HASTE: multi-Hypothesis Asynchronous Speeded-up Tracking of Events

Ignacio Alzugaray
ialzugaray@mavt.ethz.ch
Margarita Chli
mchli@ethz.ch
Vision For Robotics Lab
ETH Zürich
Zurich, Switzerland

Abstract

Feature tracking using event cameras has experienced significant progress lately, with methods achieving comparable performance to feature trackers using traditional frame-based cameras, even outperforming them on certain challenging scenarios. Most of the event-based trackers, however, still operate on intermediate, frame-like representations generated from accumulated events, on which traditional frame-based techniques can be adopted. Attempting to harness the sparsity and asynchronicity of the event stream, other approaches have emerged to process each event individually, but they lack both in accuracy and efficiency in comparison to the event-based, frame-like alternatives.

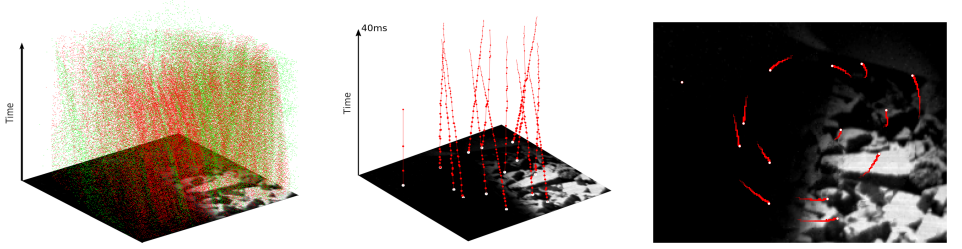
Aiming to address this shortcoming of asynchronous approaches, in this paper, we propose an asynchronous patch-feature tracker that relies solely on events and processes each event individually as soon as it gets generated. We report significant improvements in tracking quality over the state of the art in publicly available datasets, while performing an order of magnitude more efficiently than similar asynchronous tracking approaches.

Code – https://github.com/ialzugaray/haste
Video – https://youtu.be/6DZxIzrVLcI

1 INTRODUCTION

Visual feature tracking is a core component of most Visual Odometry (VO) systems (*e.g.* [10], [16]). Despite its crucial role on maintaining an accurate odometry estimation, visual tracking performance is severely compromised with degrading quality of the input images, *e.g.* blurry images captured during high-speed motion or poorly contrasted images in scenes with High Dynamic Range (HDR) of illumination. Event cameras have been demonstrated to cope well in some of these scenarios by enabling independent pixel reactions to small intensity changes. Specifically, the instant that the intensity perceived at a particular pixel varies beyond a pre-specified threshold, a new event gets generated at that pixel along with a highly accurate time-stamp. Events encode only binary information, indicating whether the intensity at a pixel has increased or decreased (referred to as the event's polarity) discarding any information on the absolute intensity values. Consequently, the output of event cameras is a sparse, asynchronously generated stream of time-stamped events encoding small, binary and incremental intensity changes on the scene at individual pixels as illustrated in Fig. 1a.

Event cameras have not only been disruptive for their sensing capabilities, but they have also stimulated research into an emerging event-driven paradigm that has the potential to overcome traditional frame-based vision in a variety of tasks. Early event-vision research, however, resulted in several methods that discretize the event stream into small batches, on which approaches similar to those applied in



(a) Event Stream (b) Spatio-Temporal Feature Tracks (c) In-plane Feature Tracks

Figure 1: Illustrative example in which each event in the stream (a) from the dataset `hdr_poster` [15] is asynchronously processed into high-quality tracks (b-c) despite the poor lighting conditions.

traditional frame-based vision can be adapted to, partially defying the distinctive characteristics of the event cameras. With this in mind, a handful of recent methods, including this work, advocate for the asynchronous processing of each event as soon at is generated, *i.e.* refraining from any form of batch processing, and thus exploiting the natural asynchronicity and sparsity of the event stream, defining what we refer to as 'event-driven' algorithms.

This paper proposes a novel event-driven tracking methodology for patch features that only relies on raw event data. In general, event-driven tracking algorithms lag both in efficiency and tracking quality with respect to their batch-processing counterparts. This work explicitly addresses the efficiency of event-driven tracking of features by reusing most of the available information between consecutive, asynchronously processed events. Specifically, we revisit the tracker described in [3] and propose a novel methodology that accounts for a speed-up by an order of magnitude without compromising the tracking quality. Inspired by this improvement, we go a step further to also propose a novel event-driven tracking formulation that retrieves more stable tracks as seen in Fig. 1, while retaining the boost in computational performance. All the proposed approaches are also compared against the most related, publicly available, event-based patch tracker in the state-of-the-art [23] on benchmarking datasets [15].

2 RELATED WORK

While some event-based VO pipelines conveniently circumvented the definition of explicit features [8, 17, 21], most of them include diverse feature tracking modules. Zhu *et al.* [23] propose a method to locally correct the event pixel location by compensating the motion of the camera based on locally estimated optical flow. This method generates a frame-like patch representation of each feature using batches of motion-compensated events, which is subsequently tracked posing a template aligning problem. The same authors extend their method in [22] to use an Inertial Measurement Unit (IMU) and perform filter-based VO. Similarly, Rebecq *et al.* [18] also compensate for the motion of the camera in the event stream but relying on an IMU and applied to the whole field of view, generating motion-compensated frame-like images, on which traditional KLT [12] is applied. Combining both event- and intensity-based sensing modalities, Kueng *et al.* [9] detect Canny edges on intensity images that are used to generated per-feature templates and matched to batches of events via Iterative Closest Point (ICP) optimization. Likewise, the authors in [6] employ a generative model to optimize for the alignment of batches of events to intensity-based template patches.

