Aggregation Functions for Simultaneous Attitude and Image Estimation With Event Cameras at High Angular Rates

Matthew Ng, Zi Min Er, Gim Song Soh, and Shaohui Foong, *Member, IEEE*

***Abstract*—For fast-moving event cameras, projection of events onto the image frame exhibits smearing of events analogous to high motion blur. For camera attitude estimation, this presents a causality dilemma where motion prior is required to unsmear events, but an image prior is required to estimate motion. This dilemma is typically circumvented by including an IMU to provide motion priors. However, IMUs limited dynamic range of $\pm 2000\,^\circ/\mathrm{s}$ are shown to be insufficient for high angular rate rotorcrafts. Contrast Maximization is an event-only optimization framework that computes the optimal motion compensation parameter while generating an event image simultaneously. This letter analyses the performance of existing aggregation functions of the contrast maximization framework and proposes a non-convolution-based aggregation function that outperforms existing implementations. The use of discrete event images for optimizers is discussed, demonstrating alternate avenues of the framework to exploit. The effect of motion blur in motion-compensated images is defined and studied for Contrast Maximisation at high angular rates. Lastly, the framework is applied to rotation datasets with angular rates exceeding $2000\,^\circ/\mathrm{s}$ to demonstrate high angular rate motion estimation without motion priors.**

***Index Terms*—Aerial systems: perception and autonomy, visual servoing.**

I. Introduction

FIRST conceived in 1991 [1], event cameras are neuromorphic cameras that mimic the neural architecture of the eye for the perception of the world. Unlike traditional cameras, which transmit pixel information frame-wise at a predefined clock speed, event cameras such as the Dynamic Vision Sensor (DVS) [2], [3] transmit pixel information pixel-wise asynchronously. Each pixel in the DVS responds to changes in light intensity. When the change exceeds a predefined threshold, an event at the pixel location is registered, time-stamped, and transmitted out. Depending on the hardware, event cameras have an event throughput between 2 million to 1.2 billion events per second.

Manuscript received September 29, 2021; accepted January 5, 2022. Date of publication February 7, 2022; date of current version February 24, 2022. This letter was recommended for publication by Associate Editor Begoña C. Arrue Arrue and Editor Pauline Pounds upon evaluation of the reviewers' comments. *(Corresponding author: Shaohui Foong.)*

The authors are with the Faculty of Engineering Product Development, Singapore University of Technology and Design, 8 Somapah Rd, Singapore 487372 (e-mail: matthew_ng@mymail.sutd.edu.sg; zimin_er@sutd.edu.sg; sohgimsong@sutd.edu.sg; shao@sutd.edu.sg).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2022.3148982, provided by the authors.

Digital Object Identifier 10.1109/LRA.2022.3148982

One well-studied application for event cameras are its use in camera attitude estimation using image alignment. However, reconstructing an image for events under motion results in a smeared image analogous to motion blur. These methods depend on a priori information to perform image motion compensation to recover a sharp image. The motion prior can be recovered from the IMU [4] or by exploiting environmental conditions to develop a priori for motion compensation [5]. This presents a causality dilemma where motion prior is required to construct an image, but an image is required to recover motion.

The Contrast Maximisation (CM) framework [6] is an open-ended optimisation algorithm that seeks to recover a motion parameter that maximises the contrast of an event image. It is an event only method that simultaneously generates both motion-compensated event image and an instantaneous state estimate. The overall computation complexity of the framework depends on 5 features.

1) The size of the event packet; Either fixed packet size or fixed time intervals (variable packet size)
2) The warp function; Depending on the expected motion to be estimated, affects the number of optimization parameters.
3) The aggregation function which defines how individual events influence the intensity of other events spatially.
4) The loss function which collects the event image into a singular metric for optimization.
5) Lastly the optimization algorithm which can be either local or global, gradient or non-gradient based.

Given the customizability of the framework, any minor changes can affect the accuracy of the warp, speed of computation, and the quality of the resultant event image. As such, this paper builds upon existing CM literature (which focuses on loss functions [7], [8]) to develop a better mathematical and computational understanding of aggregation functions. In addition, the use of non-gradient-based optimizers with a sparse aggregation function is demonstrated; Lastly, as this work focuses on applications at high angular velocity, the impact of motion blur on the CM framework.

II. Methodology

Underpinning the CM framework are aggregation functions. Any event-based method (including methods outside the CM framework) that builds an event image or Image of Warped

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2023, London, UK. Cite as RA-L paper.

Events (IWE) would apply an aggregation function [9]. The choice of aggregation function may also dictate and limit the scope of optimizers usable in the framework. As such a short discourse is dedicated in the CM methodology to introduce the various aggregation functions.

A. Contrast Maximization

The CM framework is a direct estimation method that maximizes the contrast of an event image given an event stream. It assumes, events motion compensated by an optimal motion parameter and aggregated, yields a high contrast event image due to common trajectories of events describing the same texture. Let $\mathcal{E} = \{e_k\}_{k=1}^{N}$ describe an event stream and each event $e_k = (t_k, \mathbf{x}_k, s_k)$ where t_k is the timestamp, $\mathbf{x}_k = (x_k, y_k)$ is the x-y position of the event in an event image, and s_k the sign of intensity change. In the CM framework, events are first warp to a common time reference t_{ref} for a candidate θ which parametrizes the motion of the camera. The warp, defined in (1), geometrically maps events spatially from $\mathbf{x}_k$ to a sub-pixel location $\mathbf{x'}_k$ with the weights of the map restricted temporally by $t_k - t_{ref}$.

$$\mathbf{x'}_k = \mathbf{W}\left(\mathbf{x}_k, t_k, \theta\right) \tag{1}$$

The warp events are used to build an event image, H, defined as the sum of events that fall within the pixel. Theoretically defined using a Dirac delta, since it is untenable to sum infinite for any meaningful calculation, the discrete-time approximation, the Kronecker delta function is adopted, which is 1 if $\mathbf{x} = \mathbf{x'}_k$ (see (2))

$$H(\mathbf{x'}_k) = \sum_{k=1}^{N} \boldsymbol{\delta}\left(\mathbf{x} - \mathbf{x'}_k\right) \tag{2}$$

where $\mathbf{x} = \mathbf{x'}_k \Rightarrow \boldsymbol{\delta}(\mathbf{x} - \mathbf{x'}_k) = 1$. In practice, however, since (1) warps to sub-pixel locations, the Kronecker delta function is adapted to define data associations between events. There are four possible ways:

