MVSEC dataset. Table 3 is the average of these per-sequence tables.

Table D1. Quantitative comparison of the proposed methods with the SOTA. MVSEC flying1. See Table 3.

	Algorithm	Mean Err [cm] ↓	Median Err [cm] ↓	bad-pix [%] ↓	SILog Err ×100 ↓	AErrR [%] ↓	log RMSE ×100 ↓	$\delta < 1.25$ [%] ↑	$\delta < 1.25^2$ [%] ↑	$\delta < 1.25^3$ [%] ↑	#Points [million] ↑
SOTA	EMVS ^[10] (monocular)	39.37	14.95	3.05	4.72	13.25	22.10	82.03	93.43	97.62	1.21
	ESVO ^[15]	24.04	10.21	2.54	2.94	9.76	17.17	91.43	96.53	98.55	1.95
	ESVO indep. 1s	23.39	10.03	2.18	2.79	9.78	16.72	91.57	96.84	98.79	1.41
	SGM indep. 1s	35.45	13.61	5.54	7.35	15.03	27.46	85.96	93.51	96.40	11.64
	GTS indep. 1s	700.37	38.39	32.51	79.26	111.21	91.44	54.27	67.16	73.39	0.03
Ours	$H_c \circ A_t$ (Alg 1)	22.53	9.72	1.30	1.94	7.91	14.11	93.49	97.50	99.17	0.96
	$H_c \circ A_t$ (Alg 1) + MF	23.33	9.90	1.39	2.08	8.12	14.61	93.16	97.28	99.05	3.48
	$H_c \circ H_t$	24.38	10.81	1.40	2.11	8.59	14.71	92.53	97.38	99.15	1.24
	$H_t \circ A_c$	23.78	10.37	1.26	2.03	8.36	14.41	92.72	97.54	99.21	1.02
	$A_c \circ H_t$	24.48	10.78	1.39	2.14	8.62	14.81	92.48	97.34	99.13	1.20
	$A_c \circ A_t$	21.93	9.35	1.22	1.87	7.75	13.85	93.65	97.65	99.23	0.87
	$A_t \circ H_c$	22.32	9.63	1.31	1.92	7.85	14.02	93.62	97.51	99.17	1.03
	$A_t \circ H_c$ + shuffling	23.92	10.60	1.28	2.01	8.40	14.36	92.76	97.55	99.21	1.11

Table D2. Quantitative comparison of the proposed methods with the SOTA. MVSEC flying2. See Table 3.

	Algorithm	Mean Err [cm] ↓	Median Err [cm] ↓	bad-pix [%] ↓	SILog Err ×100 ↓	AErrR [%] ↓	log RMSE ×100 ↓	$\delta < 1.25$ [%] ↑	$\delta < 1.25^2$ [%] ↑	$\delta < 1.25^3$ [%] ↑	#Points [million] ↑
SOTA	EMVS ^[10] (monocular)	31.42	13.01	6.15	4.56	13.37	21.80	84.07	94.72	97.88	1.17
	ESVO ^[15]	21.34	8.97	3.75	3.48	9.32	19.14	91.60	95.88	97.86	1.89
	ESVO indep. 1s	20.42	8.63	3.50	3.24	9.14	18.35	92.03	96.19	98.19	1.41
	SGM indep. 1s	32.94	8.75	8.29	9.50	15.82	31.54	84.40	92.33	95.48	16.95
	GTS indep. 1s	167.14	37.23	43.08	71.91	94.78	86.93	49.36	60.54	67.76	0.07
Ours	$H_c \circ A_t$ (Alg 1)	18.20	8.49	1.77	1.78	8.13	13.59	95.53	98.13	99.08	0.65
	$H_c \circ A_t$ (Alg 1) + MF	18.58	8.68	1.86	1.81	8.19	13.71	95.27	98.07	99.09	2.42
	$H_c \circ H_t$	22.47	10.13	2.92	2.46	9.41	16.07	92.66	97.19	98.79	1.39
	$H_t \circ A_c$	21.37	9.86	2.50	2.26	9.13	15.41	93.41	97.60	98.92	1.18
	$A_c \circ H_t$	22.28	10.06	2.81	2.40	9.35	15.88	92.80	97.28	98.82	1.34
	$A_c \circ A_t$	19.04	8.76	2.24	1.99	8.44	14.39	94.68	97.89	99.02	1.01
	$A_t \circ H_c$	19.82	8.91	2.55	2.09	8.60	14.77	94.16	97.55	98.94	1.17
	$A_t \circ H_c$ + shuffling	21.66	9.91	2.60	2.22	9.16	15.27	93.14	97.53	98.92	1.22

Table D3. Quantitative comparison of the proposed methods with the SOTA. MVSEC flying3. See Table 3.

