Robotics: Science and Systems 2024
Delft, Netherlands, July 15-July 19, 2024

Event-based Visual Inertial Velometer

Xiuyuan Lu[1]*, Yi Zhou[2]*, Junkai Niu[2], Sheng Zhong[2], and Shaojie Shen[1]
[1]CKS Robotic Institute, Hong Kong University of Science and Technology, Hong Kong, China
[2]School of Robotics, Hunan University, Changsha, China
Email: xluaj@connect.ust.hk; eeyzhou@hnu.edu.cn (*equal contribution)

***Abstract*—Neuromorphic event-based cameras are bio-inspired visual sensors with asynchronous pixels and extremely high temporal resolution. Such favorable properties make them an excellent choice for solving state estimation tasks under aggressive ego motion. However, failures of camera pose tracking are frequently witnessed in state-of-the-art event-based visual odometry systems when the local map cannot be updated in time. One of the biggest roadblocks for this specific field is the absence of efficient and robust methods for data association without imposing any assumption on the environment. This problem seems, however, unlikely to be addressed as in standard vision due to the motion-dependent observability of event data. Therefore, we propose a map-free design for event-based visual-inertial state estimation in this paper. Instead of estimating the position of the event camera, we find that recovering the instantaneous linear velocity is more consistent with the differential working principle of event cameras. The proposed event-based visual-inertial velometer leverages a continuous-time formulation that incrementally fuses the heterogeneous measurements from a stereo event camera and an inertial measurement unit. Experiments on both synthetic and real data demonstrate that the proposed method can recover instantaneous linear velocity in metric scale with low latency.**

Multimedia Material

Supplemental video: https://youtu.be/FCHFMGRj3So

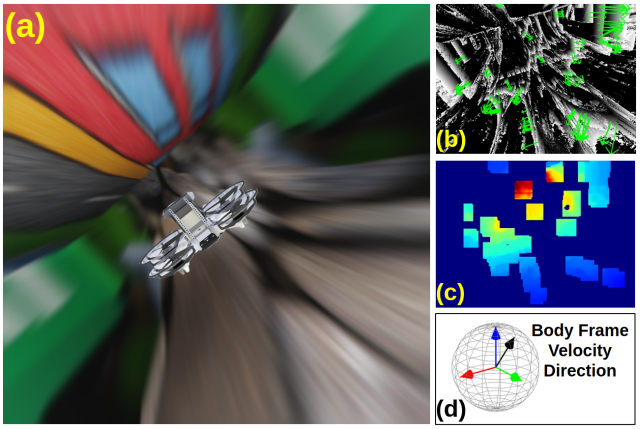


Fig. 1: The proposed system takes as input the event data from a stereo event-based camera and inertial measurements from an IMU. (a) Aggressive maneuvers of a drone through a narrow corridor. (b) Event-based normal flow estimates. (c) Corresponding depth estimates. (d) Illustration of the normalized result of instantaneous linear velocity estimation.

I. Introduction

As opposed to standard cameras that capture synchronously the absolute brightness at a fixed frame rate, event cameras are bio-inspired sensors that acquire visual information in the form of a stream of asynchronous per-pixel intensity changes (called "events"). Endowed with micro-second temporal resolution, event cameras can sense at exactly the same rate as the scene dynamics (induced by either ego motion or independent moving objects). Consequently, they do not suffer from motion blur and are qualified for state estimation tasks involving aggressive ego motion. The main challenge of building such an event-based visual state estimator (a.k.a event-based visual odometry) is to solve in real time the tracking and mapping sub-problems by exploiting photometric and geometric constraints encoded in the generative model of events.

Although event-based methods for camera pose tracking [1, 2] have been proved effective in tackling aggressive ego-motion estimation, most event-based VO systems are not always qualified. Early works using a monocular event-based camera (e.g., [3, 4]) require a very gentle motion (typically a local-loopy behavior) for the initialization of a local 3D map, based on which the camera pose can be tracked using a 3D-2D registration pipeline. To remove such a limitation on the initialization, Zhou *et al*. [5, 6] further use a stereo event-based camera to improve the efficiency and accuracy of mapping. However, tracking failure is still witnessed when the ego motion of the camera suddenly becomes violent (mainly in terms of the angular velocity). This is because the mapping sub-problem solver cannot update the local map in time.

To circumvent this issue, researchers resort to visual-inertial fusion and feature-based methods. Although the event-inertial combination seems not as complimentary as the standard-visual-inertial counterpart, it is still beneficial to fuse event data with measurements of an inertial measurement unit (IMU). First, an IMU's measurements can be used as a motion prior to propagate the estimated state. Besides, such motion prior can be leveraged to obtain a motion-compensated image of warped events (IWE), on top of which a feature-based pipeline can be simply built [7]. Obviously, the success of [7] largely relies on a sharp IWE, which is obtained via a geometric 3D-2D warping operation that requires the median depth of the local map. This warping operation, however, can easily fail if there are relatively large depth variations. To alleviate the front end's dependence on motion prior, some works [8, 9] straightforwardly establish feature association on

simple image-like representations of events (e.g., time surfaces [10]). These ad hoc solutions may work to some extent, but the success of big-baseline feature matching is hardly guaranteed. This is due to the fact that edges parallel to the epipolar line hardly trigger any events, and thus, junction-shape patterns may not be observed completely. Consequently, the appearance similarity evaluated on such naive image-like representations could be violated in a sudden variation of linear velocity.

Noticing the dependence on the first-order kinematics, [11, 12] propose a set of speed-invariant representations, which lead to constant thickness of edge patterns irrelevant to camera speed. Though we have witnessed a lot trials on event-based feature detection and tracking [12, 13, 14, 15], most of them either cannot give satisfied feature tracking results as their standard-vision counterparts [16, 17] do, or are computationally inefficient for real-time applications. It's also worth mentioning that few works [18, 19, 20] manage to establish efficiently event-to-edge association, based on which parallel tracking and mapping is achieved under aggressive motion at an ultra-frame rate. However, these solutions are only applicable in the presence of man-made geometric patterns. All these issues drive us to think about a question: Is there a more rational design of state estimation for an event-based visual-inertial system that is more consistent with the differential working principle of event cameras? To answer this question, we propose a first-order-kinematic state estimator, namely a velometer, using a stereo event-based camera and an IMU (see Fig. 1). Considering the special working principle of event cameras, we propose a continuous-time pipeline for the linear velocity estimation problem. Specifically, our approach exploits the generative model of event-based normal flow induced by the first-order kinematics of the event camera. Besides, the dynamic constraint from the IMU is also utilized. The contribution of this paper is summarized as follows:

- A novel design of state estimator for an event-based visual-inertial system, which exploits the differential nature of event cameras and recovers the first-order kinematic state (i.e., linear velocity) by fusing events and inertial measurements.
- A rigorous derivation for computing normal flow from spatial-temporal gradients of event data.
- A continuous-time framework for event-based visual-inertial fusion, which can handle asynchronous event measurements and establish data association with temporally non-aligned measurements from the accelerometer.

The rest of the paper is organized as follows. First, a literature review focusing on the topic of event-inertial state estimation is provided. We then clarify the notion and revisit several key concepts as preliminaries in Sec. III. Our method is discussed in Sec. IV, followed with the experimental evaluation in Sec. V. Finally, the conclusion is drawn in Sec. VI.

II. Related Work

Event-based state estimation intends to recover the motion parameters and a 3D map (optional) by exploiting specific constraints between ego motion and scene geometry in the generative model of event data. Due to the special working principle and sensor characteristics, event cameras also appear as a complementary sensor in some visual-inertial systems aiming to address challenging scenarios that involve high-speed maneuvers and/or high-dynamic-range (HDR) illumination. We also realize the existence of a large body of literature on the topic of event-based motion estimation, such as the optimization based methods for geometric model fitting on event data [21, 22, 23, 24, 25, 26], and learning methods that recover motion and structure [27, 28]. To be compact, our literature review focuses on two aspects: i) Fusion of event data and IMU measurements; and ii) Different choices of state variables in the problem of event-based state estimation.

