VibES: Induced Vibration for Persistent Event-Based Sensing

Vincenzo Polizzi[1] Stephen Yang[1,2] Quentin Clark[3]
Jonathan Kelly[1] Igor Gilitschenski[3] David B. Lindell[3]

[1]University of Toronto, Robotics Institute
[2]University of Toronto, Department of Mechanical and Industrial Engineering
[3]University of Toronto, Department of Computer Science
[1]{vincenzo.polizzi,jonathan.kelly}@robotics.utias.utoronto.ca
[2]styang@mie.utoronto.ca, [3]{qtcc,gilitschenski,lindell}@cs.toronto.edu

Abstract

Event cameras are a bio-inspired class of sensors that asynchronously measure per-pixel intensity changes. Under fixed illumination conditions in static or low-motion scenes, rigidly mounted event cameras are unable to generate any events and become unsuitable for most computer vision tasks. To address this limitation, recent work has investigated motion-induced event stimulation, which often requires complex hardware or additional optical components. In contrast, we introduce a lightweight approach to sustain persistent event generation by employing a simple rotating unbalanced mass to induce periodic vibrational motion. This is combined with a motion-compensation pipeline that removes the injected motion and yields clean, motion-corrected events for downstream perception tasks. We develop a hardware prototype to demonstrate our approach and evaluate it on real-world datasets. Our method reliably recovers motion parameters and improves both image reconstruction and edge detection compared to event-based sensing without motion induction.

Supplementary Material: For code and data please visit https://papers.starslab.ca/vibes/.

1. Introduction

Most computer vision algorithms rely on conventional image sensors—such as RGB or greyscale cameras—that capture synchronous image frames at fixed intervals. While widely adopted, these frame-based sensors suffer from well-known limitations, including motion blur, latency due to frame timing, and high power consumption [19]. These drawbacks are especially problematic in robotics applications where fast dynamics, low latency, and energy efficiency are critical, for example in navigation [41], feature tracking [32], and real-time scene understanding [27].

Event cameras have emerged as a promising alternative. These bio-inspired sensors asynchronously detect per-pixel changes in log-intensity, producing a sparse stream of events with microsecond latency and high dynamic range [19]. Event cameras inherently reduce motion blur, operate at lower power [20], and enable low-latency perception, making them well-suited for resource-constrained, high-speed robots.

However, event cameras do have a fundamental limitation: event generation depends on motion. In static scenes, or when edges align with the direction of the camera motion, events are not triggered. This results in event sparsity, loss of spatial information, and perceptual fading [7], which undermines long-term feature tracking, edge detection, and high-quality image reconstruction [24]. Consequently, the performance of event-based systems degrades significantly without sufficient camera or scene motion.

Interestingly, the human visual system faces a similar challenge. To prevent image fading during fixation, our eyes perform rapid, involuntary movements known as microsaccades or fixational eye movements (FEMs). These movements continuously stimulate photoreceptors, refreshing the visual input and maintaining high-resolution perception [28, 46]. This biological mechanism suggests that artificial motion could be exploited to sustain event generation.

In this work, we draw inspiration from microsaccades while adopting a deterministic, engineered approach. Instead of attempting to replicate their stochastic nature, we design a lightweight vibration mechanism that mechanically stimulates the event camera, ensuring continuous event generation in static or quasi-static scenes. Our approach, VibES, uses a simple rotating unbalanced mass within a spring–damper system to induce harmonic motion. A motion-compensation pipeline then removes the in-

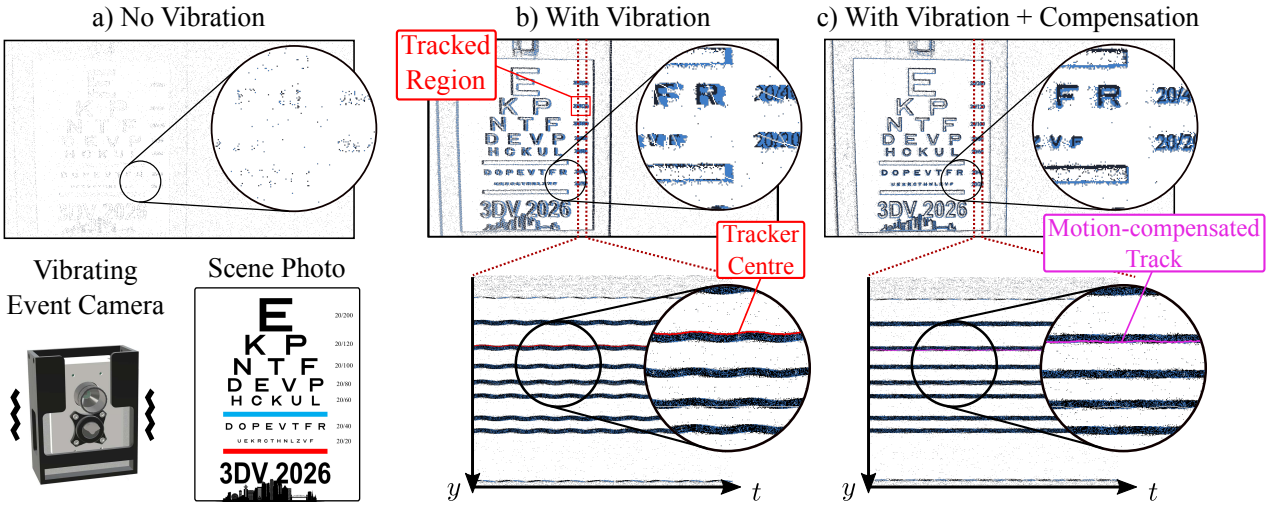

Figure 1. **Qualitative illustration of our method. (a) No Vibration.** With a static event camera and no induced motion, the accumulated event image appears blurred and lacks sharp edges, while the y–t slice shows little temporal structure. **(b) With Vibration.** Introducing controlled vibrations stimulates the sensor, increasing the number of events and producing sinusoidal traces in the y–t slice. We develop an Extended Kalman Filter (EKF) to track the vibrational motion in a region of the scene (red). **(c) With Vibration + Compensation.** We use the EKF-tracked region to estimate and compensate for the sinusoidal motion across the entire scene. The motion-compensated accumulated image recovers sharp structures, and the y–t slice aligns with the stable motion-compensated track (magenta), revealing the underlying scene. An inset (bottom left) shows the reference scene displayed in front of the camera with induced motion.

jected vibration, producing clean, motion-corrected events for downstream tasks. Compared to existing microsaccade-inspired methods based on optical systems [24] or pan–tilt actuators [13, 34, 43, 44, 49], our design requires no additional optical components or position-encoding sensors, enables real-time motion compensation on the event stream, and extends to tracking motion frequencies in a target scene. An overview of the proposed method is shown in Fig. 1.