The aforementioned approaches discretize the event stream into small batches between tracking iterations that are often processed via expensive iterative optimization schemes, defying the natural asynchronicity of the event cameras. With the emergence of asynchronous, event-driven corner features detection algorithms [2, 11, 13, 14, 20], several other works have attempted to perform feature tracking by establishing data association between those feature detections in an event-driven fashion [1, 5], but

their accuracy is yet to match previously referenced event-batching tracking methods. Without relying on a stream of finely detected feature detections, the work in [3] describes a novel generic framework that circumvents the need for expensive iterative optimization schemes and proposes a novel patch-feature tracking methodology that operates in a event-driven fashion.

In this paper, we build on top of the framework described in [3] (briefly summarized in Section 3.1) to firstly propose key modifications of the tracking method (Section 3.2) shown to achieve substantial computational advantage over the original method in Section 3.3. Based on similar assumptions, another tracking formulation is then proposed in Section 3.4 and within the same framework this is shown to improve even further the performance in tracking quality while achieving competitive results with respect to event-batching feature tracking approaches.

3 METHODOLOGY

3.1 Multi-Hypothesis Event-based Feature Tracking

The generic optimization framework described in [3] serves as the basis for this work and thus, we provide a brief description of it applied to feature tracking.

Each feature at time t_k is completely defined by an $n \times n$ template patch $\mathcal{T}^{(k)} \in \mathbb{R}^{n \times n}$ and a state $\mathbf{x}^{(k)} = \{x, y, \theta\} \in \mathcal{X}$, specifying its xy-pixel coordinates and its orientation θ in the image plane. While the complete event stream could be used to determine the optimal feature state at each time, we assume that only a small set of spatially and temporally neighboring events is sufficient for this. Thus, each feature is associated to a small window of the latest m events $\mathcal{E}^{(k)} = \{\mathbf{e}_{k-m+1}, \mathbf{e}_{k-m+2}, \ldots, \mathbf{e}_k\}$ sorted in ascending order of time-stamps (m is tuned as in [3]), generated within the feature's range until t_k, *i.e.* within a maximum of n pixels away from the current feature's position. An event occurring at time-stamp t_k is defined as $\mathbf{e}_k = \{t_k, \mathbf{p}_k\}$, indicating its 2D pixel location $\mathbf{p}_k \in \mathbb{R}^2$ (the event's polarity is ignored). Each such event location $\mathbf{p}_k$ is transformed to the feature's reference frame indicating the so-called 'template location' as $\mathbf{p}'_k(\mathbf{x}) = \mathbf{p}'(\mathbf{p}_k, \mathbf{x}) = R^T(\theta)(\mathbf{p}_k - [x, y]^T)$, such that the rotation matrix $R \in \mathrm{SO}(2)$.

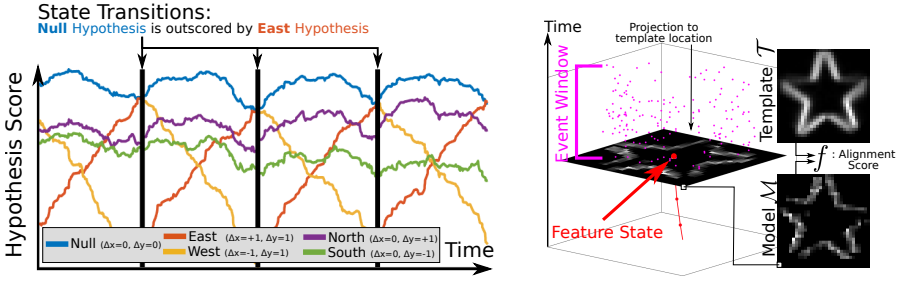
Every time a new event $\mathbf{e}_{k+1}$ gets generated within the range of a particular feature, its fixed-size event window gets updated as $\mathcal{E}^{(k+1)} = \{\mathbf{e}_{k-m+2}, \ldots, \mathbf{e}_k, \mathbf{e}_{k+1}\}$, *i.e.* the oldest event $\mathbf{e}_{k-m+1}$ is replaced by the newest one $\mathbf{e}_{k+1}$. On each new event the feature template $\mathcal{T}$ is also locally refined. The template value $\mathcal{T}[\mathbf{p}'_{\text{mid}}(\mathbf{x}^{(k)})]$, located on the template coordinates $\mathbf{p}'_{\text{mid}}(\mathbf{x}^{(k)}) = \mathbf{p}'(\mathbf{p}_{k-m/2}, \mathbf{x}^{(k)})$ of the middle event $\mathbf{e}_{k-m/2}$ in the event window $\mathcal{E}^{(k)}$, is increased by a predefined scalar value α so that $\mathcal{T}^{(k)}[\mathbf{p}'_{\text{mid}}(\mathbf{x}^{(k)})] = \mathcal{T}^{(k-1)}[\mathbf{p}'_{\text{mid}}(\mathbf{x}^{(k)})] + \alpha$. Consequently, $\mathcal{T}$ and $\mathcal{E}$ get updated as soon as an event is generated within this feature's range, leading to a new optimal feature state according to an alignment score function f of choice, described generically as

$$\mathbf{x}^{(k+1)} = \underset{\mathbf{x} \in \mathcal{X}}{\arg\max}\, f(\mathcal{E}^{(k+1)}, \mathcal{T}^{(k+1)}, \mathbf{x}) \,. \tag{1}$$