A. Event-based Visual-Inertial Odometry

There is a large body of literature on this topic, most of which are built on top of a discrete-time back end that fuses events and IMU measurements in a manner of either probabilistic filtering or nonlinear optimization. These methods differ mainly in the front tend, where event-based data association is established. Early works [29, 7] detect corners on an edge map generated by naively accumulating a set of events onto the image plane. To keep tracking those corners, a two-step expectation-maximization (EM) method [29] is proposed to estimate the event-based optical flow and update the feature template. However, the computational complexity is so high that few successive reports apply this strategy in the front end. With a prior knowledge of short-term relative motion and depth information of the local map, Rebecq *et al*. [7] leverage the idea of motion compensation [22] to restore sharp IWEs, on which corner features are constantly tracked. More recently, an increasingly popular trend (e.g., [8, 9]) is to apply traditional image-based feature detection and tracking methods on a certain image-like representation (e.g., [10, 11]) of event data. Despite their relative successes, the front end of these ad-hoc methods are not as theoretically sound as their standard-vision counterparts due to the motion-dependent nature of event data. Moreover, the publicly available datasets [30, 31, 32, 33] used for evaluation are typically collected using an agent undergoing gentle motion. The superior performance of these methods lies in the improvement in terms of robustness under low-texture or HDR conditions. Few of them (except for [7]), however, have ever demonstrated the ability to solve state estimation problems under aggressive maneuvers.

A relatively smaller body of literature looks into the design of the back end that can deal with asynchronous and temporally non-aligned data. The employment of a continuous-time framework to the event-inertial fusion is firstly reported in [34], which parametrizes the camera trajectories in SE(3) with cubic B-splines. Based on the generative model of event and inertial observations, the proposed probabilistic approach seeks the optimal estimate of the posteriori of the camera pose over a time interval, given the local map, event data, and IMU measurements. The limitation of this method consists of, on one hand, the requisite of a known 3D map. On the other hand, it does not manifest the ability to address aggressive

ego motion estimation. An alternative way to implement a continuous-time framework is using Gaussian process. Wang *et al*. [35] represent a continuous-time trajectory with states defined at different timestamps of event feature tracklets. By using a white-noise-on-acceleration motion prior, the smoothness of camera trajectories is enforced considering an agent's physical plausibility. The overall trajectory is estimated using non-parametric Gaussian process regression. Still, it does not manifest effectiveness under aggressive motion.

B. First-Order Kinematics: A Better Choice?

It's only recently that researchers began to consider this question: What is the optimal choice for state variables in the problem of event-based state estimation? Most existing event-inertial fusion pipelines target absolute camera poses. However, this choice is inconsistent with the differential nature of event data. To this end, Peng *et al*. [36] propose the first event-based state estimation method on the level of first-order kinematics. The method relies on trifocal tensor geometry, which exploits the relationship between line features and first-order kinematics of the event camera. With known angular velocity, up-to-scale linear velocity can be estimated in a closed form. A successive work [37] is presented recently. It extends [36] by a direct solution of the linear velocity, and furthermore, delivers a tight event-inertial fusion back end achieving more reliable estimation of linear velocity. These methods demonstrate their feasibility in tackling the problem of aggressive ego-motion estimation. However, a limited scope of usage is clearly seen because of the dependence on straight lines, which are only available in man-made structural environments. Besides, real-time performance is not demonstrated in [37].

Considering the limitation of existing methods, we propose a novel state estimator at the level of first-order kinematics, namely a velometer, using a stereo event-based camera and an IMU. Different from feature-based methods, our approach leverages a low-level observation, namely event-based normal flow, which is directly induced by the first-order kinematics of the event camera. To be compatible with the asynchronous property of event data, we also employ a continuous-time framework using cubic B-splines to parametrize state variables. The resulting pipeline is a map-free velocity estimator that tightly fuses event-based normal flow and IMU measurements, which can estimate linear velocity under aggressive motion in real time. Furthermore, we demonstrate in Sec. V-D that the proposed linear velocity estimator allows for recovering position information with only one integration operation, leading to more accurate dead-reckoning results.

III. Problem Statement and Preliminaries

Let $\mathcal{X}^{\mathtt{A}}$ denote a state $\mathcal{X}$ being described in the coordinate system $\mathtt{A}$; the state $\mathcal{X}$ can be position $\mathbf{r}$, linear velocity $\mathbf{v}$, and angular velocity $\boldsymbol{\omega}$, etc. The relative pose between two coordinate systems (e.g., the world frame $\mathtt{W}$ and the body frame $\mathtt{B}$) is denoted by a rigid transformation $\mathbf{T}_{\mathtt{WB}} \doteq \{\mathbf{R}_{\mathtt{WB}}, \mathbf{t}_{\mathtt{WB}}\}$, which consists of a rotation $\mathbf{R}_{\mathtt{WB}}$ and a translation $\mathbf{t}_{\mathtt{WB}}$. Given raw output from a stereo event camera and an IMU, the goal is to estimate real-scale linear velocity $\mathbf{v}^{\mathtt{B}}$ of the sensor suite. We assume that the stereo event-based camera has been pre-calibrated, including the spatial-temporal calibration between the left and right event cameras, and between the left camera and the IMU. For simplification, we further assume that the left camera's coordinate system coincides with that of the IMU, which is denoted as the body coordinate system or body frame ($\mathtt{B}$).

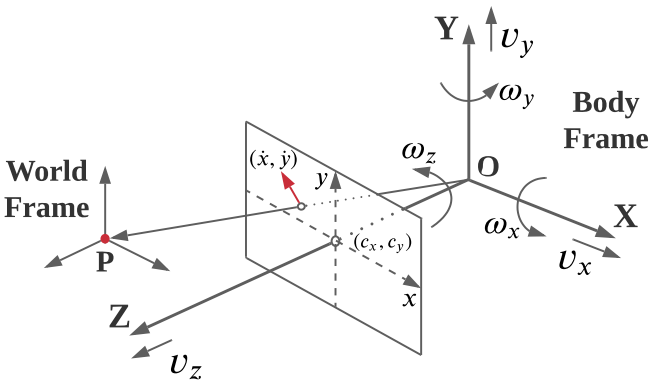


Fig. 2: Geometry and kinematics involved in the problem of motion flow. An ideal perspective camera model is used to illustrate that the 3D point $\mathbf{P}$ is projected on the image plane. The components of both linear velocity and angular velocity are marked along their corresponding axis. The resulting motion flow is denoted by the red vector $(\dot{x}, \dot{y})$.

To explain our method, we need to revisit several essential concepts as preliminaries, including *motion flow*, *normal flow*, and *event-based depth estimation*.

A. Motion Flow

Motion flow, also known as optical flow, refers to derivatives of a 2D image location w.r.t time. It describes how a 2D point would traverse on the image plane. As illustrated in Fig. 2, the motion flow of a textured 3D point $\mathbf{P}^{\mathtt{B}} = [X, Y, Z]^{\mathtt{T}}$ can be determined according to the famous equation

$$\begin{aligned}\begin{bmatrix}\dot{x}(t)\\ \dot{y}(t)\end{bmatrix} &= \begin{bmatrix}\frac{v_z x' - f v_x}{Z} + \frac{\omega_z}{f} x' y' - \omega_y (f + \frac{x'^2}{f}) + \omega_z y' \\ \frac{v_z y' - f v_y}{Z} + \omega_x (f + \frac{y'^2}{f}) - \frac{\omega_y}{f} x' y' - \omega_z x'\end{bmatrix} \\ &= \frac{1}{Z(\mathbf{x})} \mathbf{A}(\mathbf{x}) \mathbf{v}^{\mathtt{B}}(t) + \mathbf{B}(\mathbf{x}) \boldsymbol{\omega}^{\mathtt{B}}(t), \end{aligned} \tag{1}$$

where $(\dot{x}, \dot{y})$ denotes the motion flow vector at pixel $\mathbf{x} \doteq (x, y)$, $\boldsymbol{\omega}^{\mathtt{B}} = [\omega_x, \omega_y, \omega_z]$ the angular velocity, $\mathbf{v}^{\mathtt{B}} = [v_x, v_y, v_z]$ the linear velocity, and f the focal length. The relative image coordinates w.r.t the optical centre (c_x, c_y) are denoted by x' and y', namely $x' \doteq x - c_x$ and $y' \doteq y - c_y$.

