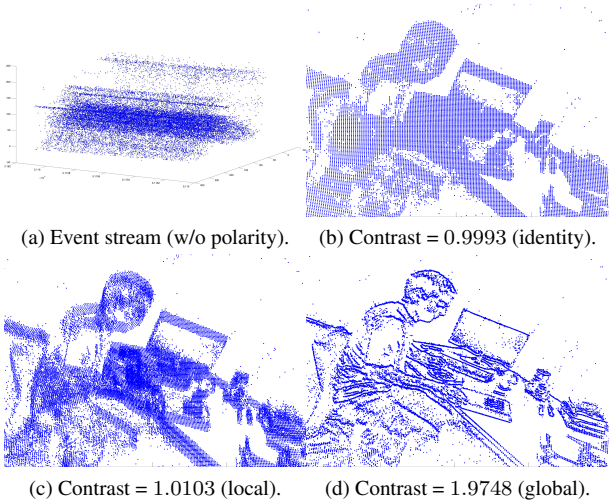
Globally Optimal Contrast Maximisation for Event-based Motion Estimation

Daqi Liu Álvaro Parra Tat-Jun Chin
School of Computer Science, The University of Adelaide
{daqi.liu, alvaro.parrabustos, tat-jun.chin}@adelaide.edu.au

Abstract

Contrast maximisation estimates the motion captured in an event stream by maximising the sharpness of the motion-compensated event image. To carry out contrast maximisation, many previous works employ iterative optimisation algorithms, such as conjugate gradient, which require good initialisation to avoid converging to bad local minima. To alleviate this weakness, we propose a new globally optimal event-based motion estimation algorithm. Based on branch-and-bound (BnB), our method solves rotational (3DoF) motion estimation on event streams, which supports practical applications such as video stabilisation and attitude estimation. Underpinning our method are novel bounding functions for contrast maximisation, whose theoretical validity is rigorously established. We show concrete examples from public datasets where globally optimal solutions are vital to the success of contrast maximisation. Despite its exact nature, our algorithm is currently able to process a $50,000$*-event input in* ≈ 300 *seconds (a locally optimal solver takes* ≈ 30 *seconds on the same input). The potential for GPU acceleration will also be discussed.*



(a) Event stream (w/o polarity). (b) Contrast = 0.9993 (identity).
(c) Contrast = 1.0103 (local). (d) Contrast = 1.9748 (global).

Figure 1. (a) 10 ms event stream under rotational motion [14]. Since event polarity is not used in our work, the events are plotted in the same colour. (b) Event image without motion compensation (identity transformation). (c)(d) Event images produced with locally and globally optimal contrast maximisation.

1. Introduction

By asynchronously detecting brightness changes, event cameras offer a fundamentally different way to detect and characterise physical motion. Currently, active research is being conducted to employ event cameras in many areas, such as robotics/UAVs [9, 21], autonomous driving[22, 31, 32], and spacecraft navigation [7, 8]. While the utility of event cameras extends beyond motion perception, e.g., object recognition and tracking [26, 28], the focus of our work is on estimating visual motion using event cameras.

Due to the different nature of the data, new approaches are required to extract motion from event streams. A recent successful framework is contrast maximisation (CM) [13]. Given an event stream, CM aims to find the motion parameters that yield the sharpest motion-compensated event image; see Fig. 1. Intuitively, the correct motion parameters will align corresponding events, thereby producing an image with high contrast. We formally define CM below.

Event image Let $\mathcal{E} = \{\boldsymbol{e}_i\}_i^N$ be an event stream recorded over time duration $\mathcal{T} = [0, t_{\max}]$. Each event $\boldsymbol{e}_i = (\mathbf{u}_i, t_i, p_i)$ contains an image position $\mathbf{u}_i$, time stamp $t_i \in \mathcal{T}$, and polarity $p_i \in \{-1, +1\}$. We assume $\mathcal{E}$ was produced under camera motion $\mathcal{M}$ over a 3D scene, thus each $\boldsymbol{e}_i$ is associated with a scene point that triggered the event.

We parameterise $\mathcal{M}$ by a vector $\boldsymbol{\omega} \in \Omega$, and let $X = \{\mathbf{x}_j\}_{j=1}^P$ be the centre coordinates of the pixels in the image plane of the event sensor. Under CM, the event image H_c is defined as a function of $\boldsymbol{\omega}$, and the intensity at pixel $\mathbf{x}_j$ is

$$H_c(\mathbf{x}_j; \boldsymbol{\omega}) = \sum_{i=1}^{N} \boldsymbol{\delta}(\mathbf{x}_j - f(\mathbf{u}_i, t_i; \boldsymbol{\omega})), \qquad (1)$$

where $\boldsymbol{\delta}$ is a kernel function (e.g., Gaussian). Following [13, Sec. 2.1], we do not use event polarities in (1). H_c

is regarded to be captured at time 0, and the function

$$\mathbf{u}'_i = f(\mathbf{u}_i, t_i; \boldsymbol{\omega}) \tag{2}$$

warps $\mathbf{u}_i$ to $\mathbf{u}'_i$ in H_c by "undoing" the motion $\mathcal{M}$ between time 0 and t_i. Intuitively, $\mathbf{u}'_i$ is the image position of the 3D scene point that triggered $\boldsymbol{e}_i$, if it was observed at time 0.

In practice, the region of support of the kernel $\boldsymbol{\delta}$ in (1) is small w.r.t. image dimensions, e.g., Gaussian kernels with bandwidth $\sigma = 1$ pixel were used in [13, Sec. 2]. This motivates the usage of "discrete" event images

$$H_d(\mathbf{x}_j; \boldsymbol{\omega}) = \sum_{i=1}^{N} \mathbb{I}(f(\mathbf{u}_i, t_i; \boldsymbol{\omega}) \text{ lies in pixel } \mathbf{x}_j), \tag{3}$$

where $\mathbb{I}$ returns 1 if the input predicate is true, and 0 otherwise. As we will show later, conducting CM using H_c (with small bandwidth) and H_d yields almost identical results.

Contrast maximisation The *contrast* of an event image H (continuous or discrete) is the variance of its pixel values. Since H depends on $\boldsymbol{\omega}$, the contrast is also a function of $\boldsymbol{\omega}$

$$C(\boldsymbol{\omega}) = \frac{1}{P} \sum_{j=1}^{P} (H(\mathbf{x}_j; \boldsymbol{\omega}) - \mu(\boldsymbol{\omega}))^2, \tag{4}$$

where $\mu(\boldsymbol{\omega})$ is the mean intensity

$$\mu(\boldsymbol{\omega}) = \frac{1}{P} \sum_{j=1}^{P} H(\mathbf{x}_j; \boldsymbol{\omega}). \tag{5}$$

CM [13] estimates $\mathcal{M}$ by maximising the contrast of H, i.e.,

$$\max_{\boldsymbol{\omega} \in \Omega} \; C(\boldsymbol{\omega}). \tag{6}$$

The intuition is that the correct $\boldsymbol{\omega}$ will allow $\mathcal{M}$ to align events that correspond to the same scene points in H, thus leading to a sharp or high-contrast event image; see Fig. 1.

