Motion and Structure from Event-based Normal Flow –Supplementary Material–

Zhongyang Ren[1*], Bangyan Liao[2*], Delei Kong[1], Jinghang Li[1], Peidong Liu[2], Laurent Kneip[3], Guillermo Gallego[4], and Yi Zhou[1✉]

[1] School of Robotics, Hunan University
[2] School of Engineering, Westlake University
[3] School of Information Science and Technology, ShanghaiTech University
[4] TU Berlin, ECDF, SCIoI Excellence Cluster and Robotics Institute Germany

Overview

In this supplementary material we provide the following:

- A detailed review of preliminaries involved, including motion field, differential geometry and event-based normal flow (Sec. I).
- Additional details about differential homography (Sec. II).
- Details of dataset and evaluation metrics utilized (Sec. III).
- Implementation details of the proposed algorithms (Sec. IV).
- More evaluations on the proposed methods, including an analysis of numerical stability, and the performance of the continuous-time nonlinear solver in textured scenes. (Sec. V).

I Supplement to Preliminaries

We supplement Sec. 3 of the main paper by providing a detailed review on three important concepts, including *motion field* (Sec. I.A), *differential geometry* across multiple views (Sec. I.B), and *event-based normal flow* (Sec. I.C).

I.A Motion Field

Consider a visual observer moving in a static environment, and let its instantaneous angular velocity and linear velocity be denoted in the observer's body frame as ${}^{\mathrm{B}}\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)^\top$ and ${}^{\mathrm{B}}\boldsymbol{\nu} = (\nu_x, \nu_y, \nu_z)^\top$, respectively. The scene flow of a 3D point ${}^{\mathrm{B}}\mathbf{P} = (X, Y, Z)^\top$ is

$$ {}^{\mathrm{B}}\dot{\mathbf{P}} = -{}^{\mathrm{B}}\boldsymbol{\omega} \times {}^{\mathrm{B}}\mathbf{P} - {}^{\mathrm{B}}\boldsymbol{\nu}. \tag{1} $$

Let $\mathbf{x} = (x, y)^\top = (X/Z, Y/Z)^\top$ be the image of $\mathbf{P}$ represented in calibrated camera coordinates. Projecting the scene flow vector onto the image plane gives

∗ equal contribution; ✉ corresponding author (eeyzhou@hnu.edu.cn).

us the relationship between the optical flow $\mathbf{u}$ (in calibrated coordinates) and the dynamics of the observer as

$$\mathbf{u} = \frac{1}{Z}\mathbf{A}(\mathbf{x})\,{}^{\mathtt{B}}\boldsymbol{\nu} + \mathbf{B}(\mathbf{x})\,{}^{\mathtt{B}}\boldsymbol{\omega}, \tag{2}$$

$$\mathbf{A}(\mathbf{x}) = \begin{bmatrix} -1 & 0 & x \\ 0 & -1 & y \end{bmatrix}, \tag{3}$$

$$\mathbf{B}(\mathbf{x}) = \begin{bmatrix} xy & -(1+x^2) & y \\ (1+y^2) & -xy & -x \end{bmatrix}, \tag{4}$$

where (2) is acknowledged as the motion field equation [4]. The depth Z is different for every pixel $\mathbf{x}$, hence we often write $Z = Z(\mathbf{x})$.

I.B Multi-View Differential Geometry

The theory of differential geometry across multiple views can be derived by taking the temporal derivatives of the standard multi-view geometry [5]. It examines the relationship between the optical flow and the instantaneous kinematics across multiple views. We focus on two specific two-view geometric properties: 1) the differential homography [9], and 2) the differential epipolar geometry [8].

Differential Homography. The standard homography is a projective transformation that describes the image motion between two views of a camera observing a planar scene. Such a mapping function, induced by the planar scene, can be expressed as

$$\hat{\mathbf{x}}' = \mathbf{H}\hat{\mathbf{x}}, \tag{5}$$

where $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ denote the corresponding pixels in terms of homogeneous calibrated image coordinates, and $\mathbf{H} \in \mathbb{R}^{3\times 3}$ is the homography matrix. The induced homography is a function of the relative pose $\{\mathtt{R}, \mathbf{t}\}$ and the structure parameters $\{\mathbf{N}, d\}$ (i.e., parameters of the plane in the scene), and can be expressed as $\mathbf{H} \doteq \mathtt{R} - \frac{1}{d}\mathbf{t}\mathbf{N}^\top$. The differential version replaces the potentially large displacement of the transformation $\mathbf{x} \mapsto \mathbf{x}'$ by its first-order continuous-time approximation $\mathbf{x} \mapsto \mathbf{x} + \Delta t\hat{\mathbf{u}}$, with point velocity given by:

$$\hat{\mathbf{u}}(\mathbf{x}) = \left(\mathbf{1} - \hat{\mathbf{x}}\mathbf{e}_3^\top\right)\mathbf{H}_d\hat{\mathbf{x}}. \tag{6}$$

Here, $\mathbf{H}_d \doteq -\left([\boldsymbol{\omega}]_\times + \frac{1}{d}\boldsymbol{\nu}\mathbf{N}^\top\right)$ denotes the differential homography matrix ($[\cdot]_\times$ denotes the cross-product, skew-symmetric matrix), $\mathbf{e}_3 = (0,0,1)^\top$, $\hat{\mathbf{x}} = (x, y, 1)^\top$, and $\hat{\mathbf{u}} = (u_x, u_y, 0)^\top$ the optical flow (in homogeneous coordinates).