The motion flow equation (Eq. 1) is first disclosed in [38]. It is clearly demonstrated that the motion flow is uniquely determined by the 3D information (Z) and the first-order kinematics of the camera ($\mathbf{v}^{\mathtt{B}}$ and $\boldsymbol{\omega}^{\mathtt{B}}$). In other words, given the angular velocity $\boldsymbol{\omega}^{\mathtt{B}}$ (e.g., obtained from an IMU's

measurements) at a time instant, the linear velocity $\mathbf{v}^{\mathsf{B}}$ can be estimated if the motion flow and depth information at no fewer than two pixels are known.

B. Normal Flow

For standard vision, the motion flow is typically determined according to the *Horn-Schunck* model [39], which assumes that the brightness of a point remains constant when observed from two spatial and temporal neighbouring perspectives. By expanding the image brightness function with first-order Taylor expansion, the brightness-constancy constraint goes as,

$$I(\mathbf{x}+d\mathbf{x}, t+dt) \approx I(\mathbf{x}, t) + \nabla_{\mathbf{x}} I(\mathbf{x}) \begin{bmatrix} dx \\ dy \end{bmatrix} + \nabla_t I(\mathbf{x}) dt. \quad (2)$$

Actually, this equation provides only one constraint such that only the partial component parallel to the image gradient direction can be determined as

$$\dot{\mathbf{x}}_n(t) = -\frac{\nabla_t I(\mathbf{x})}{\|\nabla_{\mathbf{x}} I(\mathbf{x})\|^2} \nabla_{\mathbf{x}} I(\mathbf{x}), \quad (3)$$

where $\dot{\mathbf{x}}_n$ denotes the projection of the flow $\dot{\mathbf{x}}$ on the image gradient direction $\nabla_{\mathbf{x}} I$. This is referred to as the well-known *aperture problem*, and the partial component $\dot{\mathbf{x}}_n$ is typically called *normal flow*.

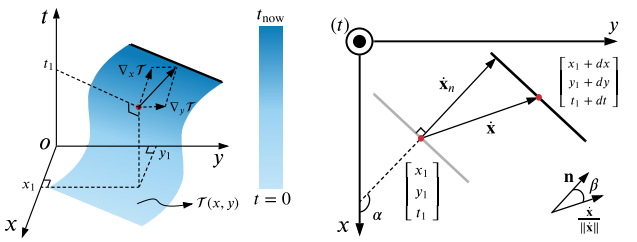
Let $\mathbf{n}$ denote the image gradient direction at $\mathbf{x}$, namely $\mathbf{n} = \frac{\nabla_{\mathbf{x}} I(\mathbf{x})}{\|\nabla_{\mathbf{x}} I(\mathbf{x})\|}$. By multiplying both sides of Eq. 1 with $\mathbf{n}^{\mathsf{T}}$, we obtain

$$\|\dot{\mathbf{x}}_n(t)\| = \frac{1}{Z(\mathbf{x})} \mathbf{n}^{\mathsf{T}} \mathbf{A}(\mathbf{x})^{\mathsf{B}} \mathbf{v}(t) + \mathbf{n}^{\mathsf{T}} \mathbf{B}(\mathbf{x})^{\mathsf{B}} \boldsymbol{\omega}(t). \quad (4)$$

Since normal flows rather than full motion flows can be estimated straightforwardly from raw events, Eq. 4 is used as a constraint in the estimation of linear velocity.

C. Event-based Depth Estimation

Existing methods of recovering depth information from event data can be classified into two categories. The first category is called the "temporal stereo" method, in which a monocular event-based camera is more often used. Temporal stereo methods assume the camera's motion to be known as a prior [40, 41]. They utilize events occurred over a temporal window to determine the 3D location of structures by searching the maximum in the disparity space image (DSI) through either voting in the discrete space [40, 41] or in a way of continuous optimization [22]. The second category is called the "instantaneous stereo" method, which typically applies a stereo event camera that have been calibrated extrinsically and synchronized temporally [42, 43, 44]. Typically, this kind of methods consists of two steps: 1) Searching corresponding events occurred in the left and right event cameras; 2) Triangulation. To enhance the ratio of true-positive matching, state-of-the-art methods [45, 46] utilize a hybrid metric to measure the similarity of two events, which jointly considers temporal coherence, epipolar constraint, and motion consistency in the spatio-temporal neighborhood. Since depth estimation is not the focus of this work, we simply apply the instantaneous stereo matching method (i.e., block matching on time surfaces [10]) implemented in [5].



(a) Temporal flow estimation as spatio-temporal planar fitting.

(b) Trigonometry in regard to motion flow $\dot{\mathbf{x}}$, normal flow $\dot{\mathbf{x}}_n$ and brightness gradient direction $\mathbf{n}$.

Fig. 3: General principles of normal flow estimation from raw event streams. (a) A time surface $\mathcal{T}(x, y)$ is spanned in the spatio-temporal domain as the edge (black) traverses. The temporal flow $\nabla_{\mathbf{x}} \mathcal{T}$ can be estimated by fitting a local plane. A color scale for the temporal values is provided aside. The darker the color, the closer the time is to the current moment. (b) The normal flow $\dot{\mathbf{x}}_n$, namely the component of the motion flow $\dot{\mathbf{x}}$ along the direction of brightness gradient $\mathbf{n}$, is parallel to the temporal flow $\nabla_{\mathbf{x}} \mathcal{T}$.

IV. Methodology

Given as input the raw events from a stereo event camera and the measurements from an IMU, the goal is to recover the instantaneous linear velocity in metric scale. In this section, we first introduce our method of event-based normal flow estimation (IV-A). Second, we present a continuous-time approach that fuses the two heterogeneous measurements by incrementally fitting a cubic B-spline in the space of linear velocity (IV-B). Finally, we discuss our initialization method (IV-C).

A. Event-based Normal Flow Estimation

To estimate visual flows from raw events, [47] proposes a local differential approach that fits local spatio-temporal planes on the time surface composed of co-active events. The time surface is defined specifically by the function $\mathcal{T}(x, y)$, which returns the timestamp by which the most recent event occurred at pixel $[x, y]^{\mathsf{T}}$. The spatio-temporal plane's gradient is calculated as the temporal derivative w.r.t the pixel coordinate, namely $\nabla_{\mathbf{x}} \mathcal{T} \doteq \left[\nabla_x \mathcal{T}, \nabla_y \mathcal{T}\right]^{\mathsf{T}}$, which reports the timestamp difference between spatially adjacent events in the direction of x and y, respectively.