Global versus local solutions By *globally optimal* (or "global") solution to CM (6), we mean $\boldsymbol{\omega}^* \in \Omega$ such that

$$C(\boldsymbol{\omega}^*) \geq C(\boldsymbol{\omega}) \;\; \forall \boldsymbol{\omega} \in \Omega. \tag{7}$$

A solution $\hat{\boldsymbol{\omega}}$ is *approximate* if $C(\hat{\boldsymbol{\omega}}) < C(\boldsymbol{\omega}^*)$. Also, a solution is *locally optimal* (or "local") if it is the maximiser of its neighbourhood [24, Chap. 2]. All global solutions are locally optimal, but the converse is not true.

1.1. Previous works

Previous works on CM (e.g., [11, 13, 28, 29]) apply nonlinear optimisation (e.g., conjugate gradient) to solve (6). Given an initial solution $\boldsymbol{\omega}^{(0)}$, the solution is successively updated until convergence to a locally optimal solution. In practice, if the local solution is a bad approximate solution, there can be significant differences in its quality compared to the global solution; see Fig. 1. This can occur when $\boldsymbol{\omega}^{(0)}$ is too distant from good solutions, or $C(\boldsymbol{\omega})$ is too nonconcave (e.g., when $\boldsymbol{\delta}$ has a very small bandwidth). Thus, algorithms that can find $\boldsymbol{\omega}^*$ are desirable.

Recent improvements to CM include modifying the objective function to better suit the targeted settings [11, 29]. However, the optimisation work horse remains locally optimal methods. Other frameworks for event processing [6, 7, 12, 18, 19, 20] conduct filtering, Hough transform, or specialised optimisation schemes; these are generally less flexible than CM [13]. There is also active research in applying deep learning to event data [27, 33, 35, 36], which require a separate training phase on large datasets.

Contributions We focus on estimating rotational motion from events, which is useful for several applications, e.g., video stabilisation [14] and attitude estimation [8].

Specifically, we propose a BnB method for *globally optimal* CM for rotation estimation. Unlike previous CM techniques, our algorithm does not require external initialisations $\boldsymbol{\omega}^{(0)}$, and can guarantee finding the global solution $\boldsymbol{\omega}^*$ to (6). Our core contributions are novel bounding functions for CM, whose theoretical validity are established. As we will show in Sec. 4, while local methods generally produce acceptable results [13, 14], they often fail during periods with fast rotational motions. On the other hand, our global method always returns accurate results.

2. Rotation estimation from events

If duration $\mathcal{T}$ is small (e.g., $t_{\max} = 10ms$), a fixed axis of rotation and constant angular velocity can be assumed for $\mathcal{M}$ [13]. Following [13, Sec. 3], $\mathcal{M}$ can be parametrised as a 3-vector $\boldsymbol{\omega}$, where the direction of $\boldsymbol{\omega}$ is the axis of rotation, and the length $\|\boldsymbol{\omega}\|_2$ of $\boldsymbol{\omega}$ is the angular rate of change. Between time 0 and t, the rotation undergone is

$$\mathbf{R}(t; \boldsymbol{\omega}) = \exp([\boldsymbol{\omega}\, t]_\times), \tag{8}$$

where $\boldsymbol{\omega} t$ is the axis-angle representation of the rotation, $[\boldsymbol{\omega} t]_\times$ is the skew symmetric form of $\boldsymbol{\omega} t$, and exp is the exponential map (see [1] for details).

Let $\mathbf{K}$ be the 3×3 intrinsic matrix of the event camera ($\mathbf{K}$ is known after calibration [2, 34]). The warp (2) is thus

$$f(\mathbf{u}_i, t_i; \boldsymbol{\omega}) = \frac{\mathbf{K}^{(1:2)} \mathbf{R}(t_i; \boldsymbol{\omega}) \tilde{\mathbf{u}}_i}{\mathbf{K}^{(3)} \mathbf{R}(t_i; \boldsymbol{\omega}) \tilde{\mathbf{u}}_i}, \tag{9}$$

where $\tilde{\mathbf{u}}_i = [\mathbf{u}_i^T \; 1]^T$ is the homogeneous version of $\mathbf{u}_i$, and $\mathbf{K}^{(1:2)}$ and $\mathbf{K}^{(3)}$ are respectively the first-two rows and third row of $\mathbf{K}$. Intuitively, (9) rotates the ray that passes through $\mathbf{u}_i$ using $\mathbf{R}(t_i; \boldsymbol{\omega})$, then projects the rotated ray onto H.

Following [13, Sec. 3], we also assume a known maximum angular rate $r_{\max}$. The domain is thus an $r_{\max}$-ball

$$\Omega = \{\boldsymbol{\omega} \in \mathbb{R}^3 \mid \|\boldsymbol{\omega}\|_2 \leq r_{\max}\}, \tag{10}$$

and our problem reduces to maximising $C(\boldsymbol{\omega})$ over this ball, based on the rotational motion model (9).

2.1. Main algorithm

Algorithm 1 summarises our BnB algorithm to achieve globally optimal CM for rotation estimation. Starting from the tightest bounding cube $\mathbb{B}$ on the $r_{\max}$-ball Ω (the initial $\mathbb{B}$ is thus of size $(2r_{\max})^3$), the algorithm recursively subdivides $\mathbb{B}$ and prunes the subcubes until the global solution is found. A lower bound $\underline{C}$ and upper bound $\overline{C}(\mathbb{B})$ are used to prune each $\mathbb{B}$. When the difference between the bounds is smaller than τ, the algorithm terminates with $\hat{\boldsymbol{\omega}}$ being the global solution $\boldsymbol{\omega}^*$ (up to error τ, which can be chosen to be arbitrarily small). See [15, 16] for details of BnB.

As alluded to above, our core contributions are novel and effective bounding functions for CM using BnB. We describe our bounding functions in the next section.

Algorithm 1 BnB for rotation estimation from events.

Require: Event stream $\mathcal{E} = \{\boldsymbol{e}_i\}_{i=1}^N$, maximum angular rate of change $r_{\max}$, convergence threshold τ.

$q \leftarrow$ Initialise priority queue.
$\mathbb{B} \leftarrow$ Cube in $\mathbb{R}^3$ of size $(2r_{\max})^3$ centred at origin.
$\boldsymbol{\omega}_c \leftarrow$ Centre of $\mathbb{B}$.
$\hat{\boldsymbol{\omega}} \leftarrow \boldsymbol{\omega}_c$.
Insert $\mathbb{B}$ into q with priority $\overline{C}(\mathbb{B})$.
while q is not empty **do**
  $\mathbb{B} \leftarrow$ Dequeue top item from q.
  If $\overline{C}(\mathbb{B}) - C(\hat{\boldsymbol{\omega}}) \leq \tau$, then terminate.
  $\boldsymbol{\omega}_c \leftarrow$ Centre of $\mathbb{B}$.
  If $C(\boldsymbol{\omega}_c) \geq C(\hat{\boldsymbol{\omega}})$, then $\hat{\boldsymbol{\omega}} \leftarrow \boldsymbol{\omega}_c$.
  Uniformly subdivide $\mathbb{B}$ into 8 subcubes $\mathbb{B}_1, \ldots, \mathbb{B}_8$.
  for $i = 1, \cdots, 8$ **do**
    if $\overline{C}(\mathbb{B}_i) \geq C(\hat{\boldsymbol{\omega}})$ **then**
      Insert $\mathbb{B}_i$ into q with priority $\overline{C}(\mathbb{B}_i)$.
    end if
  end for
end while
return $\hat{\boldsymbol{\omega}}$ as $\boldsymbol{\omega}^*$.