	Algorithm	Mean Err [cm] ↓	Median Err [cm] ↓	bad-pix [%] ↓	SILog Err ×100 ↓	AErrR [%] ↓	log RMSE ×100 ↓	$\delta < 1.25$ [%] ↑	$\delta < 1.25^2$ [%] ↑	$\delta < 1.25^3$ [%] ↑	#Points [million] ↑
SOTA	EMVS ^[10] (monocular)	30.54	15.09	2.31	3.33	11.59	18.27	88.16	96.45	98.47	1.42
	ESVO ^[15]	29.62	12.61	3.78	4.02	11.50	20.20	88.28	94.88	97.52	2.29
	ESVO indep. 1s	24.29	10.84	2.81	3.05	9.84	17.54	91.87	96.46	98.16	1.86
	SGM indep. 1s	37.86	14.69	5.33	8.52	17.65	29.46	85.67	93.31	96.21	14.81
	GTS indep. 1s	299.48	60.66	39.75	72.24	102.77	88.87	45.04	58.86	66.94	0.08
Ours	$H_c \circ A_t$ (Alg 1)	19.49	10.38	0.99	1.43	7.35	12.00	96.09	98.60	99.38	0.82
	$H_c \circ A_t$ (Alg 1) + MF	20.02	10.59	1.02	1.50	7.50	12.29	95.79	98.51	99.36	3.11
	$H_c \circ H_t$	23.48	11.99	1.36	1.95	8.58	14.01	93.79	97.90	99.14	1.79
	$H_t \circ A_c$	22.68	11.83	1.24	1.81	8.35	13.49	94.33	98.14	99.24	1.55
	$A_c \circ H_t$	23.29	11.97	1.32	1.91	8.52	13.85	93.94	97.97	99.17	1.72
	$A_c \circ A_t$	20.17	10.68	1.06	1.53	7.59	12.39	95.67	98.48	99.36	1.09
	$A_t \circ H_c$	20.61	10.74	1.13	1.59	7.68	12.64	95.40	98.35	99.30	1.23
	$A_t \circ H_c$ + shuffling	22.24	11.60	1.17	1.73	8.17	13.20	94.58	98.21	99.27	1.39

Supporting Information

Supporting Information is available from the Wiley Online Library or from the project page: https://github.com/tub-rip/dvs_mcemvs.

Conflict of Interest

The authors declare no conflict of interest.

Data Availability Statement

The data that support the findings of this study are openly available in 1) MVSEC: The Multi Vehicle Stereo Event Camera dataset at

<https://daniilidis-group.github.io/mvsec/>; 2) UZH-RPG: University of Zurich-Robotics and Perception Group stereo event dataset at https://rpg.ifi.uzh.ch/ECCV18_stereo_davis.html; 3) DSEC: A Stereo Event Camera Dataset for Driving Scenarios at <https://dsec.ifi.uzh.ch/>; 4) TUM-VIE: The TUM Stereo Visual-Inertial Event Data Set at <https://vision.in.tum.de/data/datasets/visual-inertial-event-dataset>; 5) EVIMO2v1: Motion Segmentation Dataset and Learning Pipeline for Event Cameras at https://better-flow.github.io/evimo/download_evimo_2_v1.html.

Keywords

event cameras, neuromorphic processing, robotics, spatial AI, stereo depth estimation

Received: July 21, 2022
Revised: August 22, 2022
Published online: September 23, 2022