Let's assume that the brightness gradient direction is orthogonal to edges. As shown in Fig. 3(a) and Fig. 3(b), the time surface $\mathcal{T}$ can be regarded as a temporal profile induced by the normal flow $\dot{\mathbf{x}}_n$, and we have

$$\begin{bmatrix} \nabla_x \mathcal{T} \\ \nabla_y \mathcal{T} \end{bmatrix} = \frac{1}{\|\dot{\mathbf{x}}_n\|} \begin{bmatrix} \cos(\alpha) \\ \sin(\alpha) \end{bmatrix}. \quad (5)$$

In fact, only normal flows could be recovered from temporal derivatives due to the aperture problem (β is unknown in Fig. 3(b)), and thus, the visual flow estimated by [47] does

not refer to the full motion flow $\dot{\mathbf{x}}$. The method in [47] is partially correct in the sense that the temporal gradient reveals the normal flow's direction. The reciprocal of the temporal gradient's components would not straightforwardly give correct normal flows. The amplitude of the normal flow can be determined using Eq. 5, and the normal flow vector is calculated as

$$\dot{\mathbf{x}}_n = \|\dot{\mathbf{x}}_n\|\mathbf{n} = \frac{1}{\sqrt{\nabla_x \mathcal{T}^2 + \nabla_y \mathcal{T}^2}}\mathbf{n}, \tag{6}$$

where $\mathbf{n} = \frac{\nabla_{\mathbf{x}}\mathcal{T}}{\|\nabla_{\mathbf{x}}\mathcal{T}\|}$ refers to the direction of the normal flow[1]. We will justify our derivation by comparing against the result of [47] in the experiment (Sec. V-C and V-D). It is noted that Valeiras *et al*. [48] have also pointed out the erroneous way of normal flow computation in [47] and provided the correct formula.

B. Continuous-Time Linear Velocity Estimation

Although the computation of event-based normal flows cannot achieve the frequency as high as the streaming rate of event data (*e.g*. several million per second), it still can be much higher than the frame rate of a standard camera. Besides, the computation can be carried out in an asynchronous manner, and thus, the resulting event-based normal flows typically do not coincide with the inertial measurements in time (*i.e*., temporally non-aligned data association). Therefore, we employ a continuous-time formulation, which could bound the size of the optimization problem while enabling data association at any given time. Specifically, we use a cubic B-spline based parametric model to represent the continuous-time linear velocity as

$$\mathbf{v}^{\mathtt{B}}(u(t)) = \sum_{i=0}^{3} B_{i,3}(u(t))\mathbf{c}_i, \tag{7}$$

where $B_{i,k}$ denotes the basis function, i the index of control points, k (=3) the order of the spline and $\mathbf{c}_i \in \mathbb{R}^3$ the corresponding control point defined in the space of linear velocity. The function $u(\cdot)$ is a normalization operator, which transfers time t to the spline's parameter domain by means of basis translation [49].

Our goal is to fit a cubic B-spline that simultaneously satisfies the following two criteria: 1) The predicted normal flows are maximally consistent with the asynchronous normal flow measurements; and 2) The first-order derivative of the spline complies maximally with the acceleration measurements. The first criterion can be simply evaluated using Eq. 4. The second criterion is evaluated, from another equivalent perspective, by calculating the difference between the predicted velocity increment and the pre-integration of acceleration in a local frame.

Let's denote the raw accelerometer and gyroscope measurements, $\tilde{\mathbf{a}}_t$ and $\tilde{\boldsymbol{\omega}}_t$, in the body frame at time t by

$$\begin{aligned} \tilde{\mathbf{a}}_t &= \mathbf{a}_t + \mathbf{b}_{a_t} + \mathtt{R}^t_{\mathtt{W}}\mathbf{g}^{\mathtt{W}} + \boldsymbol{\sigma}_a, \\ \tilde{\boldsymbol{\omega}}_t &= \boldsymbol{\omega}_t + \mathbf{b}_{\omega_t} + \boldsymbol{\sigma}_w, \end{aligned} \tag{8}$$

[1]Note that we abuse the notation by duplicating $\mathbf{n}$, because the direction of the normal flow is parallel to that of the local image gradient.

where $\mathbf{b}_a$ and $\mathbf{b}_\omega$ are the accelerometer and gyroscope biases, while $\boldsymbol{\sigma}_a$ and $\boldsymbol{\sigma}_w$ the corresponding additive noise. Let's further consider the IMU pre-integration [50] during the time interval $[t_i, t_{i+1}]$. Assuming the IMU biases are known, we integrate the inertial measurements in local frame $\mathtt{B}_i$ (the body frame at time t_i) as

$$\begin{aligned} \tilde{\boldsymbol{\beta}}^{\mathtt{B}_i}_{\mathtt{B}_{i+1}} &= \int_{t_i}^{t_{i+1}} \mathtt{R}^{\mathtt{B}_i}_t(\tilde{\mathbf{a}}_t - \mathbf{b}_{a_t} - \mathbf{n}_a)dt, \\ \tilde{\boldsymbol{\gamma}}^{\mathtt{B}_i}_{\mathtt{B}_{i+1}} &= \int_{t_i}^{t_{i+1}} \frac{1}{2}\boldsymbol{\Omega}(\tilde{\boldsymbol{\omega}}_t - \mathbf{b}_{\omega_t} - \mathbf{n}_w)\boldsymbol{\gamma}^{\mathtt{B}_i}_t dt, \end{aligned} \tag{9}$$

where $\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} -[\boldsymbol{\omega}]_\times & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^{\mathtt{T}} & 0 \end{bmatrix}$. Then the velocity and orientation can be propagated from t_i to t_{i+1} in the world frame by

$$\begin{aligned} \mathbf{v}^{\mathtt{W}}_{\mathtt{B}_{i+1}} &= \mathbf{v}^{\mathtt{W}}_{\mathtt{B}_i} - \mathbf{g}^{\mathtt{W}}\Delta t_i + \mathtt{R}^{\mathtt{B}_i}_{\mathtt{W}}\boldsymbol{\beta}^{\mathtt{B}_i}_{\mathtt{B}_{i+1}}, \\ \mathbf{q}^{\mathtt{W}}_{\mathtt{B}_{i+1}} &= \mathbf{q}^{\mathtt{W}}_{\mathtt{B}_i} \otimes \boldsymbol{\gamma}^{\mathtt{B}_i}_{\mathtt{B}_{i+1}}, \end{aligned} \tag{10}$$

where $\mathbf{q}$ refers to the orientation in the form of quaternion. Consequently, the predicted velocity increment during the time interval can be calculated as

$$\hat{\boldsymbol{\beta}}^{t_i}_{t_{i+1}} = \mathtt{R}(\tilde{\boldsymbol{\gamma}}^{t_i}_{t_{i+1}})\mathbf{v}^{\mathtt{B}_{i+1}}_{\mathtt{B}_{i+1}} + \mathbf{g}^{\mathtt{B}_i}\Delta t_i - \mathbf{v}^{\mathtt{B}_i}_{\mathtt{B}_i}. \tag{11}$$

Now let's discuss the formulation of the continuous-time state estimation problem. The full state vector is defined as:

$$\begin{aligned} \mathcal{X} &= [\mathbf{c}_0, \mathbf{c}_1, \cdots \mathbf{c}_n, \mathbf{b}_0, \mathbf{b}_1, \cdots \mathbf{b}_{n-3}], \\ \mathbf{b}_k &= [\mathbf{b}_{a_k}, \mathbf{b}_{\omega_k}], k \in [0, n-3], \end{aligned} \tag{12}$$

where n denotes the total number of control points, and $\mathbf{b}_k$ the IMU bias in the $k_{\mathtt{th}}$ B-spline segment. Note that there are $n-3$ cubic B-spline segments in the optimization. Since each segment of the B-spline is short, we simply assume the biases are constant within each segment.