1) *Kronecker Delta:* A naive single pixel update where $\mathbf{x'}_k$ is rounded to the nearest integer in the Kronecker delta (see (3)).

$$H_1(\mathbf{x'}_k) = \sum_{k=1}^{N} \boldsymbol{\delta}\left(\mathbf{x} - \lfloor \mathbf{x'}_k \rceil\right) \tag{3}$$

where $\lfloor \cdot \rceil$ is the rounding function. The main advantage of this method is the sharp event image produced if $\mathbf{x'}_k$ is optimally mapped since events are hard assigned to a pixel. The drawback of this method is the sparsity of the image due to discontinuities between neighbouring pixel intensities. Discontinuities between event edges propagate down the framework, limiting the effectiveness of optimizers requiring a continuous first derivative.

2) *Fully Connected Gaussian:* Replacing the Kronecker Delta function with a Gaussian function [6], [8] (see (4)) with $\sigma = 1$ pixel.

$$H_2(\mathbf{x'}_k) = \sum_{k=1}^{N} \frac{1}{2\pi\sigma^2} \exp\left(-\frac{\alpha}{2\sigma^2}\right) \tag{4}$$

where $\alpha = (x - x_k)^2 + (y - y_k)^2$. This function assumes event associations at subpixel locations to neighbouring pixels is normally distributive. It produces a smooth contrast map that is ideal for optimization but is extremely costly to compute since the corresponding Gaussian weight must be computed at every event pixel location for all $\mathbf{x'}_k$. For an event camera sensor with M pixels and N is the number of events in an event stream, it is MN operations per function call for the fully connected Gaussian. The operation has a time complexity of $\mathcal{O}(mn)$. If computation time is a nonissue, this method is ideal because event influence is not truncated in image generation.

3) *Bilinear Interpolation Kernel:* Kernelized methods replace the Kronecker Delta function with two steps: a kernel summation followed by Gaussian smoothing. The Kernel sums a subset of $\mathbf{x}$. Bilinear Interpolation Kernel [10] improves on the naive single-pixel update through bilinear voting for 4 neighbouring pixels around $\mathbf{x'}_k$. Let

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \left(\begin{bmatrix} 1 & \lfloor x'_k \rfloor & \lfloor y'_k \rfloor & \lfloor x'_k \rfloor \lfloor y'_k \rfloor \\ 1 & \lfloor x'_k \rfloor & \lceil y'_k \rceil & \lfloor x'_k \rfloor \lceil y'_k \rceil \\ 1 & \lceil x'_k \rceil & \lfloor y'_k \rfloor & \lceil x'_k \rceil \lfloor y'_k \rfloor \\ 1 & \lceil x'_k \rceil & \lceil y'_k \rceil & \lceil x'_k \rceil \lceil y'_k \rceil \end{bmatrix}^{-1} \right)^{T} \begin{bmatrix} 1 \\ x'_k \\ y'_k \\ x'_k y'_k \end{bmatrix} \tag{5}$$

where $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ are floor and ceiling functions respectively, and b the bilinear weights of the 4 pixels surrounding $\mathbf{x'}_k$. Then $I_B(\mathbf{x'}_k) = \sum_{k=1}^{N} b$ is the temporary event image built using the bilinear interpolation kernel.

$$H_3(\mathbf{x'}_k) = I_B(\mathbf{x'}_k) * G(\mathbf{x}) \tag{6}$$

To improve the smoothness of $I_B(\mathbf{x'}_k)$, a Gaussian filter with $\sigma = 1$ pixel is convolved on the image, spreading the influence of $\mathbf{x'}_k$ beyond the 4 neighbours (see (6)).

4) *3x3 Gaussian Kernel:* Replacing I_B with

$$I_G(\mathbf{x'}_k) = H_2(\mathbf{x'}_k) \forall \mathbf{x} = \lfloor \mathbf{x'}_k \rceil \pm 1 \tag{7}$$

$$H_4(\mathbf{x'}_k) = I_G(\mathbf{x'}_k) * G(\mathbf{x}) \tag{8}$$

This kernel replaces the bilinear voting with a 3x3 Gaussian kernel with $\sigma = 1$ pixel [11]. The weights of the kernel are calculated based on (4) for the immediate 9 neighbours in a 3x3 grid, following that a Gaussian smoothing (see (8)). This achieves a similar effect to convolving the event image by a larger σ.

Kernelized methods results in a $4N$ and $9N$ operation cost for I_B and I_G respectively. Gaussian smoothing functions are at worst a $\mathcal{O}(mr^2)$ complexity, where r is the radius of the smoothing kernel. This results in an overall $\mathcal{O}(n + mr^2)$ complexity for H_3 and H_4.

Lastly, the focus of the event image is calculated using a focus loss function. The notion stems from the intuition that if events

describing the same texture are motion-compensated accurately, the aggregation function yields a sharp image. As focus loss functions are well-studied, the loss function *Variance* is adopted for its fast runtime, relatively minimal performance loss as compared to the best performing loss function, and extensive use in CM literature. As such, the contrast of the event image is calculated using (9).

$$C(\theta) = \frac{1}{P_n}\sum_{i=1}^{P_n}\left(H(\mathbf{x}_i) - \mu_H\right)^2 \tag{9}$$

$$\mu_H = \frac{1}{P_n}\sum_{i=1}^{P_n}\left(H(\mathbf{x}_i)\right) \tag{10}$$

where P_n is the total number of pixels in an event image, and μ_H the mean of the event image (see (10)).