Differential Epipolar Geometry. As a relatively relaxed constraint compared to the homography, the standard epipolar geometry only defines a potential direction for searching the correspondence pair $\hat{\mathbf{x}} \leftrightarrow \hat{\mathbf{x}}'$. Such a constraint evaluates the distance between a potential match to the determined epipolar line, and can be expressed as

$$\hat{\mathbf{x}}'^\top\mathbf{E}\hat{\mathbf{x}} = 0. \tag{7}$$

The differential epipolar geometry can be derived similarly, yielding

$$\hat{\mathbf{u}}(\mathbf{x})^\top [\boldsymbol{\nu}]_\times \hat{\mathbf{x}} - \hat{\mathbf{x}}^\top \mathbf{s} \hat{\mathbf{x}} = 0, \tag{8}$$

where $\mathbf{s} = \frac{1}{2}([\boldsymbol{\nu}]_\times [\boldsymbol{\omega}]_\times + [\boldsymbol{\omega}]_\times [\boldsymbol{\nu}]_\times)$. Note that (8) can be derived directly from (2) by eliminating the depth information.

I.C Event-Based Normal Flow

Normal Flow. While the motion field is the true projection of the 3D scene velocities on the image plane, its determination from visual quantities measured at the image plane is called "optical flow". The optical flow is the apparent motion of patterns on the image plane. As is well known, the optical flow can only be a good approximation for the true motion (i.e., the motion field) at image points that are unambiguous to track, i.e., that are surrounded by edge patterns that accurately determine their location in both image plane directions (x and y). A common assumption that is leveraged to compute optical flow is "brightness constancy", that is, the approximation that image brightness remains constant as the image point $\mathbf{x}$ moves. This is an approximation for points corresponding to matt and Lambertian objects; it is not a sensible assumptions for points on shiny and specular objects.

Letting $I(\mathbf{x}, t)$ be the brightness function on the image plane as a function of time, the brightness constancy assumption can be compactly written as:

$$I(\mathbf{x}(t), t) = \text{const.} \tag{9}$$

A first-order Taylor expansion gives

$$I(\mathbf{x} + \Delta\mathbf{x}, t + \Delta t) \approx I(\mathbf{x}, t) + (\nabla I(\mathbf{x}, t))^\top \dot{\mathbf{x}} \Delta t + \partial_t I(\mathbf{x}, t) \Delta t, \tag{10}$$

where $\dot{\mathbf{x}} \equiv \dot{\mathbf{x}}(t) = (dx/dt, dy/dt)^\top$ is the velocity of image point $\mathbf{x}(t)$, $\nabla = (\partial_x, \partial_y)^\top$ are the spatial derivatives and ∂t is the temporal derivative. Moving terms around,

$$I(\mathbf{x} + \Delta\mathbf{x}, t + \Delta t) - I(\mathbf{x}, t) \approx (\nabla I(\mathbf{x}, t))^\top \dot{\mathbf{x}} \Delta t + \partial_t I(\mathbf{x}, t) \Delta t. \tag{11}$$

Brightness constancy states that for the true motion curves $\mathbf{x}(t)$, the intensity remains the same, $I(\mathbf{x} + \Delta\mathbf{x}, t + \Delta t) \approx I(\mathbf{x}, t)$, and therefore (since $\Delta t > 0$)

$$(\nabla I(\mathbf{x}, t))^\top \dot{\mathbf{x}} + \partial_t I(\mathbf{x}, t) \approx 0. \tag{12}$$

Assuming that the brightness I is known (i.e., measured by a camera), (12) is one equation in two unknowns ($\dot{\mathbf{x}} \in \mathbb{R}^2$). More equations are needed to be able to determine the velocity $\dot{\mathbf{x}}$ at point $(\mathbf{x}, t)$ in space-time. Without additional equations or information, all that can be determined is the velocity component that is parallel to $\nabla I(\mathbf{x}, t)$ (i.e., perpendicular to the edge) due to the dot product

operation. That is, decomposing $\dot{\mathbf{x}} = \dot{\mathbf{x}}_{\|} + \dot{\mathbf{x}}_{\perp}$ into its parallel and perpendicular components to the local edge ∇I, respectively, we have $\nabla I(\mathbf{x},t)^{\top}\dot{\mathbf{x}}_{\|} = 0$. Substituting in (12) gives:

$$0 \approx (\nabla I(\mathbf{x},t))^{\top}\left(\dot{\mathbf{x}}_{\|} + \dot{\mathbf{x}}_{\perp}\right) + \partial_t I(\mathbf{x},t) \tag{13}$$

$$= \underbrace{(\nabla I(\mathbf{x},t))^{\top}\dot{\mathbf{x}}_{\|}}_{0} + (\nabla I(\mathbf{x},t))^{\top}\dot{\mathbf{x}}_{\perp} + \partial_t I(\mathbf{x},t) \tag{14}$$

$$= (\nabla I(\mathbf{x},t))^{\top}\dot{\mathbf{x}}_{\perp} + \partial_t I(\mathbf{x},t) \tag{15}$$

From here, we can work out $\mathbf{n} \equiv \dot{\mathbf{x}}_{\perp} \propto \nabla I$. We know that

$$\mathbf{n} \equiv \dot{\mathbf{x}}_{\perp} = \|\dot{\mathbf{x}}_{\perp}\|\frac{\nabla I}{\|\nabla I\|}. \tag{16}$$

Substituting in (15) (and omitting the evaluation point $(\mathbf{x},t)$, for clarity),

$$(\nabla I)^{\top}\|\dot{\mathbf{x}}_{\perp}\|\frac{\nabla I}{\|\nabla I\|} = -\partial_t I, \tag{17}$$

which gives

$$\|\dot{\mathbf{x}}_{\perp}\| = -\partial_t I\frac{\|\nabla I\|}{(\nabla I)^{\top}\nabla I} = -\partial_t I\frac{1}{\|\nabla I\|}. \tag{18}$$

Substituting in (16), it finally gives

$$\mathbf{n}(\mathbf{x},t) = -\frac{\partial_t I(\mathbf{x},t)}{\|\nabla I(\mathbf{x},t)\|^2}\nabla I(\mathbf{x},t). \tag{19}$$

This component of the optical flow $\dot{\mathbf{x}}(\mathbf{x},t)$, which is perpendicular to the edge $\nabla I(\mathbf{x},t)$, is called *normal flow*. While (19) is well-known in conventional (frame-based) computer vision, it is not directly applicable to event cameras because the latter do not directly provide the means to compute all terms required (the spatial and temporal derivatives of the brightness). Next, we present how to compute the normal flow in the case of event cameras.