3. Bounds for contrast maximisation

To search for the maximum of $C(\boldsymbol{\omega})$ using BnB, a lower and upper bound on the objective are required.

The lower bound $\underline{C}$ must satisfy the condition

$$\underline{C} \leq \max_{\boldsymbol{\omega} \in \Omega} C(\boldsymbol{\omega}), \tag{11}$$

which is trivially achieved by any (suboptimal) solution. In Algorithm 1, the current best solution $\hat{\boldsymbol{\omega}}$ is used to provide $\underline{C}$, which is iteratively raised as the search progresses.

The upper bound $\overline{C}(\mathbb{B})$ is defined over a region (a subcube) $\mathbb{B}$ of Ω, and must satisfy the condition

$$\overline{C}(\mathbb{B}) \geq \max_{\boldsymbol{\omega} \in \mathbb{B}} C(\boldsymbol{\omega}). \tag{A1}$$

Also, as $\mathbb{B}$ collapses to a single point $\boldsymbol{\omega}$, $\overline{C}(\mathbb{B})$ should equate to $C(\boldsymbol{\omega})$; more formally,

$$\overline{C}(\mathbb{B}) \to C(\boldsymbol{\omega}) \text{ when } \mathbb{B} \to \boldsymbol{\omega}. \tag{A2}$$

See [16] for the rationale of the above conditions for BnB.

Deriving the upper bound is a more involved process. Our starting point is to rewrite (4) as

$$C(\boldsymbol{\omega}) = \frac{1}{P} \sum_{j=1}^{P} H(\mathbf{x}_j; \boldsymbol{\omega})^2 - \mu(\boldsymbol{\omega})^2, \tag{12}$$

which motivates a bound based on two components

$$\overline{C}(\mathbb{B}) := \frac{1}{P} \overline{S}(\mathbb{B}) - \underline{\mu}(\mathbb{B})^2, \tag{13}$$

where $\overline{S}(\mathbb{B})$ is an upper bound

$$\overline{S}(\mathbb{B}) \geq \max_{\boldsymbol{\omega} \in \mathbb{B}} \sum_{j=1}^{P} H(\mathbf{x}_j; \boldsymbol{\omega})^2 \tag{14}$$

on the "sum of squares (SoS)" component, and

$$\underline{\mu}(\mathbb{B}) \leq \min_{\boldsymbol{\omega} \in \mathbb{B}} \mu(\boldsymbol{\omega}) \tag{15}$$

is a lower bound of the mean pixel value. Given (14) and (15), then (13) satisfies A1. If equality holds in (14) and (15) when $\mathbb{B}$ is singleton, then (13) also satisfies A2.

In Secs. 3.1 and 3.2, we develop $\overline{S}(\mathbb{B})$ for continuous and discrete event images, before deriving $\underline{\mu}(\mathbb{B})$ in Sec. 3.3.

3.1. SoS bound for continuous event image

For the continuous event image H_c (1), our SoS upper bound (denoted $\overline{S}_c$) is defined as

$$\overline{S}_c(\mathbb{B}) := \sum_{j=1}^{P} \overline{H}_c(\mathbf{x}_j; \mathbb{B})^2, \tag{16}$$

where $\overline{H}_c(\mathbf{x}_j; \mathbb{B})$ is an upper bound on the value of H_c at $\mathbf{x}_j$. To obtain $\overline{H}_c(\mathbf{x}_j; \mathbb{B})$, we bound the position

$$\{\mathbf{u}_i' = f(\mathbf{u}_i, t_i; \boldsymbol{\omega}) \mid \boldsymbol{\omega} \in \mathbb{B}\} \tag{17}$$

of each warped event, under all possible $\boldsymbol{\omega} \in \mathbb{B}$ for the warping function (9). To this end, let $\boldsymbol{\omega}_\mathbf{c}$ be the centre of a cube $\mathbb{B}$, and $\boldsymbol{\omega}_\mathbf{p}$ and $\boldsymbol{\omega}_\mathbf{q}$ be opposite corners of $\mathbb{B}$. Define

$$\alpha_i(\mathbb{B}) := 0.5 \|\boldsymbol{\omega}_\mathbf{p} t_i - \boldsymbol{\omega}_\mathbf{q} t_i\|_2. \tag{18}$$

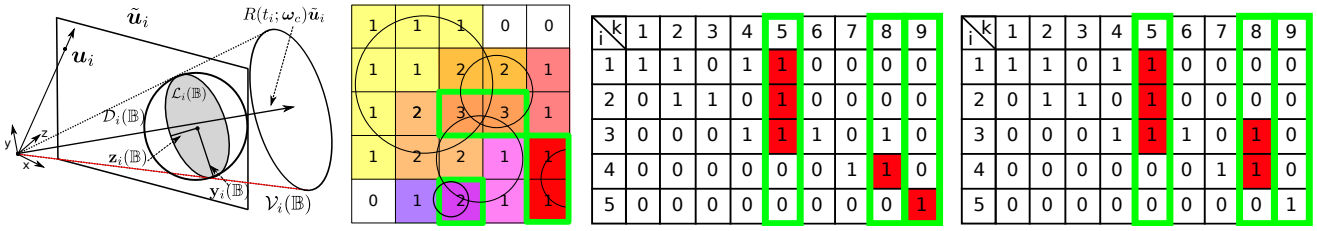


Figure 2. (a) The vector $R(t_i,\boldsymbol{\omega})\tilde{\mathbf{u}}_i$ for all $\boldsymbol{\omega}\in\mathbb{B}$ lies in the cone $\mathcal{V}_i(\mathbb{B})$, and the projection of $\mathcal{V}_i(\mathbb{B})$ onto the image plane is an elliptical region $\mathcal{L}_i(\mathbb{B})$. (b) Example with 5×5 pixel image and $N=5$ events; there are thus $N=5$ discs $\{\mathcal{D}_i(\mathbb{B})\}_{i=1}^5$ on the image. The value in a pixel is the number of discs that intersect the pixel. Pixels with the same color are from the same connected component (CC); there are 9 CCs in this example. (c)(d) Incidence matrix $\mathbf{M}$ corresponding to the example in (b). The solution of IQP and R-IQP are indicated respectively in (c) and (d), where entries (i,k) of $\mathbf{Z}$ that are set to 1 are marked in red. The dominant columns of $\mathbf{M}$ are outlined in green, and their corresponding CCs are also outlined in green in (b).

