SLAM Homework 5:
Motion Estimation with Event Camera

Prof. Laurent Kneip

May 28th 2024

Instructions:

1. Deadline: **2024-6-11 23:59:59**
2. No handwritten homework is accepted.
3. Your homework should be submitted in PDF format and packed with your code, and the naming format of the file is *studentID-name-hw5*.
4. Please submit your homework through email to *daizj2022@shanghaitech.edu.cn* with the subject line "studentID-name-hw5"

Event cameras respond primarily to edges—formed by strong gradients—and are thus particularly well-suited for line-based motion estimation. This homework extends the discussion on motion estimation with event cameras for line.

Assume a calibrated event camera undergoing an arbitrary six DoF motion, while observing a set of M lines $\{\mathbf{L}_i\}_{i=1}^{M}$. Each line generates a set of N_i events $\mathcal{E}_i = \{e_{ij}\}_{j=1}^{N_i}$ where each event $e_{ij} = (\mathbf{x}_{ij}, t_{ij}, p_{ij})$ is characterized by its pixel coordinate $\mathbf{x}_{ij}$ in the image plane, timestamp t_{ij} (with μs resolution), and polarity p_{ij}. For a small time window $[t_s - \Delta t, t_s + \Delta t]$, centered at reference time t_s, such that the camera motion can be approximated by linear dynamics, the events generated by a single line circumscribe a manifold termed eventail as fig 1.

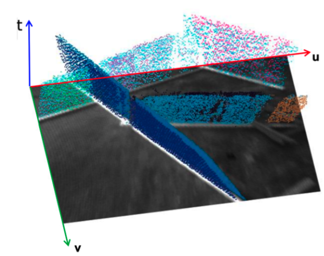

Figure 1: The eventails.

In this homework, for simplicity, we will consider the case of one line, and thus drop the index i from the variables. We furthermore express all quantities in the camera frame centered at time t_s. The incidence relation enforces that events are triggered by points on the line, such that the line $\mathbf{L}_j = [\mathbf{d}_j^\intercal \mathbf{m}_j^\intercal]^\intercal$ (in Plücker coordinates) emanating from an individual event e_j triggered at time t_j (orange line in fig 2) intersects the line $\mathbf{L} = [\mathbf{d}^\intercal \mathbf{m}^\intercal]^\intercal$ ((blue line in fig 2). The condition for intersection of two non-parallel lines is

$$\mathbf{d}^\intercal \mathbf{m}_j + \mathbf{m}^\intercal \mathbf{d}_j = 0. \tag{1}$$

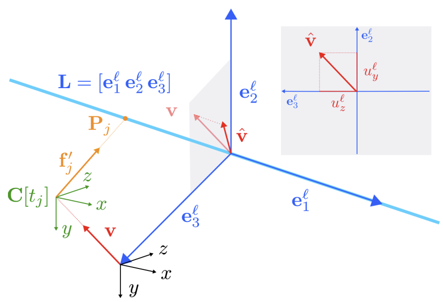

Figure 2: Incidence relationship between the line $\mathbf{L}$

Plücker line coordinates comprise the line direction $\mathbf{d} \in \mathbb{R}^3$ and moment $\mathbf{m} = \mathbf{P} \times \mathbf{d}$, with $\mathbf{P} \in \mathbb{R}^3$ being an arbitrary point on the line. For the event ray j we use the camera position $\mathbf{C}[t_j] = \mathbf{P}_j$ at time t_j, and the event bearing vector $\mathbf{f}_j' = \mathbf{d}_j$ (measurement of $\mathbf{P}_j$ on line) rotated into the reference frame at time t_s. Under first order dynamics, these are $\mathbf{C}[t_j] = t_j'\mathbf{v}$ ($\mathbf{v}$ is velocity) and $\mathbf{f}_j' = \mathbf{R}[t_j]\mathbf{f}_j$, with $t_j' = t_j - t_s$. To simplify, we assume $\mathbf{R}[t_j]$ is given. The Plücker coordinates are thus $\mathbf{L}_j = [\mathbf{f}_j'^\intercal (\mathbf{C}[t_j] \times \mathbf{f}_j')]^\intercal$ and the incidence relation becomes

$$\mathbf{d}^\intercal (\mathbf{C}[t_j] \times \mathbf{f}_j') + \mathbf{m}^\intercal \mathbf{f}_j' = 0. \tag{2}$$

We can make a transition into a minimal form, the details can be seen in [1]

$$t_j' \mathbf{f}_j'^\intercal (u_z^\ell \mathbf{e}_2^\ell - u_y^\ell \mathbf{e}_3^\ell) + \mathbf{f}_j'^\intercal \mathbf{e}_2^\ell = 0. \tag{3}$$

where $\mathbf{u}_\ell = [u_x^\ell u_y^\ell u_z^\ell]$ is camera velocity experssed in the line coordinate frame, $\mathbf{R}_\ell = [\mathbf{e}_1^\ell \mathbf{e}_2^\ell \mathbf{e}_3^\ell]$ is rotation matrix. The equation 3 has five unknowns, and each such constraint originates from a single event, this means that five events are the minimum number to solve this system. After we stack 5 equations, and system is linear in the unknowns and can be rewritten as a single matrix equation

$$\underbrace{\begin{bmatrix} t_1' \mathbf{f}_1'^\intercal & \mathbf{f}_1'^\intercal \\ \vdots & \vdots \\ t_5' \mathbf{f}_5'^\intercal & \mathbf{f}_5'^\intercal \end{bmatrix}}_{\doteq \mathbf{A} \in \mathbb{R}^{5\times 6}} \underbrace{\begin{bmatrix} u_z^\ell \mathbf{e}_2^\ell - u_y^\ell \mathbf{e}_3^\ell \\ \mathbf{e}_2^\ell \end{bmatrix}}_{\doteq \mathbf{x} \in \mathbb{R}^{6\times 1}} = \mathbf{0}. \tag{4}$$