Our contributions are threefold:

- We design a vibrating event camera using a rotating unbalanced mass that enables event generation in static scenes, addressing a key limitation of event cameras.
- We introduce a real-time motion-compensation pipeline that estimates and removes induced vibrations online, without requiring calibration or prior knowledge of physical parameters.
- We validate our method on four real-world datasets, demonstrating improved event density and higher-quality results for edge detection and image reconstruction. We also show extensions to specialized applications such as scene frequency estimation and relative depth prediction.

2. Related Work

Event cameras are increasingly used in robotics and computer vision due to their high temporal resolution and ability to capture fast scene dynamics. In this section, we review their role in perception, outline complementary hybrid sensing strategies, and discuss microsaccade-inspired methods for maintaining event generation in otherwise static scenes.

Event cameras in robotics. Event cameras offer several advantages over conventional cameras, including high dynamic range and microsecond-level temporal resolution. These properties have been exploited for feature tracking under challenging conditions [21, 32, 50].

In robotics, event cameras have been applied to visual localization [45], edge detection [5], and motion compensation [18], frequency estimation [3, 37, 42], image reconstruction [23, 26, 45] and deblurring [25, 36]. We show that for motion compensation, edge detection, and image reconstruction tasks, our method can provide a high-quality information stream, taking advantage of the induced motion.

Hybrid-modality sensing methods. Beyond microsaccades, other strategies for event generation include moving stereo event cameras at high speed for 3D reconstruction [29], pairing event cameras with conventional frames for static localization [11], and employing visual tracking for object following [22]. Machine learning-based approaches have also been developed to infer object motion from sparse event data [6], with some methods relying on continuous-motion assumptions, for example in human choreography capture applications [48]. Our system does not rely on conventional image frames or on the assumption of fast motions in the scene. Instead, we propose a fully model-based event-centric approach that exploits the intrinsic properties of events and their sensitivity to motion, generating a continuous stream of information. This stream can be leveraged for reconstruction, edge detection, and the tasks discussed above, even in otherwise static scenes.

Microsaccade-inspired event cameras. Inspired by fixational eye movements in human vision, several works have proposed artificial microsaccades to enable event generation in static scenes. Existing solutions rely on mechanically induced motion using pan–tilt units [13, 34, 43, 44, 49], mirrors [30], rotating wedge prisms [24], or polarization-based optical elements [31]. These systems typically require specialized hardware and careful calibration. Our approach employs a simpler mechanical setup and a processing pipeline that adapts automatically to system parameters, making it more suitable for real-world deployment. Concurrent work has explored motion recovery from unstructured camera jitter inspired by spacecraft vibration [2], but the lack of a motion prior limits reconstruction accuracy.

The approach most similar to our own is that of He et al. [24]. They employ a rotating wedge prism to generate events continuously and utilize position-encoding hardware to track and remove the induced motion. Our method introduces a mechanically simpler approach, requiring no custom optical elements or position encoders such as in [24]. Instead, we directly estimate the state of the sinusoidal motion from the event stream itself and perform parameter estimation online. Moreover, we demonstrate that our software stack is agnostic to the underlying hardware. On data generated by the rotating wedge-prism in [24], we show that our system performs equally well in terms of motion compensation quality and event stream consistency. Finally, our experiments show that our system enables the same downstream use cases proposed in [24]—particularly in image reconstruction and edges extraction—with simpler hardware and a more general motion-tracking framework.

3. Methodology

Our approach aims to excite the pixels of an event camera to produce a continuous stream of events, enabling high-quality image reconstruction and sharp edge detection from raw event data. We first describe the mechanical system responsible for generating vibrational motion and its effect on event generation in the camera (Sec. 3.1). We then present the motion compensation pipeline that allows us to recover the underlying scene structure (Sec. 3.2). Finally, we describe our physical hardware prototype (Sec. 3.3).

3.1. Camera Motion Model

To characterize the motion induced by our vibration mechanism, we adopt a classical mass-spring-damper model, illustrated in Fig. 2. This model captures the essential dynamics of the periodic motion imparted to the camera.

We model our system as a forced damped harmonic oscillator, which can be described by a second-order differential equation. The closed-form steady-state solution is of the form

$$y(t) = \hat{A} \cdot \sin(\hat{\omega} t - \phi), \tag{1}$$

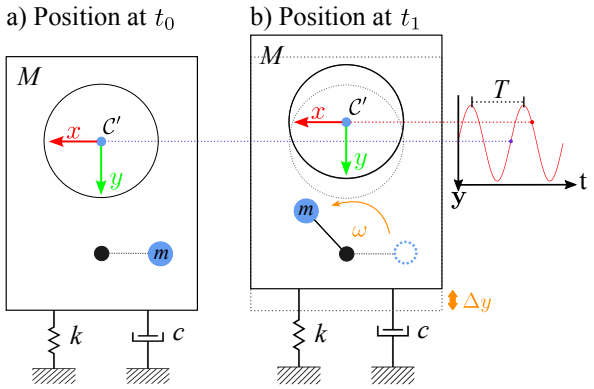

Figure 2. **Schematic of the mass–spring–damper model**. Camera setup at two time steps, (a) t_0 and (b) t_1. The rotation of an off-axis mass m with angular velocity ω induces planar displacements $\Delta x, \Delta y$ (only the vertical component is illustrated). The estimated oscillation frequency, $\frac{2\pi}{T}$, corresponds to the motion perceived by the camera. Note that this differs from the natural frequency of the off-axis mass ω, due to inertial effects.