B. Pseudo Fully Connected Gaussian

Apart from optimizer choice, the aggregation step is the most computationally expensive part of the framework [6]. Motivated to reduce the time complexity further. We propose a redefinition of the Gaussian function, that achieves linear time complexity, eliminates additional computation of the Jacobian and retains the accuracy achieved by (4). Inspired by the empirical rule of the field of statistics that states: For a normal distribution 98.9% of the observable data falls within 3 standard deviations of the mean. Extending this heuristic to event influence, it can be said that 98.9% of the event influence for $\mathbf{x'}_k$ is expended for events $\mathbf{x'}$ up to 3 standard deviations of $\mathbf{x'}_k$. As such this method replaces the Kronecker delta with a single-step Gaussian function as in (4), but calculating the Gaussian weight for neighbouring pixels $r\sigma$ from $\mathbf{x'}_k$ as shown in (11) where $1 \leq r \leq 3$.

$$H_5(\mathbf{x'}_k) = H_2(\mathbf{x'}_k) \forall \mathbf{x} = \lfloor \mathbf{x'}_k \rceil \pm r\sigma \tag{11}$$

Assuming $r = 3$, $\sigma = 1$ pixel, this means a 7x7 kernel is applied, and the convolution step is omitted since influence outside the 7x7 grid can be considered negligible. As values outside 3σ have a magnitude $\leq 6 \times 10^{-5}$, this reduces the operation cost of the aggregation step to $49N$. Resulting in an aggregation function with linear time complexity ($\mathcal{O}(n)$) which is **independent** of the sensor size.

Given the high temporal resolution of event cameras, the number of events arriving per event sequence is $\approx 20,000/10\,\mathrm{ms}$. The computation cost for a DVS240 ($M = 43,200$) and a 720p camera ($M = 921,600$) will be identical in that $10\,\mathrm{ms}$ window. In practice, it was found that the choice of r depended on the optimizer with more robust optimizers functioning well at $r = 1$ and most optimizers such as AdaGrad optimizing well at $r = 2$.

1) Analytical Derivative of $H_5(\mathbf{x'}_k)$: Since this method, shown later, is numerically equivalent to (4), it is possible to take the analytical derivative of the Gaussian function for optimization. Taking the partial derivative of $H_5(\mathbf{x'}_k)$ with respect to $\mathbf{x'}_k$:

$$\frac{\partial H_5}{\partial \mathbf{x'}_k} = \sum_{k=1}^{N}\frac{\beta}{2\pi\sigma^4}\exp\left(-\frac{\alpha}{2\sigma^2}\right) \forall \mathbf{x} = \lfloor \mathbf{x'}_k \rceil \pm r\sigma \tag{12}$$

TABLE I
SUMMARY OF DIFFERENCES BETWEEN AGGREGATION FUNCTIONS

	H_1	H_2	H_3	H_4	H_5
Time Complexity	$\mathcal{O}(n)$	$\mathcal{O}(mn)$	$\mathcal{O}(n + mr^2)$	$\mathcal{O}(n)$	
Convolution Free	yes	yes	no	no	yes
Independent of Image Size	yes	no	no	no	yes
$\frac{\partial H}{\partial \mathbf{x'}_k} \propto H$	no	yes	no	no	yes

where $\beta = (x - x_k) + (y - y_k)$. When simplified to (12) and $\sigma = 1$, the first-order derivative is shown to be linearly proportional to the zeroth-order Gaussian by β. Since the components of β are precalculated in α, no additional computation is required to generate (12).

The differences between aggregation functions are summarized in Table I. It shows compactly how the proposed function is composed of the individual benefits of the different aggregation functions.

III. EXPERIMENTS

In this section, 3 experiments are conducted, each designed to expose different characteristics of the aggregation function in the framework. First, the proposed aggregation function is validated using a common optimizer, answering the question: *"Can aggregation functions be made faster?"* Second, a discussion on the use of discrete aggregation functions and its impact on optimizer choice, answers the question: *"What other features in the framework be exploited?"* Lastly, we define and describe a type of motion blur unique to event cameras and its effects on the CM framework. All experiments are conducted on a single core i7-3840QM CPU at $2.8\,\mathrm{GHz}$.

In total, three optimizers will be demonstrated:

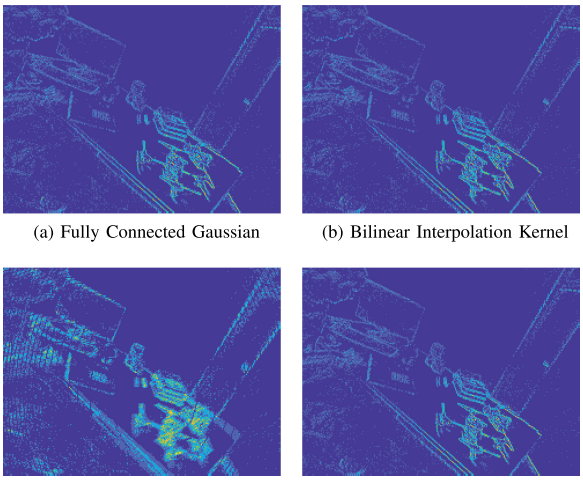
1) Gradient Descent (Local Optimizer)
2) Particle Swarm (Metaheuristic Optimizer)
3) Branch-and-Bound (Global Optimizer)

Motivated by the rich literature available and its simplicity, Particle Swarm Optimization (PSO) was chosen as a medium between the fast gradient-based local optimizers presented in [6], [10], while trading off the certainty and complexity of Branch-and-Bound (BNB) [11] for comparison in the CM framework. It was also chosen to demonstrate the fundamental non-gradient requirement needed by discrete aggregation functions for optimization.

For concise presentation of data, gradient descent (GD), BNB, and PSO are appended with an ordinal numeral referring to the aggregation function used for optimization, e.g. GD using $H_2(\mathbf{x'}_k)$ is abbreviated to GD2 and GD using $H_5(\mathbf{x'}_k)$ is abbreviated to GD5.

A. Validation of Proposed Function

In this subsection, the proposed function is validated and compared. For each method described above, a CM routine is performed on the same set of $10\,\mathrm{ms}$ event subsequence of the *'dynamic_rotation'* Event-Camera dataset [12] which were generated using a DAVIS240C [13]. This $10\,\mathrm{ms}$ subsequence is the



(a) Fully Connected Gaussian (b) Bilinear Interpolation Kernel

(c) 3x3 Gaussian Kernel (d) Proposed

Fig. 1. Event images generated using the Kronecker Delta function from the optimized value of θ for each aggregation function.