Then, the following inequality can be established

$$\angle(\mathbf{R}(t_i;\boldsymbol{\omega}_\mathbf{c})\tilde{\mathbf{u}}_i,\mathbf{R}(t_i;\boldsymbol{\omega})\tilde{\mathbf{u}}_i)\leq\alpha_i(\mathbb{B}), \tag{19}$$

which is an extension of [15, Lemma 3.2]. Intuitively, (19) states that the rotated vector $\mathbf{R}(t_i;\boldsymbol{\omega})\tilde{\mathbf{u}}_i$ under all $\boldsymbol{\omega}\in\mathbb{B}$ must lie within the cone

$$\mathcal{V}_i(\mathbb{B}):=\left\{\tilde{\mathbf{u}}\in\mathbb{R}^3\mid\angle(\mathbf{R}(t_i;\boldsymbol{\omega}_\mathbf{c})\tilde{\mathbf{u}}_i,\tilde{\mathbf{u}})\leq\alpha_i(\mathbb{B})\right\}. \tag{20}$$

Fig. 2a illustrates the cone $\mathcal{V}_i(\mathbb{B})$. Now, the pinhole projection of all the rays in $\mathcal{V}_i(\mathbb{B})$ yields the 2D region

$$\mathcal{L}_i(\mathbb{B})=\left\{\mathbf{x}=\frac{\mathbf{K}^{(1:2)}\tilde{\mathbf{u}}}{\mathbf{K}^{(3)}\tilde{\mathbf{u}}}\,\middle|\,\tilde{\mathbf{u}}\in\mathcal{V}_i(\mathbb{B})\right\}, \tag{21}$$

which is an elliptical region [23, Chap. 2]; see Fig. 2a. Further, the centre $\mathbf{c}_i(\mathbb{B})$, semi-major axis $\mathbf{y}_i(\mathbb{B})$ and semi-minor axis $\mathbf{z}_i(\mathbb{B})$ of $\mathcal{L}_i(\mathbb{B})$ can be analytically determined (see the supplementary material). We further define

$$\mathcal{D}_i(\mathbb{B})=\left\{\mathbf{x}\in\mathbb{R}^2\,\middle|\,\|\mathbf{x}-\mathbf{c}_i(\mathbb{B})\|\leq\|\mathbf{y}_i(\mathbb{B})\|\right\}, \tag{22}$$

i.e., the smallest disc that contains $\mathcal{L}_i(\mathbb{B})$.

By construction, $\mathcal{D}_i(\mathbb{B})$ fully contains the set of positions that $\mathbf{u}_i'$ can take for all $\boldsymbol{\omega}\in\mathbb{B}$, i.e., the set (17). We thus define the upper bound on the pixel values of H_c as

$$\overline{H}_c(\mathbf{x}_j;\mathbb{B})=\sum_{i=1}^N\boldsymbol{\delta}\left(\max\left(\|\mathbf{x}_j-\mathbf{c}_i(\mathbb{B})\|-\|\mathbf{y}_i(\mathbb{B})\|,0\right)\right). \tag{23}$$

Intuitively, we take the distance of $\mathbf{x}_j$ to the boundary of $\mathcal{D}_i(\mathbb{B})$ to calculate the intensity, and if $\mathbf{x}_j$ is within the disc then the distance is zero.

Lemma 1.

$$\overline{H}_c(\mathbf{x}_j;\mathbb{B})\geq\max_{\boldsymbol{\omega}\in\mathbb{B}}H_c(\mathbf{x}_j;\boldsymbol{\omega}) \tag{24}$$

with equality achieved if $\mathbb{B}$ is singleton, i.e., $\mathbb{B}=\{\boldsymbol{\omega}\}$.

Proof. See supplementary material. □

Given Lemma 1, it is clear that $\overline{S}_c(\mathbb{B})$ satisfies the conditions (see Sec. 3) to be a valid component in the upper bound (13) for the continuous event image H_c.

3.2. SoS bound for discrete event image

Given the N discs $\{\mathcal{D}_i(\mathbb{B})\}_{i=1}^N$ associated with the N events, define the intersection matrix $\mathbf{T}\in\{0,1\}^{N\times P}$:

$$\mathbf{T}_{i,j}=\begin{cases}1 & \mathcal{D}_i(\mathbb{B})\text{ intersects pixel }\mathbf{x}_j;\\ 0 & \text{otherwise.}\end{cases} \tag{25}$$

The disc-pixel intersections can be computed efficiently using established techniques [10, 30]. We assume $\sum_{j=1}^P\mathbf{T}_{i,j}>0$ for all i, i.e., each disc intersects at least one pixel. If there are discs that lie beyond the image plane, we ignore these discs without loss of generality.

A direct extension of $\overline{H}_c$ (23) to the discrete case would be to calculate the pixel upper bound value as

$$\overline{H}_d(\mathbf{x}_j;\mathbb{B})=\sum_{i=1}^N\mathbf{T}_{i,j}, \tag{26}$$

i.e., number of discs that intersect the pixel; see Fig. 2b. This can however be overly pessimistic, since the pixel value for the discrete event image (3) satisfies

$$\sum_{j=1}^P H_d(\mathbf{x}_j;\boldsymbol{\omega})\leq N\implies\sum_{j=1}^P H_d(\mathbf{x}_j;\boldsymbol{\omega})^2\leq N^2, \tag{27}$$

whereas by using (26),

$$\sum_{j=1}^P\overline{H}_d(\mathbf{x}_j;\mathbb{B})\leq PN\implies\sum_{j=1}^P\overline{H}_d(\mathbf{x}_j;\boldsymbol{\omega})^2\leq(PN)^2. \tag{28}$$

Note that P is the number of pixels (e.g., $P = 240 \times 320 = 76k$ for IniVation Davis 240C [3]), thus $(PN)^2 \gg N^2$.

To get a tighter bound, we note that the discs $\{\mathcal{D}_i(\mathbb{B})\}_{i=1}^N$ partition X into a set of connected components (CC)

$$\{\mathcal{G}_k\}_{k=1}^K, \tag{29}$$

where each $\mathcal{G}_k$ is a connected set of pixels that are intersected by the same discs; see Fig. 2b. Then, define the incidence matrix $\mathbf{M} \in \{0,1\}^{N \times K}$, where

$$\mathbf{M}_{i,k} = \begin{cases} 1 & \exists \mathbf{x}_j \in \mathcal{G}_k \text{ such that } \mathbf{T}_{i,j} = 1; \\ 0 & \text{otherwise.} \end{cases} \tag{30}$$

In words, $\mathbf{M}_{i,k} = 1$ if $\mathcal{D}_i(\mathbb{B})$ is a disc that intersect to form $\mathcal{G}_k$. We then formulate the integer quadratic program

$$\begin{aligned} \overline{S}_d^*(\mathbb{B}) = \max_{\mathbf{Z} \in \{0,1\}^{N \times K}} \quad & \sum_{k=1}^K \left(\sum_{i=1}^N \mathbf{Z}_{i,k} \mathbf{M}_{i,k} \right)^2 \\ \text{s.t.} \quad & \mathbf{Z}_{i,k} \leq \mathbf{M}_{i,k}, \quad \forall i,k, \\ & \sum_{k=1}^K \mathbf{Z}_{i,k} = 1, \quad \forall i. \end{aligned} \tag{IQP}$$

In words, choose a set of CCs that are intersected by as many discs as possible, while ensuring that each disc is selected exactly once. Intuitively, IQP warps the events (under uncertainty $\boldsymbol{\omega} \in \mathbb{B}$) into "clusters" that are populated by as many events as possible, to encourage fewer clusters and higher contrast. See Fig. 2c for a sample solution of IQP.

Lemma 2.

$$\overline{S}_d^*(\mathbb{B}) \geq \max_{\boldsymbol{\omega} \in \mathbb{B}} \sum_{j=1}^P H_d(\mathbf{x}_j; \boldsymbol{\omega})^2, \tag{31}$$

with equality achieved if $\mathbb{B}$ is singleton, i.e., $\mathbb{B} = \{\boldsymbol{\omega}\}$.

