sensors

MDPI

Article

ESVIO: Event-Based Stereo Visual-Inertial Odometry

Zhe Liu [1], Dianxi Shi [2,*], Ruihao Li [2,*] and Shaowu Yang [1]

[1] College of Computer, National University of Defense Technology, Changsha 410005, China
[2] Artificial Intelligence Research Center (AIRC), Defense Innovation Institute, Beijing 100166, China
* Correspondence: dxshi@nudt.edu.cn (D.S.); liruihao2008@gmail.com (R.L.)

Abstract: The emerging event cameras are bio-inspired sensors that can output pixel-level brightness changes at extremely high rates, and event-based visual-inertial odometry (VIO) is widely studied and used in autonomous robots. In this paper, we propose an event-based stereo VIO system, namely ESVIO. Firstly, we present a novel direct event-based VIO method, which fuses events' depth, Time-Surface images, and pre-integrated inertial measurement to estimate the camera motion and inertial measurement unit (IMU) biases in a sliding window non-linear optimization framework, effectively improving the state estimation accuracy and robustness. Secondly, we design an event-inertia semi-joint initialization method, through two steps of event-only initialization and event-inertia initial optimization, to rapidly and accurately solve the initialization parameters of the VIO system, thereby further improving the state estimation accuracy. Based on these two methods, we implement the ESVIO system and evaluate the effectiveness and robustness of ESVIO on various public datasets. The experimental results show that ESVIO achieves good performance in both accuracy and robustness when compared with other state-of-the-art event-based VIO and stereo visual odometry (VO) systems, and, at the same time, with no compromise to real-time performance.

Keywords: stereo visual-inertial odometry; event camera; sensor fusion; state estimation; event-inertial initialization

Citation: Liu, Z.; Shi, D.; Li, R.; Yang, S. ESVIO: Event-Based Stereo Visual-Inertial Odometry. *Sensors* **2023**, *23*, 1998. https://doi.org/10.3390/s23041998

Received: 15 December 2022
Revised: 30 January 2023
Accepted: 8 February 2023
Published: 10 February 2023

1. Introduction

Visual odometry (VO) is an emerging and hot research topic in the fields of robotics and computer vision [1], which aims to make the real-time estimation of camera poses and motion trajectory using only visual sensors (such as monocular camera [2], stereo camera [3], panoramic camera [4], depth camera [5], etc.). However, due to the limitation of imaging modalities, standard camera-based VO methods are not robust enough in some challenging scenarios, such as fast-moving, high dynamic range (HDR), rapid illumination changing, etc. In recent years, a new type of bio-inspired vision sensor, namely event camera, has gradually attracted the attention of researchers. It works differently from the standard camera and has independent pixels which can capture brightness changes individually and output this information as "event" asynchronously [6]. Compared to standard cameras, event cameras have the higher temporal resolution, wider dynamic range, lower latency, lower power consumption, and bandwidth [7,8]. With these advantages, researchers have developed a series of event-based VO methods [8–12] to provide correct camera state estimation in the challenging scenarios, as mentioned above. However, due to the hardware limitation of event cameras (such as noise and dynamic effects, low spatial resolution, etc.) and related research is still in the exploration stage, the event-based VO still faces the problems of low accuracy and low robustness in some scenarios.

In the traditional VO field, introducing a low-cost inertial measurement unit (IMU) to improve the accuracy and robustness of camera state estimation is a high-efficiency solution, which is called visual-inertial odometry (VIO) and has been a research highlight in the past decade [13–18]. For the event-based VO, integrating inertial measurement can also assist the

system in estimating motion and recovery scale. At present, a few works have explored event-based VIO methods and achieved accurate state estimation results, such as [19–24]. However, two common challenges exist in the event-based VIO field: (1) how to process the novel event data and fuse them with IMU measurement to estimate camera motion. Most of the existing event-based VIO methods (except for [22]) convert event stream to event image frames and execute feature-based VIO pipelines based on these frames. These methods will extract and track specific visual features from consecutive event image frames. Then, they will minimize feature reprojection errors and IMU propagation errors to estimate camera motion. However, due to the working principle and hardware limitations of event cameras, most event image frames have the disadvantages of high noise and fewer texture details, which is not conducive to visual feature extraction and tracking. (2) How to solve initialization parameters by fusing event data and IMU measurement in the initial stage of the VIO system. Visual-inertial initialization is crucial for VIO, which can help the system obtain parameters required for state estimation, including scale, gravity direction, IMU bias, etc. However, for existing event-based VIO systems, there are no initialization methods that can effectively fuse event data and IMU measurement together to estimate initialization parameters.

To this end, firstly, for the challenge (1), we propose a novel direct event-based VIO method for better fusion of event data and IMU measurement. This VIO method can directly estimate the state of the system based on event inverse depth frames, Time-Surface (TS) images, and pre-integrated inertial measurement in a sliding window non-linear optimization framework without feature tracking. Secondly, for challenge (2), we propose a semi-joint event-inertial initialization method to improve the performance of event-based VIO state estimation. This initialization method can obtain an up-to-scale camera trajectory based on the event-only initialization (since the trajectory estimated by event-based stereo VO methods, such as ESVO [8], always has a scale error, so the trajectory here is deemed up-to-scale) and solve initialization parameters using the event-inertial initial optimization. Thirdly, referring to ESVO [8], we implement an event-based stereo VIO system based on the above two proposed methods, namely ESVIO. Finally, we evaluate ESVIO on the public datasets and the experimental results show that ESVIO achieves better performance in both accuracy and robustness in different scenarios. To conclude, our contributions can be summarized as below:

- We present a direct event-based stereo VIO method for the first time, which uses the sliding window tightly coupled optimization method to directly estimate the optimal state of the system based on the event and IMU data without feature tracking.
- We propose a semi-joint event-inertial initialization method to estimate initialization parameters, including scale, gravity direction, initial velocities, accelerometer, and gyroscope biases, in two steps (event-only initialization and event-inertial initial optimization).
- We implement the event-based stereo VIO system, ESVIO, in C++ and evaluate it on four public datasets [25–28]. The results demonstrate that our system achieves good accuracy and robust performance when compared with the state-of-the-art, and, at the same time, with no compromise to real-time performance.

The rest of the paper is organized as follows. Section 2 gives a brief review of related works. The system overview of ESVIO is presented in Section 3, followed by the preliminaries in Section 4. The proposed event-inertial initialization method is introduced in Section 5, and the detailed direct event-based VIO pipeline for state estimation is given in Section 6. Section 7 provides experimental evaluations. Finally, our conclusion is drawn in Section 8.

2. Related Work

2.1. Visual-Inertial Odometry Methods

Considering different sensor fusion methods, VIO methods can be divided into loosely- and tightly coupled methods [29]. For the loosely-coupled method, firstly, it estimates the state of the system based on visual information and IMU measurement separately in two independent estimators, and, then, it solves the system state based on these two state estima-

tion results. Loosely-coupled methods are common in the early stages of VIO development, generally through an Extended Kalman Filter (EKF) to achieve sensor fusion and the final state estimation, such as [30,31]. For the tightly coupled method, it directly estimates the system state based on visual information and IMU measurement together in one estimator. Compared with the loosely coupled method, the tightly coupled method needs to consider the interaction between different sensor data, and the algorithm implementation is more complicated, but it will obtain more accurate state estimation results. Until now, tightly coupled methods mainly achieve sensor fusion and state estimation through filter-based estimation methods (e.g., EKF) [14,32,33] or optimization-based estimation methods (e.g., graph optimization) [13,15,17,34–37].

Considering visual measurement processing, VIO methods can also be divided into two different paradigms: feature-based VIO methods and direct VIO methods. For the feature-based method, firstly, it extracts and tracks specific visual features from consecutive camera frames through feature descriptors. Then, it will minimize feature reprojection errors and IMU propagation errors to estimate the system state. At present, most of the existing VIO methods are feature-based methods due to their robustness and maturity, such as [13,15,33,38–40]. Unlike the feature-based method, the direct VIO method directly uses the intensity value of pixels to construct the photometric error between two camera frames, and then estimates the system state by minimizing photometric errors and IMU propagation errors without the procedure of feature extraction and tracking. Compared with feature-based methods, direct VIO methods have faster processing speed and better performance in textureless scenarios [41], such as [16,18,42,43]. However, direct methods typically require a higher frame rate than feature-based methods to maintain good performance [44].

2.2. Event-Based Visual Odometry and SLAM

For event-based VO/SLAM methods, they can be divided into monocular ones and stereo ones. Mueggler et al. [9] presented an onboard event-based monocular system for 6-Degrees of Freedom (DoF) localization with high speed and fast rotation. Kim et al., [10] can perform real-time 3D reconstruction, 6-DoF localization, and scene intensity estimation by EKF based on a monocular event camera. Kueng et al. [45] implemented a monocular VO system by event-based feature detection. Rebecq et al. [11] proposed EVO that is implemented by an image alignment tracker via semi-dense mapping. Bryner et al. [12] introduced a maximum-likelihood framework to estimate the camera motion. Nguyen et al. [46] used the Stacked Spatial LSTM Network to estimate camera poses with event images as input. However, monocular methods always face the scale consistency problem, and stereo methods perform better in this aspect thanks to the known depth information.

For event-based stereo VO/SLAM methods, Zhou et al. [8] proposed a stereo event-based VO system called ESVO. The system achieved semi-dense 3D reconstruction [25] and 3D–2D registration tracking [47] via inverse depth map and TS. Jiao et al. [26] analyzed and compared the performance of ESVO under different event representation methods. Compared with monocular methods, stereo ones allow for more accurate and robust state estimation. However, due to the hardware limitation of the event camera, it is still difficult to obtain accurate state estimation results for VO/SLAM methods based on the event camera only in some scenes [8]. Therefore, fusing more sensor information, such as IMU measurement, is a solution to improve the performance of event-based VO/SLAM methods.

2.3. Event-Based Visual-Inertial Odometry

Until now, there are not many VIO methods based on the event camera. Zhu et al. [19] presented the event-based VIO system, namely EVIO, to provide accurate metric tracking of a camera's full 6-DoF pose. EVIO uses the sensor state and event information to track visual features within the event frame over time and fuses visual feature tracks with the IMU measurement to update the system state employing an EKF with a structureless vision model. Rebecq et al. [20] also proposed a feature-based method, but fused event and inertial measurements using a keyframe-based non-linear optimization

framework to estimate camera motion. Based on [20], Vidal et al. [21] implemented the first VIO state estimation method that fuses events, standard frames, and inertial measurements in a tightly coupled manner to achieve accurate and robust state estimation. Mueggler et al. [22] proposed a VIO method with an event camera under the continuous-time framework which allows direct integration of the asynchronous events with microsecond accuracy and inertial measurements at high frequency. Gentil et al. [23] presented an event-based VIO method which uses line features for the first time and constructs different line feature constraints to assist in state estimation. Chen et al. [24] proposed an event-based stereo VIO pipeline with sliding windows optimization, and further extended it to include standard frames, which tightly integrated stereo event streams, stereo image frames, and IMU together, to achieve accurate feature-based state estimation.

However, most of the above event-based VIO methods (except for [24]) are designed for monocular systems, and most of them (except for [22]) belong to the category of the feature-based VIO method. Compared with the standard camera frame, the event image frame constructed from raw event data has the disadvantages of low resolution, high noise, and fewer texture details due to the working principle and hardware limitations of event cameras, which makes the event-based feature extraction and tracking not as accurate and stable as needed [44]. Therefore, our proposed ESVIO can obtain events' depth by stereo constraint and adopt direct VIO method to implement the fusion of event data (event depth and event image frame) and inertial measurement without feature tracking.

2.4. Visual-Inertial Initialization Methods

To start a VIO system, some parameters need to be estimated in an initialization process, including scale, gravity direction, initial velocity, accelerometer and gyroscope biases, etc. A fast and accurate initialization is critical for the subsequent visual-inertial state estimation. Based on the classification of initialization in [48], we divide visual-inertial initialization methods into three categories: joint, disjoint, and semi-joint initialization.

Joint initialization is similar to the tightly-couple method, which directly fuses raw visual and inertial measurements and estimates all initialization parameters in one step. The joint initialization method was first proposed by Martinelli [49], who presented a closed-form solution to jointly solve feature depth (scale), gravity, accelerometer bias, and initial velocity by linear least squares. Subsequent joint initialization methods are basically extended based on this method, including [50–52]. The advantage of joint initialization is that the interaction between the parameters is fully considered, and the disadvantage is that the real-time performance and the convergence of the solution process are not stable.

Disjoint initialization is similar to the loosely coupled method, which processes visual and inertial information separately, and solves initialization parameters step by step. Disjoint initialization is based on the assumption that the up-to-scale camera trajectory can be estimated very accurately from pure monocular vision, and then use this trajectory to estimate the initialization parameters [48]. Disjoint visual-inertial initialization was pioneered by Mur-Artal et al. in ORB-SLAM-VI [35] and later adopted by Qin et al. in VINS-Mono [15,53]. All subsequent disjoint initialization methods are improvements to these two methods, such as [54–56]. Compared to joint initialization, disjoint initialization has faster solution speed and better robustness. However, due to insufficient consideration of the interaction between initialization parameters and possible visual measurement noise, disjoint initialization does not perform well in estimation accuracy.

Semi-joint initialization can be seen as a compromise between joint initialization and non-joint initialization. It first solves partial parameters based on the visual measurement and then estimates all initialization parameters in one step. This idea was first implemented by Yang et al. [34], but matured by Campos et al. in ORB-SLAM-3 [39,48]. In ORB-SLAM-3, visual-inertial initialization was formulated as two maximum a posteriori (MAP) estimations: visual-only MAP estimation responsible for obtaining up-to-scale camera poses and inertial-only MAP estimation responsible for solving all initialization

parameters. Semi-joint initialization performs well in terms of real-time performance and accuracy performance.

In this paper, we design a semi-joint event-inertial initialization method for event-based VIO to achieve better VIO state estimation results.

3. System Overview of ESVIO

The system overview of ESVIO is shown in Figure 1. The proposed ESVIO can be divided into four components: the measurements processing unit, the initialization unit, the depth estimation unit, and the VIO state estimation unit.

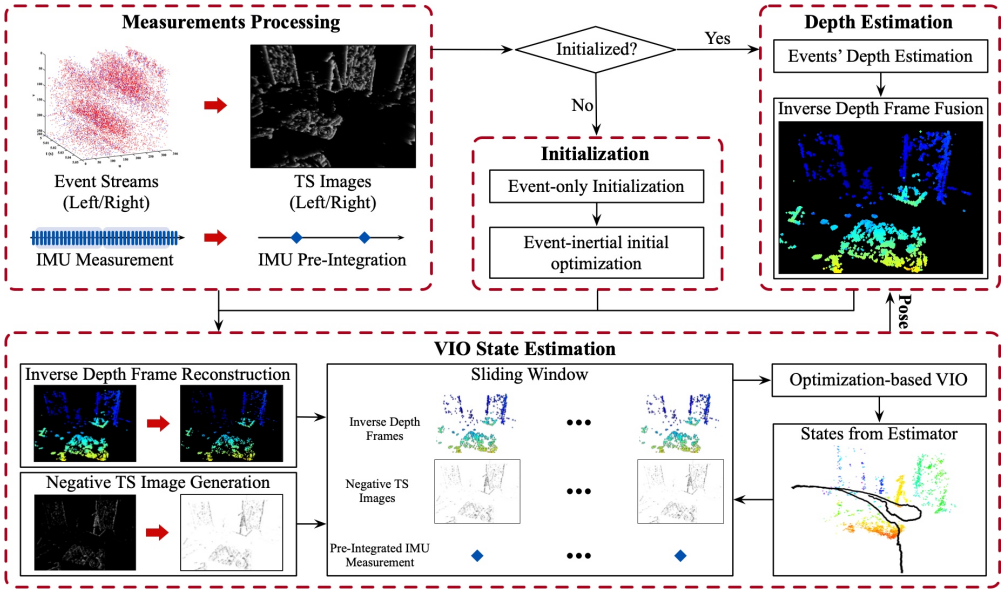


Figure 1. System overview of ESVIO. The measurements of stereo event cameras and IMU will be handled by the Measurements Processing Unit, and then the processed data will be sent to the Initialization Unit for event-only initialization and event-inertial initial optimization. After initialization, the Depth Estimation Unit will solve events' depth and fuse the inverse depth frame. Finally, the VIO State Estimation Unit will estimate the camera's 6-DoF pose, velocity, and IMU biases based on inverse depth frames, negative TS images, and pre-integrated IMU measurement.