Computation of Event-based Normal Flow. The output of an event camera is a stream of events, where each event $\boldsymbol{e}_k \doteq (\mathbf{x}_k, t_k, p_k)$ consists of the space-time coordinates $(\mathbf{x}_k, t_k)$ at which the intensity change of predefined size happened and the sign of the change (*i.e.*, polarity $p_k \in \{+1, -1\}$). Following the definition in [1], we utilize the differential mapping function $\Sigma_e : \mathbb{R}^2 \to \mathbb{R}$ that maps the pixel coordinate to the latest event's timestamp, i.e., $\mathbf{x} \mapsto t_{\text{last}}(\mathbf{x})$. Σ_e is a 2D image, also called a time map; and it is often referred to as *time surface* (TS) because when interpreting the image as an elevation map, a moving edge produces events and such points $(\mathbf{x}_k, t_{\text{last}}(\mathbf{x}_k)) \subset \mathbb{R}^3$ approximately form a surface (ignoring the discrete pixel lattice) [1, 3]. It also goes by the name of *Surface of Active Events* (SAE).

Lemma. Given the mapping function Σ_e at pixel coordinate $\mathbf{x}$ and its finite neighbourhood, the direction of the first-order partial derivative $\nabla\Sigma_e(\mathbf{x}) = \frac{\partial\Sigma_e}{\partial\mathbf{x}}(\mathbf{x})$ is identical to that of the normal flow at $\mathbf{x}$.

Proof. The directional derivative $\nabla_{\mathbf{d}}\Sigma_e(\mathbf{x})$ can be defined by the dot product of direction $\mathbf{d}$ and $\nabla\Sigma_e(\mathbf{x})$, as $\nabla_{\mathbf{d}}\Sigma_e(\mathbf{x}) \doteq \mathbf{d}\cdot\nabla\Sigma_e(\mathbf{x})$, whose maximum is achieved when $\mathbf{d}_{\max} = \frac{\nabla\Sigma_e(\mathbf{x})}{\|\nabla\Sigma_e(\mathbf{x})\|}$. The direction $\mathbf{d}_{\max}$ is proved to be perpendicular to the level set $\mathcal{S} \doteq \{\mathbf{x}|\Sigma_e(\mathbf{x}) = t\}$ which corresponds to the latent edge pixels at time t. Hence, direction $\mathbf{d}_{\max}$ is in parallel to the direction of local image gradient, namely the normal flow's direction. □

The lemma demonstrates how to determine the direction of normal flow, but there is still a unknown scale factor between the magnitude of $\mathbf{d}_{\max}$ and that of the normal flow.

Proposition. Given the mapping function Σ_e at pixel coordinate $\mathbf{x}$ and its finite neighbourhood, and the first-order partial derivative $\nabla\Sigma_e(\mathbf{x})$, normal flow $\mathbf{n}(\mathbf{x})$ at pixel $\mathbf{x}$ can be calculated by

$$\mathbf{n}(\mathbf{x}) = \frac{\nabla\Sigma_e(\mathbf{x})}{\|\nabla\Sigma_e(\mathbf{x})\|^2}. \tag{20}$$

Proof. Direction $\mathbf{d}_{\max}$ owns the algebra meaning: the maximum time increase magnitude $\Delta t_{\max} = \mathbf{d}_{\max}\cdot\nabla\Sigma_e(\mathbf{x}) = \|\nabla\Sigma_e(\mathbf{x})\|$ given a unit pixel displacement ($\|\mathbf{d}_{\max}\| = 1$). Since the magnitude of the normal flow is equal to the pixel displacement per unit time, dividing $\mathbf{d}_{\max}$ by $\Delta t_{\max}$ is the normal flow, i.e., $\mathbf{n}(\mathbf{x}) = \frac{\mathbf{d}_{\max}}{\Delta t_{\max}} = \frac{\nabla\Sigma_e(\mathbf{x})}{\|\nabla\Sigma_e(\mathbf{x})\|^2}$. □

The above normal flow calculation requires the smooth and differentiable mapping function, we can approximate its partial derivative operator with the modern image processing techniques such as *Sobel filter* or *local plane fitting*. All above-mentioned geometric elements are illustrated in Fig. 1. Based on this, we present the normal flow constraint $\mathbf{n}(\mathbf{x})^\top\mathbf{u}(\mathbf{x}) = \|\mathbf{n}(\mathbf{x})\|^2$ for robust estimation of motion and structure parameters in Sec. 4 of our paper.

II More About Differential Homography

In this section, a detailed explanation to Sec. 4 of our paper is added. In particular, we first discuss the way to recover the true differential homography from our linear solution (Sec. II.A). Then, we disclose how to retrieve the motion and scene parameters by decomposing differential homography (Sec. II.B).

II.A Recover the True Differential Homography from Our Linear Solution

As is well-known [5,9], Eq. (6) can only recover $\mathbf{H}_L = \mathbf{H}_d + \epsilon\mathbf{I}$ with an unknown scale ϵ since $\mathbf{H}_d$ has a one-dimensional null space. So our linear solver faces