TABLE II
OPTIMIZATION RESULTS OF THE INTERIOR-POINT ALGORITHM

	Avg_{FR} (seconds)	T_{FC}	Contrast
GD2	2.89	93	0.893
GD3	0.41	96	0.891
GD4	0.43	46	0.512
GD5	**0.20**	**86**	**0.893**

same 10 ms subsequence provided by [11] in the publicly available '*CMBNB-demo*'. The optimizer applied for this comparison is MATLAB's fmincon interior-point algorithm. This method was chosen because of its deterministic nature. Given the same initial conditions, event subsequence, aggregation function, and focus loss function, the optimized results are repeatable, ensuring results are due only to the changed aggregation function.

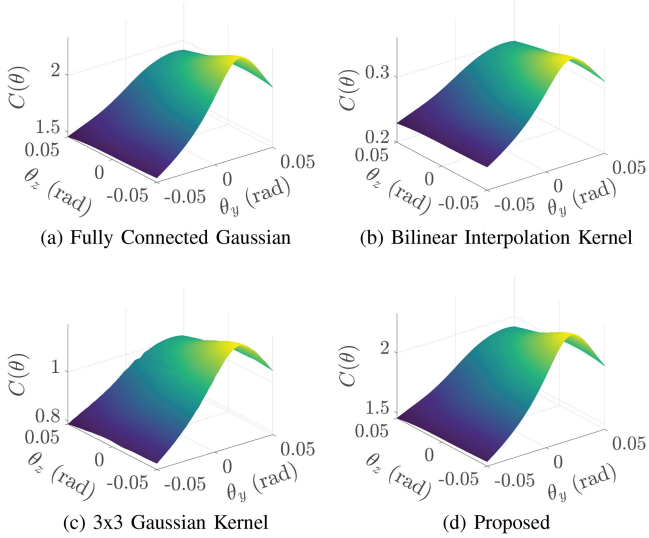
A camera under rotation motion is described in homogeneous camera calibrated coordinates by a rotation matrix, such that the warp mapping $\mathbf{x}_k$ to $\mathbf{x'}_k$ is described in (13).

$$\mathbf{W}\left(\mathbf{x}_k, t_k, \theta\right) = \mathbf{K}\mathbf{R}(t_k, \theta)\mathbf{K}^{-1}\mathbf{x}_k \tag{13}$$

$$\mathbf{R}(t_k, \theta) = \exp\left(\left[\theta(t_k - t_{ref})\right]_\times\right) \tag{14}$$

where (14) is the rotation matrix, θ the angular velocity of the camera in angle-axis representation, $[\theta(t_k - t_{ref})]_\times$ the skew symmetric matrix of $\theta(t_k - t_{ref})$, and $\mathbf{K}$ the intrinsic camera calibration matrix.

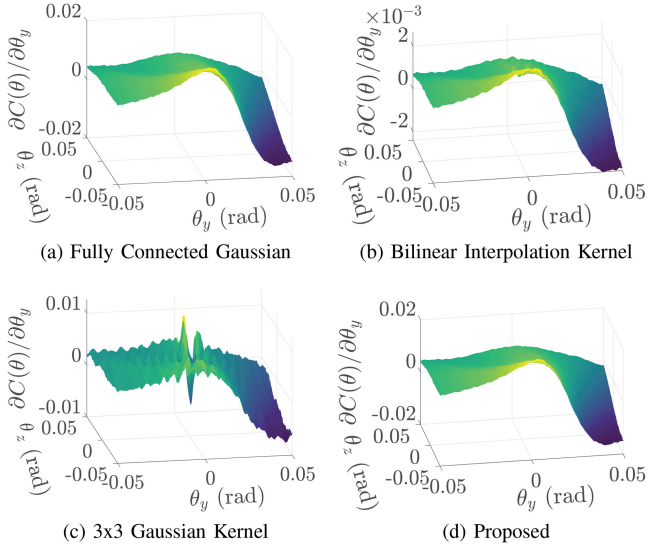
Fig. 1 shows the event image generated using the Kronecker Delta by the value of θ arrived by the interior-point algorithm for each aggregation function tested. The corresponding contrast as calculated from Fig 1, the average per-function-call runtime (Avg_{FR}) of each aggregation function, and total function calls (T_{FC}) in the optimization routine are shown in Table II. It can be seen that the proposed aggregation function yields the sharpest image for the same event subsequence in Fig. 1(d). While the kernel size is larger at 7x7, the resultant per-function-call runtime is the fastest due to the omission of the convolution step.



(a) Fully Connected Gaussian (b) Bilinear Interpolation Kernel

(c) 3x3 Gaussian Kernel (d) Proposed

Fig. 2. Contrast surface map of the 4 aggregation functions at $\theta_x \approx 0.00317$.



(a) Fully Connected Gaussian (b) Bilinear Interpolation Kernel

(c) 3x3 Gaussian Kernel (d) Proposed

Fig. 3. Contrast surface map of $\partial C(\theta)/\partial\theta_y$ of the 4 aggregation functions at $\theta_x \approx 0.00317$.

The contrast surface map is plotted to understand the aggregation function's effects on the optimiser. Since $\mathbf{R}(t_k, \theta) \in \mathrm{SO}(3)$ the resultant contrast surface plot would be 4-dimensional; therefore, a volume slice along the x-axis is taken. The chosen x-axis location, $\theta_x \approx 0.00317$ (the exact value is used in plot generation), is the x-axis rotation arrived by the algorithm running the 3x3 Gaussian Kernel function and represents a failure point for the local optimiser. The entire lower and upper bound range $\begin{bmatrix}-0.05 & 0.05\end{bmatrix}$ is generated for θ_y and θ_z.