Combining Eq. 4, Eq. 9 and Eq. 11, we finally create the following objective function,

$$\mathcal{X}^\star = \arg\min_{\mathcal{X}} \mathcal{C}, \tag{13}$$

where the cost is defined as

$$\begin{aligned} \mathcal{C} \doteq &\sum_{t\in\mathcal{S}} \left\| \|\dot{\mathbf{x}}_n(t)\| - \frac{\mathbf{n}^{\mathtt{T}}\mathbf{A}(\mathbf{x})\mathbf{v}^{\mathtt{B}}(t)}{Z_t(\mathbf{x})} - \mathbf{n}^{\mathtt{T}}\mathbf{B}(\mathbf{x})\boldsymbol{\omega}^{\mathtt{B}}(t) \right\|^2_{\mathbf{P}_c} \\ &+ \sum_{i}^{M} \|\tilde{\boldsymbol{\beta}}^{t_i}_{t_{i+1}} - \mathtt{R}(\tilde{\boldsymbol{\gamma}}^{t_i}_{t_{i+1}})\mathbf{v}^{\mathtt{B}_{i+1}}_{\mathtt{B}_{i+1}} - \hat{\mathbf{g}}^{\mathtt{B}_i}\Delta t_i + \mathbf{v}^{\mathtt{B}_i}_{\mathtt{B}_i}\|^2_{\mathbf{P}^i_{i+1}}. \end{aligned} \tag{14}$$

These two terms implement the above-mentioned two criteria, where $\mathcal{S}$ denotes the timestamp set of all involved normal flow estimates, and M the number of time intervals for IMU pre-integration. To obtain a maximum a posteriori estimate, we apply the Mahalanobis norm to both measurement residuals. The normal-flow covariance $\mathbf{P}_c$ is set empirically as a fixed one. The covariance of the IMU measurements $\mathbf{P}^i_{i+1}$ is derived from the dynamics of error terms of Eq. 9 [51]. Note that the orientation is not included in state vector $\mathcal{X}$. Thus, we simply propagate the orientation by integrating raw angular velocities, given the initial orientation. The gravity in the body

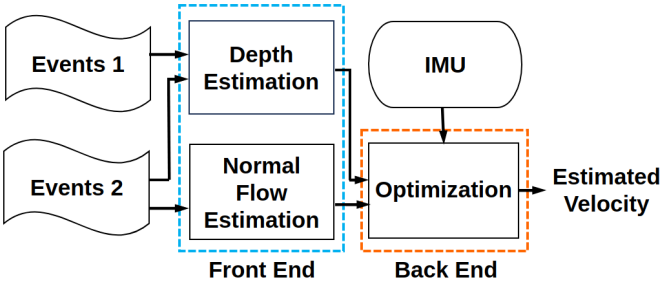
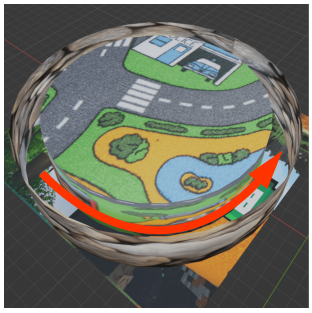
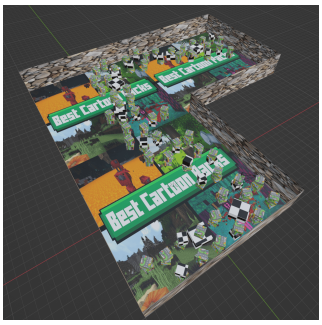


Fig. 4: Flowchart of the proposed event-based visual-inertial velometer system. The system takes as input the events from a stereo event camera and an IMU's inertial measurements, and reports the estimated linear velocity.



(a) Circular corridor.



(b) Randomly distributed boxes.

Fig. 5: Simulated scenes for synthetic data generation.

frame ($\hat{\mathbf{g}}^{\mathrm{B}_i}$) can be constantly estimated using the body-frame orientation via $\hat{\mathbf{g}}^{\mathrm{B}_i} = \mathrm{R}_{\mathrm{W}}^{\mathrm{B}_i}\,\mathbf{g}^{\mathrm{W}}$.

C. Initialization

To initialize the non-linear optimization problem (Eq. 14), we propose a linear way to coarsely determine the instantaneous linear velocity. Given as input the calculated normal flows[2], their corresponding depth information, and the corresponding measurements of instantaneous angular velocity, a linear system with as unknown linear velocity ${}^{\mathrm{B}}\mathbf{v}$ can be established based on Eq. 4:

$$\begin{bmatrix} \mathbf{n}_1^{\mathrm{T}}\mathbf{A}_1 \\ \vdots \\ \mathbf{n}_K^{\mathrm{T}}\mathbf{A}_K \end{bmatrix} {}^{\mathrm{B}}\mathbf{v} = \begin{bmatrix} Z_1(\|\dot{\mathbf{x}}_{n,1}\| - \mathbf{n}_1^{\mathrm{T}}\mathbf{B}_1\,{}^{\mathrm{B}}\boldsymbol{\omega}) \\ \vdots \\ Z_K(\|\dot{\mathbf{x}}_{n,K}\| - \mathbf{n}_K^{\mathrm{T}}\mathbf{B}_K\,{}^{\mathrm{B}}\boldsymbol{\omega}) \end{bmatrix}. \tag{15}$$

A minimal solver of Eq. 15 requires three observations, and we use RANSAC [52] for robust estimation.

V. Experiments

In this section, we first disclose the implementation details of our pipeline (Sec. V-A). Second, we introduce the datasets used in the experiments (Sec. V-B), followed with both qualitative and quantitative evaluation results and comparison against alternative solutions (Sec. V-C and Sec. V-D). Finally, we discuss the computational performance (Sec. V-E) and discuss the limitation of the work (Sec. V-F).

A. Implementation Details

The proposed event-based visual-inertial velometer system consists of two parts, as shown in Fig. 4. The front end implements the computation of event-based normal flow and corresponding depth. Particularly, the event-based normal flow is computed according to Eq. 6, given a batch of events as input. To achieve real-time performance, the spatio-temporal plane fitting operation is conducted only on a small portion of events, while the rest are culled according to the following rules:

- Events whose pixel coordinates are close (smaller than 5 pixel) to the image plane's boundary are discarded;
- Events with no more than 15 neighbouring data points in the local spatio-temporal volume defined by a 5×5 patch centred at the event's pixel coordinate are discarded;
- Events whose timestamp differs from the average timestamp of the neighbouring events by more than a threshold (5% of the event batch's duration) are discarded.

In this way, only 1% - 2% of the events are selected for subsequent normal flow estimation. We observe that obtaining 500 normal flow results approximately takes 5 ms. Note that our implementation supports controlling the size of the event batch in two ways: 1) Using a constant number of events to adapt to the event streaming rate; 2) Using a constant-length time window to ensure high-frequency execution. In our implementation, the normal flow computation is triggered by the occurrence of every 45k events. The corresponding depth information is computed as mentioned in Sec. III-C with a maximum disparity of 48 pixels and a block size of 17×17 pixels. The back end implements the continuous-time nonlinear optimization pipeline using a cubic B-spline. Specifically, a uniform rational B-spline is employed, and we set the knot interval to 0.1 s. The IMU pre-integration interval is set to 0.03 s. Ceres Solver [53] is used to solve the non-linear optimization problem (Eq. 13).

B. Datasets

To evaluate the proposed algorithm, we generate several sequences featuring aggressive maneuvers of a drone using the ESIM simulator [54]. Two synthetic scenes created are visualized in Fig. 5. The first scene is a circular corridor in which the drone undergoes a high-speed detour. The second one features a drone traveling through randomly distributed boxes. The synthetic drone is equipped rigidly with a stereo event-based camera, a stereo standard camera, and an IMU. Both of the event cameras and standard cameras share identical intrinsic and extrinsic parameters, simulating a pair of DAVIS [55] sensors. The spatial resolution of all the cameras is 346×260 pixel. The standard cameras' exposure time is set to 10 ms, with a frame rate of 30 Hz. Note that the standard cameras are only used by the comparative approaches

[2]Note that the normal flows used here are those calculated within a short time interval. In other words, their timestamps can be deemed identical.

TABLE I: Characteristics of our datasets.