The measurements processing unit is responsible for processing sensor measurements, including event processing and IMU measurement processing. For event processing, ESVIO converts event streams to TS images for subsequent depth estimation and state estimation (Section 4.2). For IMU measurements processing, ESVIO adopts the IMU pre-integration algorithm to handle IMU measurements which can efficiently reduce the complexity and calculation time cost of the IMU propagation model (Section 4.3).

The initialization unit is responsible for solving initialization parameters to improve the performance of VIO state estimation, including event-only initialization and event-inertial initial optimization. The first part will estimate an up-to-scale camera trajectory based on event data (Section 5.1). Additionally, the second part will solve initialization parameters by fusing event and IMU data in a sliding window optimization framework (Section 5.2).

The depth estimation unit is responsible for inverse depth frame construction. This unit estimates the single event's depth by stereo event matching and fuses these asynchronous depth points to construct an inverse depth frame (Section 4.4).

The VIO state estimation unit is responsible for camera state estimation by fusion of event data and inertial measurement. Firstly, it selects a depth points subset to reconstruct the inverse depth frame (Section 6.1). Secondly, it generates negative TS images for state estimation (Section 6.2). Thirdly, it estimates the state of the VIO system (including the camera's 6-DoF pose, velocity, and IMU biases) using a novel tightly coupled direct VIO

pipeline in a sliding window non-linear optimization framework based on inverse depth frames, negative TS images and pre-integrated inertial measurement (Section 6.3).

4. Preliminaries

4.1. Notations

We denote ${}^{w}(\cdot)$ as the world frame. ${}^{b}(\cdot)$ is the body (IMU) frame. ${}^{c}(\cdot)$ is the camera frame, which is equivalent to the inverse depth frame used in ESVIO. In particular, we denote ${}^{cp}(\cdot)$ and ${}^{cp(r)}(\cdot)$ as the left and right camera image plane of the camera frame ${}^{c}(\cdot)$. Note that the camera frames used in the paper are all left camera frames. We use both rotation matrix $\boldsymbol{R}$ and quaternion $\boldsymbol{q}$ to represent rotation and use translation vector $\boldsymbol{p}$ to represent translation. We consider $\boldsymbol{x}_n^m$ or $\boldsymbol{X}_n^m$ as the vector $\boldsymbol{x}$ or matrix $\boldsymbol{X}$ from frame n to frame m. m_t is the frame m with timestamp t. We also use m_i to represent the frame m while taking the i-th inverse depth frame.

4.2. Event Presentation Based on Time-Surface

The output of the event camera is a stream of asynchronous events. We use $e_k = (u_k, v_k, t_k, p_k)$ to represent the k-th event in event stream, containing the pixel coordinate $\boldsymbol{x}_k(u_k, v_k) \in \mathbb{R}^2$, timestamp t_k and polarity p_k. Referring to ESVO [8], ESVIO adopts an alternative event representation method called Time-Surface (TS). TS is a two-dimensional (2D) camera size map where each pixel stores the time information. At time t, it can be defined by

$$I(\boldsymbol{x}, t) \doteq exp(-\frac{t - t_{last}({}^{cp_t}\boldsymbol{x})}{\varphi}), \tag{1}$$

where t_{last} means the timestamp of the latest event at the pixel ${}^{cp_t}\boldsymbol{x}({}^{cp_t}u, {}^{cp_t}v) \in \mathbb{R}^2$, and φ is the constant decay rate parameter (e.g., 30 ms).

Based on Equation (1), we can convert a stream of events to a TS image whose pixel's intensity is the value of the corresponding pixel in the TS map re-scaled from [0, 1] to the range [0, 255].

TS can be considered as a kind of *fixed time window* event processing method. The length of the time window is determined by the TS image generation frequency and the average brightness (intensity) of the TS image is determined by φ in Equation (1). However, we find that the setting of φ with a fixed value cannot handle different camera motions (different occurrence frequencies of events), especially at a low speed. When the camera moves slowly (low event occurrence frequency), there are few pixels in the TS image that can provide enough intensity information for subsequent depth estimation and state estimation. In order to tackle this problem, we have made improvements to the TS method used in ESVO to cope with slow camera motion. Based on Equation (1), we count the number of events across the image plane in the latest time window as n, and define the improved TS value:

$$I(\boldsymbol{x}, t) \doteq \begin{cases} exp(-\frac{t - t_{last}({}^{cp_t}\boldsymbol{x})}{\varphi}), & n \geq N_\tau \\ exp(-\frac{t - t_{last}({}^{cp_t}\boldsymbol{x})}{\varphi_{N_\tau}}), & n < N_\tau \end{cases} \tag{2}$$

where N_τ depends on the number of events for depth estimation and we set N_τ as N in Section 4.4, φ_{N_τ} is defined by

$$\varphi_{N_\tau} = \frac{t - t_{N_\tau}}{t_w} \cdot \varphi, \tag{3}$$

where t_{N_τ} is the timestamp of the latest N_τ-th events across the image plane, and t_w is the length of the time window of TS.

The improved TS also needs to be rescaled to the range [0, 255] to create the corresponding TS image. From Equations (2) and (3), we can find that: when events occur at a normal or high frequency (normal or fast camera motion, $n \geq N_\tau$), the improved TS will work as

the TS used in ESVO; when events occur at a low frequency (slow camera motion, $n < N_\tau$), there are at least N_τ events which can provide enough intensity information in the TS image (when $t_{last}({}^{cp_t}\boldsymbol{x}) = t_{N_\tau}$, the intensity of $\boldsymbol{x}$'s corresponding pixel is $255 \times exp(-t_w/\varphi)$) for depth and state estimation. Figure 2 shows the processes of different TS methods under slow camera motion in an office scene.

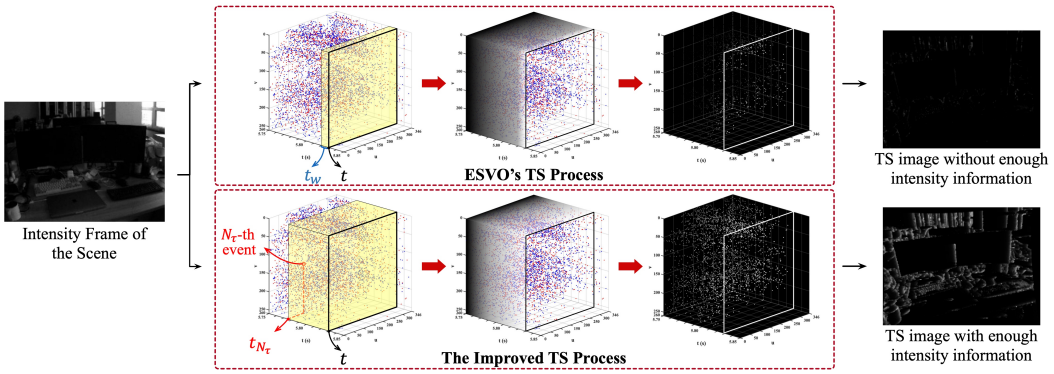


Figure 2. Different TS methods under slow camera motion in an office scene. When the camera is in slow motion, the improved TS will change the decay rate parameter based on t_{N_τ}, while the ESVO's will not. The left column is the intensity frames from the DAVIS346 event camera of the scene. The middle three columns are the TS processes: the first column shows the determination of t_{N_τ} and the events which can provide enough intensity information for TS images (in the yellow boxes); the second and third columns show how events are converted into pixels of different intensities under fixed φ (ESVO's TS) and unfixed φ_{N_τ} (improved TS). The right column is the output TS images.

4.3. IMU Pre-Integration

IMU pre-integration [57] is a computationally efficient alternative to the standard inertial measurement integration. It performs the discrete integration of the inertial measurement dynamics in a local frame of reference, thus preventing the need to reintegrate the state dynamics at each optimization step [41]. In our ESVIO, we use IMU pre-integration methods proposed in [15] to process IMU measurement.

A low-cost IMU, generally including a 3-axis accelerometer and a 3-axis gyroscope, can measure the acceleration $\boldsymbol{a}$ and the angular velocity $\boldsymbol{\omega}$ of the IMU. The raw measurements from IMU, $\hat{\boldsymbol{a}}$ and $\hat{\boldsymbol{\omega}}$, can be represented by

$$\begin{aligned} {}^b\hat{\boldsymbol{a}} &= {}^b\boldsymbol{a} + \boldsymbol{R}_w^b\,{}^w\boldsymbol{g} + {}^b\boldsymbol{b}_a + {}^b\boldsymbol{n}_a \\ {}^b\hat{\boldsymbol{\omega}} &= {}^b\boldsymbol{\omega} + {}^b\boldsymbol{b}_g + {}^b\boldsymbol{n}_g, \end{aligned} \tag{4}$$

where ${}^b\boldsymbol{b}_a$, ${}^b\boldsymbol{b}_g$ and ${}^b\boldsymbol{n}_a$, ${}^b\boldsymbol{n}_g$ are the biases and additive noises from the accelerometer and the gyroscope in the IMU frame, respectively, ${}^w\boldsymbol{g}$ is the gravity vector in the world frame. Referring to [15], we assume that the additive noises ${}^b\boldsymbol{n}_a$ and ${}^b\boldsymbol{n}_g$ are Gaussian, ${}^b\boldsymbol{n}_a \sim \mathcal{N}(0, {\sigma_a}^2)$, ${}^b\boldsymbol{n}_g \sim \mathcal{N}(0, {\sigma_g}^2)$. Acceleration bias and gyroscope bias are modeled as random walk, whose derivatives are Gaussian, ${}^b\dot{\boldsymbol{b}}_a = {}^b\boldsymbol{n}_{b_a} \sim \mathcal{N}(0, {\sigma_{b_a}}^2)$, ${}^b\dot{\boldsymbol{b}}_g = {}^b\boldsymbol{n}_{b_g} \sim \mathcal{N}(0, {\sigma_{b_g}}^2)$.

Given two IMU frames b_i and b_{i+1} that correspond to two consecutive inverse depth frames, position, velocity, and rotation in the IMU frame b_i can be computed by integrating the kinematics equation of IMU-driven algorithm within the duration $t \in [t_i, t_{i+1}]$:

$$\begin{aligned} \boldsymbol{R}_w^{b_i}\boldsymbol{p}_{b_{i+1}}^w &= \boldsymbol{R}_w^{b_i}(\boldsymbol{p}_{b_i}^w + {}^w\boldsymbol{v}_i\Delta t - \frac{1}{2}{}^w\boldsymbol{g}\Delta t^2) + \boldsymbol{\alpha}_{b_{i+1}}^{b_i} \\ \boldsymbol{R}_w^{b_i}\,{}^w\boldsymbol{v}_{i+1} &= \boldsymbol{R}_w^{b_i}({}^w\boldsymbol{v}_i - {}^w\boldsymbol{g}\Delta t) + \boldsymbol{\beta}_{b_{i+1}}^{b_i} \\ \boldsymbol{q}_w^{b_i} \otimes \boldsymbol{q}_{b_{i+1}}^w &= \boldsymbol{\gamma}_{b_{i+1}}^{b_i}, \end{aligned} \tag{5}$$

where

$$\begin{aligned}\boldsymbol{\alpha}_{b_{i+1}}^{b_i} &= \iint_{t\in[t_i,t_{i+1}]} (\boldsymbol{R}_{b_t}^{b_i}({}^{b_t}\hat{\boldsymbol{a}} - {}^{b_t}\boldsymbol{b}_a - {}^{b_t}\boldsymbol{n}_a))dt^2 \\ \boldsymbol{\beta}_{b_{i+1}}^{b_i} &= \int_{t\in[t_i,t_{i+1}]} (\boldsymbol{R}_{b_t}^{b_i}({}^{b_t}\hat{\boldsymbol{a}} - {}^{b_t}\boldsymbol{b}_a - {}^{b_t}\boldsymbol{n}_a))dt \\ \boldsymbol{\gamma}_{b_{i+1}}^{b_i} &= \int_{t\in[t_i,t_{i+1}]} \frac{1}{2}\boldsymbol{\Omega}({}^{b_t}\hat{\boldsymbol{\omega}} - {}^{b_t}\boldsymbol{b}_g - {}^{b_t}\boldsymbol{n}_g)\boldsymbol{\gamma}_{b_t}^{b_i}dt,\end{aligned} \tag{6}$$

where

$$\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} -\lfloor\boldsymbol{\omega}\rfloor_\times & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^\top & 0 \end{bmatrix}, \tag{7}$$

Δt is the duration between time interval $t \in [t_i, t_{i+1}]$, $\otimes$ is the quaternion multiplication, $\lfloor\cdot\rfloor_\times$ is the skew-symmetric matrix of a 3D vector.

$\boldsymbol{\alpha}_{b_{i+1}}^{b_i}$, $\boldsymbol{\beta}_{b_{i+1}}^{b_i}$ and $\boldsymbol{\gamma}_{b_{i+1}}^{b_i}$ in Equation (6) are called pre-integration terms. Pre-integration terms can be understood as the relative motion from IMU frame b_i to b_{i+1} without the change of the state of IMU frame b_i.