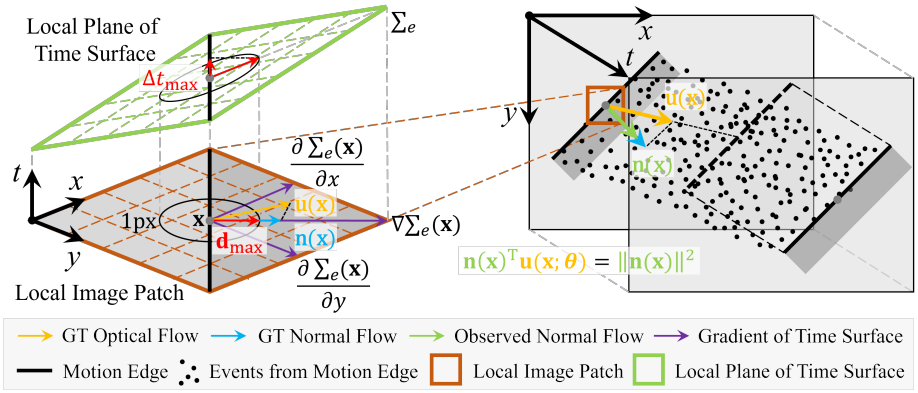


Fig. 1: Illustration of event-based normal flow, explained on the spatio-temporal profile induced by a moving edge. Given a pixel coordinate $\mathbf{x}$, when the increment $\Delta\mathbf{x}$'s direction $\mathbf{d}$ equals $\mathbf{d}_{\max}$ ($\mathbf{d}_{\max}$ is the one that maximizes the directional derivative $\nabla_{\mathbf{d}}\Sigma_e(\mathbf{x}) = \mathbf{d}\cdot\nabla\Sigma_e(\mathbf{x})$, i.e., $\mathbf{d}_{\max} = \frac{\nabla\Sigma_e(\mathbf{x})}{\|\nabla\Sigma_e(\mathbf{x})\|}$), it aligns with the direction of the time surface gradient $\nabla\Sigma_e(\mathbf{x})$, thereby determining the direction of the normal flow $\mathbf{n}(\mathbf{x})$. Since the time increment (lifetime) corresponding to $\mathbf{d}_{\max}$ is $\Delta t_{\max} = \mathbf{d}_{\max}\cdot\nabla\Sigma_e(\mathbf{x}) = \|\nabla\Sigma_e(\mathbf{x})\|$, the normal flow $\mathbf{n}(\mathbf{x}) = \frac{\mathbf{d}_{\max}}{\Delta t_{\max}} = \frac{\nabla\Sigma_e(\mathbf{x})}{\|\nabla\Sigma_e(\mathbf{x})\|^2}$.

this problem too, here we could recover the true differential homography matrix as follows: after solving for $\mathbf{H}_L$, let $\mathbf{M}_L = \mathbf{H}_L + \mathbf{H}_L^\top$, then the second largest eigenvalue of $\mathbf{M}_L$ equals 2ϵ, which enabling recovery of differential homography matrix by $\mathbf{H}_d = \mathbf{H}_L - \epsilon\mathbf{I}$.

Proof. Let

$$\begin{aligned}\mathbf{M}_L &= -\left(\frac{\boldsymbol{\nu}}{d}\mathbf{n}^\top + \mathbf{n}\frac{\boldsymbol{\nu}}{d}^\top\right) + 2\epsilon\mathbf{I} \\ &= -\mathbf{M} + 2\epsilon\mathbf{I}.\end{aligned} \tag{21}$$

We define the eigenvalue of $\mathbf{M}$ as $\lambda_{\max}, \lambda_{\text{mid}}, \lambda_{\min}$ with the decreasing order. Obviously $\mathbf{M}$ is a symmetric matrix, so according to the Rayleigh quotient theorem, for any vector $\{\mathbf{x}|\mathbf{x}\in\mathbb{R}^3, \|\mathbf{x}\| = 1\}$ we have

$$\lambda_{\min}\mathbf{x}^\top\mathbf{x} \le \mathbf{x}^\top\mathbf{M}\mathbf{x} \le \lambda_{\max}\mathbf{x}^\top\mathbf{x}. \tag{22}$$

For simplicity, we omit d and define a new variable $\mathbf{v} = \boldsymbol{\nu}/d$. Taking a vector $\mathbf{x}$ perpendicular to $(\mathbf{v}+\mathbf{n})$, and not perpendicular to $\mathbf{v}$ and $\mathbf{n}$ at the same time, we can prove that

$$\begin{aligned}\lambda_{\min}\mathbf{x}^\top\mathbf{x} &\le \mathbf{x}^\top\left(\mathbf{n}\mathbf{v}^\top + \mathbf{v}\mathbf{n}^\top\right)\mathbf{x} \\ &= \mathbf{x}^\top((\mathbf{v}+\mathbf{n})(\mathbf{v}+\mathbf{n})^\top - (\mathbf{v}\mathbf{v}^\top + \mathbf{n}\mathbf{n}^\top))\mathbf{x} \\ &= -\mathbf{x}^\top(\mathbf{v}\mathbf{v}^\top + \mathbf{n}\mathbf{n}^\top)\mathbf{x} \\ &= -[(\mathbf{x}^\top\mathbf{v})^2 + (\mathbf{x}^\top\mathbf{n})^2] < 0.\end{aligned} \tag{23}$$

And we know that the inner product of the vector itself $\mathbf{x}^\top\mathbf{x}$ is always positive. We can then conclude that $\lambda_{\min} < 0$.

Similarly, for another $\mathbf{x}$ perpendicular to $(\mathbf{v}-\mathbf{n})$, and not perpendicular to $\mathbf{v}$ and $\mathbf{n}$ at the same time we can prove that

$$\begin{aligned}\lambda_{\max}\mathbf{x}^\top\mathbf{x} &\geq \mathbf{x}^\top(\mathbf{n}\mathbf{v}^\top + \mathbf{v}\mathbf{n}^\top)\mathbf{x} \\ &= \mathbf{x}^\top((\mathbf{v}\mathbf{v}^\top + \mathbf{n}\mathbf{n}^\top) - (\mathbf{v}-\mathbf{n})(\mathbf{v}-\mathbf{n})^\top)\mathbf{x} \\ &= \mathbf{x}^\top(\mathbf{v}\mathbf{v}^\top + \mathbf{n}\mathbf{n}^\top)\mathbf{x} \\ &= [(\mathbf{x}^\top\mathbf{v})^2 + (\mathbf{x}^\top\mathbf{n})^2] > 0.\end{aligned} \tag{24}$$

Similarly, we can prove that $\lambda_{\max} > 0$.