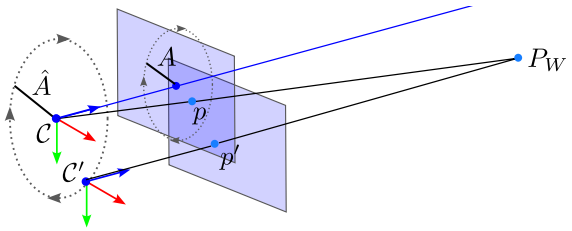

Figure 3. **Visual representation of the camera model**. The virtual camera $\mathcal{C}$ remains static, while the real camera $\mathcal{C}'$ translates along a circular path centered at $\mathcal{C}$ in the x–y image plane of $\mathcal{C}$. We denote $\mathbf{p}$ and $\mathbf{p}'$ as the projected point of $\mathbf{P}_W$ in the virtual and and real image planes respectively. $\hat{A}$ and A refer to the amplitudes of motion in the virtual and real frames respectively. The objective is to remove the resulting oscillatory motion and recover the projection of point P_W onto the image plane of $\mathcal{C}$.

where $\hat{A} = \sqrt{\hat{A}_X^2 + \hat{A}_Y^2}$ is the oscillation amplitude, with $\hat{A}_X$ and $\hat{A}_Y$ as the components along $\mathbf{x}$- and $\mathbf{y}$-axis in camera coordinates. As such, the motion of the camera when viewing a static scene follows a sinusoid. Further details are provided in the supplementary material, Sec. S1.1.

The camera motion is described by introducing a *virtual camera frame*, denoted as $\mathcal{C}$, shown in Fig. 3. In our formulation, $\mathcal{C}$ remains static and acts as the reference, while the physical camera, $\mathcal{C}'$, moves along a controlled, circular trajectory in the x–y image plane of $\mathcal{C}$.

We adopt a standard pinhole projection model to describe how a 3D point $\mathbf{P}_W$ in the world frame $\mathcal{W}$ projects onto the image plane of $\mathcal{C}$. We denote $\mathbf{p} = (u, v, 1)$ as the projected point in the virtual frame, which is the point we would like to recover. The moving camera observes point $\mathbf{p}' = (u', v', 1)$. The transformation from $\mathcal{C}$ to the moving

camera $\mathcal{C}'$ is defined as

$$\mathbf{T}(t)_{\mathcal{C}'\mathcal{C}} = \begin{bmatrix} & \hat{A}_X \cos(\hat{\omega}t + \phi_X) \\ \mathbf{I} & \hat{A}_Y \cos(\hat{\omega}t + \phi_Y) \\ & 0 \\ \mathbf{0}^\top & 1 \end{bmatrix}, \tag{2}$$

where $\mathbf{I}$ is the 3×3 identity matrix, $\hat{\omega}$ is the angular frequency of the motion, and ϕ_X, ϕ_Y are the corresponding phase shifts. Note that $\hat{\omega}$ differs from ω in Fig. 3 due to inertial factors. The $\mathbf{z}$-coordinate remains constant, as the motion is planar. The projection of $\mathbf{P}_W$ onto the moving camera $\mathcal{C}'$ is then

$$\mathbf{p}' = \mathbf{K}\mathbf{T}(t)_{\mathcal{C}'\mathcal{C}}\mathbf{T}_{\mathcal{C}W}\mathbf{P}_W. \tag{3}$$

Here, $\mathbf{K}$ is the intrinsic matrix and $\mathbf{T}_{\mathcal{C}W}$ is the extrinsic transform that expresses $\mathbf{P}_W$ in $\mathcal{C}$. To illustrate this relation more concretely, consider the vertical coordinate v of the projected point $\mathbf{p}$. In the x–y image plane, the point $\mathbf{p}$ travels along a circular path with amplitude $A = \sqrt{A_x^2 + A_y^2}$. Then expanding Eq. 3 we obtain

$$v' = f\frac{Y}{Z} + f\frac{\hat{A}_Y \cos(\hat{\omega}t + \phi_Y)}{Z} + c_Y, \tag{4}$$

where the term, $f\frac{Y}{Z} + c_Y$, corresponds to the projection v of $\mathbf{P}_W$ in the static virtual frame $\mathcal{C}$, and the second term models the time-varying displacement due to camera motion. We rewrite this as

$$v' = v + A_y \cos(\hat{\omega}t + \phi_Y), \tag{5}$$

where $A_y = f\frac{Y_0}{Z}$ is the amplitude of the induced sinusoidal motion as in Fig. 3.

From Eq. 5, we identify three key unknowns for motion compensation along one axis: the amplitude A_y (dependent on scene depth and oscillation magnitude), the phase ϕ_Y, and the coordinate v in the virtual frame. Accurate estimation of these parameters allows precise inversion of the apparent motion in the event stream and recovery of the underlying scene as if captured by a stationary camera.

3.2. Motion Compensation Setup

Once a model of the induced motion has been established, the next step is to eliminate motion from the incoming event stream. This requires estimating the dynamic motion parameters, which vary as the camera or the scene changes over time. To achieve this, we employ a series of software modules, illustrated in Fig. 4. Below, we describe each stage of the pipeline and its role in accurate motion compensation.

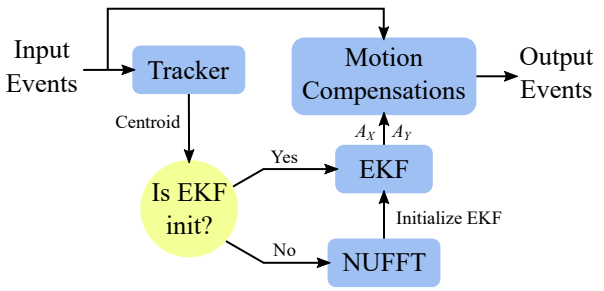

Figure 4. **Schematic representation of VibES**. The input event stream is processed by a tracker that extracts trajectories used to estimate the dominant frequency, amplitude, and phase shift of the oscillatory motion. Once the sinusoidal motion is characterized, an Extended Kalman Filter (EKF) is initialized independently for each axis to track the induced motion. The EKF estimates are then used to compensate for the induced motion in the incoming event stream.