Based on IMU pre-integration terms in Equation (6), we can establish the linear Gaussian error state recursion equation of IMU measurements, and derive the equation corresponding covariance matrix and Jacobian matrix in continuous-time and discrete-time for state estimation referring to [15]. Finally, we write down the IMU measurement model which will be used to construct the IMU residual in the initialization method (Section 5) and the VIO method (Section 6):

$$\begin{bmatrix} \hat{\boldsymbol{\alpha}}_{b_{i+1}}^{b_i} \\ \hat{\boldsymbol{\beta}}_{b_{i+1}}^{b_i} \\ \hat{\boldsymbol{\gamma}}_{b_{i+1}}^{b_i} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_w^{b_i}(\boldsymbol{p}_{b_{i+1}}^w - \boldsymbol{p}_{b_i}^w + \frac{1}{2}{}^w\boldsymbol{g}\Delta t^2 - {}^w\boldsymbol{v}_i\Delta t) \\ \boldsymbol{R}_w^{b_i}({}^w\boldsymbol{v}_{i+1} + {}^w\boldsymbol{g}\Delta t - {}^w\boldsymbol{v}_i) \\ (\boldsymbol{q}_{b_i}^w)^{-1} \otimes \boldsymbol{q}_{b_{i+1}}^w \\ {}^{b_{i+1}}\boldsymbol{b}_a - {}^{b_i}\boldsymbol{b}_a \\ {}^{b_{i+1}}\boldsymbol{b}_g - {}^{b_i}\boldsymbol{b}_g \end{bmatrix}. \tag{8}$$

4.4. Depth Estimation Based on Time-Surface

When stereo TS images are fed into the depth estimation unit, ESVIO will follow the depth estimation method proposed by ESVO [8] to estimate events' depth and fuse an inverse depth frame with the same timestamp as the corresponding TS image. At time t, the depth of event $e_{t-\varepsilon} = ({}^{cp_{t-\varepsilon}}\boldsymbol{x}, t-\varepsilon, p)$ (with $\varepsilon \in [0, \delta t]$, ${}^{cp_{t-\varepsilon}}\boldsymbol{x} \in \mathbb{R}^2$) on the left image plane can be estimated based on *the stereo temporal consistency criterion across space-time neighborhoods of the events* presented in [8]. The depth estimation optimization objective function of $e_{t-\varepsilon}$ is defined by

$$\begin{aligned}{}^{c_{t-\varepsilon}}\rho^\star &= \arg\min_{{}^{c_{t-\varepsilon}}\rho} \mathbf{C}_{depth}({}^{cp_{t-\varepsilon}}\boldsymbol{x}, {}^{c_{t-\varepsilon}}\rho, I_{left}({}^{cp_t}(\cdot), t), I_{right}({}^{cp(r)_t}(\cdot), t), \boldsymbol{T}_{c_{t-\delta t}}^{c_t}), \\ \mathbf{C}_{depth} &\doteq \sum_{{}^{cp_t}\boldsymbol{x}_{1,i}\in{}^{cp_t}W_1, {}^{cp(r)_t}\boldsymbol{x}_{2,i}\in{}^{cp(r)_t}W_2} \boldsymbol{r}_i{}^2({}^{c_{t-\varepsilon}}\rho),\end{aligned} \tag{9}$$

where ${}^{c_{t-\varepsilon}}\rho$ means the inverse depth of the event in the camera frame $c_{t-\varepsilon}$ at time $t-\varepsilon$, $I_{left}({}^{cp_t}(\cdot), t)$ and $I_{right}({}^{cp(r)_t}(\cdot), t)$ are the left and the right TS with the timestamp t, respectively, $\boldsymbol{T}_{c_{t-\delta t}}^{c_t} = (\boldsymbol{R}_{c_{t-\delta t}}^{c_t} | \boldsymbol{p}_{c_{t-\delta t}}^{c_t})$ is camera transformation matrix between camera frame $c_{t-\delta t}$ and c_t computed from the VIO state estimation results, and the depth residual $\boldsymbol{r}_i({}^{c_{t-\varepsilon}}\rho)$ is defined as

$$\boldsymbol{r}_i({}^{c_{t-\varepsilon}}\rho) \doteq I_{left}({}^{cp_t}\boldsymbol{x}_{1,i}, t) - I_{right}({}^{cp(r)_t}\boldsymbol{x}_{2,i}, t), \tag{10}$$

where ${}^{cp_t}\boldsymbol{x}_{1,i} \in \mathbb{R}^2$ and ${}^{cp(r)_t}\boldsymbol{x}_{2,i} \in \mathbb{R}^2$ are pixel coordinates on the event ($e_{t-\varepsilon}$) corresponding patches ${}^{cp_t}W_1$ and ${}^{cp(r)_t}W_2$ of the left and the right TS at time t, respectively.

Based on Equations (9) and Equation (10), we can estimate the depth of the latest N events which occur at or before time t. In ESVIO, we set N as 1500 (*MVSEC* dataset [27]) and 1000 (*RPG* datasets [25] and *ESIM* dataset [26]) based on the corresponding setting in [8] for different sensor resolutions to obtain a trade-off result between accuracy performance and time cost. Then we will fuse these asynchronous event depth points into an inverse depth frame with timestamp t based on the *probabilistic model of estimated inverse depth* and *inverse depth filters* proposed in [8]. In the real implementation of ESVIO, we fuse the inverse depth frame with timestamp t based on $3N \sim 5N$ event depth points, where N event depth points are associated with timestamp t and the rest are associated with several timestamps before t. In this way, we can construct inverse depth frames containing historical depth information for a short period of time.

5. Event-Inertial Initialization Method

Initialization is important for VIO to obtain a series of parameters, such as scale, gravity direction, and IMU bias, etc. These initialization parameters can help the system align with the world coordinate system and the scale of the real world, improving the convergence speed and estimation accuracy of state optimization in the VIO system. In the traditional VIO field, the initialization method, which combines visual information and IMU measurement, has been developed maturely, such as the disjoint initialization method used in VINS-Mono [15,53], the semi-joint method used in ORB-SLAM-3 [39,48], etc. However, for the event-based VIO field, there is no initialization method that fuses event data and IMU measurement together to estimate initialization parameters.

Therefore, we present an event-inertial initialization method for ESVIO, which can estimate scale, gravity direction, accelerometer, and gyroscope biases in the initial stage of ESVIO. This method is a kind of semi-joint initialization method and performs well in terms of real-time performance and accuracy performance. Our initialization method can be split into two steps: event-only initialization and event-inertial initial optimization.

5.1. Event-Only Initialization

The initialization procedure of ESVIO starts with the event-only initialization to obtain an up-to-scale camera trajectory. This trajectory can provide translational and rotational constraints for the event-inertial initial optimization to improve the accuracy and time performance of the optimization.

Firstly, ESVIO will apply a modified Semi-Global Matching (SGM) method [58] provided in ESVO [8] to generate an initial inverse depth frame based on the stereo TS images. Secondly, ESVIO will execute ESVO's tracking and mapping procedure bootstrapped by the initial inverse depth frame. At this period, ESVIO will maintain a sliding window that contains a series of up-to-scale camera poses estimated by the event-based stereo VO process (ESVO). The construction method of the sliding window can refer to Section 6.3. Note that, we find that the camera motion trajectory estimated by ESVO always with a scale error as 10∼15%, so the camera poses and the trajectory obtained here are called up-to-scale ones, and the subsequent event-inertial initial optimization also takes the scale factor as one of the initialization parameters. The up-to-scale camera poses are defined as $\bar{\boldsymbol{T}}_{c_i}^{c_0} = [\boldsymbol{R}_{c_i}^{c_0} | \bar{\boldsymbol{p}}_{c_i}^{c_0}], i \in [0, n]$, where we set the first camera frame ${}^{c_0}(\cdot)$ in the sliding window as the reference frame and n is the length of the sliding window. Suppose we have the extrinsic parameters $[\boldsymbol{R}_c^b | \boldsymbol{p}_c^b]$ between the event camera (left) and the IMU, we can translate poses from camera frame to IMU frame by

$$\begin{aligned} \boldsymbol{R}_{b_i}^{c_0} &= \boldsymbol{R}_{c_i}^{c_0} \cdot \boldsymbol{R}_c^{b^\top} \\ s\bar{\boldsymbol{p}}_{b_i}^{c_0} &= s\bar{\boldsymbol{p}}_{c_i}^{c_0} - \boldsymbol{R}_{c_0}^{b_i^\top} \cdot \boldsymbol{p}_c^b, \end{aligned} \tag{11}$$

where $s \in \mathbb{R}^+$ is the initialization parameter *scale* which can align the up-to-scale trajectory to the metric-scale trajectory. We will solve this parameter along with other initialization parameters in the next step.

5.2. Event-Inertial Initial Optimization

This step aims to solve an optimal estimation of the initialization parameters for ESVIO. Due to the mutual coupling between the initialization parameters, it is difficult to decouple all parameters and solve them sequentially in multiple steps. Therefore, we solve all initialization parameters through one-step optimization to obtain more accurate initialization results. When the camera motion in the sliding window maintained by the previous step (Section 5.1) is sufficient, ESVIO will carry out the event-inertial initial optimization using the sliding window optimization method based on the up-to-scale camera trajectory in the sliding window. We follow the method proposed in [53] to judge whether the camera motion is sufficient by checking the variation of pre-integration items $\boldsymbol{\alpha}_{b_{i+1}}^{b_i}$ and $\boldsymbol{\beta}_{b_{i+1}}^{b_i}$ in the sliding window.

Before event-inertial initial optimization, ESVIO will make a rough estimation of the gravity direction based on velocity pre-integration items $\boldsymbol{\beta}_{b_{i+1}}^{b_i}$ in the sliding window. Considering two consecutive IMU frames b_i and b_{i+1} in the sliding window, the velocity pre-integration item in Equation (5) can be converted to the reference frame ${}^{c_0}(\cdot)$ as:

$$\boldsymbol{R}_{b_i}^{c_0}\boldsymbol{\beta}_{b_{i+1}}^{b_i} = \boldsymbol{R}_{b_i}^{c_0}({}^{b_i}\boldsymbol{v}_{i+1} - {}^{b_i}\boldsymbol{v}_i + {}^{b_i}\boldsymbol{g}\Delta t), \tag{12}$$

where $\boldsymbol{R}_{b_i}^{c_0}$ can be obtained by Equation (11).

Adding all velocity pre-integration items as shown in Equation (12) we can obtain

$$\sum_{i=0}^{n-1}\boldsymbol{R}_{b_i}^{c_0}\boldsymbol{\beta}_{b_{i+1}}^{b_i} = {}^{c_0}\boldsymbol{v}_n - {}^{c_0}\boldsymbol{v}_0 + {}^{c_0}\boldsymbol{g}t \approx {}^{c_0}\boldsymbol{g}t, \tag{13}$$

where t is the total duration of the sliding window.

When we ignore the camera velocity changes during the sliding window, we can obtain a rough estimation of the gravity direction by normalizing the result in Equation (13).

Based on the up-to-scale camera trajectory and the rough gravity direction, ESVIO will start the event-inertial initial optimization by the sliding window optimization method. The optimal variables in the sliding window are defined by

$$\begin{aligned}\mathcal{X} &= [\boldsymbol{R}_w^{c_0}, s, \boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]^\top \\ \boldsymbol{x}_i &= [{}^{b_i}\boldsymbol{v}_i, {}^{b_i}\boldsymbol{b}_a, {}^{b_i}\boldsymbol{b}_g], i \in [0, n],\end{aligned} \tag{14}$$

where $\boldsymbol{R}_w^{c_0}$ is the gravity direction such that the gravity in the camera frame ${}^{c_0}(\cdot)$ is expressed as ${}^{c_0}\boldsymbol{g} = \boldsymbol{R}_w^{c_0}\,{}^w\boldsymbol{g}$, with ${}^w\boldsymbol{g} = (0, 0, G)^\top$ being G the Gravitational Constant, $\boldsymbol{x}_i$ is the camera initial velocity and the IMU biases in body frame at the time while taking the i-th data frame. n is the number of data frames in the sliding window.

Then, the optimization cost function $\mathbf{C}_{init}$ for the event-inertial initial optimization can be designed by

$$\mathbf{C}_{init} = \arg\min_{\mathcal{X}}\left\{\sum_{i\in\mathcal{B}}\left\|\boldsymbol{r}_{\mathcal{B}}(\hat{\boldsymbol{z}}_{b_{i+1}}^{b_i}, \mathcal{X})\right\|_{\boldsymbol{P}_{b_{i+1}}^{b_i}}^2 + \left\|\mathbf{r}_p\right\|_{\boldsymbol{P}_p}^2\right\}, \tag{15}$$

where $\boldsymbol{r}_{\mathcal{B}}(\hat{\boldsymbol{z}}_{b_{i+1}}^{b_i}, \mathcal{X})$ is the residual for IMU measurement between consecutive IMU frames b_i and b_{i+1}, $\mathbf{r}_p$ is the prior residual for the accelerometer bias as $\mathbf{r}_p = \sum||{}^{b_i}\boldsymbol{b}_a{}^2||$ referring to the initialization method [48]: if the motion does not contain enough information to estimate $\boldsymbol{b}_a$, this prior will keep its estimation close to zero; if the motion makes $\boldsymbol{b}_a$ observable, its estimation will converge towards its true value. $\boldsymbol{P}_{b_{i+1}}^{b_i}$ is the covariance matrix for the IMU measurement residual and can be computed iteratively with IMU propagation as proposed in [15]. $\boldsymbol{P}_p$ is the covariance matrix for the prior residual and can be defined as the diagonal matrix of noise ${\sigma_{b_a}}^2$. As to the gyroscope bias $\boldsymbol{b}_g$, it is observable from estimated camera orientations and gyroscope readings, so we do not need to set a prior factor for it [48].

For the IMU measurement residual $\boldsymbol{r}_{\mathcal{B}}(\hat{\boldsymbol{z}}_{b_{i+1}}^{b_i}, \mathcal{X})$, it can be defined according to the IMU measurement model defined in Equation (8) and the up-to-scale camera poses defined

in Equation (11) and has the same form as Equation (25) with different position, velocity and rotation residuals:

$$\begin{aligned}
\delta\boldsymbol{\alpha}_{b_{i+1}}^{b_i} &= \boldsymbol{R}_{c_0}^{b_i}\left(s(\bar{\boldsymbol{p}}_{b_{i+1}}^{c_0} - \bar{\boldsymbol{p}}_{b_i}^{c_0}) + \frac{1}{2}\boldsymbol{R}_w^{c_0}\,{}^{w}\boldsymbol{g}\Delta t^2 - \boldsymbol{R}_{b_i}^{c_0}\,{}^{b_i}\boldsymbol{v}_i\Delta t\right) - \hat{\boldsymbol{\alpha}}_{b_{i+1}}^{b_i} \\
\delta\boldsymbol{\beta}_{b_{i+1}}^{b_i} &= \boldsymbol{R}_{c_0}^{b_i}\left(\boldsymbol{R}_{b_{i+1}}^{c_0}\,{}^{b_{i+1}}\boldsymbol{v}_{i+1} + \boldsymbol{R}_w^{c_0}\,{}^{w}\boldsymbol{g}\Delta t - \boldsymbol{R}_{b_i}^{c_0}\,{}^{b_i}\boldsymbol{v}_i\right) - \hat{\boldsymbol{\beta}}_{b_{i+1}}^{b_i} \\
\delta\boldsymbol{\gamma}_{b_{i+1}}^{b_i} &= 2\left[\boldsymbol{q}_{b_i}^{c_0\,-1} \otimes \boldsymbol{q}_{b_{i+1}}^{c_0} \otimes (\hat{\boldsymbol{\gamma}}_{b_{i+1}}^{b_i})^{-1}\right]_{xyz}
\end{aligned} \tag{16}$$

where $\boldsymbol{R}_{c_0}^{b_i}(\boldsymbol{q}_{b_i}^{c_0})$, $\boldsymbol{q}_{b_{i+1}}^{c_0}$, $\bar{\boldsymbol{p}}_{b_i}^{c_0}$ and $\bar{\boldsymbol{p}}_{b_{i+1}}^{c_0}$ can be obtained from the up-to-scale camera trajectory estimated in the event-only initialization (Section 5.1), $[\cdot]_{xyz}$ means the real part of a quaternion which is used to approximate the three-dimensional rotational error.

The solution to the non-linear optimization problem in Equation (15) is similar to the state estimation in Equation (19), and a detailed description can be found in Section 6.3. Additionally, the Jacobian matrices of the IMU measurement residual can be found in Appendix A.1.

6. Direct Event-Based VIO Method

In this section, we propose a novel direct event-based VIO method for ESVIO to directly estimate camera state based on event data and IMU measurement without feature tracking. Firstly, the proposed method reconstructs the inverse depth frame to simplify the redundant depth information. Then, it generates the negative TS image for state estimation. Thirdly, based on inverse depth frames, negative TS images, and pre-integrated IMU measurement, the method estimates the optimal state of the system in a sliding window non-linear optimization framework. In the following, we will describe in detail the inverse depth frame reconstruction, the negative TS image generation, and the direct VIO state estimation.