For any rank deficient square matrix, the rank determine must equal to zero and there are at least one eigenvalue equals to zero. The rank of the $\mathbf{M}$ is less than two since it contains two outer product of vectors

$$\operatorname{rank}(\mathbf{M}) = \operatorname{rank}\left(\frac{\boldsymbol{\nu}}{d}\mathbf{n}^\top + \mathbf{n}\frac{\boldsymbol{\nu}}{d}^\top\right) \leq 2. \tag{25}$$

Besides, $\lambda_{\max} > 0$ and $\lambda_{\min} < 0$, we can come to the conclusion that

$$\lambda_{\text{mid}} = 0. \tag{26}$$

Consequently the second largest value of $\mathbf{M}_L = -\mathbf{M} + 2\epsilon\mathbf{I}$ equals to 2ϵ. □

II.B Retrieving Motion and Structure Parameters by Decomposing Differential Homography

After recovering the real differential homography matrix $\mathbf{H}_d$, we introduce the decomposition method to get the motion and structure parameters in this section.

Firstly, the eigen-decomposition of $\mathbf{M}$ can be written as:

$$\mathbf{M} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\top. \tag{27}$$

where $\mathbf{Q} = \{\mathbf{q}_{\max}, \mathbf{q}_{\text{mid}}, \mathbf{q}_{\min}\}$ is the orthogonal matrix which contains all eigenvectors of $\mathbf{M}$ and $\operatorname{diag}(\boldsymbol{\Lambda}) = \{\lambda_{\max}, \lambda_{\text{mid}}, \lambda_{\min}\}$ which collects all eigenvalues in the diagonal elements.

We then can define two auxiliary vectors $\mathbf{j}$ and $\mathbf{k}$

$$\begin{aligned}\mathbf{j} &= \sqrt{\frac{\lambda_{\max}}{2}}\mathbf{q}_{\max} + \sqrt{\frac{-\lambda_{\min}}{2}}\mathbf{q}_{\min}, \\ \mathbf{k} &= \sqrt{\frac{\lambda_{\max}}{2}}\mathbf{q}_{\max} - \sqrt{\frac{-\lambda_{\min}}{2}}\mathbf{q}_{\min},\end{aligned} \tag{28}$$

such that

$$\mathbf{M} = \mathbf{j}\mathbf{k}^\top + \mathbf{k}\mathbf{j}^\top. \tag{29}$$

Proof.

In Sec. II.A, we already have the conclusion that the egienvalues of $\mathbf{M}$ have following properties:

$$\lambda_{\max} > \lambda_{\mathrm{mid}} = 0 > \lambda_{\min}. \tag{30}$$

Then, we can convert Eq. (27) to

$$\begin{aligned}
\mathbf{M} &= \begin{bmatrix}\mathbf{q}_{\max} & \mathbf{q}_{\mathrm{mid}} & \mathbf{q}_{\min}\end{bmatrix} \begin{bmatrix}\lambda_{\max} & & \\ & 0 & \\ & & \lambda_{\min}\end{bmatrix} \begin{bmatrix}\mathbf{q}_{\max}{}^{\top} \\ \mathbf{q}_{\mathrm{mid}}{}^{\top} \\ \mathbf{q}_{\min}{}^{\top}\end{bmatrix} \\
&= \lambda_{\max}\mathbf{q}_{\max}\mathbf{q}_{\max}{}^{\top} + \lambda_{\min}\mathbf{q}_{\min}\mathbf{q}_{\min}{}^{\top} \\
&= \frac{\lambda_{\max}}{2}\mathbf{q}_{\max}\mathbf{q}_{\max}{}^{\top} + \frac{\lambda_{\max}}{2}\mathbf{q}_{\max}\mathbf{q}_{\max}{}^{\top} + \\
&\quad \frac{\lambda_{\min}}{2}\mathbf{q}_{\min}\mathbf{q}_{\min}{}^{\top} + \frac{\lambda_{\min}}{2}\mathbf{q}_{\min}\mathbf{q}_{\min}{}^{\top} \\
&= (\frac{\lambda_{\max}}{2}\mathbf{q}_{\max}\mathbf{q}_{\max}{}^{\top} + \sqrt{\frac{-\lambda_{\min}\lambda_{\max}}{4}}\mathbf{q}_{\min}\mathbf{q}_{\max}{}^{\top} \\
&\quad - \sqrt{\frac{-\lambda_{\min}\lambda_{\max}}{4}}\mathbf{q}_{\max}\mathbf{q}_{\min}{}^{\top} - \frac{-\lambda_{\min}}{2}\mathbf{q}_{\min}\mathbf{q}_{\min}{}^{\top}) \\
&\quad + (\frac{\lambda_{\max}}{2}\mathbf{q}_{\max}\mathbf{q}_{\max}{}^{\top} - \sqrt{\frac{-\lambda_{\min}\lambda_{\max}}{4}}\mathbf{q}_{\min}\mathbf{q}_{\max}{}^{\top} \\
&\quad + \sqrt{\frac{-\lambda_{\min}\lambda_{\max}}{4}}\mathbf{q}_{\max}\mathbf{q}_{\min}{}^{\top} - \frac{-\lambda_{\min}}{2}\mathbf{q}_{\min}\mathbf{q}_{\min}{}^{\top}).
\end{aligned} \tag{31}$$