It can be seen from Fig. 2 that all 4 functions generates a similarly smooth contrast map. However taking the partial derivative of the contrast map along θ_y shows that the 3x3 Gaussian Kernel in Fig. 3(c) is irregular compared to the Fully Connected Gaussian in Fig. 3(a). As the interior point algorithm is gradient-based, it is understandable why the optimizer terminated at $\theta \approx (0.00317, 6.76 \times 10^{-5}, -0.00267)$ evident by

TABLE III
ESTIMATED CONTRAST ACROSS 4 DATASETS. A HIGHER CONTRAST THAN INPUT CONTRAST IS BETTER

	'boxes_rotation'	*'dynamic_rotation'*	*'poster_rotation'*	*'shapes_rotation'*
Input Contrast	0.86954	0.20919	0.93329	0.12538
Fully Connected Gaussian	1.5872	0.23838	2.0943	0.18666
3x3 Gaussian Kernel	1.6484	0.20919	0.93329	0.12538
3x3 Gaussian Kernel w/o Gaussian Filter	1.7154	0.20919	2.0928	0.18774
Bilinear Interpolation Kernel	1.6918	0.20919	2.0901	0.18323
Bilinear Interpolation Kernel w/o Gaussian Filter	0.86954	0.20919	0.93329	0.12538
Pseudo Fully Connected Gaussian	1.6533	0.2397	2.0922	0.18657

the deep valley in Fig. 3(c). Considering the probability of event influence at $\mu \pm 4\sigma$ is approximately $3.35 \times 10^{-4}\,\%$, and decreases exponentially with higher σ, Fig. 3(d) shows that the assumption of negligibility is a safe assumption as the partial derivative contrast map between Fig. 3(d) and Fig. 3(a) are similar with the Pseudo Fully Connected Gaussian having the computational advantage of not calculating values near $0\,\%$ influence, yielding an almost $1300\,\%$ faster function runtime. Since both the 3x3 Gaussian Kernel and Bilinear Interpolation Kernel do not compute values outside of 1σ, the proposed method yields a $100\,\%$ faster function runtime by excluding the convolution step.

To verify the robustness of the proposed Pseudo Fully Connected Gaussian for application in the CM framework, GD5 is applied to the entire *'dynamic_rotation'* dataset. The *'dynamic_rotation'* dataset was chosen because it contains a person moving separately in the scene, such that events describing the person and the scene would have different velocities. It directly challenges the primary assumption of CM that all events encode similar velocities (trajectories and speed). Let θ and $\bar{\theta}$ be the ground truth and estimated angular velocity of the event camera. Then ϵ is the percentage error in estimating angular velocity:

$$\epsilon = \left| \frac{\theta_y - \bar{\theta_y}}{\bar{\theta_y}} \right| \cdot 100\% \tag{15}$$

Fig. 4 plots the angular velocity estimated by GD5 on the entire sequence and a selected zoomed-in section to highlight the close tracking performance of the function at high angular velocities of the dataset. The entire sequence is grouped into $10\,\mathrm{ms}$ event packets. The angular velocity is estimated for each packet, and its percentage error is calculated using (15). GD5 has an average percentage error of $\epsilon = \{3.13\%, 2.85\%, 1.93\%\}$ about the x, y, and z-axis respectively across the entire sequence, performing well within the error bounds as presented in [10], [11].

Lastly the optionality of Gaussian filtering in $H_3(\mathbf{x'}_k)$ and $H_4(\mathbf{x'}_k)$ is discussed. The CM routine is performed for each aggregation function on the 4 datasets published by [12]. $H_3(\mathbf{x'}_k)$ and $H_4(\mathbf{x'}_k)$ have an additional case where the contrast is maximized without Gaussian filtering. The $10\,\mathrm{ms}$ event subsequence is extracted from the 30 - 30.01 s of each dataset. It was chosen for two reasons: It is assumed that any motion in the scene is developed (i.e. the camera or the actors have started moving), and it is the midpoint of all datasets. Table III shows that for $H_4(\mathbf{x'}_k)$ Gaussian smoothing is an essential step in the CM process. Additionally, as $H_5(\mathbf{x'}_k)$ where $r = 1$ is equivalent to $H_3(\mathbf{x'}_k)$ without the Gaussian filter, the results shows further

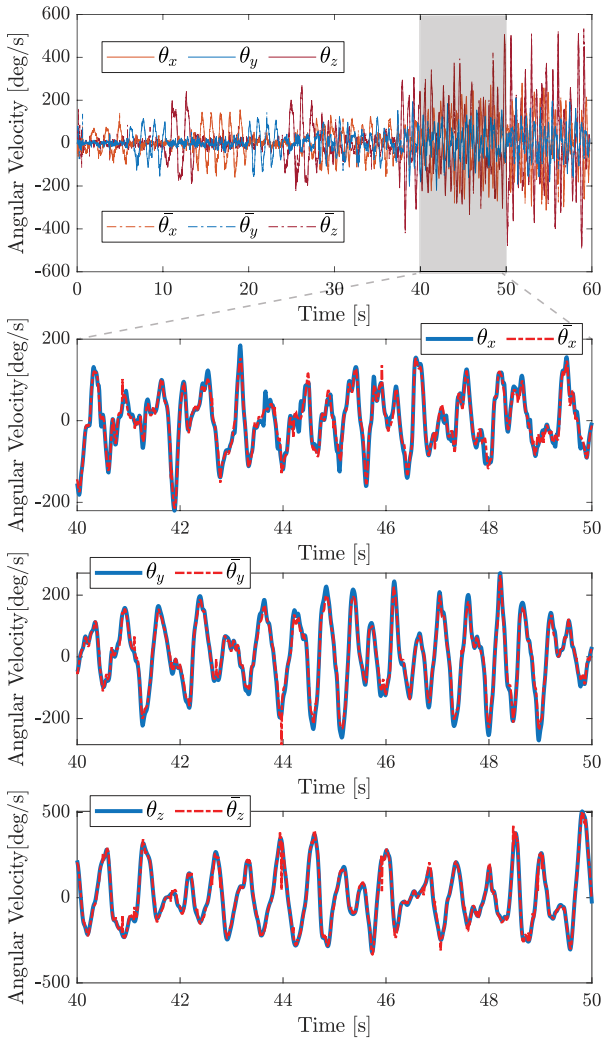
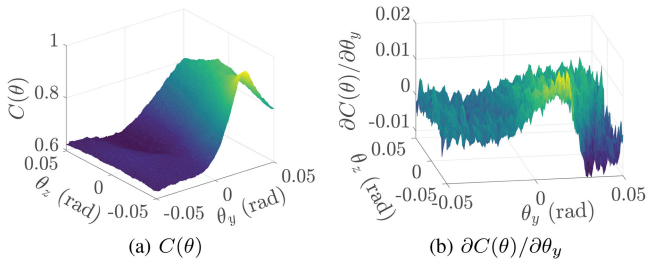


Fig. 4. Comparison of estimated angular velocity (dashed line) against the groundtruth as provided by [12] for the *'dynamic_rotation'* sequence.

that in some cases, the framework is able to perform better with a smaller kernel size. Lastly for particularly challenging scenes in *'dynamic_rotation'*, $H_5(\mathbf{x'}_k)$ and $H_2(\mathbf{x'}_k)$ were the only method able to exit the initial condition, demonstrating the occasional insufficiency of $r = 1$.