6.1. Inverse Depth Frame Reconstruction

For ESVIO, the inverse depth frame provided by the depth estimation unit sometimes contains too many depth points with redundant depth information, which will bring additional computation overhead. To tackle this problem, we propose an inverse depth frame reconstruction method to filter the depth point subset according to the distribution of depth points and reconstruct the inverse depth frame based on these depth points. As illustrated in Figure 3, the method can be achieved following the below four steps:

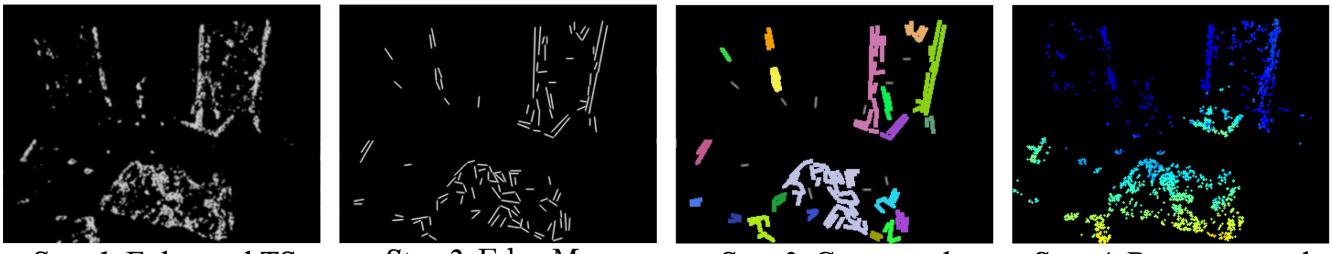


Figure 3. Four steps of the inverse depth frame reconstruction. Step 1: Enhance the TS image based on the inverse depth frame. Step 2: Extract the edge map from the enhanced TS image. Step 3: Extract connected regions in the edge map and cluster depth points. Step 4: Select depth points from each connected region according to the distribution of depth points to reconstruct the inverse depth frame.

- *Step 1*: Project all depth points of the inverse depth frame to its corresponding TS image (same timestamp) to enhance the intensity information in this TS image.

- *Step 2*: Use EDLines [59] to extract line segments in the enhanced TS image to obtain the edge map (taking less than 1.5 ms per TS image in datasets used in Section 7).
- *Step 3*: Extract connected regions in the edge map based on the contour retrieval algorithm [60]. According to the connected regions and projected coordinates in the TS image, cluster the depth points into each connected region.
- *Step 4*: According to the proportion of the number of depth points in each connected region to the total number of depth points, determine the number of points to select in each connected region, and then randomly select this number of depth points from each connected region to reconstruct the inverse depth frame.

6.2. Negative TS Image Generation

In the TS image, the pixel with a large intensity value represents the recently triggered event. Additionally, in a certain direction of this pixel, the pixel's intensity value will decrease as the distance increases, representing the same event triggered at multiple previous moments. Due to this characteristic, the TS image can be interpreted as a kind of anisotropic distance field which is usually used in edge-based VO systems [8]. Referring to ESVO [8], we invert the TS image to the negative one and utilize it to construct the minimization optimization problem for the VIO state estimation. The negative TS image can be created from a TS map by

$$\bar{I}(\boldsymbol{x}, t) = 1 - I(\boldsymbol{x}, t), \tag{17}$$

then re-scale the pixel's value from [0, 1] to the range [0, 255].

In ESVIO, the negative TS image will be used to construct event measurement residual and participates in the state estimation.

6.3. Direct VIO State Estimation Based on Event Data and IMU Measurement

For the direct method in the traditional VIO field, it constructs the visual measurement residual by the photometric error between two camera frames, which will utilize the whole image information for state estimation. However, the event can only reflect the brightness change, not the brightness intensity, and event-based VIO can hardly construct visual residual by the photometric error. Therefore, inspired by ESVO [8], for the direct method in ESVIO, we construct the event data residual based on the projection error of the inverse depth frame on the negative TS image, which can utilize the information of event inverse depth frames and event image frames (negative TS images) more comprehensively than feature-based methods.

After achieving the inverse depth frame reconstruction (Section 6.1) and the negative TS image generation (Section 6.2), based on the reconstructed inverse depth frame, ESVIO will construct *data frame* which contains the corresponding negative TS image with the same timestamp and pre-integrated IMU measurement between two consecutive inverse depth frames. The sliding window will consist of a fixed number of *data frames* as shown in Figure 4. Note that, referring to [8], we only use the left negative TS for VIO because incorporating the right negative TS does not significantly increase accuracy while it doubles the computation cost.

The full state variables in the sliding window while taking the i-th data frame are defined by

$$\begin{aligned} \mathcal{X} &= [\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]^\top \\ \boldsymbol{x}_i &= [\boldsymbol{p}_{b_i}^w, {}^w\boldsymbol{v}_i, \boldsymbol{q}_{b_i}^w, {}^{b_i}\boldsymbol{b}_a, {}^{b_i}\boldsymbol{b}_g], i \in [0, n], \end{aligned} \tag{18}$$

where $\boldsymbol{x}_i$ is described as the system position, velocity, and orientation in the world frame at the time while taking the i-th data frame and the IMU biases in the corresponding IMU frame. n is the number of data frames in the sliding window.

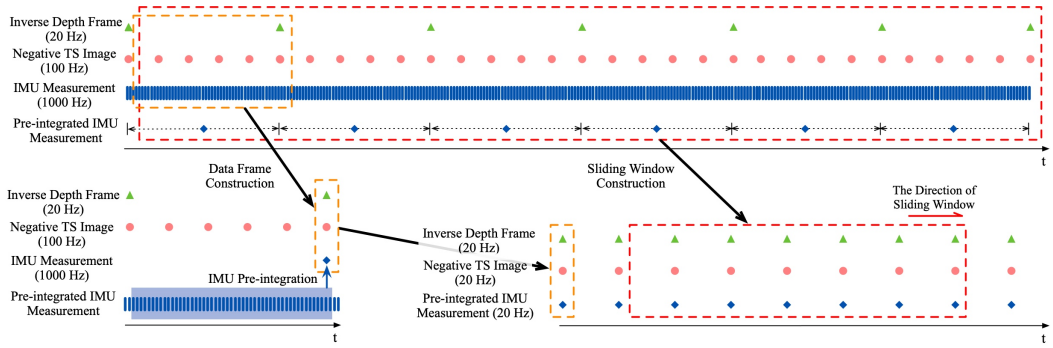


Figure 4. The sliding window construction of ESVIO. ESVIO construct *data frame* based on the frequency of inverse depth frames and construct the sliding window by a fixed number of *data frames*.

We achieve the optimization of the state estimation based on inverse depth frames, negative TS images, and pre-integrated IMU measurement. All the state variables are optimized in the sliding window by minimizing the sum of the cost terms from all the measurement residuals, including IMU measurement residuals and event data residuals. We design the optimization cost function $\mathbf{C}_{state}$ for state by

$$\mathbf{C}_{state} = \arg\min_{\mathcal{X}} \left\{ \sum_{i \in \mathcal{B}} \left\| \boldsymbol{r}_{\mathcal{B}}(\hat{\boldsymbol{z}}^{b_i}_{b_{i+1}}, \mathcal{X}) \right\|^2_{\boldsymbol{P}^{b_i}_{b_{i+1}}} + \sum_{j \in \mathcal{D}, l \in \mathcal{T}} \rho\left(\left\| \boldsymbol{r}_{\mathcal{D},\mathcal{T}}(\hat{\boldsymbol{z}}^{c_l}_{j}, \mathcal{X}) \right\|^2_{\boldsymbol{P}^{c_l}_{j}} \right) \right\}, \tag{19}$$

where $\boldsymbol{r}_{\mathcal{B}}(\hat{\boldsymbol{z}}^{b_i}_{b_{i+1}}, \mathcal{X})$ is the residual for IMU measurement and $\mathcal{B}$ is the set of pre-integrated IMU measurement in the sliding windows. $\boldsymbol{r}_{\mathcal{D},\mathcal{T}}(\hat{\boldsymbol{z}}^{c_l}_{j}, \mathcal{X})$ is the residual for event data. $\mathcal{D}$ and $\mathcal{T}$ are the sets of inverse depth frames and corresponding negative TS images in the sliding window, respectively. $\boldsymbol{P}^{b_i}_{b_{i+1}}$ and $\boldsymbol{P}^{c_l}_{j}$ are covariance matrices for the IMU measurement residual and the event data residual. $\boldsymbol{P}^{b_i}_{b_{i+1}}$ can be computed iteratively with IMU propagation as proposed in [15], and $\boldsymbol{P}^{c_l}_{j}$ is set as the identity matrix during the implementation of ESVIO. ρ is the Huber loss function [61], which is used to down-weight the large event data residual to improve the efficiency and reduce the impacts of noise events or incorrect depth points, defined as

$$\rho(r) = \begin{cases} r & , r \leq k^2 \\ 2k\sqrt{r} - k & , r > k^2 \end{cases} \tag{20}$$

where k is the scale of the Huber loss and we set k as 20 in ESVIO by trial-and-error method.

ESVIO uses Levenberg–Marquard (LM) algorithm to solve the minimization non-linear optimization problem in Equation (19). The optimal state vector $\mathcal{X}$ will be found by iteratively minimizing the Mahalanobis distance of all the residuals. At each LM iteration step, the state increment $\delta\mathcal{X}$ can be solved by following equation based on Equation (19):

$$(\boldsymbol{H}_{\mathcal{B}} + \boldsymbol{H}_{\mathcal{D},\mathcal{T}})\delta\mathcal{X} = (\boldsymbol{b}_{\mathcal{B}} + \boldsymbol{b}_{\mathcal{D},\mathcal{T}}), \delta\mathcal{X} = [\delta\boldsymbol{x}_0, \delta\boldsymbol{x}_1, \ldots, \delta\boldsymbol{x}_n]^\top, \tag{21}$$

where $\boldsymbol{H}_{(\cdot)}$ is the Hessian matrix of residuals vector $\boldsymbol{r}_{(\cdot)}$, we have $\boldsymbol{H}_{(\cdot)} = \boldsymbol{J}^\top_{(\cdot)}\boldsymbol{P}^{-1}_{(\cdot)}\boldsymbol{J}_{(\cdot)}$ and $\boldsymbol{b}_{(\cdot)} = -\boldsymbol{J}^\top_{(\cdot)}\boldsymbol{P}^{-1}_{(\cdot)}\boldsymbol{r}_{(\cdot)}$, where $\boldsymbol{J}_{(\cdot)}$ is the Jacobian matrix of residuals vector $\boldsymbol{r}_{(\cdot)}$ with respect to $\mathcal{X}$, and $\boldsymbol{P}_{(\cdot)}$ is the covariance matrix of measurements.

For position, velocity, and bias items of state vector $\mathcal{X}$, the update operator and increments at each LM iteration step can be defined as

$$\boldsymbol{p}' = \boldsymbol{p} + \delta\boldsymbol{p}, \boldsymbol{v}' = \boldsymbol{v} + \delta\boldsymbol{v}, \boldsymbol{b}' = \boldsymbol{b} + \delta\boldsymbol{b}. \tag{22}$$

For the rotation item of state vector $\mathcal{X}$, since the four-dimensional rotation quaternion $\boldsymbol{q}$ is over-parameterized, we use a pertubation $\delta\boldsymbol{\theta} \in \mathbb{R}^3$ as the rotation increment referring to [15]. Therefore, the update operator of $\boldsymbol{q}$ at each LM iteration step can be defined as

$$\boldsymbol{q}' \approx \boldsymbol{q} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}\delta\boldsymbol{\theta} \end{bmatrix}. \tag{23}$$

We can also write the Equation (23) as the rotation matrix form:

$$\boldsymbol{R}' \approx \boldsymbol{R}(\boldsymbol{I} + \lfloor \delta\boldsymbol{\theta} \rfloor_\times). \tag{24}$$

Considering the real-time performance, we do not set a specific marginalization step to marginalize old states. Formulations of measurements' residuals will be defined in the following, and the Jacobian matrices can be found in Appendix A.2.

For the IMU measurement residual between two consecutive frames b_i and b_{i+1} in the sliding window, according to the IMU measurement model defined in Equation (8), it can be defined by

$$\boldsymbol{r}_{\mathcal{B}}(\hat{\boldsymbol{z}}^{b_i}_{b_{i+1}}, \mathcal{X}) = \begin{bmatrix} \delta\boldsymbol{\alpha}^{b_i}_{b_{i+1}} \\ \delta\boldsymbol{\beta}^{b_i}_{b_{i+1}} \\ \delta\boldsymbol{\gamma}^{b_i}_{b_{i+1}} \\ \delta\boldsymbol{b}_a \\ \delta\boldsymbol{b}_g \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^{b_i}_w(\boldsymbol{p}^w_{b_{i+1}} - \boldsymbol{p}^w_{b_i} + \frac{1}{2}{}^w\boldsymbol{g}\Delta t^2 - {}^w\boldsymbol{v}_i\Delta t) - \hat{\boldsymbol{\alpha}}^{b_i}_{b_{i+1}} \\ \boldsymbol{R}^{b_i}_w({}^w\boldsymbol{v}_{i+1} + {}^w\boldsymbol{g}\Delta t - {}^w\boldsymbol{v}_i) - \hat{\boldsymbol{\beta}}^{b_i}_{b_{i+1}} \\ 2\left[\boldsymbol{q}^{w\,-1}_{b_i} \otimes \boldsymbol{q}^w_{b_{i+1}} \otimes (\hat{\boldsymbol{\gamma}}^{b_i}_{b_{i+1}})^{-1}\right]_{xyz} \\ {}^{b_{i+1}}\boldsymbol{b}_a - {}^{b_i}\boldsymbol{b}_a \\ {}^{b_{i+1}}\boldsymbol{b}_g - {}^{b_i}\boldsymbol{b}_g \end{bmatrix}. \tag{25}$$

For the event data residual between two data frames, inspired by ESVO [8], it can be defined as the depth projection error of one's inverse depth frame on another's negative TS image. Considering the j-th inverse depth frame and the l-th negative TS image ($j < l$) in the sliding window, the event data residual can be defined by

$$\begin{aligned} \boldsymbol{r}_{\mathcal{D},\mathcal{T}}(\hat{\boldsymbol{z}}^{c_l}_j, \mathcal{X}) &= \sum_{m\in\mathcal{D}_j} \bar{I}_{left}(({}^{cp_l}u_m, {}^{cp_l}v_m), t_l) = \sum_{m\in\mathcal{D}_j} \bar{I}_{left}(\pi({}^{c_l}\boldsymbol{P}_m), t_l), j < l \\ {}^{c_l}\boldsymbol{P}_m &= \begin{bmatrix} {}^{c_l}x_m \\ {}^{c_l}y_m \\ {}^{c_l}z_m \end{bmatrix} = \boldsymbol{R}^c_b(\boldsymbol{R}^{b_l}_w(\boldsymbol{R}^w_{b_j}(\boldsymbol{R}^b_c({}^{c_j}\boldsymbol{P}_m) + \boldsymbol{p}^b_c) + \boldsymbol{p}^w_{b_j} - \boldsymbol{p}^{b_l}_w) - \boldsymbol{p}^b_c), \end{aligned} \tag{26}$$

where ${}^{c_j}\boldsymbol{P}_m \in \mathbb{R}^3$ is the m-th depth point in the j-th inverse depth frame, ${}^{c_l}\boldsymbol{P}_m \in \mathbb{R}^3$ is the same depth point but in the l-th camera frame coordinate system. Additionally, $({}^{cp_l}u_m, {}^{cp_l}v_m)$ is the projected pixel location of ${}^{c_l}\boldsymbol{P}_m$ in the image plane of the l-th negative TS image. π is the projection function that turns a 3D point to a 2D pixel location using event camera intrinsic parameters.