Each event conveys such a reduced amount of information that the difference between consecutive optimal states $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(k+1)}$ is negligible. Event-batching methods explicitly exploit this, delaying the evaluation of the optimization problem until enough events are aggregated to perform a 'tracking iteration'. An event-driven alternative approach, *i.e.* without event-batching, is proposed in [3], approximating the optimization problem over a discretized space of solutions or states $\mathcal{X}$ instead.

$$\mathbf{x}^{(k+1)} = \underset{\mathbf{x} \in \mathcal{H}(\mathbf{x}^{(k)})}{\arg\max}\, f(\mathcal{E}^{(k+1)}, \mathcal{T}^{(k+1)}, \mathbf{x}) \,, \tag{2}$$

where $\mathcal{H}(\mathbf{x}^{(k)}) \subset \mathcal{X}$ is a set of discrete hypothetical states or *hypotheses* spawning from the previously known optimum $\mathbf{x}^{(k)}$. The simplest set of discrete hypotheses is computed by slightly perturbing each dimension of a given state $\mathbf{x}$ in the following form $\mathcal{H}(\mathbf{x}) = \{(\mathbf{x} = \{x, y, \theta\}) \cup \{x \pm \Delta x, y \pm \Delta y, \theta \pm \Delta\theta\}\}$



(a) Hypotheses-based Feature Tracking. (b) Template $\mathcal{T}$ and model $\mathcal{M}$.

Figure 2: In (a), each event updates the score of each hypothesis, where the top scoring hypothesis (in blue) is equivalent to the state of the tracker, *i.e.* the null hypothesis. The tracker state changes when the null hypothesis is outscored, generating a new set of hypotheses (5-hypotheses in this example). In (b), the feature's template $\mathcal{T}$ and model $\mathcal{M}$ are illustrated.

(see [3] for further details). Note that $\mathcal{H}(\mathbf{x})$ also includes the state $\mathbf{x}$, referred to as the null hypothesis ($\Delta x = \Delta y = 0$, $\Delta\theta = 0°$). In this paper, we consider the 8-neighbouring locations ($\Delta x = \Delta y = \pm 1$, $\Delta\theta = 0°$) and centered rotations ($\Delta x = \Delta y = 0$, $\Delta\theta = \pm 4°$) as hypothetical states, accounting for a total of 11 hypotheses.

Considering a discretized space of solutions, most newly generated events within the feature range will not result in a state transition in Eq. (2), *i.e.* $\mathbf{x}^{(k)} = \mathbf{x}^{(k+1)}$, and they are referred to here as 'regular' events. The alignment score f of each hypothesis is tracked (see Fig. 2a) and simultaneously updated on each regular event generated within the feature's range. When a hypothesis outscores the null hypothesis, *i.e.* the previously known optimum, it becomes the new optimal state, generating new set of hypotheses and effectively tracking the feature in the discrete space of solutions. In this paper, we consider the null hypothesis outscored if another one scores higher by at least 5%. The events that finally lead to a state transition, *i.e.* $\mathbf{x}^{(k)} \neq \mathbf{x}^{(k+1)}$, are referred to as 'state' events. Note that the discretized states retrieved can potentially be further refined on-demand using the original continuous formulation. The main motivation of this paper is to provide an efficient, event-driven approach to evaluate the alignment score function f for each hypothesis.

3.2 Alignment Score based on Correlation

The alignment score function f proposed in [3] is as follows,

$$f(\mathcal{E},\mathcal{T},\mathbf{x}) = \sum_{\mathbf{e}_i \in \mathcal{E}} w_i \mathcal{T}[\mathbf{p}'_i(\mathbf{x})] \,, \tag{3}$$

where each event $\mathbf{e}_i \in \mathcal{E}$ is projected into its template location $\mathbf{p}'_i(\mathbf{x})$ for each hypothesis $\mathbf{x}$. The value of the template at this location denoted by $\mathcal{T}[\mathbf{p}'_i(\mathbf{x})]$ is sampled via bilinear interpolation. The alignment score is the sum of sampled values weighted according to a Gaussian-like scheme $w_i = \frac{1}{N}\exp(-\frac{1}{2}((i-\frac{m}{2})/(\frac{m}{6}))^2)$, where $i = \{1,\ldots,m\}$ indicates the relative order of each event in $\mathcal{E}$ and N is a normalization factor such that $\sum_i^m w_i = 1$.

We rewrite the sampling operation as $\mathcal{T}[\mathbf{p}'_i(\mathbf{x})] = \sum_\Omega \mathcal{T} \odot S_i(\mathbf{x})$, using the operator $\odot$ to denote element-wise multiplication of the template values in $\mathcal{T}$ with a sparse sampling kernel $S_i(\mathbf{x}) \in \mathbb{R}^{n\times n}$. Then all the template locations, *i.e.* the domain Ω, are summed up. All the entries in the kernel S_i are zero except for those required for the interpolation of $\mathcal{T}[\mathbf{p}'_i(\mathbf{x})]$ (only four non-zero entries when using bilinear interpolation). As a result, Eq. (3) is reformulated as follows,

$$f(\mathcal{E},\mathcal{T},\mathbf{x}) = \sum_{\mathbf{e}_i \in \mathcal{E}} w_i \sum_\Omega \mathcal{T} \odot S_i(\mathbf{x}) = \sum_\Omega \mathcal{T} \odot \sum_{\mathbf{e}_i \in \mathcal{E}} w_i S_i(\mathbf{x}) = \sum_\Omega \mathcal{T} \odot \mathcal{M}(\mathcal{E},\mathbf{x}) \,, \tag{4}$$

where $\mathcal{M}(\mathcal{E},\mathbf{x}) = \sum_{\mathbf{e}_i\in\mathcal{E}} \mathcal{M}_i(\mathbf{x}) = \sum_{\mathbf{e}_i\in\mathcal{E}} w_i S_i(\mathbf{x})$ corresponds to the model, *i.e.* an approximation of the template $\mathcal{T}$ generated solely from events in $\mathcal{E}$ according to a hypothesis $\mathbf{x}$. Eq. (4) reveals that method described in [3] boils down to a tracker based on cross-correlation between $\mathcal{T}$ and $\mathcal{E}$ and for this reason, it is referred here as the '**Correlation**' tracker.