It is easy to prove

$$\begin{aligned}
&\mathbf{j}\mathbf{k}^{\top} = \\
&(\frac{\lambda_{\max}}{2}\mathbf{q}_{\max}\mathbf{q}_{\max}{}^{\top} + \sqrt{\frac{-\lambda_{\min}\lambda_{\max}}{4}}\mathbf{q}_{\min}\mathbf{q}_{\max}{}^{\top} \\
&- \sqrt{\frac{-\lambda_{\min}\lambda_{\max}}{4}}\mathbf{q}_{\max}\mathbf{q}_{\min}{}^{\top} - \frac{-\lambda_{\min}}{2}\mathbf{q}_{\min}\mathbf{q}_{\min}{}^{\top}), \\
&\mathbf{k}\mathbf{j}^{\top} = \\
&(\frac{\lambda_{\max}}{2}\mathbf{q}_{\max}\mathbf{q}_{\max}{}^{\top} - \sqrt{\frac{-\lambda_{\min}\lambda_{\max}}{4}}\mathbf{q}_{\min}\mathbf{q}_{\max}{}^{\top} \\
&+ \sqrt{\frac{-\lambda_{\min}\lambda_{\max}}{4}}\mathbf{q}_{\max}\mathbf{q}_{\min}{}^{\top} - \frac{-\lambda_{\min}}{2}\mathbf{q}_{\min}\mathbf{q}_{\min}{}^{\top}).
\end{aligned} \tag{32}$$

We then can conclude that $\mathbf{M} = \mathbf{j}\mathbf{k}^{\top} + \mathbf{k}\mathbf{j}^{\top}$. □

Since $\mathbf{M}$ is a symmetric matrix, we cannot directly distinguish $\boldsymbol{\nu}/d$ and $\mathbf{n}$ from each other. Note that $\mathbf{n}$ is a normal vector which has a unique norm. We can thus decompose all candidates as

$$\begin{cases}
\frac{1}{d}\boldsymbol{\nu} = \mathbf{j}\|\mathbf{k}\|, \quad \mathbf{n} = \dfrac{\mathbf{k}}{\|\mathbf{k}\|}, \quad \boldsymbol{\omega} = -(\mathbf{H}_d + \mathbf{j}\mathbf{k}^{\top})^{\vee} \\
\frac{1}{d}\boldsymbol{\nu} = \mathbf{k}\|\mathbf{j}\|, \quad \mathbf{n} = \dfrac{\mathbf{j}}{\|\mathbf{j}\|}, \quad \boldsymbol{\omega} = -(\mathbf{H}_d + \mathbf{k}\mathbf{j}^{\top})^{\vee},
\end{cases} \tag{33}$$

where the $(.)^\vee$ denotes the inverse operator of $(.)_\times$, which converts a skew-symmetric matrix to a vector.

Above all, the decomposition of the differential homography matrix results in two sets of possible solutions that are indistinguishable in the absence of prior scene or motion information. However, when integrated with data from other sensors, this decomposition remains valuable, enabling the estimation of otherwise unknown motion and scene parameters.

III Dataset and Evaluation Metrics

To supplement Sec. 5 of our paper, we detail our dataset as well as the evaluation metrics used in our experiments.

III.A Our Dataset

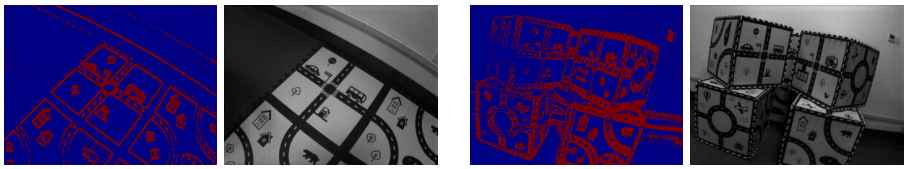
Real Data. For our study on rotational motion estimation, we initially performed experiments using the ECD dataset [6], which was captured using an event camera [2] with a relatively low resolution (240×180 px). To assess our proposed algorithm with more advanced sensors, we recorded two similar sequences, *ground_rotation* and *boxed_rotation*, using an event camera (iniVation DAVIS346) with a higher spatial resolution. Additionally, we employ an Xsens Mti-630 IMU to provide ground-truth angular velocity. Fig. 2 offers an insight into the recorded data. Meanwhile, Fig. 3 depicts the ground-truth angular velocity information, featuring high dynamics of the event camera in each sequence.

Synthetic Data. In our paper, we conduct a comprehensive set of experiments using a synthetic dataset created with a simulator [7]. This dataset comprises sequences, including *two_wall_translation*, *patterns_rotation*, *cubes_rotation*, *patterns_6dof*, and *cubes_6dof*, synthesized using an event camera with a 640×480 px spatial resolution. Additionally, the simulator provides us with ground truth data of optical flow, depth and camera trajectory for each sequence. All sequences are recorded over a span of 0.5 seconds under different motion patterns. Specifically, sequences *patterns_6dof* and *cubes_6dof* feature a combined motion pattern of a linear translation plus a rotation around a certain axis. In contrast, sequences *patterns_rotation* and *cubes_rotation* involve pure rotation at a constant angular velocity, while sequence *two_wall_translation* exhibits a pure translation. Fig. 4 shows the simulated scenes and the pose of the camera.

III.B Evaluation Metrics

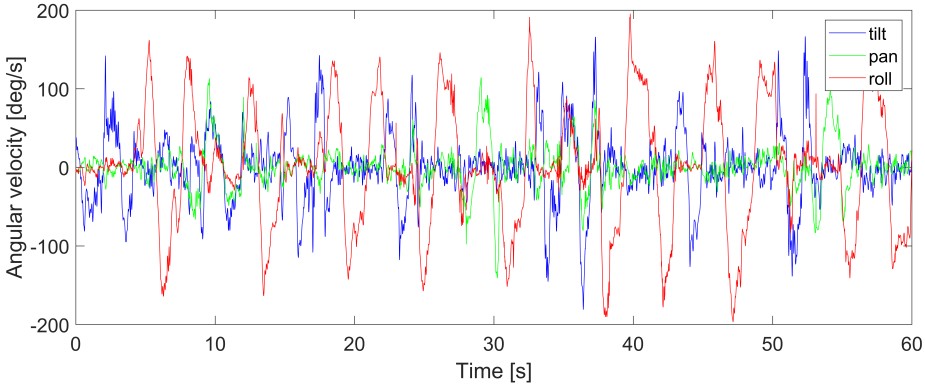
In our quantitative evaluation, we utilize the average angular velocity error (e_w) and root mean square error (RMSE) to measure the angular velocity estimation error, which is defined as

$$\mathrm{RMSE}_\omega = \sqrt{\frac{1}{3m}\sum_{i=1}^{m}(e_{\omega_x i}^2 + e_{\omega_y i}^2 + e_{\omega_z i}^2)}. \tag{34}$$