7. Experimental Evaluation

We implement the ESVIO system referring to the implementation of the ESVO system [8] based on Robot Operating System (ROS) [62] in C++. In the ESVIO system, there are three different independent threads, namely *time_surface* thread, *depth_estimation* thread, and *VIO* thread. They run concurrently to achieve the VIO state estimation. *time_surface* thread implements the conversion of the event stream into TS images, which is a part of the measurements processing unit (in Figure 1). *depth_estimation* thread implements the depth estimation unit (in Figure 1). *VIO* thread implements the pre-integration of IMU measurement (another part of the measurements processing unit shown in Figure 1), the initialization unit (in Figure 1), and the VIO state estimation unit (in Figure 1). Each thread has a different running rate accordingly to ensure reliable operation and the real-time performance of the whole system.

To evaluate the proposed ESVIO system, we perform corresponding quantitative and qualitative experimental evaluations on the public *RPG* dataset [25] and *MVSEC*

dataset [27]. The data provided by the *RPG* dataset were collected with a handheld stereo event camera in an indoor environment. Additionally, the sequences we used for 6-DoF pose estimation from *MVSEC* dataset were collected using a stereo event camera mounted on a drone flying in a capacious indoor environment. We also evaluate the ESVIO on the *ESIM* dataset, which is generated by Jiao et al. [26] using the event camera simulator ESIM [63]. In addition, we try to evaluate ESVIO on the driving dataset *DSEC* [28] where the data were collected from a stereo event camera mounted on a driving car with a higher resolution. These four datasets all provide stereo event stream, IMU measurement, intrinsic parameters of event cameras, extrinsic parameters between event cameras and the IMU. The ground truth 6-DoF poses are also provided except *DSEC* dataset. However, none of these four datasets provide ground truth IMU bias, which can be used for the initialization evaluation. We list the parameters of sensors in each dataset in Table 1.

Table 1. Parameters of sensors in three datasets used in experimental evaluations. The IMU noise specs of MPU-6150 are from its product specification. The power spectral density is the square of noise amplitude density.

Dataset	Event Camera	Resolution (Pixel)	IMU	IMU Freq. (Hz)	IMU Noise Specs
RPG [25]	DAVIS240C	240 × 180	MPU-6150	1000	total RMS noise of gyr: 0.2°/s-rms power spectral density of acc: 400 $\mu g/\sqrt{Hz}$
MVSEC [27]	DAVIS346	346 × 260	MPU-6150	1000	
ESIM [26]	Simulator [63]	346 × 260	Simulator [63]	1000	gyr noise density: 1.86×10^{-4} rad/s$\sqrt{Hz}$ gyr random walk: 2.66×10^{-5} rad/s$^2\sqrt{Hz}$ acc noise density: 1.86×10^{-3} m/s$^2\sqrt{Hz}$ acc random walk: 4.33×10^{-4} m/s$^3\sqrt{Hz}$
DSEC [28]	PPS3MVCD	640 × 480	MPU-9250	1000	gyr noise density: 5.07×10^{-3} rad/s$\sqrt{Hz}$ gyr random walk: 5.69×10^{-5} rad/s$^2\sqrt{Hz}$ acc noise density: 7.81×10^{-2} m/s$^2\sqrt{Hz}$ acc random walk: 1.33×10^{-3} m/s$^3\sqrt{Hz}$

Based on the above datasets, firstly, we show the performance of the event-inertial initialization proposed in ESVIO. Secondly, we compare ESVIO with state-of-the-art methods in terms of accuracy performance. Thirdly, we show the improvement brought by the addition of IMU to ESVIO. Fourthly, we show the results of ESVIO on sequences from the driving dataset *DSEC*. Finally, we analyze the real-time performance of our system. Considering the randomness of the algorithm results [26], the results of quantitative evaluations (motion estimation, scale error, and time cost) are all average results of 10-trial tests. Additionally, considering the possible scale error in the adopted comparison methods, we align the estimated trajectory with ground truth using Sim(3) Umeyama alignment [64] in all evaluations. All experiments run on a computer equipped with Intel® Core™ i9-10900K CPU @ 3.70 GHz and Ubuntu 20.04 LTS operation system.

7.1. Performance of the Event-Inertial Initialization

To prove the effectiveness of our proposed event-inertial initialization method for ESVIO, we compare ESVIO and its variant without the proposed event-inertial initialization (simply "ESVIO without initialization") on the *RPG* and *MVSEC* datasets which are collected in the real world. The ESVIO without initialization is achieved by the direct VIO state estimation method provided in Section 6 with the initial guesses for gravity direction (solved by Equation (13)), accelerometer bias (zero), gyroscope bias (zero), velocities, and depth (recovered from the event-only initialization method in Section 5.1).

Because *RPG* and *MVSEC* datasets do not provide IMU calibration parameters and ground truth IMU bias, we choose the scale error and the mean absolute translational error (ATE) as the metric for evaluations. We compare the scale error at the beginning of the VIO system (5 s after the system starts running) and at the end of the system for

ESVIO and ESVIO without initialization. The scale error at the beginning of the system can directly reflect the influence of initialization on scale recovery. The scale error at the end of the system and the mean ATE can represent the effect of initialization on the VIO state estimation. As shown in Table 2, ESVIO demonstrates lower scale errors and better accuracy performance than ESVIO without initialization in most sequences. The results in Table 2 show that the event-inertial initialization can effectively recover the scale and improve the accuracy performance for ESVIO state estimation. In addition, note that although the lack of initialization will make the VIO system have a scale error in the initial stage, due to the addition of IMU, the scale error will decrease after the system works for a while.

Table 2. Quantitative comparison results of ESVIO and ESVIO without initialization on the *RPG* and *MVSEC* datasets. The scale error and the mean absolute translational error (ATE) are used as the metric. The scale error includes the error at the beginning of the system (5s) and at the end of the system. The best result is highlighted in **bold.**

Sequence	Duration (s)	Method	Scale Error (%)		Mean ATE (cm)
			5s	End	
RPG_bin_edited	16.995	ESVIO	**1.8**	**0**	**2.0**
		ESVIO_w/o_initialization	8.2	0.1	2.2
RPG_boxes_edited	14.995	ESVIO	**8.7**	**2.2**	**4.0**
		ESVIO_w/o_initialization	15.8	4.0	4.7
RPG_desk_edited	12.995	ESVIO	**2.0**	**1.7**	**1.8**
		ESVIO_w/o_initialization	2.9	2.2	2.5
RPG_monitor_edited	22.985	ESVIO	**2.3**	**0.2**	**2.6**
		ESVIO_w/o_initialization	15.3	2.6	3.8
MVSEC_indoor_flying1_edited	25.976	ESVIO	**8.7**	**5.1**	**7.7**
		ESVIO_w/o_initialization	9.8	5.9	9.4
MVSEC_indoor_flying3_edited	24.984	ESVIO	**6.4**	3.8	**4.3**
		ESVIO_w/o_initialization	9.6	**3.6**	7.1
Mean	19.822	ESVIO	**5.0**	**2.2**	**3.7**
		ESVIO_w/o_initialization	10.3	3.1	5.0

7.2. Evaluation of the VIO State Estimation Accuracy for ESVIO

To show the state estimation accuracy performance of ESVIO, we perform quantitative and qualitative evaluation experiments on the public *ESIM*, *RPG*, *MVSEC* datasets.

Firstly, the state estimation results of ESVIO evaluated on *ESIM*, *RPG*, and *MVSEC* datasets are shown in Figure 5. As shown in Figure 5, the trajectories (the fourth row) estimated by ESVIO have a good accuracy performance compared to the ground truth. Furthermore, the TS images (the second row) and inverse depth frames (the third row) from Figure 5 show the performance of ESVIO for event processing and depth estimation.

Secondly, we compare ESVIO with the state-of-the-art event-based stereo VO methods, including ESVO [8] and variants of ESVO with different event processing methods (refer to [26]), on *ESIM*, *RPG*, and *MVSEC* datasets. Table 3 shows the mean ATE under 12 sequences for ESVIO, ESVO, and variants of ESVO. The subscript of ESVO represents the different event-processing methods. *TS* represents TS ($ESVO_{TS}$ equals to the original ESVO). *EM*2000, *EM*3000, *EM*4000, *EM*5000 represent Event Map with 2000, 3000, 4000, 5000 events per map. *EMTS* represents a combination of TS and Event Map. We refer to [26] for the results of ESVO and variants of ESVO. As shown in Table 3, ESVIO demonstrates better accuracy performance in terms of motion estimation on all sequences compared with the state-of-the-art event-based stereo VO methods. In addition, we plot the absolute transla-

tional and rotational error statistics of ESVIO and ESVO in Figure 6. As shown in Figure 6, for absolute translational error, ESVIO performs better than ESVO on all sequences. As to the rotational error, ESVIO performs better than ESVO in both rotational error median and distribution on most sequences. From Figure 6, the experimental results demonstrate that the proposed ESVIO not only has the advantages of accuracy performance, but also has the advantages of robustness compared with the event-based stereo VO method.

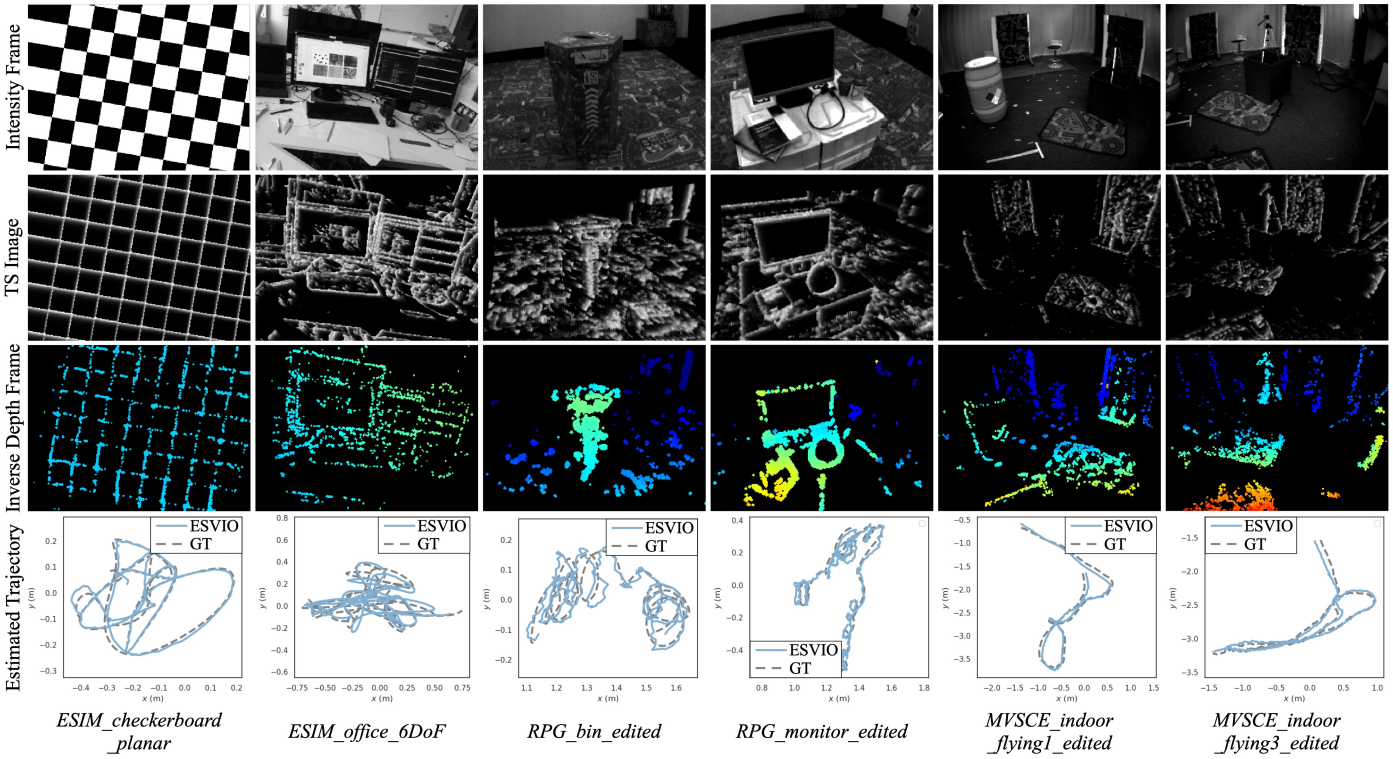


Figure 5. Results of ESVIO evaluated on *ESIM*, *RPG*, and *MVSEC* datasets. The first row shows intensity frames from the event camera, the second and the third rows show the TS images and the inverse depth frames generated by ESVIO, and the fourth row shows the trajectories estimated by ESVIO. Inverse depth frames are color-coded from red (close) to blue (far) over a black background.

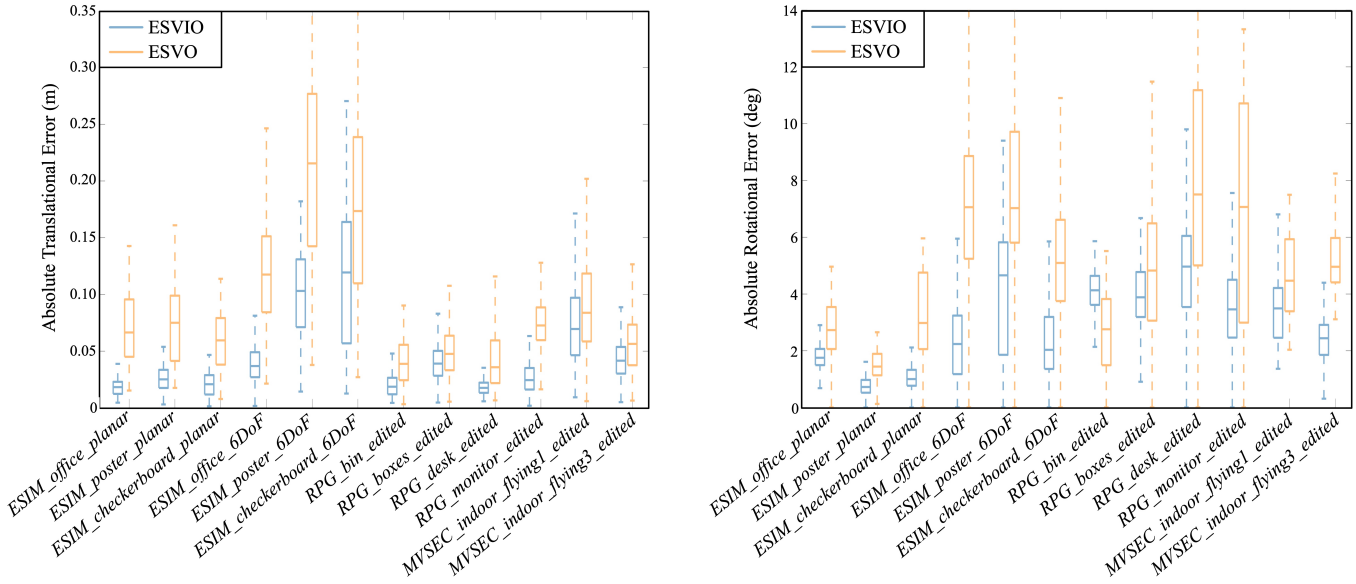
Thirdly, we compare ESVIO with the state-of-the-art event-based VIO method. At present, event-based VIO methods [19–23] are all designed for monocular systems and evaluated on the public dataset [65] which only provides monocular event data. Among these methods, only Ultimate SLAM [21] is open source method which is a feature-based VIO method and provides different VIO modes based on different sensor combinations. Therefore, we compare ESVIO and Ultimate SLAM on the same sequences used in Table 3, and the results are shown in Table 4. When we run Ultimate SLAM, we use its all parameter settings except the sensor calibration parameters. The results of Ultimate SLAM are collected under "event (E) + IMU" and "event (E) + IMU + frame (Fr)" two modes. As shown in Table 4, the performance of Ultimate SLAM is not as good as ESVIO. In most sequences, Ultimate SLAM cannot correctly estimate the state due to the failure of feature tracking (features can be extracted, but cannot be tracked stably). In the remaining sequences, although state estimation can be performed, the accuracy performance of Ultimate SLAM is still not as good as ESVIO due to the large estimation error in the initial stage. From the results in Table 4, we can find that ESVIO has better accuracy and robustness performances compared with Ultimate SLAM.