- [1] D. Berco, D. S. Ang, *Adv. Intell. Syst.* **2021**, 3, 2100025.
- [2] G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, D. Scaramuzza, *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, 44, 154.
- [3] D. Berco, D. Shenp Ang, *Adv. Intell. Syst.* **2019**, 1 1900003.
- [4] P. Lichtsteiner, C. Posch, T. Delbruck, *IEEE J. Solid-State Circuits* **2008**, 43, 566.
- [5] T. Finatou, A. Niwa, D. Matolin, K. Tsuchimoto, A. Mascheroni, E. Reynaud, P. Mostafalu, F. Brady, L. Chotard, F. LeGoff, H. Takahashi, H. Wakabayashi, Y. Oike, C. Posch, in *IEEE Intl. Solid-State Circuits Conf. (ISSCC)*, IEEE, Piscataway, NJ **2020**, pp. 112–114.
- [6] Y. Suh, S. Choi, M. Ito, J. Kim, Y. Lee, J. Seo, H. Jung, D.-H. Yeo, S. Namgung, J. Bong, J. Seok Kim, P. K. J. Park, J. Kim, H. Ryu, Y. Park, in *IEEE Int. Symp. Circuits Syst.*, IEEE, Piscataway, NJ **2020**, pp. 1–5.
- [7] T. Delbruck, M. Lang, *Front. Neurosci.* **2013**, 7 223.
- [8] N. Matsuda, O. Cossairt, M. Gupta, in *IEEE Int. Conf. Comput. Photography*, IEEE, Piscataway, NJ **2015** pp. 1–10.
- [9] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. D. Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza, J. Kusnitz, M. Debole, S. Esser, T. Delbruck, M. Flickner, D. Modha, in *IEEE Conf. Comput. Vis. Pattern Recog.*, IEEE, Piscataway, NJ **2017** pp. 7388–7397.
- [10] H. Rebecq, G. Gallego, E. Mueggler, D. Scaramuzza, *Int. J. Comput. Vis.* **2018**, 126, 1394.
- [11] W. Wang, E. Covi, A. Milozzi, M. Farronato, S. Ricci, C. Sbandati, G. Pedretti, D. Ielmini, *Adv. Intell. Syst.* **2021**, 3, 2000224.
- [12] M. E. Sayed, J. O. Roberts, K. Donaldson, S. T. Mahon, F. Iqbal, B. Li, S. Franco Aixela, G. Mastorakis, E. T. Jonasson, M. P. Nemitz, S. Bernardini, A. A. Stokes, *Adv. Intell. Syst.* **2022**, 4, 2000227.
- [13] H. Rebecq, T. Horstschäfer, G. Gallego, D. Scaramuzza, *IEEE Robot. Autom. Lett.* **2017**, 2, 593.
- [14] H. Kim, S. Leutenegger, A. J. Davison, in *Eur. Conf. Comput. Vis. (ECCV)*, Springer, Cham **2016**, pp. 349–364.
- [15] Y. Zhou, G. Gallego, S. Shen, *IEEE Trans. Robot.* **2021**, 37, 1433.
- [16] G. Gallego, H. Rebecq, D. Scaramuzza, in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, IEEE, Piscataway, NJ **2018**, pp. 3867–3876.
- [17] G. Gallego, M. Gehrig, D. Scaramuzza, in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, IEEE, Piscataway, NJ **2019**, pp. 12272–12281.
- [18] T. Stoffregen, L. Kleeman, in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, IEEE, Piscataway, NJ **2019**, pp. 12292–12300.
- [19] T. Stoffregen, G. Gallego, T. Drummond, L. Kleeman, D. Scaramuzza, in *Int. Conf. Comput. Vis. (ICCV)*, IEEE, Piscataway, NJ **2019**, pp. 7243–7252.
- [20] Z. W. Wang, P. Duan, O. Cossairt, A. Katsaggelos, T. Huang, B. Shi, in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, IEEE, Piscataway, NJ **2020**, pp. 1606–1616.
- [21] U. M. Nunes, Y. Demiris, *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, <https://ieeexplore.ieee.org/abstract/document/9625712>.
- [22] X. Peng, L. Gao, Y. Wang, L. Kneip, *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, 44, 3479.
- [23] S. Shiba, Y. Aoki, G. Gallego, in *Eur. Conf. Comput. Vis. (ECCV)*, **2022**, Unpublished.
- [24] J. J. Hagenaars, F. Paredes-Valles, G. C. H. E. de Croon, in *Advances in Neural Information Processing Systems (NeurIPS)*, Vol 34, **2021**, pp. 7167–7179.
- [25] S. Shiba, Y. Aoki, G. Gallego, *Sensors* **2022**, 22, 1.
- [26] H. Seok, J. Lim, in *IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, IEEE, Piscataway, NJ **2020**, pp. 1647–1656.