3.2.1 Event Tracker

We first extract N centroid trajectories $(u'_i, v'_i, t_i)_{i=1}^N$ using an event tracker. In this work, we use HASTE [1], a lightweight algorithm that operates directly on the raw event stream. HASTE tracks local spatiotemporal patterns without requiring additional image processing, providing robust estimates of centroid positions over time. These trajectories capture the periodic motion induced by the camera and are subsequently used to estimate the unknown parameters of Eq. 5. To initialize the HASTE trackers, we manually select regions with high texture or edge content.

3.2.2 Frequency Estimation with NUFFT

Since the camera motion is oscillatory, recovering its frequency is essential. However, events are sampled irregularly in time, which makes standard Fourier analysis unsuitable. To obtain an initial estimate of the angular velocity parameter in Eq. 5, we employ the non-uniform fast Fourier transform (NUFFT) [4]. The NUFFT decomposes a signal into its constituent frequencies but is specifically designed to operate on irregularly sampled data—making it ideal for event-based inputs that are inherently asynchronous and non-uniform in time. Before applying the NUFFT, we remove the DC component from the tracker signal (Sec. 3.2.1) to isolate oscillatory motion from the static offset introduced by the patch location. Temporal samples t_i are then rescaled to $[-\pi, \pi]$, yielding normalized timestamps $\bar{t}_i$. This produces a zero-mean, time-normalized samples $(\bar{u}'_i, \bar{v}'_i, \bar{t}_i)_{i=1}^N$ which are passed to the NUFFT. The transform returns the top M dominant frequencies, from which we extract the angular frequency $\hat{\omega}$ of the camera motion. Estimation is performed independently for the X and Y axes. Using $\hat{\omega}$, we fit the tracked samples to a sinusoidal model using least-squares optimization, recover-

ing the amplitudes and phase offsets of the motion. These parameters initialize the extended Kalman filter (EKF) described in Sec. 3.2.3.

3.2.3 Extended Kalman Filter Setup

An extended Kalman filter (EKF) refines the motion parameters over time by incorporating a dynamical model and accounting for observation noise. This yields temporally consistent parameter estimates, which are essential for reliable motion compensation. We model all motion parameters dynamically in the EKF, but they are updated differently. Oscillatory frequency and phase are assumed to be globally constant across the image plane, whereas the oscillation amplitude depends on the scene depth. We make the assumption that all objects in the same scene share the same depth plane, although our implementation allows for multiple trackers which may have regions with different depths. Thus, we vary the u', v' parameters with each specific tracker to allow for tracking at different depths.

The camera frame's angular position θ evolves as

$$\theta_t = \theta_{t-1} + \hat{\omega}\Delta t, \tag{6}$$

where Δt is the time elapsed since the last update. The projected event position in the Y direction is modeled as

$$v' = a_y \sin(\theta) + b_y \cos(\theta) + v, \tag{7}$$

which is equivalent to Eq. 5 but avoids the numerical instability that arises from directly modeling the phase shift ϕ or explicit time dependence. This formulation also mitigates ambiguities caused by trigonometric wrapping. The parameters for the y-axis expression in Eq. 5 can be recovered as

$$A_Y = \sqrt{a_y^2 + b_y^2},\ \phi_Y = \arctan 2(b_y, a_y). \tag{8}$$

The EKF state vector is defined as

$$\mathbf{x} = \begin{bmatrix} \theta & \hat{\omega} & a_y & b_y & v \end{bmatrix}^\top. \tag{9}$$

This state is updated as new tracker measurements arrive, with the corresponding Jacobians provided in Supplementary Sec. S2. The filter can operate on the entire image plane or, more effectively, on localized patches centered around feature points. Our implementation allows multiple trackers to be instantiated in parallel, enabling flexible motion parameter estimation across the scene.

The EKF enables us to predict the expected oscillatory amplitude of the incoming event stream at any given time, and subsequently remove it to obtain a motion-compensated event stream, as illustrated in panel (c) of Fig. 1.

3.3. Hardware Prototype

For all real-world data collection and experiments, we use a Prophesee EVK3 event camera equipped with an IMX636 sensor (1280 × 720 pixels). To induce motion, we rigidly attach a DC motor [14] to the camera body and mount an off-centre mass on the motor shaft. The mass is custom-machined; however, any off-axis weight with sufficient eccentricity and mass is sufficient to generate the desired motion.

The event camera is enclosed in a 3D-printed casing wrapped in foam, which acts as a passive spring–damper system and enables the motion model described in Sec. 3.1. The case constrains the motion to a 2D plane and provides a rigid interface for real-world mounting. Additional details on the mechanical prototype are provided in the supplementary material (Sec. S2.3).

4. Results

We evaluated VibES on three real-world scenes captured with the Prophesee EVK3 (1280x720 px), demonstrating efficient motion-compensated event generation while preserving reconstruction quality and edge detail. Additional validation on the AMI-EV dataset [24], recorded with a DVXplorer (640×480 px), confirms VibES effectiveness and hardware independence.

4.1. Data Collection

We recorded real-world scenes and produced output data for three settings. First, we captured the scene using a standard event camera (S-EV). Second, we capture the scene with vibration (V-EV). Finally, we apply our motion compensation algorithm to the acquired V-EV data (VibES). Overall, we evaluated our method on four real-world scenes: *AMI-EV* [24], *Logo*, *Pattern Checkerboard*, and *Pattern*. Each scene consists of a textured pattern moving in front of the camera. The *Logo* pattern contains text and rich textures, moving slowly while performing partial rotations. The *Pattern Checkerboard* and *Pattern* sequences involve planar patterns moving at different speeds, with the *Pattern* exhibiting faster motions. Images of all patterns are provided in Supplementary Sec. S2.4. To ensure a fair comparison in the real-world setting, we employed a Franka robotic arm to move various patterns in front of the camera repeatably for the V-EV and S-EV settings. We mounted the camera on the physical prototype mass-spring-damper system described in Sec. 3.3.