3.3 Incremental Alignment Score based on Correlation

In [3], Eq. (3) is fully re-evaluated with the arrival of each regular event generated within the feature's range to update the score of each tracked hypothesis, making this a costly operation and unsuitable for application to high event-rate streams. More efficient alternatives relying on a set of carefully selected approximations are described below, enabling the reuse of most of the available information between consecutive events and achieving significantly better efficiency.

Eq. (3) is formulated as a vector product $f(\mathcal{E},\mathcal{T},\mathbf{x}) = W^T V(\mathcal{E},\mathcal{T},\mathbf{x})$ of the weighting scheme $W = \begin{bmatrix} w_1 & \dots & w_m \end{bmatrix}^T$, which is constant as $\mathcal{E}$ is fixed-size, and a vector V that aggregates sampled values from $\mathcal{T}$. This vector at time-instant t_k is $V^{(k)} = V(\mathcal{E}^{(k)},\mathcal{T}^{(k)},\mathbf{x}^{(k)}) = \begin{bmatrix} v_{k-m+1}^{(k)} & \dots & v_{k-1}^{(k)} & v_k^{(k)} \end{bmatrix}^T$, aggregating the template-sampling elements $v_i^{(k)} = v_i(\mathcal{T}^{(k)},\mathbf{x}^{(k)}) = \mathcal{T}^{(k)}[\mathbf{p}_i'(\mathbf{x}^{(k)})]$.

The alignment score f of each hypothesis is concurrently tracked in this paper (see Fig. 2a). For each of these hypotheses, $\mathbf{x}$ is constant and thus each i-event is always projected into the same template location $\mathbf{p}_i'(\mathbf{x})$. Each event remains in the event-window $\mathcal{E}$ and influences V, only until m new events are generated within the feature's range. For such a short lifespan, we assume the values of $\mathcal{T}$ remain invariant and thus, here each i-event only samples $\mathcal{T}$ at the location $\mathbf{p}_i'(\mathbf{x})$ when they are first included in $\mathcal{E}$, effectively approximating $v_i^{(k)} \approx v_i = v_i(\mathcal{T}^{(i)},\mathbf{x}) = \mathcal{T}^{(i)}[\mathbf{p}_i'(\mathbf{x})]$ as a constant. After event $\mathbf{e}_{k+1}$ gets included in $\mathcal{E}$, the alignment score can be computed as follows,

$$f^{(k+1)} = f(\mathcal{E}^{(k+1)},\mathcal{T}^{(k+1)},\mathbf{x}^{(k+1)}) = W^T V^{(k+1)} = W^T g(V^{(k)},\mathbf{e}_{k+1}) \,, \tag{5}$$

with g the function that updates the approximated vector $V^{(k)} \approx \begin{bmatrix} v_{k-m+1} & v_{k-m+2} & \dots & v_{k-1} & v_k \end{bmatrix}^T$ into $V^{(k+1)} \approx \begin{bmatrix} v_{k-m+2} & \dots & v_{k-1} & v_k & v_{k+1} \end{bmatrix}^T$ by just replacing the contribution of the oldest event $\mathbf{e}_{k-m+1}$ with the one related to the newest event $\mathbf{e}_{k+1}$ in the fixed-size $\mathcal{E}$.

Eq. (5) enables an efficient re-evaluation of the score of each individual hypothesis after each new event by only retaining the most recent vector of sampled values V for each hypothesis. This approach is referred to as the '**HasteCorrelation**' tracker, denoting the use of multi-Hypothesis Asynchronous Speeded-up Tracking of Events (HASTE) for superior computational efficiency over the original Correlation tracker. Note that, when the feature's state changes (*i.e.* after a state event) and a new set of hypotheses is initialized, all elements in V must be sampled from the most recent template values $\mathcal{T}$ for each hypothesis, essentially reverting to the original approach described in [3].

HasteCorrelation is designed to perform similarly to the original Correlation tracker [3]. However, both approaches rely on an underlying cross-correlation comparing the unnormalized template $\mathcal{T}$ to the model $\mathcal{M}$, potentially leading to poor tracking performance while being computationally expensive due to the employed Gaussian-like weighting scheme. Addressing these issues, here we propose the '**HasteCorrelation***' variant. On the initialization of a new tracked hypothesis, a constant normalized template $\hat{\mathcal{T}} = \mathcal{T}/(\sum_\Omega \mathcal{T})$ is computed. For each event in $\mathcal{E}$, $\hat{\mathcal{T}}$ is sampled as $\hat{v}_i = v_i(\hat{\mathcal{T}},\mathbf{x})$ and weighted uniformly $w_i = \frac{1}{m}$, so that Eq. (5) can be written as follows,

$$f^{(k+1)} = f^{(k)} + w_i(\hat{v}_{k+1} - \hat{v}_{k-m+1}) \,, \quad \text{initially } f^{(m)} = w_i \sum_{\mathbf{e}_i\in\mathcal{E}^{(m)}} \hat{v}_i \,, \tag{6}$$

providing an even more efficient way to update the score while using normalized data for $\hat{\mathcal{T}}$ and $\mathcal{M}$.