Proof. See supplementary material. □

Solving IQP is challenging, not least because $\{\mathcal{G}_k\}_{k=1}^K$ and $\mathbf{M}$ are costly to compute and store (the number of CCs is exponential in N). To simplify the problem, first define the *density* of a CC $\mathcal{G}_k$ (corresponding to column $\mathbf{M}_{:,k}$) as

$$\Delta_k = \sum_{i=1}^N \mathbf{M}_{i,k}. \tag{32}$$

We say that a column $\mathbf{M}_{:,\eta}$ of $\mathbf{M}$ is *dominant* if there exists a subset $\Lambda \subset \{1, \dots, K\}$ (including $\Lambda = \emptyset$) such that

$$\mathbf{M}_{i,k} \leq \mathbf{M}_{i,\eta} \quad \forall i \in \{1, \dots, N\}, \forall k \in \Lambda, \tag{33}$$

whereas for all $k \notin \Lambda$, the above does not hold. In words, the 1 elements of columns in Λ is a subset of the 1 elements of $\mathbf{M}_{:,\eta}$. Geometrically, a dominant column $\mathbf{M}_{:,\eta}$ corresponds to a CC $\mathcal{G}_\eta$ such that for all discs that intersect to form the CC, $\mathcal{G}_\eta$ is the densest CC that they intersect with; mathematically, there exists $\mathcal{D}_i(\mathbb{B}) \supseteq \mathcal{G}_\eta$ such that

$$\max_{\mathbf{x}_j \in \mathcal{D}_i(\mathbb{B})} \sum_{i=1}^N \mathbf{T}_{i,j} = \sum_{i=1}^N \mathbf{M}_{i,\eta}. \tag{34}$$

Figs. 2b and 2c illustrate dominant columns.

Let $\mathbf{M}' \in \{0,1\}^{N \times K'}$ contain only the dominant columsn of $\mathbf{M}$. Typically, $K' \ll K$, and $\mathbf{M}'$ can be computed directly without first building $\mathbf{M}$, as shown in Algorithm 2. Intuitively, the method loops through the discs and incrementally keeps track of the densest CCs to form $\mathbf{M}'$.

Algorithm 2 Computing dominant columns $\mathbf{M}'$.

Require: Pixels $\{\mathbf{x}_j\}_{j=1}^P$, set of discs $\{\mathcal{D}_i(\mathbb{B})\}_{i=1}^N$ (22).
$\mathbf{T} \leftarrow N \times P$ intersection matrix (25) from discs.
$\{\overline{H}_d(\mathbf{x}_j; \mathbb{B})\}_{j=1}^P \leftarrow$ Pixel upper bound image (26).
$\{a_j\}_{j=1}^P \leftarrow$ Array of P elements initialised to 0.
$\mathbf{M}' \leftarrow [\,]$ (empty matrix).
for $i = 1, \dots, N$ **do**
  $c_{max} \leftarrow \max_{\mathbf{x}_j \in \mathcal{D}_i(\mathbb{B})} \overline{H}_d(\mathbf{x}_j; \mathbb{B})$.
  $\mathcal{R} \leftarrow \{\mathbf{x}_j \in \mathcal{D}_i(\mathbb{B}) \mid \overline{H}_d(\mathbf{x}_j; \mathbb{B}) = c_{max}, a_j = 0\}$.
  while $\mathcal{R}$ is not empty **do**
    Pick a pixel $\mathbf{x}_j \in \mathcal{R}$ and $a_j \leftarrow 1$.
    $\mathbf{M}' \leftarrow [\mathbf{M}' \quad \mathbf{T}_{:,j}]$ and $\mathcal{R} \leftarrow \mathcal{R} \setminus \{\mathbf{x}_j\}$.
    for $\mathbf{x}_\ell \in \mathcal{R}$ **do**
      if $\mathbf{T}_{:,\ell} = \mathbf{T}_{:,j}$ **then**
        $a_\ell \leftarrow 1$ and $\mathcal{R} \leftarrow \mathcal{R} \setminus \{\mathbf{x}_\ell\}$.
      end if
    end for
  end while
end for
return $\mathbf{M}'$.

Lemma 3. *Problem IQP has the same solution if $\mathbf{M}$ is replaced with $\mathbf{M}'$.*

Proof. See supplementary material. □

It is thus sufficient to formulate IQP based on the dominant columns $\mathbf{M}'$. Further, we relax IQP into

$$\begin{aligned} \overline{S}_d(\mathbb{B}) = \max_{\mathbf{Z} \in \{0,1\}^{N \times K'}} \quad & \sum_{k=1}^{K'} \left(\sum_{i=1}^N \mathbf{Z}_{i,k} \mathbf{M}'_{i,k} \right)^2 \\ \text{s.t.} \quad & \mathbf{Z}_{i,k} \leq \mathbf{M}'_{i,k}, \quad \forall i,k, \\ & \sum_{k=1}^{K'} \sum_{i=1}^N \mathbf{Z}_{i,k} = N. \end{aligned} \tag{R-IQP}$$

where we now allow discs to be selected more than once. Since enforcing $\sum_{k=1}^{K'} \mathbf{Z}_{i,k} = 1$ for all i implies

$\sum_{k=1}^{K'}\sum_{i=1}^{N}\mathbf{Z}_{i,k} = N$, R-IQP is a valid relaxation. See Fig. 2d for a sample result of R-IQP, and cf. Fig. 2c.

Lemma 4.

$$\overline{S}_d(\mathbb{B}) \geq \overline{S}^*_d(\mathbb{B}) \tag{35}$$

with equality achieved if $\mathbb{B}$ *is singleton, i.e.,* $\mathbb{B} = \{\boldsymbol{\omega}\}$.

Proof. See supplementary material. □

Bound computation and tightness R-IQP admits a simple solution. First, compute the densities $\{\Delta_k\}_{k=1}^{K'}$ of the columns of $\mathbf{M}'$. Let $\Delta_{(k)}$ be the k-th highest density, i.e.,

$$\Delta_{(k_1)} \geq \Delta_{(k_2)} \text{ if } k_1 < k_2. \tag{36}$$

Obtain γ as the largest integer such that

$$\sum_{k=1}^{\gamma}\Delta_{(k)} < N. \tag{37}$$

Then, the SoS upper bound for the discrete event image is

$$\overline{S}_d(\mathbb{B}) = \sum_{k=1}^{\gamma}\Delta_{(k)}^2 + \left(N - \sum_{k=1}^{\gamma}\Delta_{(k)}\right)^2. \tag{38}$$

Intuitively, the procedure greedily takes the densest CCs while ensuring that the quota of N discs is not exceeded. Then, any shortfall in the number of discs is met using the next largest CC partially. Given $\mathbf{M}'$, the costliest routine is just the sorting of the column sums of $\mathbf{M}'$.

Given the final constraint in R-IQP, it is clear that $\overline{S}_d(\mathbb{B}) \leq N^2$. This represents a much tighter SoS upper bound than $\sum_{j=1}^{P}\overline{H}_d(\mathbf{x}_j;\boldsymbol{\omega})^2$; see (28).