Solving eq 4 can be done with a singular value decomposition of $\mathbf{A}$ and then selecting the last column of $\mathbf{V}$ corresponding to the smallest singular value of $\mathbf{A}$. Let us denote this solution with $\hat{\mathbf{x}}$. We need to recover the unknowns from $\hat{\mathbf{x}}$.

$$\hat{\mathbf{x}} = \begin{bmatrix} \lambda \hat{\mathbf{x}}_{1:3} \\ \lambda \hat{\mathbf{x}}_{4:6} \end{bmatrix} = \begin{bmatrix} u_z^\ell \mathbf{e}_2^\ell - u_y^\ell \mathbf{e}_3^\ell \\ \mathbf{e}_2^\ell \end{bmatrix}. \tag{5}$$

we assume that this normalization is done beforehand, and thus ignore this scaling factor by setting $\lambda = 1$

$$\mathbf{e}_2^\ell = \hat{\mathbf{x}}_{4:6} \tag{6a}$$

$$u_z^\ell = \hat{\mathbf{x}}_{1:3}^\mathsf{T} \hat{\mathbf{x}}_{4:6} \tag{6b}$$

$$u_y^\ell \mathbf{e}_1^\ell = \hat{\mathbf{x}}_{1:3} \times \hat{\mathbf{x}}_{4:6}. \tag{6c}$$

$$u_y^\ell = \|\hat{\mathbf{x}}_{1:3} \times \hat{\mathbf{x}}_{4:6}\|, \mathbf{e}_1^\ell = \frac{\hat{\mathbf{x}}_{1:3} \times \hat{\mathbf{x}}_{4:6}}{\|\hat{\mathbf{x}}_{1:3} \times \hat{\mathbf{x}}_{4:6}\|}, \mathbf{e}_3^\ell = \mathbf{e}_1^\ell \times \mathbf{e}_2^\ell. \tag{7}$$

There are three tasks for you:

1) Given clean events camera synthetic data and no self-rotation, there is only one line on it, try to recover the 2 dimension camera velocity $\mathbf{u}_\ell = [0 u_y^\ell u_z^\ell]$ and rotation $\mathbf{R}_\ell = [\mathbf{e}_1^\ell \mathbf{e}_2^\ell \mathbf{e}_3^\ell]$. (30%)

2) Given clean events synthetic data with self-rotation, there is only one line on it, try to recover the 2 dimension camera velocity $\mathbf{u}_\ell = [0 u_y^\ell u_z^\ell]$ and rotation $\mathbf{R}_\ell = [\mathbf{e}_1^\ell \mathbf{e}_2^\ell \mathbf{e}_3^\ell]$. (30%)

3) Given clean & noisy events synthetic data with self-rotation, there is only one line on it, you will need to use RANSAC to fit over all inliers and get the line, try to recover the 2 dimension camera velocity $\mathbf{u}_\ell = [0 u_y^\ell u_z^\ell]$ and rotation $\mathbf{R}_\ell = [\mathbf{e}_1^\ell \mathbf{e}_2^\ell \mathbf{e}_3^\ell]$. (40%)

Data Description:

The data consist three txt files. Details on each data package:

- **data_package1.txt** a set of 50 clean events, with no self-rotation.
- **data_package2.txt** a set of 50 clean events, with self-rotation, camera angular velocity will be given in first line.
- **data_package3.txt** a set of 50 clean & noisy events, with self-rotation, camera angular velocity will be given in first line.

Moreover, universal setup across all data packages:
–IMAGE_WIDTH=640
–IMAGE_HEIGHT=480
–FOCAL_LENGTH=320
–TIME_INTERVAL=[-0.25,0.25]
Camera's ego motion always sticks to the first order dynamics.

Notes:

1) Both C++ and Python are acceptable. You can only use third-party libraries for matrix computation (i.e. NumPy and Eigen) and visualization (i.e. evo or Open3D or Matplotlib).

2) Attach your implementation with pdf in the zip. In the package, you also need to include a file named README.txt/md to identify the function of each file. Make sure that your codes can run and are consistent with your homework. We would then arrange a meeting after the deadline in which we would ask each one of you to come in for 10 minutes to demonstrate your solution on your own computer.

3) If submitted after the deadline but still within 24hrs, a 50% penalty is applied. If submitted more than 24hrs after the deadline, a zero score will be given. In special case, please contact Prof Kneip.

References

[1] Ling Gao, Daniel Gehrig, Hang Su, Davide Scaramuzza, and Laurent Kneip. An n-point linear solver for line and motion estimation with event cameras, 2024. 2