Table 3. Accuracy performance with ESVIO and different event-based stereo VO methods. The public *RPG*, *MVSEC*, *ESIM* datasets are adopted for comparison. The mean ATE (cm) is used as the metric. The best result is highlighted in **bold** and the second best result is highlighted with underline.

Sequence	Duration (s)	Length (m)	$ESVO_{TS}$	$ESVO_{EM2000}$	$ESVO_{EM3000}$	$ESVO_{EM4000}$	$ESVO_{EM5000}$	$ESVO_{EMTS}$	ESVIO
ESIM_office_planar	15.999	5.014	4.7	4.0	3.9	3.7	4.1	4.9	**1.8**
ESIM_poster_planar	15.999	5.014	4.7	3.7	4.3	4.6	5.0	5.2	**2.5**
ESIM_checkerboard_planar	15.999	5.014	4.2	2.9	2.2	2.3	2.4	3.5	**2.0**
ESIM_office_6DoF	29.999	15.003	9.1	25.3	21.0	16.6	15.8	9.4	**3.8**
ESIM_poster_6DoF	29.999	15.003	18.2	15.4	16.3	16.8	17.4	17.8	**10.2**
ESIM_checkerboard_6DoF	29.999	15.003	23.0	17.0	14.0	15.1	13.4	22.4	**11.5**
RPG_bin_edited	16.995	4.923	3.4	22.4	16.6	8.0	14.1	3.5	**2.0**
RPG_boxes_edited	14.995	6.686	6.5	5.3	17.1	13.7	9.8	22.1	**4.0**
RPG_desk_edited	12.995	3.926	3.4	2.9	3.3	3.2	2.9	3.6	**1.8**
RPG_monitor_edited	22.985	6.251	7.2	5.3	5.2	7.4	7.3	7.3	**2.6**
MVSEC_indoor_flying1_edited	25.976	11.761	18.5	22.0	16.7	16.0	22.1	14.9	**7.7**
MVSEC_indoor_flying3_edited	24.984	10.646	20.9	10.8	11.9	14.0	15.0	10.6	**4.3**
Mean	21.41	8.687	10.3	11.4	11.0	10.1	10.8	10.4	**4.5**

Table 4. Accuracy performance with ESVIO and Ultimate SLAM [21] on the *ESIM*, *RPG*, and *MVSEC* datasets. The mean ATE (cm) is used as the metric. The results of Ultimate SLAM are collected under "event (E) + IMU (I)" and "event (E) + IMU (I) + frame (Fr)" two modes. The symbol "-" means that the system can not correctly estimate states. The best result is highlighted in **bold.**

Sequence	ESVIO	Ultimate SLAM (E + I)	Ultimate SLAM (E + I + Fr)
ESIM_office_planar	**1.8**	-	9.2
ESIM_poster_planar	**2.5**	5.8	7.3
ESIM_checkerboard_planar	**2.0**	-	7.0
ESIM_office_6DoF	**3.8**	-	-
ESIM_poster_6DoF	**10.2**	-	-
ESIM_checkerboard_6DoF	**11.5**	-	-
RPG_bin_edited	**2.0**	2.5	2.8
RPG_boxes_edited	**4.0**	-	-
RPG_desk_edited	**1.8**	-	-
RPG_monitor_edited	**2.6**	3.8	3.8
MVSEC_indoor_flying1_edited	**7.7**	-	-
MVSEC_indoor_flying3_edited	**4.3**	-	-



(**a**) (**b**)

Figure 6. Boxplots of absolute translational (**a**) and rotational (**b**) error statistics for ESVIO and ESVO. The middle box spans the first and third quartiles, while the whiskers are the upper and lower limits. (**a**) Absolute Translational Error. (**b**) Absolute Rotational Error.

7.3. Evaluation of the Impact of Adding IMU to ESVIO

To validate the improvement brought by the addition of IMU to ESVIO, we present the quantitative and qualitative experimental evaluations in this subsection. We evaluate ESVIO, ESVIO without IMU (state estimation implemented based on only event measurement residual), and ESVO on the public *ESIM*, *RPG*, and *MVSEC* datasets. The mean ATE and scale error are chosen as the evaluation metric. The evaluation results are shown in Table 5. The results show that ESVIO with IMU has more accurate state estimation accuracy and lower scale error, which demonstrates that the addition of IMU improves the state estimation accuracy performance for ESVIO. Additionally, we notice that ESVIO will not reduce to ESVO when no IMU measurement is used due to the different event processing methods (improved TS vs. TS) and different state estimation methods (sliding window optimization

framework vs. 3D–2D registration between two consecutive frames). Note that, because [26] does not provide the scale error of ESVO, the results of ESVO in Table 5 are obtained by our own evaluation.

Table 5. Quantitative comparison results of ESVIO with and without IMU measurement on the *ESIM*, *RPG*, and *MVSEC* datasets. The scale error and the mean ATE are used as the metric. The best result is highlighted in **bold.**

Sequence	Scale Error (%)			Mean ATE (cm)		
	ESVIO	**ESVIO_w/o_IMU**	**ESVO**	**ESVIO**	**ESVIO_w/o_IMU**	**ESVO**
ESIM_office_planar	**2.4**	7.9	13.1	**1.8**	6.5	4.1
ESIM_poster_planar	**1.6**	12.4	11.0	**2.5**	6.7	4.7
ESIM_checkerboard_planar	**3.3**	10.7	13.0	**2.0**	5.3	4.9
ESIM_office_6DoF	**1.7**	11.6	24.3	**3.8**	9.1	11.5
ESIM_poster_6DoF	**2.0**	24.8	21.7	**10.2**	21.5	15.6
ESIM_checkerboard_6DoF	**10.7**	17.2	14.0	**11.5**	17.8	19.7
RPG_bin_edited	**0.0**	11.8	16.4	**2.0**	3.9	3.1
RPG_boxes_edited	**2.2**	11.1	24.7	**4.0**	5.0	8.8
RPG_desk_edited	**1.7**	6.7	7.7	**1.8**	3.2	3.6
RPG_monitor_edited	**0.2**	8.1	9.9	**2.6**	5.2	7.2
MVSEC_indoor_flying1_edited	**5.1**	9.5	12.7	**7.7**	13.1	14.1
MVSEC_indoor_flying3_edited	**3.8**	9.8	12.8	**4.3**	9.7	27.8
Mean	**2.9**	11.8	15.1	**4.5**	8.9	10.4

Then we choose two sequences *RPG_monitor_edited* and *MVSEC_indoor_flying1_edited* to show the details of translational and rotational errors of ESVIO and ESVIO without IMU measurement. The *RPG_monitor_edited* sequence contains more rotations, while the *MVSEC_indoor_flying1_edited* sequence is collected by a drone in a larger indoor space with a faster camera motion. As shown in Figure 7, the top subfigures are the estimated trajectories, the middle subfigures are the translational errors over time, and the bottom subfigures are the rotational errors over time. From the trajectories subfigures, we can discover that the trajectories estimated by ESVIO have a lower scale error, which are more consistent with the ground truth trajectories. From the translational error subfigures and the rotational error subfigures, we can intuitively find that ESVIO always has a lower value of the translational and rotational errors. Furthermore, from Figure 7, we can discover that the addition of IMU can significantly reduce rotation estimation error when the camera keeps rotating (e.g., 5–10 s in the *RPG_monitor_edited* sequence) or make a sharp turn (e.g., 3–7 s in the *MVSEC_indoor_flying1_edited* sequence).

7.4. Experiments on DSEC Dataset

To show the performance of ESVIO in large-scale scenes, we perform qualitative evaluation experiments on the public driving dataset *DSEC*. Driving scenarios are challenging for event-based sensors because forward motions typically produce considerably fewer events in the center of the image (where apparent motion is small) than in the periphery [66]. Additionally, the higher sensor resolution (640 × 480), the large-scale outdoor scenes and the dynamic objects (moving cars) are also challenging for ESVIO.

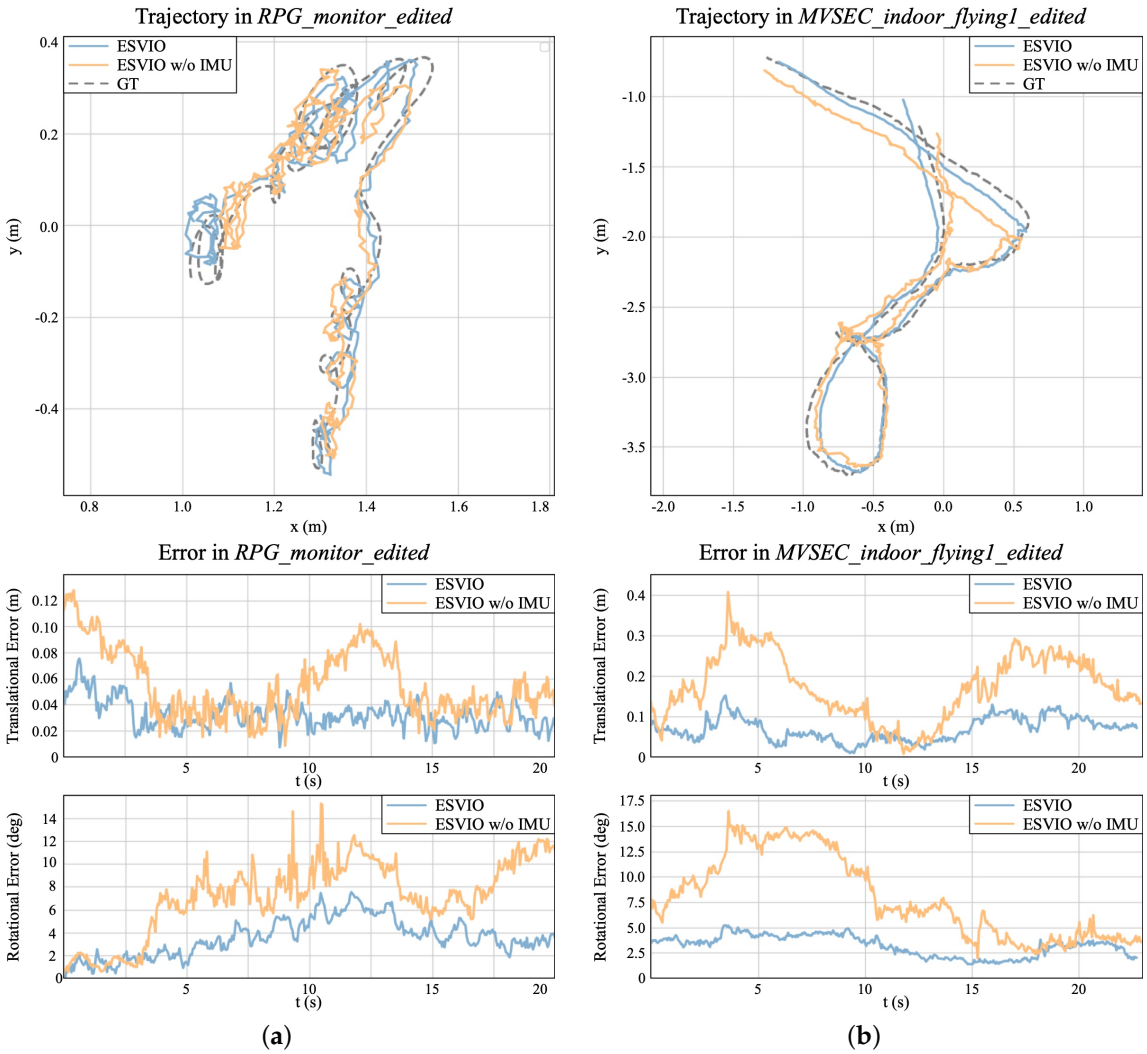
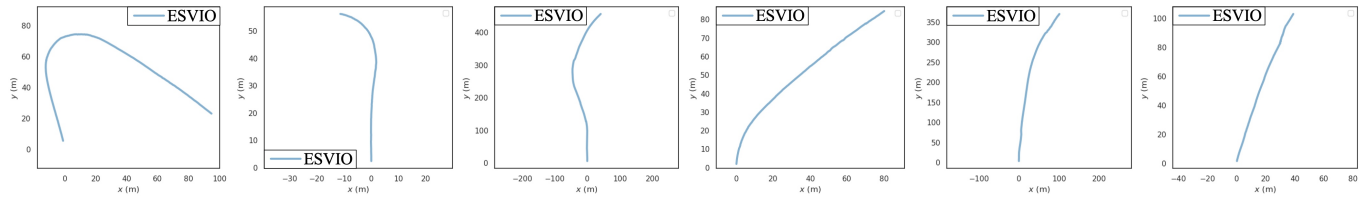


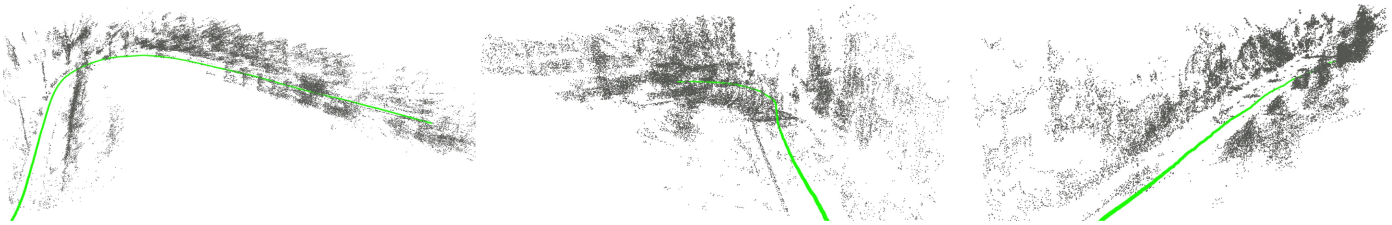
Figure 7. (**a**): Trajectories and the corresponding translational errors and rotational errors on the *RPG_monitor_edited* sequence. (**b**): Trajectory and the corresponding translational errors and rotational errors on the *MVSEC_indoor_flying1_edited* sequence. The blue line represents the results of the ESVIO, and the yellow line represents the results of ESVIO without IMU measurement. (**a**) *RPG_monitor_edited* sequence. (**b**) *MVSEC_indoor_flying1_edited* sequence.

Since the *DSEC* dataset does not provide the ground truth 6-DoF poses, we only show the results of ESVIO for the *DSEC* dataset sequences *zurich_city_04* and *zurich_city_11_a* in Figure 8. As shown in Figure 8g–l, ESVIO can successfully achieve the camera pose estimation not only on the medium-length driving sequences ((g), (h), (j), (l)), but also on the long driving sequences of up to more than 300 m ((i), (k)). Furthermore, the reconstructed 3D maps and estimated trajectories from Figure 8m,n show that ESVIO can complete semi-dense depth estimation and state estimation in the case of sparse and dense events (events in *zurich_city_04_a* and *zurich_city_04_b* are denser than in *zurich_city_11_a*, due to the richer texture). However, because of the very heavy events load, ESVIO can not run in real-time on the *DSEC* dataset, so we slow down the playback of the *rosbag* when performing the corresponding experiments in this subsection.

