EMVSI: Event based Multi-Velocity-Scale Imaging

Qiyao Gao[1,2] **Xinyu Zhou**[3,4] **Peiqi Duan**[3,4*] **Chu Zhou**[2*] **Boxin Shi**[3,4] **Imari Sato**[1,2]

[1]Graduate School of Information Science and Technology, The University of Tokyo [2]National Institute of Informatics
[3]State Key Laboratory of Multimedia Information Processing, School of Computer Science, Peking University
[4]National Engineering Research Center of Visual Technology, School of Computer Science, Peking University

gaoqy@g.ecc.u-tokyo.ac.jp zhou_chu@hotmail.com
{zhouxiny,duanqi0001,shiboxin}@pku.edu.cn imarik@nii.ac.jp

Abstract

Real-world scenes contain static, slow, and fast components with heterogeneous sensing requirements. Frame cameras share global exposure and sampling settings, trading off fast-motion fidelity, light collection, and acquisition bandwidth. Event cameras provide asynchronous sensing, but a single global contrast-threshold setting can either suppress weak responses or produce excessive events. We propose multi-threshold event imaging with motion-compatible thresholds and temporal supports for different scene components. A periodic global threshold-switching pipeline approximates the ideal spatially adaptive model on existing event cameras, followed by motion-layer association, temporal fusion, and intensity reconstruction. Experiments on controlled and real-captured scenes demonstrate improved recovery of heterogeneous motion components and reduced redundancy relative to the evaluated fixed-threshold configurations.

1 Introduction

Real-world scenes often contain motion spanning multiple velocity scales [1][2], from static background structures to slow object motion and fast dynamic components within the same field of view, as shown in Fig. 1(a). Capturing such scenes faithfully is fundamentally challenging because different motion regimes impose incompatible sensing requirements. Fast motion demands high temporal resolution to avoid overlap and blur, whereas static and slow content require sufficient temporal accumulation to preserve structural information [3][4]. We refer to this task as *multi-velocity-scale imaging (MVSI)*[1]: sensing and reconstructing heterogeneous motion while preserving fidelity across temporal scales.

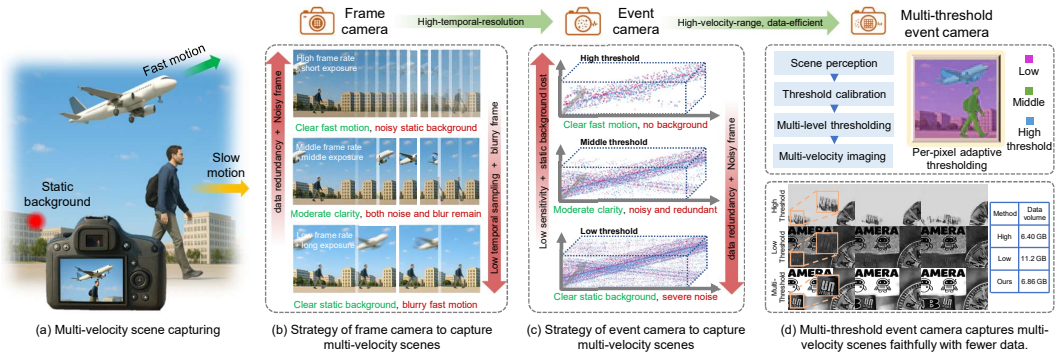
Conventional frame-based cameras are inherently mismatched to this task. As shown in Fig. 1(b), their acquisition mechanism is governed by a global exposure and a fixed frame rate, forcing all scene components to share the same temporal sampling schedule [5]. In multi-velocity scenes, the sensing parameters must therefore be chosen with respect to the fastest motion to avoid blur, which inevitably oversamples static and slow regions and results in redundancy [6]. This trade-off is structural rather than algorithmic, and cannot be resolved by reconstruction alone.

Event cameras offer a more suitable sensing paradigm. Instead of periodically reading out full frames, they asynchronously report per-pixel brightness changes with microsecond temporal resolution and low latency [7][8]. This event-driven paradigm avoids unnecessary global sampling and is, in principle, well aligned with the heterogeneous temporal demands of MVSI. However, directly deploying a standard event camera does not solve the problem. Existing event cameras operate with a

*Co-corresponding authors: Peiqi Duan and Chu Zhou.

[1]Here, *imaging* denotes the complete event-sensing and intensity-reconstruction pipeline, rather than direct image output from an event camera.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) Multi-velocity scene capturing (b) Strategy of frame camera to capture multi-velocity scenes (c) Strategy of event camera to capture multi-velocity scenes (d) Multi-threshold event camera captures multi-velocity scenes faithfully with fewer data.