- [27] S. Gao, Y. Dai, A. Nathan, *Adv. Intell. Syst.* **2022**, 4, 2100074.
- [28] S. Ghosh, G. D'Angelo, A. Glover, M. Iacono, E. Niebur, C. Bartolozzi, *Sci. Rep.* **2022**, 12, 7645.
- [29] M. Mahowald, *Ph.D. Thesis*, California Institute of Technology, Pasadena, CA, **1992**.
- [30] M. Mahowald, *The Silicon Retina*, 4–65 Springer US, Boston, MA **1994**, ISBN 978-1-4615-2724-4.
- [31] D. Marr, T. Poggio, *Science* **1976**, 194, 283.
- [32] E. Piatkowska, A. N. Belbachir, M. Gelautz, in *Int. Conf. Comput. Vis. Workshops (ICCVW)*, **2013** pp. 45–50.
- [33] M. Firouzi, J. Conradt, *Neural Proc. Lett.* **2016**, 43, 311.
- [34] M. Osswald, S.-H. Ieng, R. Benosman, G. Indiveri, *Sci. Rep.* **2017**, 7, 1.
- [35] L. Steffen, D. Reichard, J. Weinland, J. Kaiser, A. Rönna, R. Dillmann, *Front. Neurobot.* **2019**, 13 28.
- [36] J. Furmonas, J. Liobe, V. Barzdenas, *Sensors* **2022**, 22, 1201.
- [37] Y. Zhou, G. Gallego, H. Rebecq, L. Kneip, H. Li, D. Scaramuzza, in *Eur. Conf. Comput. Vis. (ECCV)*, Springer, Cham **2018** pp. 242–258.
- [38] J. N. P. Martel, J. Müller, J. Conradt, Y. Sandamirskaya, in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, IEEE, Piscataway, NJ **2018** pp. 1–5.
- [39] M. Muglikar, G. Gallego, D. Scaramuzza, in *Int. Conf. 3D Vision (3DV)*, **2021** pp. 1165–1174.
- [40] G. Haessig, X. Berthelon, S.-H. Ieng, R. Benosman, *Sci. Rep.* **2019**, 9, 3744.
- [41] J. Carneiro, S.-H. Ieng, C. Posch, R. Benosman, *Neural Netw.* **2013**, 45 27.
- [42] S.-H. Ieng, J. Carneiro, M. Osswald, R. Benosman, *Front. Neurosci.* **2018**, 12 442.
- [43] R. Benosman, S.-H. Ieng, C. Clercq, C. Bartolozzi, M. Srinivasan, *Neural Netw.* **2012**, 27 32.
- [44] E. Piatkowska, A. N. Belbachir, M. Gelautz, *Meas. Sci. Technol.* **2014**, 25, 055108.
- [45] D. Gehrig, H. Rebecq, G. Gallego, D. Scaramuzza, *Int. J. Comput. Vis.* **2020**, 128 601.
- [46] X. Lagorce, G. Orchard, F. Gallupi, B. E. Shi, R. Benosman, *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, 39, 1346.
- [47] R. T. Collins, in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, IEEE, Piscataway, NJ **1996**, pp. 358–363.
- [48] D. Yuan, R. Chung, in *IEEE Int. Conf. Robot. Autom. (ICRA)*, Vol. 2 IEEE, Piscataway, NJ **2003** pp. 1688–1693.
- [49] S. Lehmann, A. P. Bradley, J. Williams, P. J. Kootsookos, L. Clarkson, *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, 29, 82.
- [50] J. Aloimonos, J.-Y. Herve, *IEEE Trans. Pattern Anal. Mach. Intell.* **1990**, 12, 504.
- [51] H. Cho, J. Jeong, K.-J. Yoon, *IEEE Robot. Autom. Lett.* **2021**, 6, 6709.
- [52] C. Brandli, R. Berner, M. Yang, S.-C. Liu, T. Delbruck, *IEEE J. Solid-State Circuits* **2014**, 49, 2333.
- [53] C.-A. Deledalle, L. Denis, F. Tupin, *Int. J. Comput. Vis.* **2012**, 99, 86.
- [54] G. Gallego, D. Scaramuzza, *IEEE Robot. Autom. Lett.* **2017**, 2, 632.
- [55] M. Gehrig, W. Aarents, D. Gehrig, D. Scaramuzza, *IEEE Robot. Autom. Lett.* **2021**, 6, 4947.
- [56] A. Z. Zhu, D. Thakur, T. Ozaslan, B. Pfrommer, V. Kumar, K. Daniilidis, *IEEE Robot. Autom. Lett.* **2018**, 3, 2032.
- [57] A. Mitrokhin, C. Ye, C. Fermuller, Y. Aloimonos, T. Delbruck, in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, IEEE, Piscataway, NJ **2019**, pp. 6105–6112.
- [58] S. Klenk, J. Chui, N. Demmel, D. Cremers, in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, IEEE, Piscataway, NJ **2021** pp. 8601–8608.
- [59] H. Rebecq, D. Gehrig, D. Scaramuzza, in *Conf. on Robotics Learning (CoRL)*, Vol 87, of Proc. Machine Learning Research. PMLR, **2018**, pp. 969–982, <https://proceedings.mlr.press/v87/rebecq18a.html>.