3.3. Lower bound of mean pixel value

For the continuous event image (1), the lower bound of the pixel value is the "reverse" of the upper bound (23), i.e.,

$$\underline{H}_c(\mathbf{x}_j;\mathbb{B}) = \sum_{i=1}^{N}\boldsymbol{\delta}\left(\|\mathbf{x}_j - \mathbf{c}_i(\mathbb{B})\| + \|\mathbf{y}_i(\mathbb{B})\|\right), \tag{39}$$

whereby for each $\mathcal{D}_i(\mathbb{B})$, we take the maximum distance between $\mathbf{x}_j$ and a point on the disc. Then, the lower bound of the mean pixel value is simply

$$\underline{\mu}_c(\mathbb{B}) = \frac{1}{P}\sum_{j=1}^{P}\underline{H}_c(\mathbf{x}_j;\mathbb{B}). \tag{40}$$

In the discrete event image (3), if all the N discs lie fully in the image plane, the lower bound can be simply calculated as N/P. However, this ideal case rarely happens, hence the the lower bound on the mean pixel vale is

$$\underline{\mu}_d(\mathbb{B}) = \frac{1}{P}\sum_{i=1}^{N}\mathbb{I}(\mathcal{D}_i \text{ fully lie in the image plane}). \tag{41}$$

See the supplementary material for proofs of the correctness of the above lower bounds.

3.4. Computational cost and further acceleration

Our BnB method is able to process $N \approx 50,000$ events in $\approx$ 300 seconds. While this does not allow online low latency event processing, it is nonetheless useful for event sensing applications that permit offline computations, e.g., video stabilisation with post-hoc correction. Note that a local method can take up to 30 seconds to perform CM on the same input, which also does not enable online processing[1] (Sec. 4 will present more runtime results).

There is potential to speed-up our algorithm using GPUs. For example, in the bound computations for the discrete event image case, the disc-pixel intersection matrix $\mathbf{T}$ (25) could be computed using GPU-accelerated ray tracing [5, 25], essentially by backprojecting each pixel and intersecting the ray with the cones (20) in parallel. We leave GPU acceleration as future work.

4. Results

We first examine the runtime and solution quality of our algorithms, before comparing against state-of-the-art methods in the literature. The results were obtained on a standard desktop with a 3.0GHz Intel i5 CPU and 16GB RAM.

4.1. Comparison of bounding functions

The aim here is to empirically compare the performance of BnB (Algorithm 1) with continuous and discrete event images. We call these variants **CMBnB1** and **CMBnB2**.

For this experiment, a 10 ms subsequence (which contains about $N = 50,000$ events) of the *boxes* data [14] was used. The underlying camera motion was a pure rotation.

For CMBnB1, a Gaussian kernel with bandwidth 1 pixel was used (following [13, Sec. 2]). Fig. 3 plots the upper and lower bound values over time in a typical run of Algorithm 1. It is clear that the discrete case converged much faster than the continuous case; while CMBnB2 terminated at about $12k$ iterations, CMBnB1 requried no fewer than $30k$ iterations. It is evident from Fig. 3 that this difference in performance is due to the much tighter bounding in the discrete case. The next experiment will include a comparison of the solution quality of CMBnB1 and CMBnB2.

4.2. Qualitative comparisons

To highlight the importance of globally optimal CM, we tested on select 10 ms subsequences (about $N = 50k$ events each) from the *boxes* data [14]—in the next experiment, a more comprehensive experiment and quantitative benchmarking will be described. Here, on the subsequences chosen, we compared BnB against the following methods:

[1] Since the implementation of [13] was not available, we used the conjugate gradient solver in `fmincon` (Matlab) to solve CM locally optimally. Conjugate gradient solvers specialised for CM could be faster, though the previous works [11, 13, 17, 28] did not report online performance.

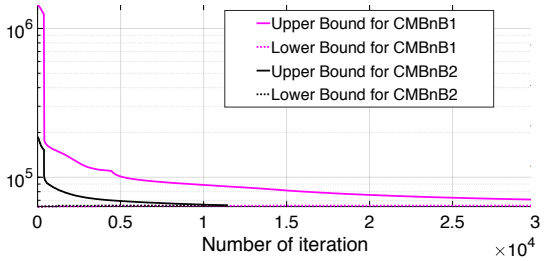


Figure 3. Upper and lower bound evolution in branch-and-bound.

- **CMGD1**: locally optimal solver (`fmincon` from Matlab) was used to perform CM with initialisation $\boldsymbol{\omega} = 0$ (equivalent to identity rotation).
- **CMGD2**: same as above, but initialised with the optimised $\boldsymbol{\omega}$ from the previous 10 ms time window.

Both local methods were executed on the continuous event image with Gaussian kernel of bandwidth 1 pixel.

Fig. 4 depicts motion compensated event images from two subsequences (Subseq 1 and Subseq 2); see supplementary material for more results. The examined cases show that the local methods (both CMGD1 and CMGD2) can indeed often converge to bad local solutions. Contrast this to BnB which always produced sharp event images.

These results also show that CM based on continuous and discrete event images yield practically identical solutions. Since CMBnB2 usually converges much faster than CMBnB1, we use CMBnB2 in the remaining experiments.

4.3. Quantitative benchmarking

We performed benchmarking using publicly available datasets [8, 22]. We introduced two additional variants to CMGD1 and CMGD2:

- **CMRW1**: A variant of CM [29] that uses a different objective function (called reward):

$$Rw(\boldsymbol{\omega}) = C(\boldsymbol{\omega}) + \frac{1}{P}\sum_{j=1}^{P}(e^{(-H(\mathbf{x}_j;\boldsymbol{\omega}))} + e^{(H(\mathbf{x}_j;\boldsymbol{\omega}))}).$$

  The initial solution is taken as $\boldsymbol{\omega} = 0$.
- **CMRW2**: Same as CMRW1 but initialised with the optimised $\boldsymbol{\omega}$ from the previous subsequence.

We also compared against EventNet [27], which is based on deep learning. However, similar to the error reported in [27], we found that the error for EventNet was much higher than the error of the CM methods (e.g., the translated angular velocity error of the maximum of EventNet is 17.1%, while it is around 5% for CM). The lack of publicly available implementation also hampered objective testing of EventNet. We thus leave comparisons against deep learning methods as future work.

4.3.1 Rotational motion in indoor scene

We used event sequences *poster*, *boxes* and *dynamic* from [14, 22], which were recorded using a Davis 240C [4] under rotational motion over a static indoor scene. The ground truth motion was captured using a motion capture system. Each sequence has a duration of 1 minute and around 100 million events. For these sequences, the rotational motion was minor in a large part of the sequences (thereby producing trivial instances to CM), thus in our experiment we used only the final 15 seconds of each sequence, which tended to have more significant motions.

We split each sequence into contiguous 10 ms subsequences which were then subject to CM. For *boxes* and *poster*, each CM instance was of size $N \approx 50k$, while for *dynamic*, each instance was of size $N \approx 25k$. For each CM instance, let $\bar{\boldsymbol{\omega}}$ and $\hat{\boldsymbol{\omega}}$ be the ground truth and estimated parameters. An error metric we used is

$$\epsilon = \|\bar{\boldsymbol{\omega}} - \hat{\boldsymbol{\omega}}\|_2. \tag{42}$$

Our second error metric, which considers only differences in angular rate, is

$$\phi = |\|\bar{\boldsymbol{\omega}}\|_2 - \|\hat{\boldsymbol{\omega}}\|_2|. \tag{43}$$

Fig. 5a plots ϵ over all the CM instances for the *boxes* sequence, while Table 1 shows the average (μ) and standard deviation (σ) of ϵ and ϕ over all CM instances.