(**a**) *zurich_city_04_a* (**b**) *zurich_city_04_b* (**c**) *zurich_city_04_c* (**d**) *zurich_city_04_e* (**e**) *zurich_city_04_f* (**f**) *zurich_city_11_a*



(**g**) *zurich_city_04_a* (**h**) *zurich_city_04_b* (**i**) *zurich_city_04_c* (**j**) *zurich_city_04_e* (**k**) *zurich_city_04_f* (**l**) *zurich_city_11_a*



(**m**) Depth reconstruction in *zurich_city_04_a* (**n**) Depth reconstruction in *zurich_city_04_b* (**o**) Depth reconstruction in *zurich_city_11_a*

Figure 8. (**a**–**f**): Scene images of different sequences in *DSEC* dataset. (**g**–**l**): The trajectories produced by ESVIO with *DSEC* dataset sequences. (**m**–**o**): The reconstructed 3D maps and trajectories with *DSEC* dataset. The trajectories are in green and the depth points are in gray.

7.5. Real-Time Performance Analysis

In this section, we analyze the real-time performance of our proposed ESVIO system.

Firstly, we present the time cost of the initialization of ESVIO. For the event-only initialization, ESVIO will take about 12 ms to execute the modified SGM method to generate an initial inverse depth frame. Then ESVIO will take about 10 ms and 46 ms to execute the tracking and mapping procedure of ESVO, respectively, for estimating up-to-scale camera poses. For event-inertial initialization, ESVIO will take less than 1 ms for the rough estimation of the gravity direction and about 95 ms for the event-inertial initialization optimization in the slide window optimization framework (including 10 data frames). Considering the visual observation conditions required for the modified SGM method and the collection and accumulation time of the data frames in the sliding window, the initialization will be completed within 2 s after the ESVIO starts.

Secondly, we show the real-time performance of ESVIO. Table 6 lists the detailed execution times of different threads of ESVIO under *MVSEC_indoor_flying1_edited* sequence. As shown in Table 6, for ESVIO, the *time_surface* thread takes about 12 ms ($\sim$ 83 Hz) to create the TS image, the *depth_estimation* thread takes about 46 ms ($\sim$ 22 Hz) to estimate 1500 events' depth and fuse 6000 depth points to the inverse depth frame, the *VIO* thread takes about 24 ms ($\sim$ 42 Hz) to perform state estimation in the slide window optimization framework (including 6 data frames) based on 500 depth points per reconstructed inverse depth frame.

Table 6. Mean execution time of ESVIO's different threads with *MVSEC_indoor_flying1* sequence of *MVSEC Dataset*.

Thread	Method	Times (ms)	Rate (Hz)
time_surface	Event processing	12.0	83.3
depth_estimation	Depth estimation	45.8	21.8
VIO	IMU pre-integration	0.4	41.5
	Data preprocessing	2.1	
	Nonlinear optimization	21.6	

Regarding the choice for the size of the sliding window (6 as mentioned above), we justify it by showing its influence on the state estimation accuracy and computation cost. As shown in Table 7, we have investigated the effect of sliding window size (4, 6, 8, and 10 frames, respectively) in terms of state estimation accuracy and computation cost of the *VIO* thread under *MVSEC_ indoor_flying1_ edited* sequence. The results show that the sliding window with 4 data frames presents the worst performance in accuracy, and the sliding window with 6/8/10 data frames demonstrates similar accuracy performance. However, the sliding window with 8/10 data frames has a larger computation cost. Considering the real-time performance, we set the sliding window size to 6 in the state estimation.

Table 7. State estimation accuracy and computation cost of the *VIO* thread under different sizes of the sliding window optimization framework with *MVSEC_indoor_flying1* sequence. The mean ATE (cm) is used as the metric for accuracy comparison.

Size of Sliding Window	State Estimation Accuracy (cm)	Computation Cost of *VIO* Thread (ms)
4	10.2	16.9
6	7.7	24.1
8	7.5	36.5
10	8.3	44.2

Regarding the choice for the number of depth points selected in the state estimation (500 as mentioned above), we implement the same experiments under *MVSEC_ indoor_flyin- g1_ edited* sequence. Table 8 shows the state estimation accuracy and computation cost of the *VIO* thread with selecting 300, 500, 1000, and 1500 depth points per reconstructed inverse depth frame. As shown in Table 8, while increasing the number of depth points can effectively improve the state estimation accuracy, however, it also brings more computation cost. Considering the real-time performance, we choose to select 500 depth points per reconstructed inverse depth frame in the state estimation.

Table 8. State estimation accuracy and computation cost of the *VIO* thread under different numbers of depth points per reconstructed inverse depth frame in the state estimation with *MVSEC_indoor_flying1* sequence. The mean ATE (cm) is used as the metric for accuracy comparison.

Number of Depth Points	State Estimation Accuracy (cm)	Computation Cost of *VIO* Thread (ms)
300	9.9	16.4
500	7.7	24.1
1000	7.8	43.8
1500	7.3	66.5

Regarding the number of events for depth estimation and the number of the depth points for depth fusion in the *depth_estimation* thread (1500 and 6000 as mentioned above),

we refer to the corresponding settings in ESVO [8] and make some adjustments on this basis to obtain inverse depth frames with more depth information on different datasets.

8. Conclusions

This paper proposes a novel event-based stereo VIO system, namely ESVIO. Firstly, we present a direct event-based stereo VIO pipeline for the first time, which can directly estimate camera motion based on event data and IMU measurement in a tightly coupled sliding window non-linear optimization framework. Secondly, we design a semi-joint event-inertial initialization method that can solve initialization parameters in two steps at the initial stage of the VIO system. Based on the VIO and the initialization methods, we implement the ESVIO system. Corresponding experimental evaluations are conducted to prove the effectiveness of the proposed system, and the results show that ESVIO achieves good accuracy and robustness performance when compared with other event-based monocular VIO and stereo VO systems, and, at the same time, with no compromise to real-time performance.

However, ESVIO still has some limitations. Firstly, the performance of ESVIO is still limited by the hardware of event cameras. Secondly, ESVIO lacks the long-term data association mechanism, such as loop closure, which enables relocalization and drifts elimination.

In the future, we will introduce the standard camera frame to enhance the performance of ESVIO and implement loop closure for ESVIO. We will also implement a specific marginalization step for ESVIO. We would also like to record an event camera dataset that is more suitable for event-based VIO systems and event-inertial initialization methods to test and evaluate.

Author Contributions: Conceptualization, Z.L.; methodology, Z.L.; software, Z.L.; validation, Z.L.; formal analysis, Z.L.; investigation, Z.L.; resources, D.S.; data curation, Z.L.; writing—original draft preparation, Z.L.; writing—review and editing, Z.L., D.S. and R.L.; visualization, Z.L.; supervision, D.S. and R.L.; project administration, D.S. and S.Y.; funding acquisition, D.S., R.L. and S.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant No. 91948303) and the National Natural Science Foundation of China (Grant No. 61903377).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank Yi Zhou for sharing their code of ESVO and edited rosbag files of *RPG* and *MVSEC* datasets. We also thank Jianhao Jiao for providing the ESVO and the variants of ESVO baselines [26] and edited rodbag files of *ESIM* dataset. We also thank Antoni Rosinol Vidal for sharing the code of Ultimate SLAM.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Appendix A.1. Jacobian Matrices in Initialization

The Jacobian matrices of the IMU measurement residual with respect to state increments $\delta \boldsymbol{x}_i$ and $\delta \boldsymbol{x}_{i+1}$ are, respectively, computed by

$$\frac{\partial \boldsymbol{r}_{\mathcal{B}}}{\partial \delta \boldsymbol{x}_i} = \begin{bmatrix} -\Delta t & -\boldsymbol{J}^{\boldsymbol{\alpha}}_{b_i \boldsymbol{b}_a} & -\boldsymbol{J}^{\boldsymbol{\alpha}}_{b_i \boldsymbol{b}_g} \\ -\boldsymbol{I} & -\boldsymbol{J}^{\boldsymbol{\beta}}_{b_i \boldsymbol{b}_a} & -\boldsymbol{J}^{\boldsymbol{\beta}}_{b_i \boldsymbol{b}_g} \\ 0 & 0 & \left[\hat{\boldsymbol{\gamma}}^{b_i}_{b_{i+1}} \otimes \boldsymbol{q}^{c_0}_{b_{i+1}}{}^{-1} \otimes \boldsymbol{q}^{c_0}_{b_i}\right]_L \boldsymbol{J}^{\boldsymbol{\gamma}}_{b_i \boldsymbol{b}_g} \\ 0 & -\boldsymbol{I} & 0 \\ 0 & 0 & -\boldsymbol{I} \end{bmatrix}, \frac{\partial \boldsymbol{r}_{\mathcal{B}}}{\partial \delta \boldsymbol{x}_{i+1}} = \begin{bmatrix} 0 & 0 & 0 \\ \boldsymbol{R}^{b_i}_{c_0} \boldsymbol{R}^{c_0}_{b_{i+1}} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \boldsymbol{I} & 0 \\ 0 & 0 & \boldsymbol{I} \end{bmatrix}, \quad \text{(A1)}$$

where $[\cdot]_L$ is the left-quaternion-product matrices [67], $\boldsymbol{J}^{(\cdot)}_{b_i \boldsymbol{b}_a} = \frac{\partial (\cdot)^{b_i}_{b_{i+1}}}{\partial \delta^{b_i} \boldsymbol{b}_a}$ and $\boldsymbol{J}^{(\cdot)}_{b_i \boldsymbol{b}_g} = \frac{\partial (\cdot)^{b_i}_{b_{i+1}}}{\partial \delta^{b_i} \boldsymbol{b}_g}$ are the Jacobian matrices of pre-integration terms with respect to accelerometer bias increment and gyroscope bias increment, respectively, referring to [15], such as $\boldsymbol{J}^{\boldsymbol{\alpha}}_{b_i \boldsymbol{b}_a} = \frac{\partial \boldsymbol{\alpha}^{b_i}_{b_{i+1}}}{\partial \delta^{b_i} \boldsymbol{b}_a}$.

The Jacobian matrices of the IMU measurement residual with respect to the gravity direction increment $\delta \boldsymbol{R}^{c_0}_w$ and the scale increment δs are, respectively, computed by

$$\frac{\partial \boldsymbol{r}_{\mathcal{B}}}{\partial \delta \boldsymbol{R}^{c_0}_w} = \begin{bmatrix} \frac{1}{2} \Delta t^2 \boldsymbol{R}^{b_i}_{c_0} \boldsymbol{R}^{c_0}_w \lfloor {}^w \boldsymbol{g} \rfloor_\times \\ \frac{1}{2} \Delta t \boldsymbol{R}^{b_i}_{c_0} \boldsymbol{R}^{c_0}_w \lfloor {}^w \boldsymbol{g} \rfloor_\times \\ 0 \\ 0 \\ 0 \end{bmatrix}, \frac{\partial \boldsymbol{r}_{\mathcal{B}}}{\partial \delta s} = \begin{bmatrix} \boldsymbol{R}^{b_i}_{c_0} (\bar{\boldsymbol{p}}^{c_0}_{b_{i+1}} - \bar{\boldsymbol{p}}^{c_0}_{b_i}) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{A2}$$

Appendix A.2. Jacobian Matrices in State Estimation

Appendix A.2.1. Jacobian Matrices of IMU Measurement Residual

During the non-linear optimization, the Jacobian matrix of the IMU measurement residual with respect to the state increment $\delta \boldsymbol{x}_i$ is computed by

$$\frac{\partial \boldsymbol{r}_{\mathcal{B}}}{\partial \delta \boldsymbol{x}_i} = \begin{bmatrix} -\boldsymbol{R}^{b_i}_w & -\boldsymbol{R}^{b_i}_w \Delta t & \lfloor \boldsymbol{R}^{b_i}_w (\boldsymbol{p}^w_{b_{i+1}} - \boldsymbol{p}^w_{b_i} + \frac{1}{2} {}^w \boldsymbol{g} \Delta t^2 - {}^w \boldsymbol{v}_i \Delta t) \rfloor_\times & -\boldsymbol{J}^{\boldsymbol{\alpha}}_{b_i \boldsymbol{b}_a} & -\boldsymbol{J}^{\boldsymbol{\alpha}}_{b_i \boldsymbol{b}_g} \\ 0 & -\boldsymbol{R}^{b_i}_w & \lfloor \boldsymbol{R}^{b_i}_w ({}^w \boldsymbol{v}_{i+1} + {}^w \boldsymbol{g} \Delta t - {}^w \boldsymbol{v}_i) \rfloor_\times & -\boldsymbol{J}^{\boldsymbol{\beta}}_{b_i \boldsymbol{b}_a} & -\boldsymbol{J}^{\boldsymbol{\beta}}_{b_i \boldsymbol{b}_g} \\ 0 & 0 & -[\boldsymbol{q}^{w\,-1}_{b_i} \otimes \boldsymbol{q}^w_{b_{i+1}}]_L [\hat{\boldsymbol{\gamma}}^{b_i}_{b_{i+1}}]_R & 0 & [\hat{\boldsymbol{\gamma}}^{b_i}_{b_{i+1}} \otimes \boldsymbol{q}^{w\,-1}_{b_{i+1}} \otimes \boldsymbol{q}^w_{b_i}]_L \boldsymbol{J}^{\boldsymbol{\gamma}}_{b_i \boldsymbol{b}_g} \\ 0 & 0 & 0 & -\boldsymbol{I} & 0 \\ 0 & 0 & 0 & 0 & -\boldsymbol{I} \end{bmatrix}, \tag{A3}$$

where $[\cdot]_R$ is the right-quaternion-product matrices [67].

The Jacobian matrix of the IMU measurement residual with respect to the state increment $\delta \boldsymbol{x}_{i+1}$ is computed by

$$\frac{\partial \boldsymbol{r}_{\mathcal{B}}}{\partial \delta \boldsymbol{x}_{i+1}} = \begin{bmatrix} \boldsymbol{R}^{b_i}_w & 0 & 0 & 0 & 0 \\ 0 & \boldsymbol{R}^{b_i}_w & 0 & 0 & 0 \\ 0 & 0 & [\boldsymbol{q}^{w\,-1}_{b_i} \otimes \boldsymbol{q}^w_{b_{i+1}} \otimes (\hat{\boldsymbol{\gamma}}^{b_i}_{b_{i+1}})^{-1}]_L & 0 & 0 \\ 0 & 0 & 0 & \boldsymbol{I} & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{I} \end{bmatrix}. \tag{A4}$$

Appendix A.2.2. Jacobian Matrices of Event Data Residual

The Jacobian matrices of the event measurement residual with respect to system state increments $\delta \boldsymbol{x}_j$ and $\delta \boldsymbol{x}_l$ can be solved by the chain rule:

$$\left[\frac{\partial \boldsymbol{r}_{\mathcal{D},\mathcal{T}}}{\partial \delta \boldsymbol{x}_j}, \frac{\partial \boldsymbol{r}_{\mathcal{D},\mathcal{T}}}{\partial \delta \boldsymbol{x}_l} \right] = \sum_{m \in \mathcal{D}} \frac{\partial \boldsymbol{r}_{\mathcal{D},\mathcal{T}}}{\partial ({}^{cp_l} u_m, {}^{cp_l} v_m)} \cdot \frac{\partial ({}^{cp_l} u_m, {}^{cp_l} v_m)}{\partial {}^{cl} \boldsymbol{P}_m} \cdot \left[\frac{\partial {}^{cl} \boldsymbol{P}_m}{\partial \delta \boldsymbol{x}_j}, \frac{\partial {}^{cl} \boldsymbol{P}_m}{\partial \delta \boldsymbol{x}_l} \right]. \tag{A5}$$

The first part in Equation (A5), $\frac{\partial \boldsymbol{r}_{\mathcal{D},\mathcal{T}}}{\partial ({}^{cp_l} u_m, {}^{cp_l} v_m)}$, can be considered as the gradient at the pixel location $({}^{cp_l} u_m, {}^{cp_l} v_m)$ in the l-th negative TS image:

$$\frac{\partial \boldsymbol{r}_{\mathcal{D},\mathcal{T}}}{\partial ({}^{cp_l} u_m, {}^{cp_l} v_m)} = \begin{bmatrix} \nabla u_{({}^{cp_l} u_m, {}^{cp_l} v_m)} & \nabla v_{({}^{cp_l} u_m, {}^{cp_l} v_m)} \end{bmatrix}. \tag{A6}$$

Additionally, the second part $\frac{\partial({}^{cp_l}u_m, {}^{cp_l}v_m)}{\partial {}^{c_l}\boldsymbol{P}_m}$ can be regarded as the derivative result of the back projection function:

$$\frac{\partial({}^{cp_l}u_m, {}^{cp_l}v_m)}{\partial {}^{c_l}\boldsymbol{P}_m} = \begin{bmatrix} \frac{\boldsymbol{\mathcal{P}}_{(1,1)}}{{}^{c_l}z_m} & 0 & -\frac{\boldsymbol{\mathcal{P}}_{(1,1)} \cdot {}^{c_l}x_m}{({}^{c_l}z_m)^2} \\ 0 & \frac{\boldsymbol{\mathcal{P}}_{(2,2)}}{{}^{c_l}z_m} & -\frac{\boldsymbol{\mathcal{P}}_{(1,1)} \cdot {}^{c_l}y_m}{({}^{c_l}z_m)^2} \end{bmatrix}, \tag{A7}$$

where $\boldsymbol{\mathcal{P}}$ is the projection matrix of the event camera.