4.2. Metrics

We evaluated the performance of our method for image quality, edges, and texture detection using the following metrics. We provide mathematical definitions of all metrics in our supplementary material, Sec. S2.5.

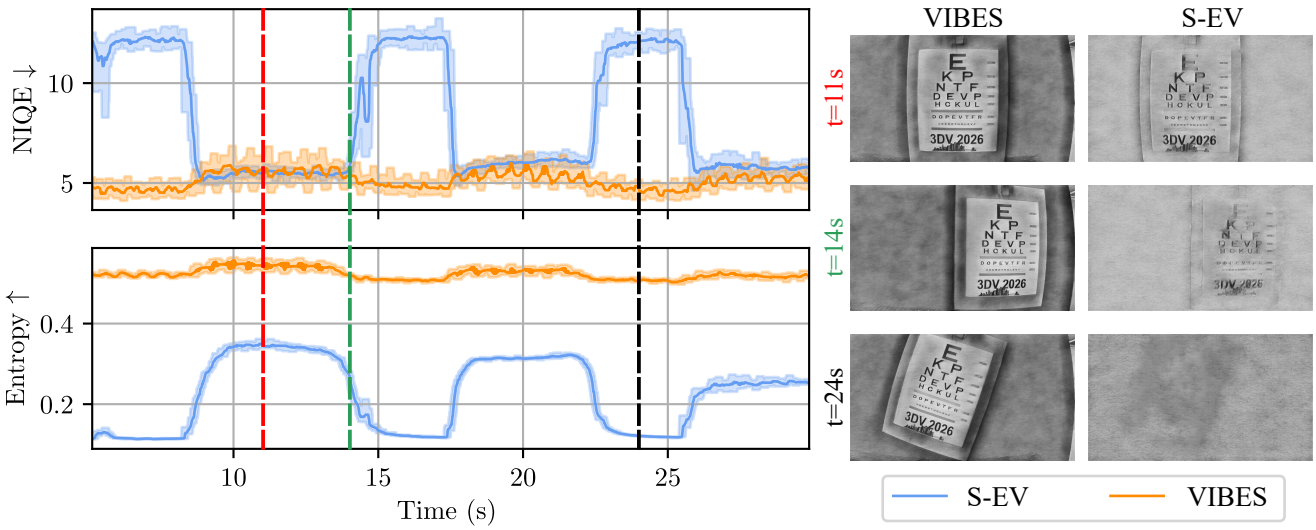

Figure 5. **Shannon entropy and NIQE scores** of reconstructed frames of binary accumulated event frames, computed over 10 ms time windows on the *Logo* real-world scene. The shaded range denotes the minimum and maximum values within every 10-frame window. Temporal screenshots of both the VibES and S-EV reconstructions, generated using E2VID [38, 39], are shown on the right. The regions where the S-EV method produces low-quality images are those where the scene is completely static or there is slow motion.

- **Shannon entropy** [40]**.** The entropy of binary images accumulated over a 10 ms window indicates the amount of information captured by the sensor. Higher values correspond to richer input data [24].
- **Natural Image Quality Evaluator (NIQE)** [33]**.** We compute NIQE on reconstructed images from E2VID [38, 39] with a 10 ms event stream window. Lower values indicate reconstructions closer to the statistical properties of natural images and higher perceptual quality.
- **Variance of image pixel values.** We count per-pixel events over a 10 ms window and compute the variance of the resulting image pixel values. High variance indicates well-aligned events forming sharp edges, while low variance reflects blurred or dispersed events [18].
- **Gradient magnitude of image pixel values.** We count per-pixel events over a 10 ms window and calculate the gradient magnitude of the resulting image pixel values. High values indicate strong edges and texture.
- **Edges continuity and fragmentation.** We count per-pixel events over a 33 ms window, smooth the resulting image with a Gaussian blur, binarize, and thin using the Zhang–Suen algorithm [51]. From the resulting edge maps, connected components are extracted [15, 47], and their number reflects fragmentation (many small components imply broken or noisy edges, fewer longer ones imply better continuity). We additionally compute the average edge length [35] and count edge junctions, which together serve as indicators of edge coherence and texture reconstruction quality.

4.3. Image Quality Evaluation

To evaluate the consistency of the output, we compute the entropy of accumulated event frames. The plot in Fig. 5 reports the entropy for the *Logo* scene and results for all scenes are shown in Tab. 1. Our results show that our induced motion, although small (less than a millimeter in the x- and y-directions), is sufficient to yield a higher information gain compared to the static event camera. It is worth noting that the variance of the entropy information is low when a vibratory motion is induced, meaning that we have a consistent and stable information input that does not depend on scene dynamics (Tab. 1).

Scene	Method	Entropy ↑
AMI-EV [24]	S-EV	0.08 ± 0.05
	VibES	$\mathbf{0.24 \pm 0.04}$
Logo	S-EV	0.21 ± 0.09
	VibES	$\mathbf{0.52 \pm 0.01}$
Pattern Checkerboard	S-EV	0.29 ± 0.13
	VibES	$\mathbf{0.59 \pm 0.04}$
Pattern	S-EV	0.30 ± 0.12
	VibES	$\mathbf{0.39 \pm 0.08}$

Table 1. **Shannon entropy** values (mean, standard deviation) computed on binary accumulated event frames over 10 ms windows. Higher entropy indicates a richer and more informative event stream.

Fig. 5 shows the NIQE and entropy results for the *Logo* scene, while per-scene NIQE values are provided in Tab. 2. The results demonstrate that induced motion consistently leads to improved reconstructions compared to a static event camera. A clear correlation emerges between entropy and image quality: higher entropy yields better NIQE scores in the S-EV setup. Conversely, when entropy is low — such

Scene	Method	NIQE ↓
AMI-EV [24]	S-EV	10.4 ± 5.0
	V-EV	$\mathbf{9.1} \pm 1.0$
Logo	S-EV	8.6 ± 9.4
	V-EV	$\mathbf{5.1} \pm 0.3$
Pattern Checkerboard	S-EV	8.2 ± 9.0
	V-EV	$\mathbf{6.3} \pm 0.2$
Pattern	S-EV	9.4 ± 5.0
	V-EV	$\mathbf{8.2} \pm 0.6$

Table 2. **Natural Image Quality Evaluator (NIQE)** average values (mean, standard deviation) computed using E2VID [38] at 100 Hz.