3.4 Incremental Alignment Score based on Differences

Building on the approximations introduced for HasteCorrelation*, we also propose '**HasteDifference***' that employs an alignment score function based on the sum of the differences between normalized template $\hat{\mathcal{T}}$ and model $\mathcal{M}$ as follows,

$$f(\mathcal{E},\mathcal{T},\mathbf{x}) = -\sum_{\Omega}|\hat{\mathcal{T}}-\mathcal{M}|^{\gamma} = -\sum_{\Omega}|\hat{\mathcal{T}}-\sum_{\mathbf{e}_i\in\mathcal{E}}\mathcal{M}_i|^{\gamma} = -\sum_{\Omega}|\mathcal{D}|^{\gamma}\,, \tag{7}$$

where all the involved operations (subtraction, absolute value and exponentiation) are element-wise. Here events are also equally weighted, *i.e.* $w_i = \frac{1}{m}$, so that the model $\mathcal{M} = \sum_{\mathbf{e}_i\in\mathcal{E}}\mathcal{M}_i = \sum_{\mathbf{e}_i\in\mathcal{E}} w_i S_i$ is normalized. The absolute differences in patch $\mathcal{D}\in\mathbb{R}^{n\times n}$ are penalized using a factor $\gamma = 2$ and summed to compute the score. The negative sign preceding the expression aims to minimize the differences in $\mathcal{D}$ in the score maximization framework of Eq. (2).

Tracking each hypothesis $\mathbf{x}$ independently and concurrently, each of their respective models $\mathcal{M}^{(k)} = \mathcal{M}(\mathcal{E}^{(k)},\mathbf{x})$ can be locally and incrementally updated as follows,

$$\mathcal{M}^{(k+1)} = \sum_{\mathbf{e}_i\in\mathcal{E}^{(k+1)}}\mathcal{M}_i = \mathcal{M}_{k-m+2}+\cdots+\mathcal{M}_{k+1} = \mathcal{M}^{(k)}-\mathcal{M}_{k-m+1}+\mathcal{M}_{k+1}\,. \tag{8}$$

The normalized template $\hat{\mathcal{T}}$, obtained upon the generation of each new hypothesis, is assumed constant and thus the difference patch $\mathcal{D}$ can be locally updated as follows,

$$\mathcal{D}^{(k+1)} = \mathcal{D}^{(k)}+\mathcal{M}_{k-m+1}-\mathcal{M}_{k+1}\,, \tag{9}$$

which allows the incremental update of the alignment score from the previous score as

$$f^{(k+1)} = f^{(k)}+|\mathcal{D}^{(k)}[\mathbf{p}'_{k-m+1}]|^{\gamma}+|\mathcal{D}^{(k)}[\mathbf{p}'_{k+1}]|^{\gamma}-|\mathcal{D}^{(k+1)}[\mathbf{p}'_{k-m+1}]|^{\gamma}-|\mathcal{D}^{(k+1)}[\mathbf{p}'_{k+1}]|^{\gamma}\,, \tag{10}$$

where $\mathcal{D}[\mathbf{p}'_i]$ is the value (or values, when using bilinear interpolation) of the location $\mathbf{p}'_i$ in the patch of differences $\mathcal{D}$. This procedure is equivalent to the one proposed for HasteCorrelation where, upon the update of the event window to $\mathcal{E}^{(k+1)}$, the influence of the newest event $\mathbf{e}_{k+1}$ replaces the influence of the oldest one $\mathbf{e}_{k-m+1}$, enabling an efficient score update by retaining the most recent $\mathcal{D}$ for each hypothesis. As for HasteCorrelation*, when the feature's state changes, *i.e.* $\mathbf{x}^{(k)}\neq\mathbf{x}^{(k+1)}$, after a state event $\mathbf{e}_{k+1}$ and a set of hypotheses is newly generated, the template $\hat{\mathcal{T}}$ is generated from the most recent values of $\mathcal{T}$ and $\mathcal{D}$ is then re-computed from scratch for each hypothesis.

The alignment score function in Eq. (7) and its variants have widespread application in both event and non-event feature tracking approaches (*e.g.* [7, 13]). The incremental score update proposed here allows such methods to be adapted to the hypothesis-based optimization framework described in [3] while remaining computationally competitive.

4 EXPERIMENTAL EVALUATION

We compare the proposed HASTE trackers with our re-implementation of the original Correlation tracker [3]. Aside from these event-driven methods, we compare against the most related, event-based tracking method [23] for patch features with a publicly available implementation[1] that is referred to as the Event-based Optical-Flow ('**EOF**') tracker. In contrast to other alternatives that employ IMU [18, 22] or intensity images [7, 9, 19], the EOF tracker operates directly on raw events only and process them in batches in consecutive tracking iterations. The default parameters of the EOF tracker are used, except from the patch size, which is adapted to match all the compared methods (*i.e.* 31×31 px), and the number of tracked features in each iteration, which is set to 15.

All approaches are benchmarked on the Event Camera Dataset [15] and, specifically, on the `shapes`, `poster` and `boxes` scenes (sorted by increasing order of visual clutter and complexity).