Amongst the local methods, the different objective functions did not yield significant differences in quality. The more important observation was that CMGD2/CMRW2 gave solutions of much higher quality than CMGD1/CMRW1, which confirms that good initialisation is essential for the local methods. Due to its exact nature, CMBnB provided the best quality in rotation estimation; its standard deviation of ϵ is also lower, indicating a higher stability over the sequences. Moreover, CMBnB does not require any initialisation, unlike the local methods.

For the three sequences used (*dynamic*, *boxes*, *poster*), the maximum absolute angular velocities are 500, 670 and 1000 deg/s respectively [14, 22]. The average ϕ error of CMBnB of 10.09, 17.97 and 46.34 deg/s thus translate into 2.2%, 2.7% and 4.6% of the maximum, respectively.

Runtime The average runtimes of CMBnB over all instances in the three sequences (*dynamic*, *boxes*, *poster*) were 163.2, 278.3 and 320.6 seconds. CMGD optimised with the conjugate gradient solver in `fmincon` has average runtimes of 20.2, 31.1 and 35.3 seconds.

4.3.2 Attitude estimation

We repeated the above experiment on the event-based star tracking (attitude estimation) dataset of [7, 8], which contains 11 event sequences of rotational motion over a star field. Each sequence has a constant angular velocity of 4

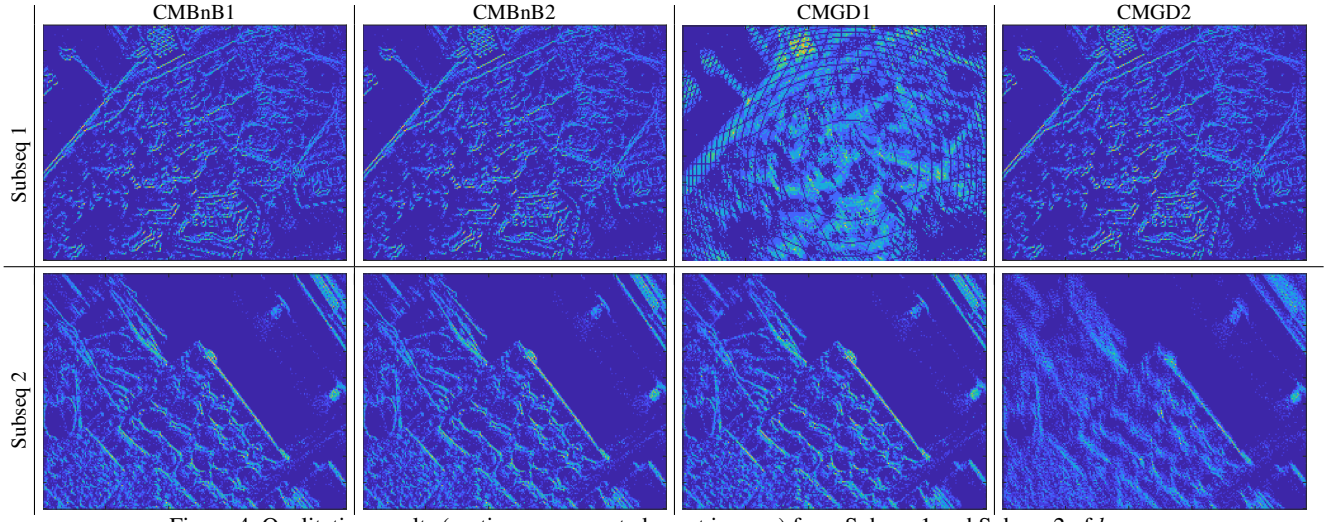
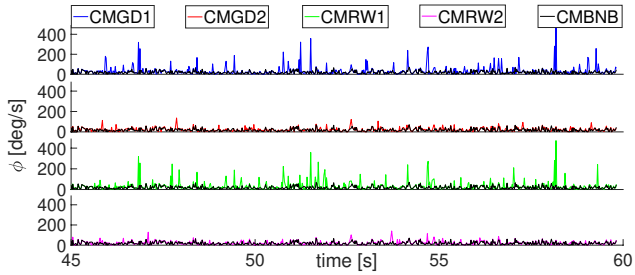


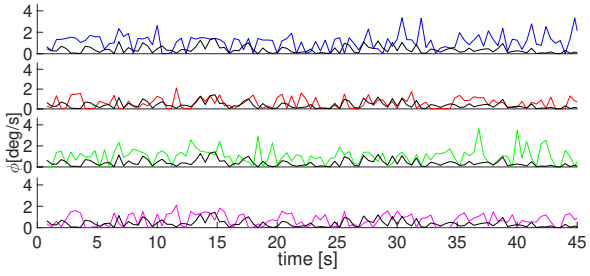
Figure 4. Qualitative results (motion compensated event images) from Subseq 1 and Subseq 2 of *boxes*.

Method	*dynamic*				*boxes*				*poster*			
	$\mu(\epsilon)$	$\mu(\phi)$	$\sigma(\epsilon)$	$\sigma(\phi)$	$\mu(\epsilon)$	$\mu(\phi)$	$\sigma(\epsilon)$	$\sigma(\phi)$	$\mu(\epsilon)$	$\mu(\phi)$	$\sigma(\epsilon)$	$\sigma(\phi)$
CMGD1	21.52	20.07	24.38	31.13	31.29	31.47	34.30	45.94	56.58	54.92	47.03	58.95
CMGD2	15.09	13.31	10.39	12.08	22.01	21.70	12.79	18.83	49.64	50.12	35.93	42.77
CMRW1	21.03	18.59	25.41	28.83	32.28	32.23	36.11	46.01	59.03	58.71	49.49	60.87
CMRW2	14.55	12.29	9.85	11.21	21.95	21.41	13.71	18.42	49.49	50.04	37.51	43.35
CMBnB	**11.93**	**10.09**	**7.82**	**8.74**	**18.76**	**17.97**	**10.06**	**14.66**	**44.34**	**46.34**	**24.79**	**36.79**

Table 1. Average and standard deviation of ϵ and ϕ over all CM instances in *boxes*, *dynamic*, and *poster* (best result bolded).



(a) *boxes*



(b) Sequence 1

Figure 5. Error ϕ of all CM methods (plotted against time) for (a) *boxes* sequence [14] and (b) Sequence 1 of [8]. In each subplot, the error of CMBNB is superimposed for reference.

deg/s over a duration of 45 seconds and around 1.5 million events. We split each sequence into 400 ms subsequences, which yielded $N \approx 15,000$ events per subsequence. Fig. 5b

Method	$\mu(\epsilon)$	$\mu(\phi)$	$\sigma(\epsilon)$	$\sigma(\phi)$
CMGD1	0.448	0.652	0.314	0.486
CMGD2	0.294	0.423	0.232	0.323
CMRW1	0.429	0.601	0.346	0.468
CMRW2	0.318	0.461	0.234	0.341
CMBnB	**0.174**	**0.234**	**0.168**	**0.217**

Table 2. Average and standard deviation of ϵ and ϕ over all CM instances in the star tracking dataset (best result bolded).

plots the ϕ errors for Sequence 1 in the dataset. The average errors and standard deviation over all CM instances are shown in Table 2. Again, CMBnB gave the highest quality solutions; its average ϕ error of 0.234 deg/s translate into 5.8% of the maximum. The average runtime of CMBnB and CMGD over all instances were 80.7 and 11.1 seconds.