Scene	Method	Variance ↑	Grad. Mag. ↑
AMI-EV [24]	S-EV	0.06 ± 0.05	0.04 ± 0.02
	V-EV	0.32 ± 0.06	0.09 ± 0.02
	VibES	$\mathbf{0.44 \pm 0.10}$	$\mathbf{0.12 \pm 0.02}$
Logo	S-EV	0.11 ± 0.08	0.18 ± 0.07
	V-EV	2.34 ± 0.13	0.48 ± 0.03
	VibES	$\mathbf{3.86 \pm 0.34}$	$\mathbf{0.56 \pm 0.02}$
Pattern Checkerboard	S-EV	0.16 ± 0.11	0.22 ± 0.11
	V-EV	1.9 ± 0.15	0.57 ± 0.04
	VibES	$\mathbf{3.16 \pm 0.33}$	$\mathbf{0.60 \pm 0.04}$
Pattern	S-EV	0.22 ± 0.18	0.19 ± 0.09
	V-EV	0.76 ± 0.16	$\mathbf{0.30 \pm 0.06}$
	VibES	$\mathbf{1.11 \pm 0.25}$	$\mathbf{0.30 \pm 0.06}$

Table 3. **Quantitative evaluation of motion compensation.** We report image variance and average gradient magnitude, expressed as mean and standard deviation across frames. Higher values indicate sharper images and better-preserved scene structure.

as in static or low-motion scenes — our reconstructions remain stable and do not fade, ensuring consistent information output over time.

4.4. Edge Extraction Evaluation

We evaluate the sharpness of the accumulated frames by computing the variance and the gradient magnitude for the three setups: S-EV, V-EV, and VibES. Tab. 3 reports the average and variance of these metrics across all scenes.

The results indicate that our method effectively compensates for induced motion, achieving higher variance and gradient magnitude values, which correspond to sharper edges and stronger structural cues. We evaluate edge quality using both continuity and fragmentation metrics. In Tab. 4, we report statistics for the described metrics. The evaluation shows that VibES produces clean, well-connected edge maps, making it well-suited for preserving object boundaries and overall shape structure. In contrast, the standard event camera without induced vibration tends to produce noisier and more fragmented edges.

4.5. Runtime Evaluation

Our software stack, implemented in C++ with the Metavision API by Prophesee, is evaluated by running the compensation script ten times per dataset. The average processing time is 15.28 ± 3.7 ns per event (~ 65 Mev/s), including undistortion and motion compensation. The initial parameter estimation (Sec. 3.2.2), requires 104.8 ± 0.03 ms, but remains non-blocking thanks to a multithreaded design that tracks incoming events while buffering and performing motion compensation. A video demonstration is provided in the supplementary material. Although runtime is fast, real-time performance depends on sensor resolution and scene dynamics; in our experiments, event rates peaked at 45 Mev/s.

5. Applications

In this section, we demonstrate two additional applications of our pipeline beyond its primary goal of showing that induced motion can generate a stable event stream suitable for high-quality image reconstruction and edge extraction.

5.1. Frequency Estimation

While our software is designed to detect the frequency of the camera's oscillatory motion, it can also be applied to estimate the vibration frequency of an object in the scene with a static camera.

To evaluate this, we display a triangle moving along a small circular path on a computer monitor, point the event camera at the screen, and apply VibES's initialization stack to estimate the triangle's motion frequency. Given the monitor's refresh rate of 75 Hz, we test a range of frequencies from 7.5 Hz to 22.5 Hz, noting that higher frequencies are affected by aliasing due to temporal sampling limitations. For each target frequency, we repeat the estimation process 10 times, reporting the mean estimated frequency and the corresponding 3σ error.

The results in Tab. 5 demonstrate that VibES achieves accurate frequency estimation with low bias and variance. We note, however, a slight increase in bias at higher frequencies, with deviations of up to 0.1 Hz between the ground truth and the predicted values. This effect is likely due to aliasing, as we are able to estimate higher frequencies in both simulated data Sec. 5.2 and real-world data.

5.2. Relative Depth Awareness

We explore how parallax induced by the camera vibration can encode information about scene depth. Note that this type of depth estimation is not possible in setups like the wedge-prism solution proposed by [24], because the camera center of projection does not change.

Scene	Method	Avg. Num. Components ↓	Avg. Contour Length ↑	Avg. Junction Count
AMI-EV [24]	S-EV	91.1 ± 17.2	4.2 ± 2.4	63.3 ± 54.6
	VibES	$\mathbf{82.9 \pm 14.1}$	$\mathbf{8.3 \pm 1.6}$	147.9 ± 31.8
Logo	S-EV	265.8 ± 221.4	3.6 ± 3.1	136.5 ± 165.8
	VibES	$\mathbf{175.0 \pm 52.5}$	$\mathbf{32.1 \pm 10.8}$	979.3 ± 349.2
Pattern Checkerboard	S-EV	300.3 ± 232.9	6.7 ± 5.3	193.2 ± 176.3
	VibES	$\mathbf{140.7 \pm 40.2}$	$\mathbf{49.9 \pm 17.3}$	691.3 ± 63.2
Pattern	S-EV	138.7 ± 102.3	5.6 ± 4.2	226.9 ± 197.5
	VibES	$\mathbf{116.3 \pm 35.1}$	$\mathbf{15.6 \pm 4.2}$	304.4 ± 127.6

Table 4. **Edge extraction evaluation on captured scenes.** We report the number of connected components (lower is better), the average contour length in pixels (higher is better), and the number of junctions, expressed as mean ± standard deviation across frames. Across most scenes, VibES reduces edge fragmentation and increases contour continuity compared to S-EV, indicating cleaner, more coherent edge maps.