[1] https://github.com/daniilidis-group/event_feature_tracking

These scenes were recorded under a wide range of handheld camera motions. Here, we choose to illustrate the tracking results on mainly translational (`trans`) and mainly rotational motions (`rot`) or a combination of both (`6dof`) as well as in adverse lighting conditions (`hdr`). The recording sensor is a DAVIS240 [4] that captures both events and intensity images at fixed rate. Here, intensity images are only shown for illustration purposes and are not used in any of the compared methods. The selected datasets last for approximately one minute and come with 6-Degrees of Freedom (DoF) camera poses captured at 200Hz using an indoor positioning system.

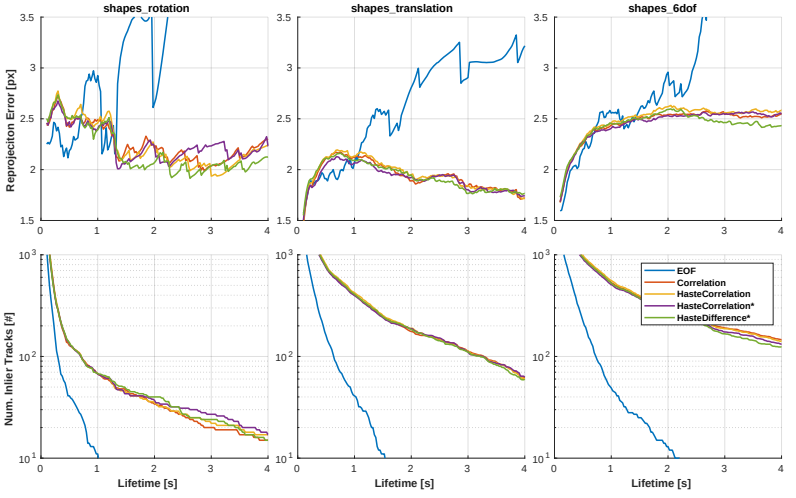
4.1 Tracking Quality

In previous works [2, 7, 23], event-based tracks were evaluated in comparison to intensity-based tracks and thus, in practice, such evaluations are only possible with datasets exhibiting no substantial image degradation, *e.g.* no motion blur and good illumination, which only accounts for a minimal part of the selected datasets. Finally, while various event-based tracking strategies could be have been compared as part of the same VO pipeline here, no asynchronous event-driven VO system is publicly available at submission time to enable this.

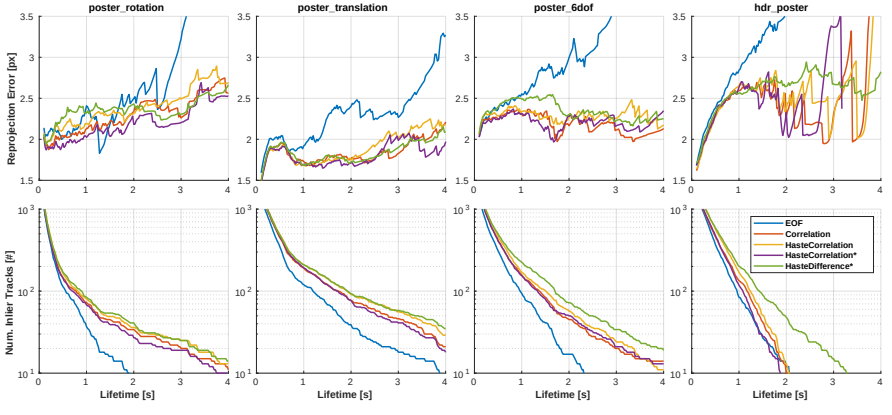
Following these considerations, we opt for the evaluation methodology described in [10, 3], where each instance of a particular feature in image space is combined with the ground-truth camera poses to triangulate a potential 3D point in space giving rise to this feature track. The quality of each track is measured as the mean reprojection error of such 3D triangulated points over all the track. Note that, while a 3D point might not be accurately triangulated (*e.g.* due to low parallax), the mean reprojection error would still indicate how well its track conveys the camera motion according to ground-truth. As all compared methods are prone to drift, the mean error is evaluated as a function of a feature's lifetime (*i.e.* the length of time that its track is alive) and consider that a track becomes an outlier as soon as the mean reprojection error exceeds 5px. Following a buffer time of 0.1s from the initialisation of a new feature track, this track enters the evaluation process, such that noisy results of short-lived feature tracks are minimized. Fig. 3 illustrates the evolution of the mean reprojection error of all the inlier tracks and the number of inlier tracks with respect to track lifetime.

In contrast to the event-driven trackers compared here, the EOF tracker comprises a complete visual front end capable of detecting new features and pruning failing tracks. As a result, all methods compared here get initialized with the same set of feature detections as used in the EOF tracker for fairness. The compared event-driven approaches do not currently implement any strategy to remove features that are failing and thus they track them until they leave the field instead. Since in our evaluation we only consider each tracked feature until it becomes an outlier, the reported results for the event-driven methods present the maximum potential that could be achieved with the same set of detected features as in the EOF tracker in a real setup.