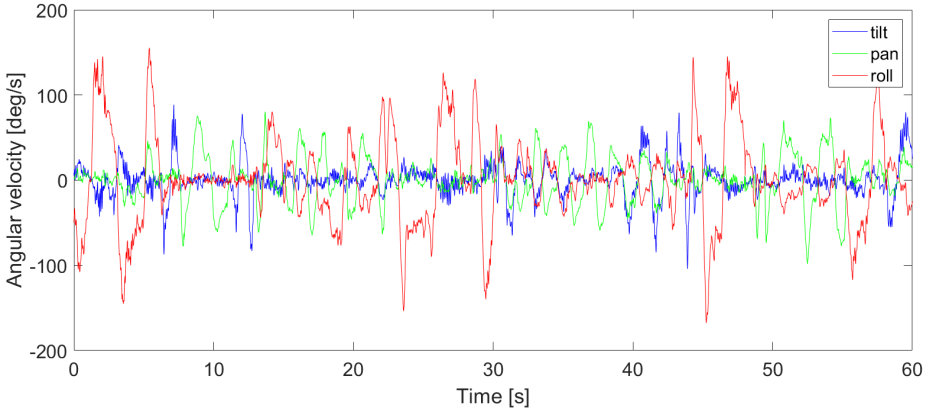


(a) *ground_rotation* **(b)** *boxes_rotation*

Fig. 2: Event data (represented with a naive accumulation) and corresponding frames from our data collected using a DAVIS-346 event camera.

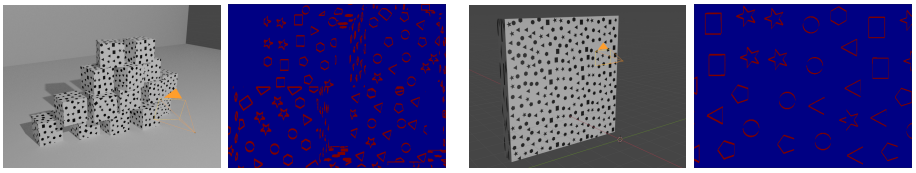


(a) *ground_rotation*



(b) *boxes_rotation*

Fig. 3: Angular velocity measurements used as ground truth from an IMU.



(a) *cubes_rotation* **(b)** *patterns_rotation*

Fig. 4: An event camera (orange) capturing different sequences in simulated scenes.

For linear velocity evaluation, we also evaluate error by means of the average (e_v) and the RMSE

$$\text{RMSE}_v = \sqrt{\frac{1}{3m}\sum_{i=1}^{m}(e_{v_x i}^2 + e_{v_y i}^2 + e_{v_z i}^2)}. \tag{35}$$

Retrieving the motion and structure parameters from the decomposition of differential homography is non-trivial, and it is out of the scope of this paper. Therefore, instead of comparing motion and structure parameters with those derived from other methods, we assess the errors in the estimated differential homography matrix by means of the Frobenius norm of the difference with respect to the ground truth.

IV Implementation Details

In this section, we add implementation details to supplement Sec. 5 of our paper. The following sections will introduce normal flow extraction, linear solver, continuous-time nonlinear solver, hyperparameter settings in our experiments.

Normal Flow Extraction. Since our minimal solver is built on normal flow constraint, the pre-stage normal flow calculation becomes crucial. Here, we directly calculate gradients from raw event volume with local plane fitting. To further refine these calculations, RANSAC is applied. Ultimately, we employ (20) to accurately determine the partial normal flow.

Linear Solver. From the event batch, we extracted numerous partial normal flow constraints, which enabled us to construct an overdetermined system of equations. However, the estimated normal flow is still subject to noise due to the limitations of existing methods. To address this, we employed RANSAC for robust sampling.

Continuous-Time Nonlinear Solver. Based on B-Spline and above mentioned normal flow constraint, we designed a continuous motion estimation solver. This solver utilizes the estimation results from our linear solver as initial

values, incorporating a sliding window mechanism to refine the control points for optimization. To enhance robustness in our estimations, we substitute the squared norm error with robust functions, specifically M-estimators. Consequently, this approach enables us to derive motion parameters at any given time from the continuous-time B-Spline curve, ensuring a more accurate and flexible representation of motion across time.

Hyper-Parameter Settings. Parameters including window size and threshold may have crucial influence on the performance of each solver, which need a clear clarification. Here, we have configured the spatial and temporal window sizes for normal flow extraction at 7px $\times$ 7px $\times$ 0.04s, respectively. Additionally, the RANSAC threshold of plane fitting is set at 10^{-5}, while the RANSAC threshold of linear solver is established at 10^{-4}.

Hardware Configurations. All experiments are conducted on a standard computer with Intel i7-13700F CPU, 16G memory and implemented with MATLAB (version R2022a).

V Extensive Results

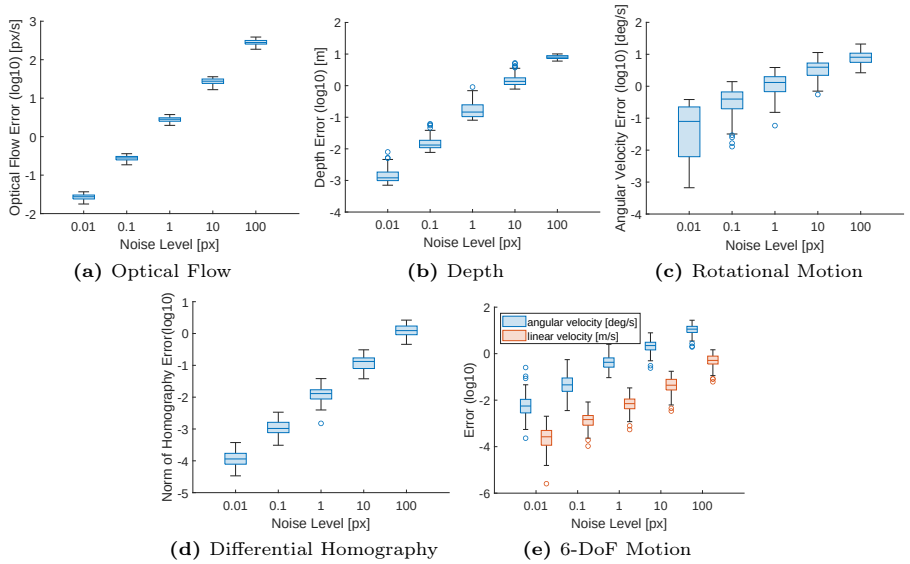
To further investigate and discuss the performance of our solver, we conduct some extensive experiments to supplement Sec. 5 of our paper.