- [60] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, D. Scaramuzza, *Int. J. Robot. Research* **2017**, 36, 142.
- [61] M. Muglikar, M. Gehrig, D. Gehrig, D. Scaramuzza, in *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, IEEE, Piscataway, NJ **2021**, pp. 1403–1409.
- [62] A. Geiger, P. Lenz, R. Urtasun, in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, IEEE, Piscataway, NJ **2012**, pp. 3354–3361.
- [63] C. Ye, A. Mitrokhin, C. Parameshwara, C. Fermüller, J. A. Yorke, Y. Aloimonos, in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, IEEE, Piscataway, NJ **2020**, pp. 5831–5838.
- [64] M. Pizzoli, C. Forster, D. Scaramuzza, in *IEEE Int. Conf. Robot. Autom. (ICRA)*, IEEE, Piscataway, NJ **2014**, pp. 2609–2616.
- [65] H. Hirschmuller, *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, 30, 328.
- [66] C. Brandli, T. Mantel, M. Hutter, M. Höpfinger, R. Berner, R. Siegwart, T. Delbruck, *Front. Neurosci.* **2014**, 7 275.
- [67] A. Z. Zhu, Y. Chen, K. Daniilidis, in *Eur. Conf. Comput. Vis. (ECCV)*, Springer, Cham **2018**, pp. 438–452.
- [68] S. Tulyakov, F. Fleuret, M. Kiefel, P. Gehler, M. Hirsch, in *Int. Conf. Comput. Vis. (ICCV)*, IEEE, Piscataway, NJ **2019**, pp. 1527–1537.
- [69] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, D. Scaramuzza, *IEEE Trans. Robot.* **2017**, 33, 249.
- [70] V. Usenko, N. Demmel, D. Schubert, J. Stueckler, D. Cremers, *IEEE Robot. Autom. Lett.* **2020**, 5, 422.
- [71] L. Burner, A. Mitrokhin, C. Fermüller, Y. Aloimonos, arXiv e-prints **2022**.
- [72] B. Son, Y. Suh, S. Kim, H. Jung, J.-S. Kim, C. Shin, K. Park, K. Lee, J. Park, J. Woo, Y. Roh, H. Lee, Y. Wang, I. Ovsiannikov, H. Ryu, in *IEEE Intl. Solid-State Circuits Conf. (ISSCC)*, IEEE, Piscataway, NJ **2017**, pp. 66–67.
- [73] Prophesee Evaluation Kits, <https://www.prophesee.ai/event-based-evk/>, (accessed: 2020).
- [74] G. Taverni, D. P. Moeys, C. Li, C. Cavaco, V. Motsnyi, D. S. S. Bello, T. Delbruck, *IEEE Trans. Circuits Syst. II* **2018**, 65, 677.
- [75] A. Z. Zhu, L. Yuan, K. Chaney, K. Daniilidis, in *Robotics: Science and Systems (RSS)*, **2018** pp. 1–9, Robotics: Science and Systems <http://www.roboticsproceedings.org/rss14/p62.html>.
- [76] J. Hidalgo-Carrió, G. Gallego, D. Scaramuzza, in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, IEEE, Piscataway, NJ **2022** pp. 5781–5790.
- [77] Z. Wang, D. Yuan, Y. Ng, R. Mahony, in *IEEE Int. Conf. Robot. Autom. (ICRA)*, IEEE, Piscataway, NJ **2022**, pp. 398–404.
- [78] A. J. Davison, arXiv e-prints **2018**.
- [79] P. Dudek, T. Richardson, L. Bose, S. Carey, J. Chen, C. Greatwood, Y. Liu, W. Mayol-Cuevas, *Sci. Robot.* **2022**, 7, eabl7755.