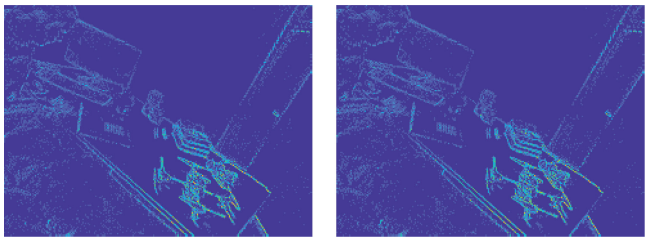
B. Discrete Aggregation Functions

The aggregation function chosen is highly dependent on the type of optimizer used in the CM framework. Consider gradient-based methods that require a continuous first derivative to drive



(a) $C(\theta)$ (b) $\partial C(\theta)/\partial\theta_y$

Fig. 5. Contrast surface map of the Kronecker Delta function (3) at $\theta_x \approx 0.00317$.



(a) Pseudo Fully Connected Gaussian (Contrast = 0.893) (b) Kronecker Delta (Contrast = 0.902)

Fig. 6. Event image generated using the Kronecker Delta function from the optimized value of θ arrived by PSO.

the direction and termination point of the optimization. Out of the 5 functions tested, the Kronecker Delta has the fastest function runtime at 1.1 ms. Practically, however, it is unusable in gradient-based optimization because its first derivative contrast map is highly irregular due to the sparsity of events describing the scene (see Fig. 5(b)). In the case of the tested subsequence of the *'dynamic_rotation'* dataset, the optimizer was not able to exit its initial condition.

First mentioned in [11], discrete event images are generated using hard assign aggregation functions such as (2). The method has been shown to yield similar results with clearly superior performance in the BNB global optimization method compared to the 3x3 Gaussian Kernel (a soft assign aggregation function). However, discrete event images limit the scope of optimizers available to non-gradient-based methods. Consider PSO, an optimization method that does not require objective functions to be differentiable. Since the contrast surface map of the Kronecker Delta function in Fig. 5(a) is similar to the smooth contrast surface maps in Fig. 2. The intuition would be to apply PSO to the CM framework since particles of PSO surf on said contrast map towards the "global" minimum, and any concerns for the smoothness of the Jacobian of $C(\theta)$ can be conveniently disregarded.

Two aggregation functions, PSO1 and PSO5, are compared. From Fig. 6 it can be seen that both aggregation function yield similar results. Plotting the contrast of the overall best estimate of θ in the swarm (see Fig. 7) shows both methods converging at similar rates with the Pseudo Fully Connected Gaussian terminating almost 100 iterations earlier. While not indicating a superior performance since the termination condition of PSO can be further tuned by changing the inertia weight or terminating condition, the optimizer was not specifically tuned to favour

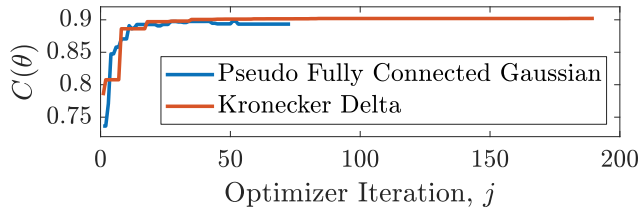


Fig. 7. Plot of the convergence rate for 2 aggregation functions in PSO.

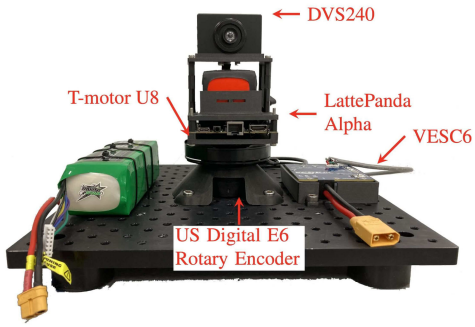
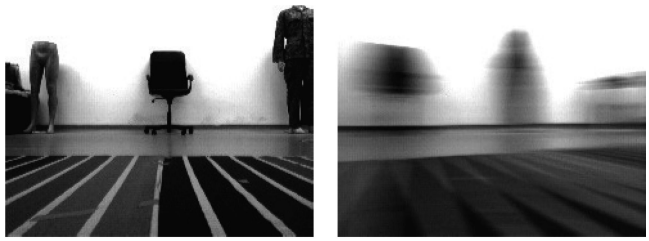


Fig. 8. System setup to capture event data at different rotational speeds.



(a) Static (b) Camera rotating at 360 °/s

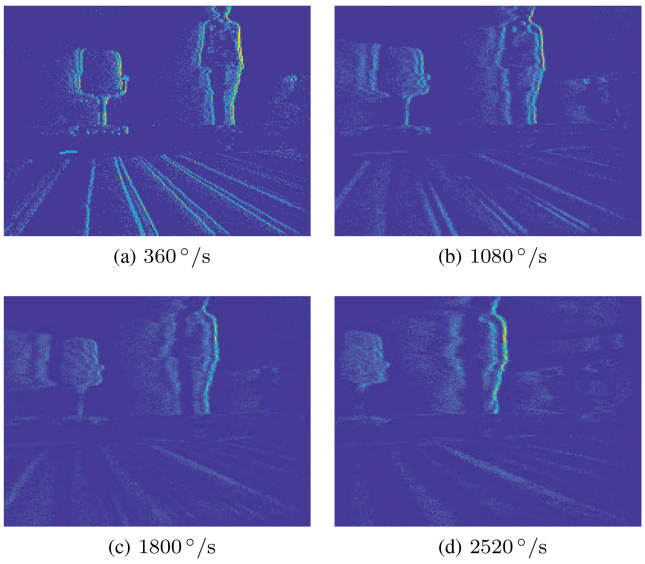
Fig. 9. Grayscale image taken using DVS240 of a chair and mannequin.

either function. Instead, the similar convergence rate gives confidence that discrete event images do not impede the convergence rate of non-gradient based optimizer and, in the case of PSO, can benefit from the faster function runtime of the Kronecker Delta with proper tuning.