Figure 1: (a) Multi-velocity scale scenes contain components spanning different motion scales, including static regions, slow motion, and fast motion. (b) Frame cameras share global exposure and sampling settings, creating trade-offs among motion blur, per-frame light collection, and acquisition bandwidth. (c) Standard event cameras use a global contrast threshold, leading to a trade-off between sensitivity and data efficiency: low thresholds preserve weak signals but generate redundant events, while high thresholds reduce data but lose static or slow-varying details. (d) We approximate a spatially adaptive sensing model on standard hardware through periodic global threshold switching, motion-layer association, and reconstruction. In the evaluated multi-velocity scenes, scheduled acquisition preserves complementary motion responses while reducing redundant event volume relative to the low-threshold comparison.

single global threshold, which controls event triggering uniformly across all pixels [9]. As shown in Fig. 1(c), such a global threshold is fundamentally incompatible with multi-velocity scenes: a low threshold preserves weak responses from static or slow components but produces excessive events under fast motion [10], while a high threshold suppresses redundancy at the cost of losing static and low-speed structure [11][12]. As a result, a single-threshold configuration cannot simultaneously match the sensing requirements of static, slow, and fast dynamics.

To address this limitation, we advocate a multi-threshold imaging model for event-based MVSI, as shown in Fig. 1(d). Instead of assigning one global threshold to the entire sensor, the ideal multi-threshold event camera should adapt the threshold according to local scene dynamics and appearance, so that different motion regimes are sampled with compatible sensitivity. Intuitively, static or weakly changing regions require lower thresholds to maintain sufficient responsiveness, whereas rapidly changing regions benefit from higher thresholds to avoid redundant triggering and bandwidth saturation [13][14]. From this perspective, the quality of MVSI reconstruction is fundamentally tied to the sensing process, particularly whether the event stream is acquired under motion-compatible thresholds.

Realizing this idea is nontrivial and raises three key challenges. **First, how should the optimal threshold be determined for each motion regime?** Multi-velocity scenes contain heterogeneous dynamics, and threshold selection directly governs sensing quality and data efficiency. Moreover, static and dynamic event generation follow fundamentally different triggering behaviors, rendering a unified selection criterion inadequate. We therefore develop separate threshold estimation strategies for static and dynamic components, enabling scene-adaptive threshold assignment under heterogeneous motion conditions. **Second, how can event streams arising from different motion scales be unified for reconstruction?** Because different scene components are best captured under different temporal supports, their event streams are inherently heterogeneous in density and temporal extent. A single-window representation is therefore mismatched to multi-velocity scenes, as it necessarily under-represents some motions while oversampling others. To address this issue, we introduce a multi-stream fusion strategy that aligns and aggregates heterogeneous event streams into a unified representation while preserving motion-specific structure. **Third, how can multi-threshold imaging be realized with existing hardware?** Current event sensors do not support true pixel-wise threshold programmability. We therefore propose a practical acquisition pipeline that enables threshold-aware capture of multi-velocity scenes with a standard event camera. This provides a hardware-compatible realization of the proposed imaging paradigm without requiring new sensor fabrication.

Our contributions are summarized as follows:

- We formulate multi-velocity-scale imaging as a sensing and reconstruction problem for event cameras, and propose a multi-threshold imaging model to address the threshold mismatch inherent in single-threshold sensing.
- We develop a scene-adaptive threshold estimation and multi-stream fusion framework that determines motion-compatible thresholds for heterogeneous scene components and unifies their event representations across different temporal windows.
- We present a practical realization strategy for multi-threshold event imaging using standard hardware, including a threshold-aware capturing pipeline, which enables deployable MVSI without requiring new sensor fabrication.

2 Related Work

Frame-based multi-velocity-scale imaging. Frame-based cameras remain the dominant sensing modality in computer vision, benefiting from mature CMOS sensor design [15, 16] and powerful reconstruction pipelines [17, 18, 19]. However, their globally synchronized acquisition is inherently mismatched to multi-velocity-scale scenes. Each frame integrates radiance over a finite exposure and samples the scene at a fixed frame rate, leading to motion blur, temporal aliasing, and loss of fast dynamics under high-speed motion [20, 21, 22]. Increasing the frame rate can reduce blur, but it also oversamples static and slow regions, causing substantial redundancy in bandwidth, storage, and computation. Hybrid frame–event systems have been explored to alleviate these limitations [23, 24], but the globally sampled nature of frame-based sensing remains inefficient for scenes containing static, slow, and fast components simultaneously.