Target (Hz)	Est. (Hz)	Avg. Abs. Error ↓	3σ ↓
7.5	7.486	0.0617	0.202
10.0	10.074	0.0754	0.177
12.6	12.684	0.0923	0.234
15.2	15.255	0.0764	0.264
17.4	17.476	0.0829	0.225
20.0	20.100	0.1000	0.189
22.6	22.644	0.0517	0.147

Table 5. **Scene frequency estimation**. VibES performs precise estimation of frequency, although performance degrades as the frequency increases. Reported results are based on 10 samples.

We test three synthetic scenes containing two staggered patterns at different depth planes, as in Fig. 6. To create synthetic data, we generate RGB frames using Blender [8] at 1000 FPS and convert them to events using V2E [9]. Camera motion is simulated according to the sinusoidal motion model described in Sec. 3.1. We instantiate two trackers per pattern. Then we store the amplitudes estimated by the EKF for each of the patterns and average them across the entire scene duration. Finally, we compute the ratio between the average amplitudes to get the estimation of the relative depth. Our simulated results show that our system reliably predicts the relative depth of the two patterns: for ground-truth relative depth ratios of 0.33, 0.50, and 0.66 the predicted ratios are 0.41 ± 0.01, 0.59 ± 0.02, and 0.80 ± 0.03. Please see Supplementary Sec. S2.6 for additional details.

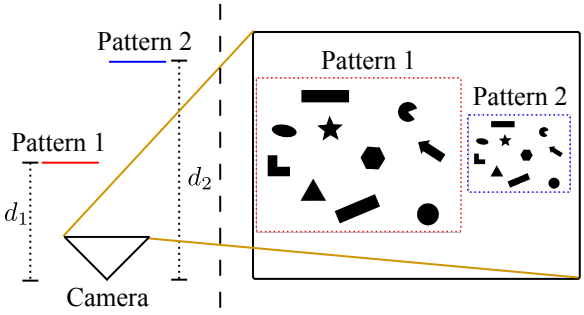

Figure 6. **Illustration of depth prediction setup**. The two patterns are placed at different distances from the camera and we estimate the ratio $\frac{d_1}{d_2}$ (or vice versa) by observing the amplitudes of the perceived motion of the camera measured by the tracker (Sec. 3.2.1).

6. Conclusion

In this work, we introduced VibES, a bio-inspired approach that enhances event cameras by actively inducing motion. Our method generates persistent event streams in static scenes, addressing a key limitation of event-based vision.

We see several promising avenues for future work. For example, the vibration system could be made adaptive—that is, disabled in highly dynamic scenes with dense event streams, and reactivated when the event density drops. Such an approach would preserve the low-power benefits of event cameras while ensuring a continuous visual signal when needed. Additionally, although our method already handles high event rates, downsampling or other event filtering techniques may be considered to further improve the real time performance, given the high event bandwidth generated by the induced motion. Our approach could also be combined with other event-based algorithms to improve performance on downstream tasks.

Acknowledgments

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) through the RGPIN program and the Canada Research Chairs program, the Canada Foundation for Innovation, and the Ontario Research Fund.

References

[1] Ignacio Alzugaray and Margarita Chli. HASTE: multi-hypothesis asynchronous speeded-up tracking of events. In *Brit. Mach. Vis. Conf. (BMVC)*, 2020. 4

[2] Samya Bagchi, Peter Anastasiou, Matthew Tetlow, Tat-Jun Chin, and Yasir Latif. Event-based star tracking under spacecraft jitter: the e-sturt dataset. *arXiv e-prints*, 2025. 3

[3] Dwijay Bane, Anurag Gupta, and Manan Suri. Non-invasive qualitative vibration analysis using event camera. *arXiv e-prints*, 2024. 2

[4] Alexander H Barnett, Jeremy Magland, and Ludvig af Klinteberg. A parallel nonuniform fast fourier transform library based on an "exponential of semicircle" kernel. *SIAM J. Sci. Comput.*, 2019. 4

[5] Christian Brändli, Jonas Strubel, Susanne Keller, Davide Scaramuzza, and Tobi Delbruck. Elised — an event-based line segment detector. In *Int. Conf. Event-Based Control, Commun. Signal Proc. (EBCCSP)*, 2016. 2

[6] Zhiwen Chen, Jinjian Wu, Junhui Hou, Leida Li, Weisheng Dong, and Guangming Shi. Ecsnet: Spatio-temporal feature learning for event camera. *IEEE TCSVT*, 2022. 2

[7] FJJ Clarke. A study of troxler's effect. *Opt. Acta: Int. J. Opt.*, 1960. 1

[8] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 8, 13

[9] Tobi Delbruck, Yuhuang Hu, and Zhe He. V2e: From video frames to realistic dvs event camera streams. *arXiv e-prints*, 2020. 8

[10] DJI. Dji robomaster m2006 p36 brushless dc gear motor, 2025. 15

[11] Yan Dong and Tao Zhang. Standard and event cameras fusion for feature tracking. In *Proc. Int. Conf. Mach. Vis. and Appl. (ICMVA)*, 2021. 2

[12] Dynamixel. Dynamixel mx-28t. http://emanual.robotis.com/docs/en/dxl/mx/mx-28t/. 15

[13] Giulia D'Angelo, Victoria Clerico, Chiara Bartolozzi, Matej Hoffmann, P Michael Furlong, and Alexander Hadjiivanov. Wandering around: A bioinspired approach to visual attention through object motion sensitivity. *Neuromorphic Comput. Eng.*, 2025. 2, 3

[14] SparkFun Electronics. Geared hobby motor. https://www.sparkfun.com/hobby-motor-gear.html, 2025. 5, 15

[15] Christophe Fiorio and Jens Gustedt. Two linear time union-find strategies for image processing. *Theor. Comput. Sci.*, 1996. 6

[16] FLIR. Ptu d46. https://www.sustainable-robotics.com/reference/PTU/ptu_d46_datasheet.pdf. 15, 16

[17] FLIR. Ptu e46. https://www.sustainable-robotics.com/reference/PTU/PTU-manual-E46-1.00-SMALL.pdf, 2014. 15, 16

[18] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 2, 6, 14

[19] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J. Davison, Jörg Conradt, Kostas Daniilidis, and Davide Scaramuzza. Event-based vision: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2022. 1