C. Motion Blur in Event Cameras

This subsection explores the impact of motion blur on the CM framework. The DVS240 event camera is mounted onto a rotational rig for accurate rotation. The rotational rig comprises a T-motor U8, US Digital E6 Rotary Encoder, and VESC6 that provides closed-loop rotational control of the U8 motor. The DVS240 is interfaced via USB to a LattePanda Alpha. Fig. 8 illustrates the setup.

Motion blur is the apparent streaking of texture of objects under motion. For a moving camera under high rotation, all objects within the camera view can exhibit motion blur. This is illustrated in Fig. 9 where Fig. 9(a) shows the static scene as taken using a DVS240, and Fig. 9(b) the same scene as the camera rotates 360 °/s. Existing studies on image deblurring in event cameras focus on deblurring the low frame rate intensity image of the DAVIS camera using the high temporal resolution event data [14]–[16].



(a) 360 °/s (b) 1080 °/s

(c) 1800 °/s (d) 2520 °/s

Fig. 10. Event Image generated of the same scene using the Kronecker Delta function for DVS240 rotating at various rotation rates.

TABLE IV
CONTRAST CALCULATED FROM THE ESTIMATED VALUE OF θ

	360 °/s	1080 °/s	1800 °/s	2520 °/s
GD2	1.9696	4.9325	6.5843	7.5880
GD3	1.9713	4.9213	6.5822	7.5923
GD4	1.9372	4.9196	6.5417	7.2871
GD5	1.9701	4.9306	6.5750	7.5839
PSO1	**1.9926**	**4.9868**	**6.6301**	**7.6722**
PSO5	1.9637	4.9229	6.5764	7.5953
BNB1	1.9143	4.9579	6.5343	7.6410

In the CM framework, by nature of definition, motion blur is the result of poor motion compensation of event data when mapping from spatiotemporal events to spatial images using $H(\mathbf{x'}_k)$. Event cameras are not exempt from motion blur despite their high temporal resolution. Pixel front-end noise and commonly adopted timestamping-during-readout scheme result in latency of actual event detection time among projected pixels describing the same texture [17]. In the CM framework, which uses the temporal difference to weight motion compensation, this results in apparent motion blur. Fig. 10 shows event images generated using the Kronecker Delta function for events recorded at various rates. Note the increasingly washed-out profile of the chair and mannequin, as events are not propagated back to the correct pixel location in the event image due to different timestamps describing the same texture.

The impact is quantitatively assessed through two measures. First, a comparison of contrast as it is the main metric used to terminate optimization. Secondly, the percentage error in estimating angular velocity since one of the use cases for the CM framework is to estimate the event camera's attitude.

It can be seen from Table IV that all optimizers with the various aggregation functions perform similarly with an average contrast standard deviation of 0.0301. This shows that the optimizers continue to perform well despite losses in event data describing the same texture due to misstamped time. Misstamped times are described as losses because aggregation functions accumulate event data mapped to the same pixel area. An inability to map to the same pixel area results in a loss in accumulated event magnitude. This does not faze the framework because both aggregation and loss function work in tandem to disproportionally reward event locations of high magnitude.

TABLE V
PERCENTAGE ERROR OF θ_y AS CALCULATED USING (15), WHERE θ_y IS THE ESTIMATED ANGULAR VELOCITY ABOUT THE Y-AXIS OF THE EVENT CAMERA

	360 °/s	1080 °/s	1800 °/s	2520 °/s
GD2	3.4906	2.0905	0.0371	0.1188
GD3	3.4200	2.3080	0.0987	**0.0673**
GD4	3.0954	2.3015	0.5734	1.8642
GD5	3.4965	2.1515	**0.0255**	0.1461
PSO1	2.7341	1.9901	0.3639	0.5584
PSO5	3.8647	2.3161	0.0408	0.1644
BNB1	**0.4985**	**1.5464**	1.7671	0.3598

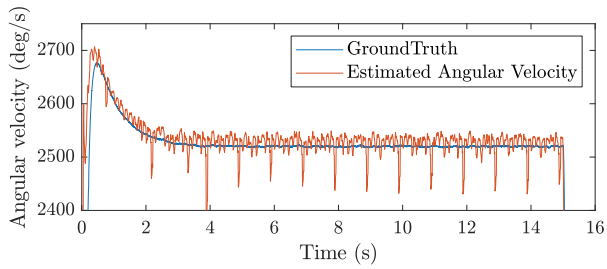


Fig. 11. Instantaneous estimation of angular velocity without a priori for each 10 ms event packet.

By directly estimating using (3), PSO1 has the advantage to outperform the other optimizers in contrast maximization. This performance, however, does not equate to the best-estimated warp as shown in Table V. While all estimates are within $4\,\%$ accurate, the results seem to suggest that a combination that maximizes warp accuracy with implicit event image generation for tasks like motion segmentation or scene feature tracking would be ideal since the contrast estimates are so similar. This observation, interestingly, would run contrary to conventional wisdom where sharp images are critical to the performance of a computer vision algorithm.

IV. INSTANTANEOUS ESTIMATION AT HIGH ROTATIONS

Consider high rotational systems such as a damaged quadrotor [18] and freerotors [19], where rotations about a single axis dominates motion estimates. It has been shown that for low rotations rates of $1150\,°/\mathrm{s}$ it is still possible to perform motion-compensation using attitude recovered from the onboard IMU [18]. However, IMUs typically found on off-the-shelf flight controllers such as the MPU-9250 [20] are shown to have an insufficient sensing range of $\pm 2000\,°/\mathrm{s}$ compared to the angular dynamic range of freerotors which exceeds $2160\,°/\mathrm{s}$ [21] up to $4680\,°/\mathrm{s}$ [22].