V.A Numerical Stability Analysis

We conduct the numerical stability analysis of the linear solver in the five problems tackled: 1) Optical flow estimation, 2) Depth estimation, 3) Rotational estimation, 4) Differential homography estimation, and 5) 6-DoF motion tracking. We gradually increase the noise level from 0.01 px to 100 px on the normal flow observations, and observe how sensitive of our linear solver is in each problem. As shown in Fig. 5, for optical flow estimation, the error increases with the noise level and our linear solver could estimate optical flow under 1-pixel error in normal flow observation. Also, our linear solver could estimate depth with high accuracy under the same noise level. For tasks involving rotation, differential homography, and 6-DoF motion estimation, our linear solver demonstrates greater resilience, effectively functioning under more severe noise conditions. In general, accurate results are witnessed in all five problems under a noise level of 1 pixel, thereby substantiating the robustness of our linear solver.

V.B Continuous-Time Nonlinear Solver

Our experiments on sequence *shapes_ rotation* from the dataset [6] have already demonstrated the superiority of our continuous-time nonlinear solver in handling aggressive motion. However, the relatively simple texture of this sequence



(a) Optical Flow **(b)** Depth **(c)** Rotational Motion

(d) Differential Homography **(e)** 6-DoF Motion

Fig. 5: Numerical stability analysis on the tasks of optical flow, depth estimation, rotational motion estimation, differential homography, and 6-DoF motion tracking, respectively.

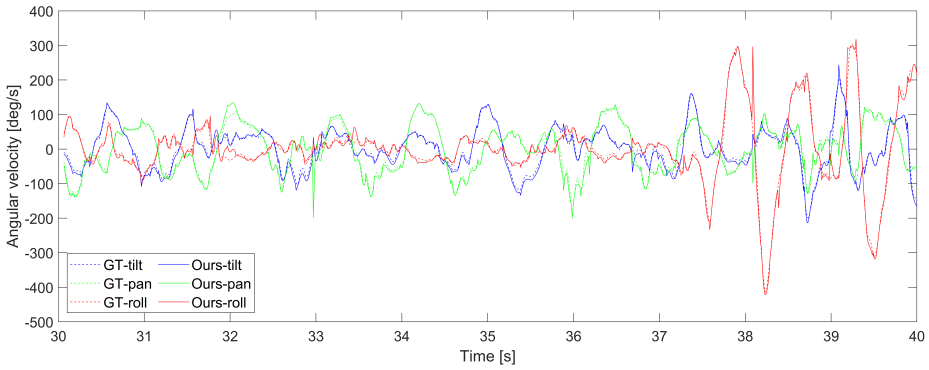


Fig. 6: Results of the continuous-time angular velocity estimator on sequence *dynamic_rotation*.

raises questions about our solver's performance in more complex visual environments. To address this, we conduct an additional experiment on sequence *dynamic_rotation*, also from the dataset [6], which features a high-texture scene. The results, illustrated in Fig. 6, confirm the solver's capability to perform accurately in everyday-textured scenes, indicating its robustness across diverse scene complexities.

References

1. Benosman, R., Clercq, C., Lagorce, X., Ieng, S.H., Bartolozzi, C.: Event-based visual flow. IEEE Trans. Neural Netw. Learn. Syst. **25**(2), 407–417 (2014). https://doi.org/10.1109/TNNLS.2013.2273537
2. Brandli, C., Berner, R., Yang, M., Liu, S.C., Delbruck, T.: A 240x180 130dB 3μs latency global shutter spatiotemporal vision sensor. IEEE J. Solid-State Circuits **49**(10), 2333–2341 (2014). https://doi.org/10.1109/JSSC.2014.2342715
3. Lagorce, X., Orchard, G., Gallupi, F., Shi, B.E., Benosman, R.: HOTS: A hierarchy of event-based time-surfaces for pattern recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**(7), 1346–1359 (Jul 2017). https://doi.org/10.1109/TPAMI.2016.2574707
4. Longuet-Higgins, H.C., Prazdny, K.: The interpretation of a moving retinal image. Proceedings of the Royal Society of London. Series B. Biological Sciences **208**(1173), 385–397 (1980). https://doi.org/10.1098/rspb.1980.0057
5. Ma, Y., Soatto, S., Košecká, J., Sastry, S.: An Invitation to 3-D Vision: From Images to Geometric Models. Springer (2004)
6. 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. Research **36**(2), 142–149 (2017). https://doi.org/10.1177/0278364917691115
7. Rebecq, H., Gehrig, D., Scaramuzza, D.: ESIM: an open event camera simulator. In: Conf. on Robotics Learning (CoRL) (2018)
8. Zhuang, B., Cheong, L.F., Lee, G.H.: Rolling-shutter-aware differential sfm and image rectification. In: 2017 IEEE International Conference on Computer Vision (ICCV). pp. 948–956 (2017). https://doi.org/10.1109/ICCV.2017.108
9. Zhuang, B., Tran, Q.H.: Image stitching and rectification for hand-held cameras. In: European Conference on Computer Vision (2020). https://doi.org/10.1007/978-3-030-58571-6_15