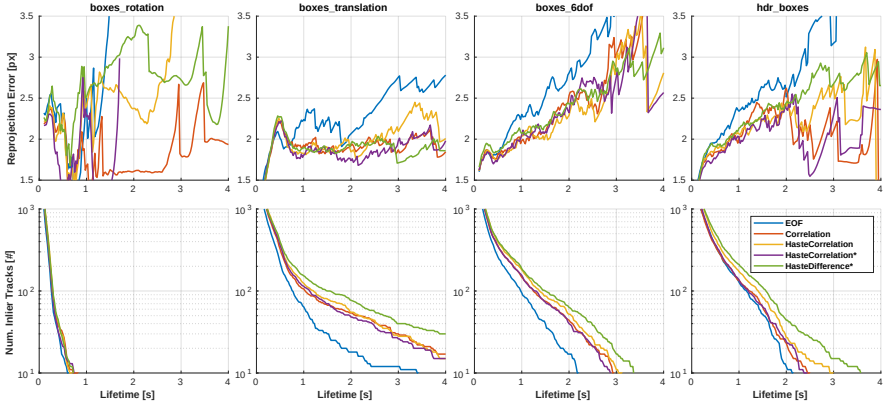
All compared methods exhibit generally similar mean reprojection error levels on short-lived tracks. Notably, the EOF tracks degrade faster leaving only a handful of features to be tracked for longer lifetimes, which, in general, perform worse in terms of error than the event-driven alternatives. Contrary to our expectation that HasteCorrelation would exhibit significantly worse tracking quality than the original Correlation tracker [3] due the several approximations that were required for its definition, we observe no significant differences in the mean inlier track error nor the features' lifetime. Similarly, the reported performance for HasteCorrelation* matches that of the original Correlation tracker and the proposed HasteCorrelation, which is evidence that HasteCorrelation* can still perform competitively even when removing the smooth weighting scheme, while being significantly more efficient. Interestingly, HasteDifference* is, in general, capable of retrieving more stable tracks, consistently achieving longer lifetimes than the rest of the methods in the most cluttered and complex datasets while achieving competitive mean reprojection errors even on the long-lived tracks. We believe that the superior performance of HasteDifference* over the other event-driven alternatives arise from using the information of all the locations in the template $\mathcal{T}$ when comparing it to the model $\mathcal{M}$ (see Eq. (7)), whereas the proposed correlation-based methods only consider the locations where events are being projected (see Eq. (3)).



(a) `shapes` scene.

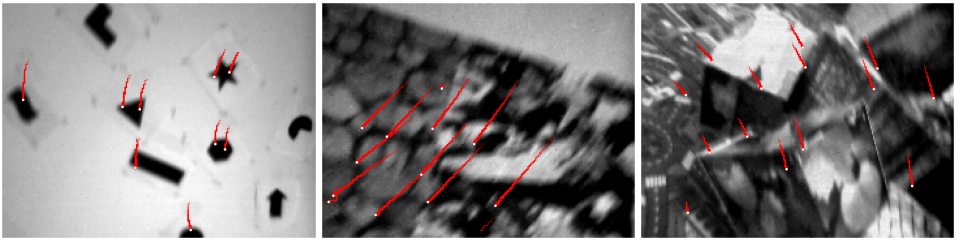


(b) `poster` scene.



(c) `boxes` scene.

Figure 3: Mean reprojection error and number of inlier tracks relative to the feature lifetime.



(a) `shapes_6dof` (b) `poster_6dof` (c) `boxes_6dof`

Figure 4: Example feature tracks with the HasteDifference* tracker over 40ms.

	Regular Event		State Event		All
Tracker	Num. [%]	Time [μs/ev]	Num. [%]	Time [μs/ev]	Time [μs/ev]
Correlation [3]	98.88	47.91	1.12	95.79	48.45
HasteCorrelation	98.77	2.98	1.23	56.15	3.64
HasteCorrelation*	99.05	**1.51**	0.95	**40.47**	**1.88**
HasteDifference*	98.18	1.99	1.82	61.36	3.07

Table 1: Timings for 'regular' and 'state' events, and their combined performance, indicating the percentage of the events used for the computation. The best figure is shown in boldface.

Visual examples of tracks retrieved with HasteDifference* are presented in Fig. 1 and 4, explicitly choosing challenging instants, where traditional frame-based vision is likely to fail.

4.2 Computational Performance

The main contribution of this work is the drastic drop in computational cost of event-driven tracking as reported in Table 1, enabling its use in practical applications. These timings are generated by running all methods on an Intel Xeon E3-1505m CPU at 2.8GHz with 16GB using a single-threaded C++ implementation. No computational performance metric is provided for the EOF tracker as the only publicly available implementation is coded in MATLAB, taking up to several hours to complete a single dataset using multi-threading. For completeness, we refer the reader to the IMU-based EOF tracker's successor in [22], where the authors claim that a non-public C++ implementation is able to track up to 15 features (same as specified in this manuscript) in real-time on a 6 core CPU, but only under moderate optical flow, signifying an overall high computational cost.

In Table 1, processed events within the feature's range are distinguished as either regular or state events, as the latter involve a significant computational overhead derived from the initialization of new hypotheses. The average processing time per event and the percentage of their occurrence in the stream are reported ('All' including both regular and state events). We observe that all the HASTE variants computationally outperform the original Correlation tracker [3] by an order or magnitude. The main advantage arises from the efficient processing of regular events, which account for the vast majority of the processed events, in a far more efficient manner by re-using most information between consecutive events. Among the HASTE variants, HasteCorrelation perform computationally the worst as it retain the original Guassian-like weighting scheme and it needs to be applied in the dot product of Eq. (5). The computational differences between HasteCorrelation* and HasteDifference* arise from the fact that the former must retain and access only a vector of sampled values V of m elements for each hypothesis whereas the latter requires the retaining and accessing of a full $n \times n$ patch $\mathcal{D}$ for each hypothesis. The reported results indicate that HasteCorrelation* could be a valid alternative to HasteDifference* for systems with constrained resources at expenses of a diminished tracking quality.