Seq. Name	Scene / Motion Pattern	Linear Vel.	Angular Vel.
Synthetic_1	Circular Corridor	5.68 m/s	0.66 rad/s
Synthetic_2	Circular Corridor	11.0 m/s	1.45 rad/s
Synthetic_3	Circular Corridor	18.9 m/s	2.50 rad/s
Synthetic_4	Randomly Distributed Boxes	2.77 m/s	2.05 rad/s
Synthetic_5	Randomly Distributed Boxes	3.19 m/s	3.11 rad/s
Real_1	Horizontal spin	2.17 m/s	4.82 rad/s
Real_2	Vertical spin	4.94 m/s	13.30 rad/s
Real_3	Loopy cross shape	4.84 m/s	11.26 rad/s

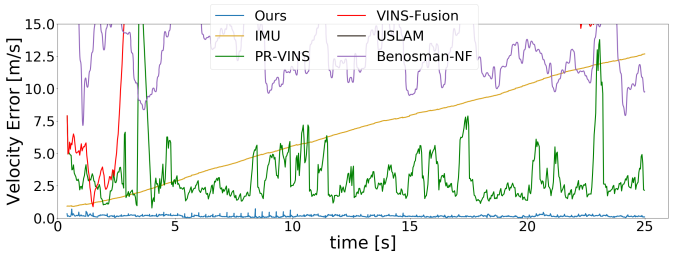
in Sec. V-C. Regarding the configuration of event generation model, we keep the setting as default. The eps value is 0.001 and the contrast threshold is 0.5. The IMU runs at 200 Hz. The standard deviations associated to the white noise of the accelerometer and gyroscope are set to $1.86 \times 10^{-2}\ m/s^2$ and $1.86 \times 10^{-3}\ rad/s$, and the standard deviations of the accelerometer and gyroscope bias are set to $4.33 \times 10^{-3}\ m/s^2$ and $2.66 \times 10^{-4}\ rad/s$, respectively.

Due to the absence of publicly available datasets[3] that capture aggressive ego motion of a stereo event camera, we collect our own real data. Similar to the data used in [7], we attach a stereo event camera to a leash and spin it. The stereo event camera is spun horizontally, vertically, and in a loopy cross shape, respectively. An example of the spinning motion is shown at the top left of Fig. 8. A pair of DAVIS-346 cameras are used in data collection, and blur of images is clearly observed (see the top middle of Fig. 8). The real data is collected in a $9\ m \times 6\ m \times 3.5\ m$ room equipped with a motion capture (MoCap) system. Some boxes are randomly distributed in the environment. The person spinning the stereo event camera stands in the center of the room. By attaching several visual markers to the stereo rig, the MoCap system can report 6-DoF poses ($\mathbf{R}_{\mathtt{WM}}$ and $\mathbf{t}_{\mathtt{WM}}$) of the marker coordinate system at 200 Hz and with sub-millimeter accuracy. Differentiating the position information with respect to time, we can obtain the linear velocity in the world coordinate system (*i.e.*, a MoCap-specific reference coordinate system). With known transformation between the markers and the left camera, we are able to obtain the velocity in the left camera's coordinate system by ${}^{\mathtt{B}}\mathbf{v} = \mathbf{R}_{\mathtt{BM}}\mathbf{R}_{\mathtt{MW}}{}^{\mathtt{W}}\mathbf{v}$. Data interpolation may be required to obtain ground truth velocity at the exact moment the estimation is conducted. The corresponding kinematic characteristics of all the data are summarized in Table. I.

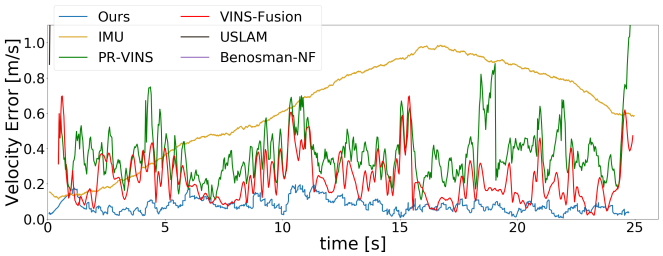
C. Evaluation on Synthetic Data

To demonstrate the effectiveness of our method, we compare it against several alternative solutions that can determine the linear velocity. The first one is the IMU integration, which only requires the initial orientation and linear velocity as a prior. The second one is the VINS-Fusion pipeline [51], which is widely applied in state estimation tasks of small unmanned aerial vehicles. To make a fair comparison, the VINS-Fusion pipeline is evaluated after being successfully initialized. The third one is a periodically re-initialized VINS-Fusion (PR-VINS) pipeline, which is to alleviate the unfairness caused by the lost camera tracking in the original VINS-Fusion. The fourth one, called Ultimate SLAM (USLAM) [57], is the only open-source visual-inertial odometry method for event cameras. It uses as input the data from a standard camera, an event camera, and an IMU. The last one (Benosman-NF) replaces the method for computing event-based normal flow in our pipeline with Benosman's method [47].

[3]Note that the aggressive data provided in [56] are captured with a monocular event camera, and thus, not applicable to our task.



(a) Absolute velocity estimation error of *Synthetic_3*.



(b) Absolute velocity estimation error of *Synthetic_5*.

Fig. 6: *Representative AVE results.* Note that the AVE of *VINS-Fusion* and *USLAM* in (a) and the AVE of *Benosman-NF* and *USLAM* in (b) are too large to be visualized inside the plot.

TABLE II: Average AVE of each synthetic sequence (m/s).

Seq. Name	IMU	VINS-Fusion	PR-VINS	USLAM	Benosman-NF	Ours
Synthetic_1	2.67	0.48	1.31	Failed	10.78	**0.09**
Synthetic_2	5.73	0.87	1.33	23.29	13.57	**0.16**
Synthetic_3	7.83	19.80	2.99	75.33	13.14	**0.19**
Synthetic_4	0.54	0.14	0.27	19.27	6.71	**0.06**
Synthetic_5	0.63	0.23	0.37	Failed	7.75	**0.08**

To quantitatively evaluate the results, we apply as evaluation metrics the Absolute Velocity Error (AVE) and Relative Velocity Error (RVE). The latter one is defined as the ratio of the AVE and the magnitude of ground truth velocity, namely

$$\text{RVE} = \frac{|\mathbf{v}_{gt} - \mathbf{v}_{est}|}{|\mathbf{v}_{gt}|} \times 100\%. \tag{16}$$

To have a close-up view of each method's performance, we select as representatives the results of two sequences that feature the most aggressive maneuvers in each scene. As illustrated in Fig. 6, obvious accumulated errors are witnessed in the velocity estimates from the IMU-integration method. VINS-Fusion performs normally on *Synthetic_5* but fails on *Synthetic_3* due to a failure of camera tracking caused by

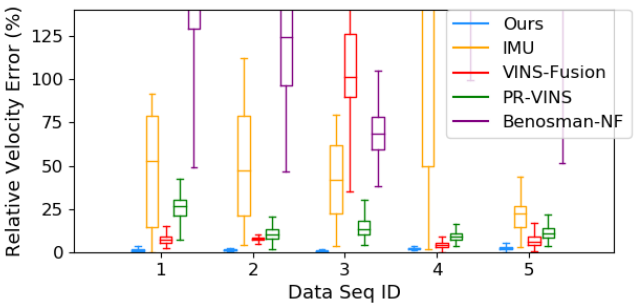


Fig. 7: RVE results on our synthetic dataset. Note that the RVE of *USLAM* is too large to be visualized inside the plot.

TABLE III: Average AVE of each real sequence (m/s).

Seq. Name	IMU	VINS-Fusion	PR-VINS	USLAM	Benosman-NF	Ours
Real_1	1.39	2.30	3.86	1.79	4.43	**0.72**
Real_2	9.21	8.43	7.62	7.48	4.89	**1.83**
Real_3	8.89	7.62	7.46	7.49	5.97	**1.65**

image blur. The AVE of PR-VINS is typically bounded to some extent, but it is still less accurate than our method. The unsatisfactory results of USLAM, according to our investigation, are due to the lack of sufficient features being detected and tracked. Note that the ego motion is notably more aggressive in the synthetic data, and thus, the USLAM can hardly achieve comparable performance as using real data (as seen in V-D), even if it has been manually initialized. Finally, Benosman-NF typically gives bad results, which, from another perspective, justifies our derivation of event-based normal flow computation.