[20] Daniel Gehrig and Davide Scaramuzza. Low-latency automotive vision with event cameras. *Nature*, 2024. 1

[21] Daniel Gehrig, Henri Rebecq, Guillermo Gallego, and Davide Scaramuzza. Eklt: Asynchronous photometric feature tracking using events and frames. *Int. J. Comput. Vis.*, 2020. 2

[22] Arren Glover and Chiara Bartolozzi. Robust visual tracking with a freely-moving event camera. In *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*. IEEE, 2017. 2

[23] Guangsha Guo, Yang Feng, Hengyi Lv, Yuchen Zhao, Hailong Liu, and Guoling Bi. Event-guided image super-resolution reconstruction. *Sensors*, 2023. 2

[24] Botao He, Ze Wang, Yuan Zhou, Jingxi Chen, Chahat Deep Singh, Haojia Li, Yuman Gao, Shaojie Shen, Kaiwei Wang, Yanjun Cao, Chao Xu, Yiannis Aloimonos, Fei Gao, and Cornelia Fermüller. Microsaccade-inspired event camera for robotics. *Sci. Robot.*, 2024. 1, 2, 3, 5, 6, 7, 8, 15

[25] Zhe Jiang, Yu Zhang, Dongqing Zou, Jimmy Ren, Jiancheng Lv, and Yebin Liu. Learning event-based motion deblurring. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2

[26] Yongcheng Jing, Yiding Yang, Xinchao Wang, Mingli Song, and Dacheng Tao. Turning frequency to resolution: Video super-resolution via event cameras. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 2

[27] Lingdong Kong, Youquan Liu, Lai Xing Ng, Benoit R Cottereau, and Wei Tsang Ooi. Openess: Event-based semantic scene understanding with open vocabularies. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2024. 1

[28] Bart Krekelberg. Microsaccades. *Curr. Biol.*, 2011. 1

[29] Siqi Li, Zhikuan Zhou, Zhou Xue, Yipeng Li, Shaoyi Du, and Yue Gao. 3d feature tracking via event camera. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2024. 2

[30] Maximilian PR Löhr and Heiko Neumann. Contrast detection in event-streams from dynamic vision sensors with fixational eye movements. In *IEEE Int. Symp. Circuits Syst. (ISCAS)*. IEEE, 2018. 3

[31] Ryota Maeda, Yunseong Moon, and Seung-Hwan Baek. Event ellipsometer: Event-based mueller-matrix video imaging. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2025. 3

[32] Nico Messikommer, Carter Fang, Mathias Gehrig, and Davide Scaramuzza. Data-driven feature tracking for event cameras. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2023. 1, 2

[33] A. Mittal, R. Soundararajan, and A. C. Bovik. Making a "completely blind" image quality analyzer. *IEEE Sign. Process. Letters*, 2013. 6, 14

[34] Garrick Orchard, Ajinkya Jayawant, Gregory K. Cohen, and Nitish Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. *Front. Neurosci.*, 2015. 2, 3, 15

[35] W Pabst and E Gregorova. Characterization of particles and particle systems. *ICT Prague*, 2007. 6

[36] Liyuan Pan, Cedric Scheerlinck, Xin Yu, Richard Hartley, Miaomiao Liu, and Yuchao Dai. Bringing a blurry frame alive at high frame-rate with an event camera. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2

[37] Bernd Pfrommer. Frequency cam: Imaging periodic signals in real-time. *arXiv e-prints*, 2022. 2

[38] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. Events-to-video: Bringing modern computer vision to event cameras. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 6, 7

[39] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. High speed and high dynamic range video with an event camera. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019. 6

[40] C. E. Shannon. A mathematical theory of communication. *Bell Sys. Tech. J*, 1948. 6, 14

[41] Waseem Shariff, Mehdi Sefidgar Dilmaghani, Paul Kielty, Mohamed Moustafa, Joe Lemley, and Peter Corcoran. Event cameras in automotive sensing: A review. *IEEE Access*, 2024. 1

[42] Radim Spetlik, Tereza Uhrová, and Jiří Matas. Efficient real-time quadcopter propeller detection and attribute estimation with high-resolution event camera. In *Scandinavian Conf. on Im. Analysis (SCIA)*. Springer, 2025. 2

[43] Simone Testa, Giacomo Indiveri, and Silvio P Sabatini. Dynamic detectors of oriented spatial contrast from isotropic fixational eye movements. SciTePress, 2020. 2, 3, 15

[44] Simone Testa, Silvio P Sabatini, and Andrea Canessa. Active fixation as an efficient coding strategy for neuromorphic vision. *Sci. Rep.*, 2023. 2, 3, 15

[45] 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. In *IEEE Robot. Autom. Lett.*, 2018. 2

[46] Eric G Wu, Nora Brackbill, Colleen Rhoades, Alexandra Kling, Alex R Gogliettino, Nishal P Shah, Alexander Sher, Alan M Litke, Eero P Simoncelli, and EJ Chichilnisky. Fixational eye movements enhance the precision of visual information transmitted by the primate retina. *bioRxiv*, 2023. 1

[47] Kesheng Wu, Ekow Otoo, and Arie Shoshani. Optimizing connected component labeling algorithms. In *Im. Proc.* SPIE, 2005. 6

[48] Lan Xu, Weipeng Xu, Vladislav Golyanik, Marc Habermann, Lu Fang, and Christian Theobalt. Eventcap: Monocular 3d capture of high-speed human motions using an event camera. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2

[49] Amirreza Yousefzadeh, Garrick Orchard, Teresa Serrano-Gotarredona, and Bernabé Linares-Barranco. Active perception with dynamic vision sensors. minimum saccades with optimum recognition. *IEEE Trans. Biomed. Circuits Syst.*, 2018. 2, 3

[50] Jiqing Zhang, Bo Dong, Haiwei Zhang, Jianchuan Ding, Felix Heide, Baocai Yin, and Xin Yang. Spiking transformers for event-based single object tracking. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 2

[51] T. Y. Zhang and C. Y. Suen. A fast parallel algorithm for thinning digital patterns. *Commun. ACM*, 1984. 6