Although the average event-rate of these datasets ranges between 300K-3M events per second, only the events within the range of an active feature are processed, and thus, real-time performance depends on the number of concurrently tracked features and their overlap. The presented results indicate that the strategy proposed here is, to the best of our knowledge, the most efficient, event-driven patch-based feature tracking methodology to date.

5 Conclusions

This paper extends the capabilities of the hypothesis-based patch tracking methods relying solely on events, by revisiting the tracker described in [3] and proposing two novel variants, HasteCorrelation and HasteCorrelation*, that achieve over an order of magnitude speed-up without any significant trade-off in tracking quality.

Building on the approximations introduced for the correlation-based HASTE trackers, we also propose the novel the HasteDifference* tracker, exhibiting even better tracking quality than the aforementioned methods while retaining the computational advantage of the HASTE variants.

The contributions of this paper represent key advances that enable the application of event-driven methods in realistic, high-speed scenarios, opening up interesting research directions towards the development of a completely event-driven VO pipeline in the future.

References

[1] I. Alzugaray and M. Chli. ACE: An efficient asynchronous corner tracker for event cameras. In *2018 International Conference on 3D Vision (3DV)*, pages 653–661, 2018.

[2] I. Alzugaray and M. Chli. Asynchronous corner detection and tracking for event cameras in real time. *IEEE Robotics and Automation Letters*, 3(4):3177–3184, Oct 2018. doi: 10.1109/LRA.2018.2849882.

[3] I. Alzugaray and M. Chli. Asynchronous multi-hypothesis tracking of features with event cameras. In *2019 International Conference on 3D Vision (3DV)*, pages 269–278, Sep. 2019. doi: 10.1109/3DV.2019.00038.

[4] C. Brandli, R. Berner, M. Yang, S. C. Liu, and T. Delbruck. A 240×180 130db 3 μs latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits*, 49(10):2333–2341, Oct 2014. ISSN 0018-9200. doi: 10.1109/JSSC.2014.2342715.

[5] Xavier Clady, Jean-Matthieu Maro, Sébastien Barré, and Ryad B Benosman. A motion-based feature for event-based pattern recognition. *Frontiers in Neuroscience*, 10:594, 2017.

[6] Daniel Gehrig, Henri Rebecq, Guillermo Gallego, and Davide Scaramuzza. Asynchronous, photometric feature tracking using events and frames. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[7] Daniel Gehrig, Henri Rebecq, Guillermo Gallego, and Davide Scaramuzza. EKLT: Asynchronous photometric feature tracking using events and frames. *International Journal of Computer Vision*, 128(3):601–618, 2020. ISSN 1573-1405. doi: 10.1007/s11263-019-01209-w. URL https://doi.org/10.1007/s11263-019-01209-w.

[8] Hanme Kim, Stefan Leutenegger, and Andrew J Davison. Real-time 3D reconstruction and 6-DoF tracking with an event camera. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 349–364, 2016.

[9] B. Kueng, E. Mueggler, G. Gallego, and D. Scaramuzza. Low-latency visual odometry using event-based feature tracks. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 16–23, Oct 2016. doi: 10.1109/IROS.2016.7758089.

[10] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334, 2015.

[11] R. Li, D. Shi, Y. Zhang, K. Li, and R. Li. Fa-harris: A fast and asynchronous corner detector for event cameras. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6223–6229, Nov 2019. doi: 10.1109/IROS40897.2019.8968491.

[12] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc. URL http://dl.acm.org/citation.cfm?id=1623264.1623280.

[13] Jacques Manderscheid, Amos Sironi, Nicolas Bourdis, Davide Migliore, and Vincent Lepetit. Speed invariant time surface for learning to detect corner points with event-based cameras. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[14] Elias Mueggler, Chiara Bartolozzi, and Davide Scaramuzza. Fast event-based corner detection. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2017.

[15] Elias Mueggler, Henri Rebecq, Guillermo Gallego, Tobi Delbruck, and Davide Scaramuzza. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *International Journal of Robotics Research (IJRR)*, 36(2):142–149, 2017. doi: 10.1177/0278364917691115.

[16] T. Qin, P. Li, and S. Shen. VINS-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, Aug 2018. ISSN 1552-3098. doi: 10.1109/TRO.2018.2853729.

[17] Henri Rebecq, Timo Horstschaefer, Guillermo Gallego, and Davide Scaramuzza. EVO: A geometric approach to event-based 6-DOF parallel tracking and mapping in real time. *IEEE Robotics and Automation Letters*, 2(2):593–600, 2017.

[18] Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza. Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2017.

[19] D. Tedaldi, G. Gallego, E. Mueggler, and D. Scaramuzza. Feature detection and tracking with the dynamic and active-pixel vision sensor (davis). In *International Conference on Event-Based Control, Communication and Signal Processing (EBCCSP)*, pages 1–7, June 2016. doi: 10.1109/EBCCSP.2016.7605086.

[20] V. Vasco, A. Glover, and C. Bartolozzi. Fast event-based harris corner detection exploiting the advantages of event-driven cameras. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 4144–4149, Oct 2016. doi: 10.1109/IROS.2016.7759610.

[21] David Weikersdorfer, Raoul Hoffmann, and Jörg Conradt. Simultaneous localization and mapping for event-based vision systems. In *Computer Vision Systems*, pages 133–142, 2013. ISBN 978-3-642-39402-7.

[22] A Zhu, Nikolay Atanasov, and Kostas Daniilidis. Event-based visual inertial odometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[23] A. Z. Zhu, N. Atanasov, and K. Daniilidis. Event-based feature tracking with probabilistic data association. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4465–4470, May 2017. doi: 10.1109/ICRA.2017.7989517.