More detailed average AVE results can be found in Table. II. Correspondingly, the RVE results are illustrated using box plots in Fig. 7. In conclusion, our method always achieves the best performance in terms of AVE and RVE.

D. Real-World Experiments

We also evaluate our method using the self-collected real data. The quantitative results are listed in Table. III, and the evaluation metric used is AVE. The best results are highlighted in bold, and the conclusion is consistent with that in the evaluation on synthetic data. Note that using the IMU alone outperforms other image-based solutions in sequence *Real_1*. This is because the motion blur effect observed in sequence *Real_1* is much more severe compared to sequences *Real_2* and *Real_3*, leading to less accurate feature matching results on images.

Additionally, we perform a qualitative evaluation by illustrating the trajectories recovered using different methods (see Fig. 8). Both VINS-Fusion and USLAM can directly return the camera trajectories. For the pure IMU-based solution and our method, dead reckoning is carried out. Note that our method leads to the most reliable trajectory, USLAM performs the second best, while the others notably drift away.

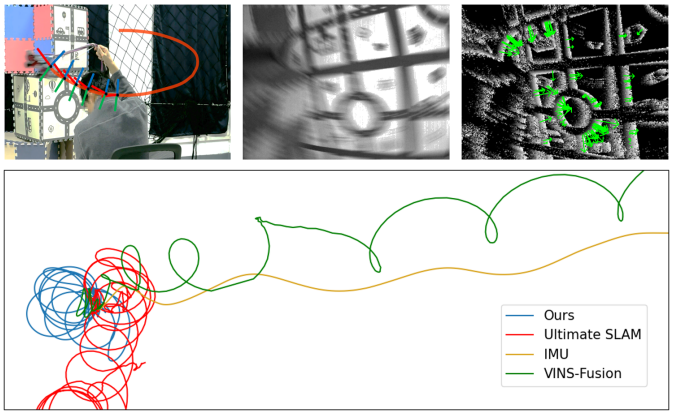


Fig. 8: Self-collected data under aggressive motion. Top left: Person spinning horizontally a stereo event camera attached to a leash. Top middle: Blurry images captured by the DAVIS sensor. Top right: A visualization of event-based normal flows (green arrows). Bottom: A bird-eye view of the dead-reckoning results of different methods.

TABLE IV: Computation time of our algorithm

Module	Data Preprocessing	Normal Flow & Depth Est.	Back-end Optimization	Overall
Avg. Time (ms)	3	10	10	23

The primary cause of drift in VINS-Fusion is the inaccurate feature tracking results obtained on blurry images. The quality of feature tracking downgrades significantly under aggressive motion. This ultimately leads to inaccurate pose estimation results that incrementally accumulate to a drift in the trajectory. By comparison, our method does not suffer from motion blur and can exploit visual cues from event-based normal flow. Compared to the dead-reckoning result of an IMU, our method requires only a single integration. We demonstrate that the accurate linear velocity recovered by our method leads to better trajectories in terms of global consistency, indicating our method a better solution to the dead-reckoning task.

E. Computational Performance

Our pipeline is implemented in C++ on ROS and all experiments are performed in real time on a desktop PC with an Intel Core 13700K CPU. The computation time of the main modules is summarized in Table. IV. The normal flow and depth estimations are performed simultaneously in two independent threads. Fed with the front-end results, the back end ultimately solves the state estimation problem using another separated thread in parallel. The whole system can output the body-frame linear velocity up to 75 Hz. Note that efficiency can be further improved by optimizing the code.

F. Discussion

1) Limitations: The proposed method has some limitations. First, the event-based normal flow is computed asynchronously

in the sense that the computation is triggered by the occurrence of a certain number of events. In other words, the event-based normal flows are computed in an asynchronous batch manner. Therefore, a batch of flow estimates may share the identical timestamps, but the time interval between two successive batches is not a constant. Although the resulting temporal non-alignment between data can be effectively tackled by the proposed continuous-time framework, we need to investigate that if a fully asynchronous computation (i.e., an event-by-event manner) will further unlock the potential of event cameras. The way depth information is computed will be adjusted accordingly. Second, the uniform cubic B-spline parametrization would be replicated by a non-uniform one to deal with sudden changes of velocity. This will be investigated as another future work.

2) Why choose linear velocity as the only unknowns?: The angular velocity can be definitely included in the state vector as unknowns and solved using our method. However, it will result in worse performance in terms of accuracy and efficiency. First, it is well known that decoupling rotation and translation estimation, if applicable, leads to better accuracy. This is because the ambiguity in visual cues is circumvented when computing the rotation and translation separately. Moreover, the estimated 3-D angular velocity from events, as reported in [21], is just as accurate as the gyroscope's measurements at its best performance. Thus, it is reasonable to use the angular velocity measurements as input for granted. Second, it is obvious that the involvement of angular velocity in the state vector will complicate the system, and thus, slows down the computation. In fact, only the accurate estimates of linear velocity are key to performing a dead-reckoning operation with less drift under high-speed maneuvers. This is why we choose linear velocity as the only unknowns.

3) Frame-based methods vs Event-based methods: It is also worth thinking about which is the optimal design for handling state estimation under aggressive maneuvers. For standard frame-based cameras, the motion blur can be mitigated by adaptively reducing the integration time. However, the adaptive exposure control complicates the system and is thus not widely applied in the community. As for event-based cameras, they are naturally qualified for high-speed perception tasks due to the high temporal resolution. Besides, the system delay induced by an event-based camera is even negligible compared to that caused by the blind period between successive exposures of a standard frame-based camera. Nonetheless, we would like to clarify that although event cameras have a very high dynamic range, event-based methods are not applicable in very dark environments.

VI. Conclusion

We present a novel strategy for fusing event data and inertial measurements for state estimation tasks under aggressive maneuvers. The proposed system is called event-based visual-inertial velometer because the state estimation problem is solved on the level of the first-order kinematics, namely linear velocity. Our front end takes as input events acquired by a pair of event cameras in stereo configuration, estimating the normal flow and corresponding depth. The event-based normal flow directly communicates the linear velocity, closely aligned with the differential working principle of event cameras. The back end is built on top of a continuous-time framework, which can handle data associations across heterogeneous and temporally non-aligned measurements. Experiments demonstrate the effectiveness of our method and endorse our method as a better choice for dead-reckoning operation. We hope our work inspires future research in event-based state estimation.

Acknowledgment

The authors would like to thank Prof. Guillermo Gallego for the discussion at the early stage of this research, and Dr. Yi Yu for proof reading. We also thank the anonymous reviewers for their suggestions, which led us to improve the paper. This work was supported by the National Key Research and Development Project of China under Grant 2023YFB4706600.

References

[1] G. Gallego, J. E. Lund, E. Mueggler, H. Rebecq, T. Delbruck, and D. Scaramuzza, "Event-based, 6-dof camera tracking from photometric depth maps," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 10, pp. 2402–2412, 2017.

[2] S. Bryner, G. Gallego, H. Rebecq, and D. Scaramuzza, "Event-based, direct camera tracking from a photometric 3d map using nonlinear optimization," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 325–331.

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

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

[5] Y. Zhou, G. Gallego, and S. Shen, "Event-based stereo visual odometry," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1433–1450, 2021.

[6] J. Niu, S. Zhong, and Y. Zhou, "Imu-aided event-based stereo visual odometry," *arXiv preprint arXiv:2405.04071*, 2024.

[7] H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization," in *British Mach. Vis. Conf. (BMVC)*, 2017.

[8] A. Hadviger, I. Cvišić, I. Marković, S. Vražić, and I. Petrović, "Feature-based event stereo visual odometry," in *2021 European Conference on Mobile Robots (ECMR)*. IEEE, 2021, pp. 1–6.