Finally, the third part of Equation (A5) can be, respectively, defined by

$$\begin{aligned} \frac{\partial {}^{c_l}\boldsymbol{P}_m}{\partial \delta \boldsymbol{x}_j} &= \begin{bmatrix} \boldsymbol{R}_b^c \boldsymbol{R}_w^{b_l} & 0 & -\boldsymbol{R}_b^c \boldsymbol{R}_w^{b_l} \boldsymbol{R}_{b_j}^w \lfloor \boldsymbol{R}_c^b ({}^{c_j}\boldsymbol{P}_m) + \boldsymbol{p}_c^b \rfloor_\times & 0 & 0 \end{bmatrix}, \\ \frac{\partial {}^{c_l}\boldsymbol{P}_m}{\partial \delta \boldsymbol{x}_l} &= \begin{bmatrix} -\boldsymbol{R}_b^c \boldsymbol{R}_w^{b_l} & 0 & \boldsymbol{R}_b^c \lfloor \boldsymbol{R}_w^{b_l} (\boldsymbol{R}_{b_j}^w (\boldsymbol{R}_c^b ({}^{c_j}\boldsymbol{P}_m) + \boldsymbol{p}_c^b) + \boldsymbol{p}_{b_j}^w - \boldsymbol{p}_w^{b_l}) \rfloor_\times & 0 & 0 \end{bmatrix}. \end{aligned} \tag{A8}$$

References

1. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. [CrossRef]
2. Gomez-Ojeda, R.; Moreno, F.A.; Zuñiga-Noël, D.; Scaramuzza, D.; Gonzalez-Jimenez, J. PL-SLAM: A stereo SLAM system through the combination of points and line segments. *IEEE Trans. Robot.* **2019**, *35*, 734–746. [CrossRef]
3. Wang, R.; Schworer, M.; Cremers, D. Stereo DSO: Large-scale direct sparse visual odometry with stereo cameras. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 3903–3911.
4. Huang, H.; Yeung, S.K. 360vo: Visual odometry using a single 360 camera. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 5594–5600.
5. Fontan, A.; Civera, J.; Triebel, R. Information-driven direct rgb-d odometry. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 4929–4937.
6. Gallego, G.; Delbruck, T.; Orchard, G.; Bartolozzi, C.; Taba, B.; Censi, A.; Leutenegger, S.; Davison, A.; Conradt, J.; Daniilidis, K.; et al. Event-based vision: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *44*, 154–180. [CrossRef]
7. Mostafavi, M.; Wang, L.; Yoon, K.J. Learning to reconstruct HDR images from events, with applications to depth and flow prediction. *Int. J. Comput. Vis.* **2021**, *129*, 900–920. [CrossRef]
8. Zhou, Y.; Gallego, G.; Shen, S. Event-based stereo visual odometry. *IEEE Trans. Robot.* **2021**, *37*, 1433–1450. [CrossRef]
9. Mueggler, E.; Huber, B.; Scaramuzza, D. Event-based, 6-DoF pose tracking for high-speed maneuvers. In Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Chicago, IL, USA, 14–18 September 2014; pp. 2761–2768.
10. Kim, H.; Leutenegger, S.; Davison, A.J. Real-time 3D reconstruction and 6-DoF tracking with an event camera. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016; pp. 349–364.
11. Rebecq, H.; Horstschäfer, T.; Gallego, G.; Scaramuzza, D. EVO: A geometric approach to event-based 6-DoF parallel tracking and mapping in real time. *IEEE Robot. Autom. Lett.* **2016**, *2*, 593–600. [CrossRef]
12. Bryner, S.; Gallego, G.; Rebecq, H.; Scaramuzza, D. Event-based, direct camera tracking from a photometric 3D map using nonlinear optimization. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 325–331.
13. Leutenegger, S.; Lynen, S.; Bosse, M.; Siegwart, R.; Furgale, P. Keyframe-based visual–inertial odometry using nonlinear optimization. *Int. J. Robot. Res.* **2015**, *34*, 314–334. [CrossRef]
14. Bloesch, M.; Omari, S.; Hutter, M.; Siegwart, R. Robust visual inertial odometry using a direct EKF-based approach. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots And Systems (IROS), Hamburg, Germany, 28 September–2 October 2015; pp. 298–304.
15. Qin, T.; Li, P.; Shen, S. VINS-Mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [CrossRef]
16. Von Stumberg, L.; Usenko, V.; Cremers, D. Direct sparse visual-inertial odometry using dynamic marginalization. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 2510–2517.
17. Usenko, V.; Demmel, N.; Schubert, D.; Stückler, J.; Cremers, D. Visual-inertial mapping with non-linear factor recovery. *IEEE Robot. Autom. Lett.* **2019**, *5*, 422–429. [CrossRef]
18. Von Stumberg, L.; Cremers, D. DM-VIO: Delayed Marginalization Visual-Inertial Odometry. *IEEE Robot. Autom. Lett.* **2022**, *7*, 1408–1415. [CrossRef]
19. Zihao Zhu, A.; Atanasov, N.; Daniilidis, K. Event-based visual inertial odometry. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5391–5399.

20. Rebecq, H.; Horstschaefer, T.; Scaramuzza, D. Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization. In Proceedings of the British Machine Vision Conference, London, UK, 4–7 September 2017.
21. Vidal, A.R.; Rebecq, H.; Horstschaefer, T.; Scaramuzza, D. Ultimate SLAM? Combining events, images, and IMU for robust visual SLAM in HDR and high-speed scenarios. *IEEE Robot. Autom. Lett.* **2018**, *3*, 994–1001. [CrossRef]
22. Mueggler, E.; Gallego, G.; Rebecq, H.; Scaramuzza, D. Continuous-time visual-inertial odometry for event cameras. *IEEE Trans. Robot.* **2018**, *34*, 1425–1440. [CrossRef]
23. Le Gentil, C.; Tschopp, F.; Alzugaray, I.; Vidal-Calleja, T.; Siegwart, R.; Nieto, J. Idol: A framework for imu-dvs odometry using lines. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October–24 January 2021; pp. 5863–5870.
24. Chen, P.; Guan, W.; Lu, P. ESVIO: Event-based Stereo Visual Inertial Odometry. *arXiv* **2022**, arXiv:2212.13184.
25. Zhou, Y.; Gallego, G.; Rebecq, H.; Kneip, L.; Li, H.; Scaramuzza, D. Semi-dense 3D reconstruction with a stereo event camera. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 235–251.
26. Jiao, J.; Huang, H.; Li, L.; He, Z.; Zhu, Y.; Liu, M. Comparing Representations in Tracking for Event Camera-based SLAM. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Virtual Conference, 19–25 June 2021; pp. 1369–1376.
27. Zhu, A.Z.; Thakur, D.; Özaslan, T.; Pfrommer, B.; Kumar, V.; Daniilidis, K. The multivehicle stereo event camera dataset: An event camera dataset for 3D perception. *IEEE Robot. Autom. Lett.* **2018**, *3*, 2032–2039. [CrossRef]
28. Gehrig, M.; Aarents, W.; Gehrig, D.; Scaramuzza, D. Dsec: A stereo event camera dataset for driving scenarios. *IEEE Robot. Autom. Lett.* **2021**, *6*, 4947–4954. [CrossRef]
29. Corke, P.; Lobo, J.; Dias, J. An introduction to inertial and visual sensing. *Int. J. Robot. Res.* **2007**, *26*, 519–535. [CrossRef]
30. Weiss, S.; Achtelik, M.W.; Lynen, S.; Chli, M.; Siegwart, R. Real-time onboard visual-inertial state estimation and self-calibration of MAVs in unknown environments. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation, St Paul, MN, USA, 14–18 May 2012; pp. 957–964.
31. Lynen, S.; Achtelik, M.W.; Weiss, S.; Chli, M.; Siegwart, R. A robust and modular multi-sensor fusion approach applied to MAV navigation. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 3923–3929.
32. Mourikis, A.I.; Roumeliotis, S.I. A multi-state constraint Kalman filter for vision-aided inertial navigation. In Proceedings of the 2007 IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2007; pp. 3565–3572.
33. Yang, Y.; Geneva, P.; Eckenhoff, K.; Huang, G. Visual-Inertial Odometry with Point and Line Features. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, 4–8 November 2019; pp. 2447–2454.
34. Yang, Z.; Shen, S. Monocular visual–inertial state estimation with online initialization and camera–IMU extrinsic calibration. *IEEE Trans. Autom. Sci. Eng.* **2016**, *14*, 39–51. [CrossRef]
35. Mur-Artal, R.; Tardós, J.D. Visual-inertial monocular SLAM with map reuse. *IEEE Robot. Autom. Lett.* **2017**, *2*, 796–803. [CrossRef]
36. Liu, Z.; Shi, D.; Li, R.; Qin, W.; Zhang, Y.; Ren, X. PLC-VIO: Visual-Inertial Odometry Based on Point-Line Constraints. *IEEE Trans. Autom. Sci. Eng.* **2021**, *19*, 1880–1897. [CrossRef]
37. Mo, J.; Sattar, J. Continuous-time spline visual-inertial odometry. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 9492–9498.
38. He, Y.; Zhao, J.; Guo, Y.; He, W.; Yuan, K. PL-VIO: Tightly-coupled monocular visual–inertial odometry using point and line features. *Sensors* **2018**, *18*, 1159. [CrossRef]
39. Campos, C.; Elvira, R.; Rodríguez, J.J.G.; Montiel, J.M.; Tardós, J.D. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Trans. Robot.* **2021**, *37*, 1874–1890. [CrossRef]
40. Xu, B.; Wang, P.; He, Y.; Chen, Y.; Chen, Y.; Zhou, M. Leveraging structural information to improve point line visual-inertial odometry. *IEEE Robot. Autom. Lett.* **2022**, *7*, 3483–3490. [CrossRef]
41. Huang, G. Visual-inertial navigation: A concise review. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 9572–9582.
42. Usenko, V.; Engel, J.; Stückler, J.; Cremers, D. Direct visual-inertial odometry with stereo cameras. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; pp. 1885–1892.
43. Eckenhoff, K.; Geneva, P.; Huang, G. Direct visual-inertial navigation with analytical preintegration. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 1429–1435.
44. Hidalgo-Carrió, J.; Gallego, G.; Scaramuzza, D. Event-aided Direct Sparse Odometry. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 5781–5790.
45. Kueng, B.; Mueggler, E.; Gallego, G.; Scaramuzza, D. Low-latency visual odometry using event-based feature tracks. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Republic of Korea, 9–14 October 2016; pp. 16–23.
46. Nguyen, A.; Do, T.T.; Caldwell, D.G.; Tsagarakis, N.G. Real-time 6-DoF pose relocalization for event cameras with stacked spatial LSTM networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, CA, USA, 16–17 June 2019.
47. Zhou, Y.; Li, H.; Kneip, L. Canny-VO: Visual Odometry with RGB-D Cameras Based on Geometric 3D-2D Edge Alignment. *IEEE Trans. Robot.* **2018**, *35*, 184–199. [CrossRef]

48. Campos, C.; Montiel, J.M.; Tardós, J.D. Inertial-only optimization for visual-inertial initialization. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 51–57.
49. Martinelli, A. Closed-form solution of visual-inertial structure from motion. *Int. J. Comput. Vis.* **2014**, *106*, 138–152. [CrossRef]
50. Kaiser, J.; Martinelli, A.; Fontana, F.; Scaramuzza, D. Simultaneous state initialization and gyroscope bias calibration in visual inertial aided navigation. *IEEE Robot. Autom. Lett.* **2016**, *2*, 18–25. [CrossRef]
51. Campos, C.; Montiel, J.M.; Tardós, J.D. Fast and robust initialization for visual-inertial SLAM. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 1288–1294.
52. Liu, H.; Qiu, J.; Huang, W. Integrating Point and Line Features for Visual-Inertial Initialization. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 9470–9476.
53. Qin, T.; Shen, S. Robust initialization of monocular visual-inertial estimation on aerial robots. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 4225–4232.
54. Huang, W.; Liu, H. Online initialization and automatic camera-IMU extrinsic calibration for monocular visual-inertial SLAM. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 5182–5189.
55. Li, J.; Bao, H.; Zhang, G. Rapid and robust monocular visual-inertial initialization with gravity estimation via vertical edges. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macao, 4–8 November 2019; pp. 6230–6236.
56. Huang, W.; Wan, W.; Liu, H. Optimization-based online initialization and calibration of monocular visual-inertial odometry considering spatial-temporal constraints. *Sensors* **2021**, *21*, 2673. [CrossRef]
57. Lupton, T.; Sukkarieh, S. Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions. *IEEE Trans. Robot.* **2011**, *28*, 61–76. [CrossRef]
58. Hirschmuller, H. Stereo processing by semiglobal matching and mutual information. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *30*, 328–341. [CrossRef]
59. Akinlar, C.; Topal, C. EDLines: A real-time line segment detector with a false detection control. *Pattern Recognit. Lett.* **2011**, *32*, 1633–1642. [CrossRef]
60. Suzuki, S. Topological structural analysis of digitized binary images by border following. *Comput. Vision, Graph. Image Process.* **1985**, *30*, 32–46. [CrossRef]
61. Huber, P.J. Robust estimation of a location parameter. In *Breakthroughs in Statistics*; Springer: Cham, Switzerland, 1992; pp. 492–518.
62. Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A.Y. ROS: An open-source Robot Operating System. In Proceedings of the ICRA Workshop on Open Source Software, Kobe, Japan, 12–17 May 2009; Volume 3, p. 5.
63. Rebecq, H.; Gehrig, D.; Scaramuzza, D. ESIM: An open event camera simulator. In Proceedings of the Conference on Robot Learning, Zürich, Switzerland, 29–31 October 2018; pp. 969–982.
64. Umeyama, S. Least-squares estimation of transformation parameters between two point patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **1991**, *13*, 376–380. [CrossRef]
65. Mueggler, E.; Rebecq, H.; Gallego, G.; Delbruck, T.; Scaramuzza, D. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM. *Int. J. Robot. Res.* **2017**, *36*, 142–149. [CrossRef]
66. Ghosh, S.; Gallego, G. Multi-Event-Camera Depth Estimation and Outlier Rejection by Refocused Events Fusion. *Adv. Intell. Syst.* **2022**, *4*, 2200221. [CrossRef]
67. Sola, J. Quaternion kinematics for the error-state Kalman filter. *arXiv* **2017**, arXiv:1711.02508.