To circumvent the limitations of the IMU and demonstrate its effectiveness, the CM framework is applied to the entire $2520\,°/\mathrm{s}$ rotation dataset. The rotation for each $10\,\mathrm{ms}$ event packet is estimated without motion priors and plotted in Fig. 11. The ground truth is the angular speed reported by VESC6. At $2520\,°/\mathrm{s}$ the framework is shown to be capable of estimating at

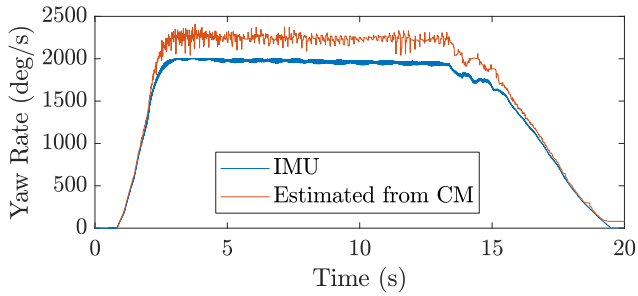


Fig. 12. In-flight estimation of angular velocity without a priori for each 10 ms event packet.

twice the angular speed found in the literature with a median percentage error of $0.386\,\%$.

This is further corroborated with in-flight state estimation on a freerotor. The camera was mounted in a similar side-facing manner. At rotations greater than $2000\,^\circ/\mathrm{s}$, features in the scene do not persist for more than 2 frames and present a secondary necessity for the use of this framework. Fig. 12 plots the yaw rate estimated from events captured using a DVS240 and the corresponding imu data from the event camera. Similar to Fig. 11, the motion prior for each event packet is assumed unknown. This effectively restarts the algorithm to zero initial condition, demonstrating estimation reliability across the entire flight range. It can be seen that the framework continues to estimate the yaw rate from events above the sensing threshold of the onboard IMU, which saturates just below $2000\,^\circ/\mathrm{s}$.

V. Conclusion

This paper analyzed the impact of aggregation functions in the CM framework, how it affects optimizer choice, and the effect of latency in time-stamping on event image generation. The proposed convolution-free aggregation function was shown to yield superior performance over existing aggregation functions with an average percentage error of less than $3.2\,\%$ for the entire *'dynamic_rotation'* dataset. The advantages of the proposed aggregation function are summarized in Table I. Motion-blur was also shown to have a negligible impact on the framework's accuracy, in contrast to conventional computer vision algorithms, which are typically known to degrade estimation accuracy. Lastly, the CM framework was applied to high rotation static and flight datasets demonstrating its ideal application as a simultaneous image and state estimator for high rotation problems.

References

[1] M. Mahowald, "The silicon retina," in *Proc. An Analog VLSI Syst. Stereoscopic Vis.*, 1994, pp. 4–65.

[2] J. A. Leñero-Bardallo, T. Serrano-Gotarredona, and B. Linares-Barranco, "A 3.6 μs latency asynchronous frame-free event-driven dynamic-vision-sensor," *IEEE J. Solid-State Circuits*, vol. 46, no. 6, pp. 1443–1455, Jun. 2011.

[3] T. Serrano-Gotarredona and B. Linares-Barranco, "A 128 $\times$ 128 1.5% contrast sensitivity 0.9% FPN 3 μs latency 4 mW asynchronous frame-free dynamic vision sensor using transimpedance preamplifiers," *IEEE J. Solid-State Circuits*, vol. 48, no. 3, pp. 827–838, Mar. 2013.

[4] A. R. Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate SLAM? combining events, images, and IMU for robust visual SLAM in HDR and high-speed scenarios," *IEEE Robot. Automat. Lett.*, vol. 3, no. 2, pp. 994–1001, Apr. 2018.

[5] E. Mueggler, B. Huber, and D. Scaramuzza, "Event-based, 6-DOF pose tracking for high-speed maneuvers," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 2761–2768.

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

[7] T. Stoffregen, G. Gallego, T. Drummond, L. Kleeman, and D. Scaramuzza, "Event-based motion segmentation by motion compensation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 7244–7253.

[8] G. Gallego, M. Gehrig, and D. Scaramuzza, "Focus is all you need: Loss functions for event-based vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12280–12289.

[9] H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization," in *BMVC*, 2017.

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

[11] D. Liu, A. Parra, and T.-J. Chin, "Globally optimal contrast maximisation for event-based motion estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 6349–6358.

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

[13] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240$\times$ 180 130 db 3 μs latency global shutter spatiotemporal vision sensor," *IEEE J. Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, Oct. 2014.

[14] L. Pan, C. Scheerlinck, X. Yu, R. Hartley, M. Liu, and Y. Dai, "Bringing a blurry frame alive at high frame-rate with an event camera," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6820–6829.

[15] L. Pan, R. Hartley, C. Scheerlinck, M. Liu, X. Yu, and Y. Dai, "High frame rate video reconstruction based on an event camera," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–1, 2020. doi: 10.1109/TPAMI.2020.3036667.

[16] Z. Jiang, Y. Zhang, D. Zou, J. Ren, J. Lv, and Y. Liu, "Learning event-based motion deblurring," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 3320–3329.

[17] "Understanding the performance of neuromorphic event-based vision sensors" iniVation AG, Zurich, Switzerland, White Paper, 05 2020. [Online]. Available: https://inivation.com/wp-content/uploads/2020/05/White-Paper-May-2020.pdf

[18] S. Sun, G. Cioffi, C. De Visser, and D. Scaramuzza, "Autonomous quadrotor flight despite rotor failure with onboard vision sensors: Frames vs. events," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 580–587, Apr. 2021.

[19] M. Ng, E. Tang, G. S. Soh, and S. Foong, "SHIFT: Selective heading image for translation an onboard monocular optical flow estimator for fast constantly rotating UAVs," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 8525–8531.

[20] T. S. Lembono, J. E. Low, L. S. T. Win, S. Foong, and U.-X. Tan, "Orientation filter and angular rates estimation in monocopter using accelerometers and magnetometer with the extended kalman filter," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 4537–4543.

[21] C. H. Tan, D. S. bin Shaiful, E. Tang, G. S. Soh, and S. Foong, "High angular rates estimation using numerical phase-locked loop method," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics*, 2020, pp. 1516–1521.

[22] E. R. Ulrich, D. J. Pines, and J. S. Humbert, "From falling to flying: The path to powered flight of a robotic samara nano air vehicle," *Bioinspiration Biomimetics*, vol. 5, no. 4, 2010, Art. no. 45009.