5. Conclusions

We proposed a novel globally optimal algorithm for CM based on BnB. The theoretical validity of our algorithm has been established, and the experiments showed that it greatly outperformed local methods in terms of solution quality.

Acknowledgements

This work was supported by ARC DP200101675. We thank S. Bagchi for the implementation of CM and E. Mueggler for providing the dataset used in experiments.

References

[1] http://en.wikipedia.org/w/index.php?title=Exponential%20map%20(Lie%20theory)&oldid=925866493. 2

[2] Event-based vision resources. https://github.com/uzh-rpg/event-based_vision_resources. 2

[3] Jean-Charles Bazin, Hongdong Li, In So Kweon, Cédric Demonceaux, Pascal Vasseur, and Katsushi Ikeuchi. A branch-and-bound approach to correspondence and grouping problems. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1565–1576, 2012. 5

[4] Christian Brandli, Raphael Berner, Minhao Yang, Shih-Chii Liu, and Tobi Delbruck. A 240× 180 130 db 3 μs latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits*, 49(10):2333–2341, 2014. 7

[5] Nathan A Carr, Jared Hoberock, Keenan Crane, and John C Hart. Fast gpu ray tracing of dynamic meshes using geometry images. In *Proceedings of Graphics Interface 2006*, pages 203–209. Canadian Information Processing Society, 2006. 6

[6] Andrea Censi and Davide Scaramuzza. Low-latency event-based visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 703–710. IEEE, 2014. 2

[7] Tat-Jun Chin and Samya Bagchi. Event-based star tracking via multiresolution progressive hough transforms. *arXiv preprint arXiv:1906.07866*, 2019. 1, 2, 7

[8] Tat-Jun Chin, Samya Bagchi, Anders Eriksson, and Andre van Schaik. Star tracking using an event camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 1, 2, 7, 8

[9] Tobi Delbruck and Manuel Lang. Robotic goalie with 3 ms reaction time at 4% cpu load using event-based dynamic vision sensor. *Frontiers in neuroscience*, 7:223, 2013. 1

[10] James D Foley, Foley Dan Van, Andries Van Dam, Steven K Feiner, John F Hughes, J Hughes, and Edward Angel. *Computer graphics: principles and practice*, volume 12110. Addison-Wesley Professional, 1996. 4

[11] Guillermo Gallego, Mathias Gehrig, and Davide Scaramuzza. Focus is all you need: Loss functions for event-based vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12280–12289, 2019. 2, 6

[12] Guillermo Gallego, Jon EA Lund, Elias Mueggler, Henri Rebecq, Tobi Delbruck, and Davide Scaramuzza. Event-based, 6-dof camera tracking from photometric depth maps. *IEEE transactions on pattern analysis and machine intelligence*, 40(10):2402–2412, 2017. 2

[13] Guillermo Gallego, Henri Rebecq, and Davide Scaramuzza. A unifying contrast maximization framework for event cameras, with applications to motion, depth, and optical flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3867–3876, 2018. 1, 2, 6

[14] Guillermo Gallego and Davide Scaramuzza. Accurate angular velocity estimation with an event camera. *IEEE Robotics and Automation Letters*, 2(2):632–639, 2017. 1, 2, 6, 7, 8

[15] Richard I Hartley and Fredrik Kahl. Global optimization through rotation space search. *International Journal of Computer Vision*, 82(1):64–79, 2009. 3, 4

[16] Reiner Horst and Hoang Tuy. *Global optimization: deterministic approaches*. Springer, 1996. 3

[17] Mina A Khoei, Sio-hoi Ieng, and Ryad Benosman. Asynchronous event-based motion processing: From visual events to probabilistic sensory representation. *Neural computation*, 31(6):1114–1138, 2019. 6

[18] Hanme Kim, Ankur Handa, Ryad Benosman, Sio-Hoi Ieng, and Andrew J Davison. Simultaneous mosaicing and tracking with an event camera. *J. Solid State Circ*, 43:566–576, 2008. 2

[19] Hanme Kim, Stefan Leutenegger, and Andrew J Davison. Real-time 3d reconstruction and 6-dof tracking with an event camera. In *European Conference on Computer Vision*, pages 349–364. Springer, 2016. 2

[20] Beat Kueng, Elias Mueggler, Guillermo Gallego, and Davide Scaramuzza. Low-latency visual odometry using event-based feature tracks. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 16–23. IEEE, 2016. 2

[21] Elias Mueggler, Basil Huber, and Davide Scaramuzza. Event-based, 6-dof pose tracking for high-speed maneuvers. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2761–2768. IEEE, 2014. 1

[22] Elias Mueggler, Henri Rebecq, Guillermo Gallego, Tobi Delbruck, and Davide Scaramuzza. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *The International Journal of Robotics Research*, 36(2):142–149, 2017. 1, 7

[23] David Mumford. *Algebraic geometry I: complex projective varieties*. Springer Science & Business Media, 1995. 4

[24] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006. 2

[25] Stefan Popov, Johannes Günther, Hans-Peter Seidel, and Philipp Slusallek. Stackless kd-tree traversal for high performance gpu ray tracing. In *Computer Graphics Forum*, volume 26, pages 415–424. Wiley Online Library, 2007. 6

[26] Bharath Ramesh, Hong Yang, Garrick Michael Orchard, Ngoc Anh Le Thi, Shihao Zhang, and Cheng Xiang. Dart: distribution aware retinal transform for event-based cameras. *IEEE transactions on pattern analysis and machine intelligence*, 2019. 1

[27] Yusuke Sekikawa, Kosuke Hara, and Hideo Saito. Eventnet: Asynchronous recursive event processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3887–3896, 2019. 2, 7

[28] Timo Stoffregen, Guillermo Gallego, Tom Drummond, Lindsay Kleeman, and Davide Scaramuzza. Event-based motion segmentation by motion compensation. *arXiv preprint arXiv:1904.01293*, 2019. 1, 2, 6

[29] Timo Stoffregen and Lindsay Kleeman. Event cameras, contrast maximization and reward functions: An analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2, 7

[30] Jerry R Van Aken. An efficient ellipse-drawing algorithm. *IEEE Computer Graphics and Applications*, 4(9):24–35, 1984. 4

[31] Antoni Rosinol Vidal, Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza. Ultimate slam? combining events,

images, and imu for robust visual slam in hdr and high-speed scenarios. *IEEE Robotics and Automation Letters*, 3(2):994–1001, 2018. 1

[32] David Weikersdorfer, David B Adrian, Daniel Cremers, and Jörg Conradt. Event-based 3d slam with a depth-augmented dynamic vision sensor. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 359–364. IEEE, 2014. 1

[33] Chengxi Ye. *Learning of dense optical flow, motion and depth, from sparse event cameras*. PhD thesis, 2019. 2

[34] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22, 2000. 2

[35] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Ev-flownet: Self-supervised optical flow estimation for event-based cameras. *arXiv preprint arXiv:1802.06898*, 2018. 2

[36] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised event-based learning of optical flow, depth, and egomotion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 989–997, 2019. 2