[9] P. Chen, W. Guan, and P. Lu, "Esvio: Event-based stereo visual inertial odometry," *IEEE Robotics and Automation Letters*, 2023.

[10] X. Lagorce, G. Orchard, F. Gallupi, B. E. Shi, and R. Benosman, "HOTS: A hierarchy of event-based time-surfaces for pattern recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1346–1359, Jul. 2017.

[11] J. Manderscheid, A. Sironi, N. Bourdis, D. Migliore, and V. Lepetit, "Speed invariant time surface for learning to detect corner points with event-based cameras," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.

[12] A. Glover, A. Dinale, L. D. S. Rosa, S. Bamford, and C. Bartolozzi, "luvharris: A practical corner detector for event-cameras," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, pp. 10 087–10 098, 2021.

[13] I. Alzugaray and M. Chli, "Asynchronous corner detection and tracking for event cameras in real time," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 3177–3184, Oct. 2018.

[14] ——, "ACE: An efficient asynchronous corner tracker for event cameras," in *3D Vision (3DV)*, 2018, pp. 653–661.

[15] L. Dardelet, R. Benosman, and S.-H. Ieng, "An event-by-event feature detection and tracking invariant to motion direction and velocity," *TechRxiv*, 2021. [Online]. Available: https://doi.org/10.36227/techrxiv.17013824.v1

[16] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *Int. Conf. Comput. Vis. (ICCV)*. IEEE, 2011, pp. 2564–2571.

[17] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *Int. Conf. Comput. Vis. (ICCV)*. IEEE, 2011, pp. 2548–2555.

[18] W. O. Chamorro Hernandez, J. Andrade-Cetto, and J. Solá Ortega, "High-speed event camera tracking," in *British Mach. Vis. Conf. (BMVC)*, 2020.

[19] W. Chamorro, J. Solà, and J. Andrade-Cetto, "Event-based line slam in real-time," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8146–8153, 2022.

[20] ——, "Event-imu fusion strategies for faster-than-imu estimation throughput," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2023, pp. 3975–3982.

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

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

[23] U. M. Nunes and Y. Demiris, "Entropy minimisation framework for event-based vision model estimation," in *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham: Springer International Publishing, 2020, pp. 161–176.

[24] Y. Zhou, G. Gallego, X. Lu, S. Liu, and S. Shen, "Event-based motion segmentation with spatio-temporal graph cuts," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 8, pp. 4868–4880, 2021.

[25] U. M. Nunes and Y. Demiris, "Robust event-based vision model estimation by dispersion minimisation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, pp. 9561–9573, 2022.

[26] U. M. Nunes, L. U. Perrinet, and S.-H. Ieng, "Time-to-contact map by joint estimation of up-to-scale inverse depth and global motion using a single event camera," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 23 653–23 663.

[27] A. Z. Zhu and L. Yuan, "Ev-flownet: Self-supervised optical flow estimation for event-based cameras," in *Robotics: Science and Systems*, 2018.

[28] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "Unsupervised event-based learning of optical flow, depth, and egomotion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 989–997.

[29] A. Z. Zhu, N. Atanasov, and K. Daniilidis, "Event-based visual inertial odometry," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017, pp. 5816–5824.

[30] A. Z. Zhu, D. Thakur, T. Ozaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The multivehicle stereo event camera dataset: An event camera dataset for 3D perception," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2032–2039, Jul. 2018.

[31] M. Gehrig, W. Aarents, D. Gehrig, and D. Scaramuzza, "Dsec: A stereo event camera dataset for driving scenarios," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4947–4954, 2021.

[32] L. Gao, Y. Liang, J. Yang, S. Wu, C. Wang, J. Chen, and L. Kneip, "Vector: A versatile event-centric benchmark for multi-sensor slam," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8217–8224, 2022.

[33] K. Chaney, F. Cladera, Z. Wang, A. Bisulco, M. A. Hsieh, C. Korpela, V. Kumar, C. J. Taylor, and K. Daniilidis, "M3ed: Multi-robot, multi-sensor, multi-environment event dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 4015–4022.

[34] E. Mueggler, G. Gallego, H. Rebecq, and D. Scaramuzza, "Continuous-time visual-inertial odometry for event cameras," *IEEE Trans. Robot.*, vol. 34, no. 6, pp. 1425–1440, Dec. 2018.

[35] J. Wang and J. D. Gammell, "Event-based stereo visual odometry with native temporal resolution via continuous-time gaussian process regression," *IEEE Robotics and Automation Letters*, 2023.

[36] P. Xin, X. Wanting, Y. Jiaqi, and K. Laurent, "Continuous event-line constraint for closed-form velocity initialization," in *British Machine Vision Conference (BMVC)*, 2021.

[37] W. Xu, X. Peng, and L. Kneip, "Tight fusion of events and inertial measurements for direct velocity estimation," *IEEE Transactions on Robotics*, 2023.

[38] H. C. Longuet-Higgins and K. Prazdny, "The interpre-

tation of a moving retinal image," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 208, no. 1173, pp. 385–397, 1980.

[39] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[40] H. Rebecq, G. Gallego, and D. Scaramuzza, "EMVS: Event-based multi-view stereo," in *British Mach. Vis. Conf. (BMVC)*, 2016.

[41] H. Rebecq, G. Gallego, E. Mueggler, and D. Scaramuzza, "EMVS: Event-based multi-view stereo—3D reconstruction with an event camera in real-time," *Int. J. Comput. Vis.*, vol. 126, no. 12, pp. 1394–1414, Dec. 2018.

[42] J. Kogler, M. Humenberger, and C. Sulzbachner, "Event-based stereo matching approaches for frameless address event stereo data," in *Int. Symp. Adv. Vis. Comput. (ISVC)*, 2011, pp. 674–685.

[43] P. Rogister, R. Benosman, S.-H. Ieng, P. Lichtsteiner, and T. Delbruck, "Asynchronous event-based binocular stereo matching," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 2, pp. 347–353, 2012.

[44] L. A. Camunas-Mesa, T. Serrano-Gotarredona, S. H. Ieng, R. B. Benosman, and B. Linares-Barranco, "On the use of orientation filters for 3D reconstruction in event-driven stereo vision," *Front. Neurosci.*, vol. 8, p. 48, 2014.

[45] S.-H. Ieng, J. Carneiro, M. Osswald, and R. Benosman, "Neuromorphic event-based generalized time-based stereovision," *Front. Neurosci.*, vol. 12, p. 442, 2018.

[46] Y. Zhou, G. Gallego, H. Rebecq, L. Kneip, H. Li, and D. Scaramuzza, "Semi-dense 3D reconstruction with a stereo event camera," in *Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 242–258.

[47] R. Benosman, C. Clercq, X. Lagorce, S.-H. Ieng, and C. Bartolozzi, "Event-based visual flow," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 2, pp. 407–417, 2014.

[48] D. R. Valeiras, X. Clady, S.-H. Ieng, and R. Benosman, "Event-based line fitting and segment detection using a neuromorphic visual sensor," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 4, pp. 1218–1230, 2018.

[49] C. De Boor, "On calculating with b-splines," *Journal of Approximation theory*, vol. 6, no. 1, pp. 50–62, 1972.

[50] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," in *Robotics: Science and Systems (RSS)*, 2015.

[51] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, 2018.

[52] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

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

[54] H. Rebecq, D. Gehrig, and D. Scaramuzza, "ESIM: an open event camera simulator," in *Conf. on Robotics Learning (CoRL)*, 2018.

[55] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240x180 130dB 3μs latency global shutter spatiotemporal vision sensor," *IEEE J. Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, 2014.

[56] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM," *Int. J. Robot. Research*, vol. 36, no. 2, pp. 142–149, 2017.

[57] A. R. Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate slam? combining events, images, and imu for robust visual slam in hdr and high-speed scenarios," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 994–1001, 2018.