Event-based multi-velocity-scale imaging. Event cameras report asynchronous brightness changes once the logarithmic intensity variation at each pixel exceeds a contrast threshold. Their microsecond temporal resolution, high dynamic range, and motion-blur-free sensing make them well suited for fast dynamics and high-contrast scenes [25, 26, 27, 28]. To bridge the gap between sparse events and conventional visual representations, event-to-video reconstruction has been extensively studied. E2VID [29] introduced recurrent reconstruction from events, followed by improvements in datasets [25], lightweight architectures [30], and HDR or high-frame-rate reconstruction [31, 32, 33]. Nevertheless, reconstruction remains challenging when event activity is insufficient, especially for static or low-motion regions. Hybrid frame–event systems [34, 35] can provide additional intensity cues, but require extra hardware and reduce the data-efficiency advantage of event sensing.

Lowering sensor thresholds can reveal useful responses in static scenes, enabling unified reconstruction of static and dynamic content [10, 36], but one operating setting remains a sensitivity–redundancy compromise. Prior work addresses variable event density and motion with sparse event selection [37], bounding-box memory and filtering [38], adaptive temporal decay [39], and frame integration at different durations [40]. Sensor-level STC/Trail filtering also suppresses selected triggered events [41]. Time-surface representations likewise re-encode previously generated events and introduce their own temporal-scale choices. Unlike existing methods that process or filter events after they are generated, our proposed imaging model directly optimizes the acquisition process through adaptive threshold scheduling, mitigating information loss from overly high thresholds and readout saturation from excessive events at overly low thresholds.

3 Method

3.1 Multi-threshold Imaging Model

Event reconstruction requires a threshold–window pair suited to the scene and representation. Short windows help limit fast-motion overlap, while longer integration may support weak or slowly changing content; the appropriate duration also depends on contrast, noise, and the reconstruction task. Once an integration window is selected, sensor bias becomes a dominant factor in imaging quality by directly regulating the event triggering rate: overly insensitive settings suppress useful responses, whereas overly sensitive settings generate redundant events and may saturate sensor readout. Thus, heterogeneous motion benefits from different acquisition settings.

We verify this behavior through the controlled experiment in Fig. 2. Specifically, we move a printed grayscale pattern at three representative image-plane speeds of 1000, 400, and 10 pixels/s, and

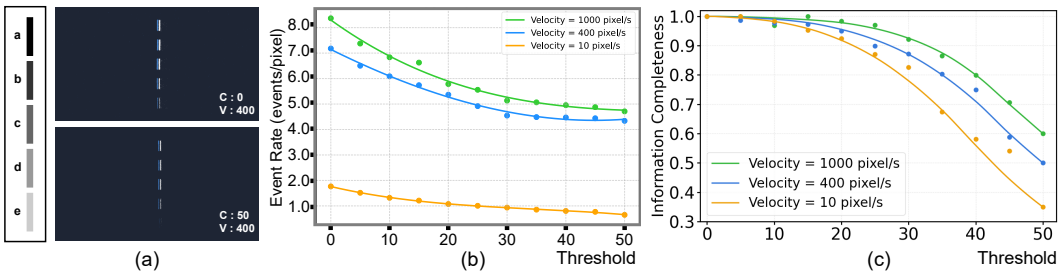


Figure 2: (a) Event visualizations of the pattern moving at $v = 400$ pixel/s under two threshold settings, $C = 0$ and $C = 50$, showing that high thresholds suppress weakly triggered structures. (b) Event rate as a function of threshold for different motion speeds. The distinct decay profiles indicate that the event response is strongly motion-dependent. (c) Information completeness, computed as the fraction of denoised valid event pixels retained under each threshold relative to the denoised reference at $C = 0$, decreases with increasing threshold at different rates for different velocities, suggesting that each motion scale requires its own operating threshold.

record the resulting events with a Prophesee EVK4 event camera. The sensor uses a tunable contrast threshold C: increasing C suppresses event triggering, while decreasing C makes the sensor more responsive. For each speed, we sweep the operating threshold from $C = 0$ to $C = 50$ and record the corresponding event stream. Fig. 2(a) shows representative event visualizations at $v = 400$ pixels/s. When the threshold is increased from $C = 0$ to $C = 50$, weak responses from the pattern are visibly suppressed, indicating that high thresholds may remove not only noise or redundancy but also valid scene structures.

We first quantify the threshold response using the event rate, measured as the number of triggered events per pixel within the integration window associated with each speed. Fig. 2(b) shows that the event rate decreases as the threshold increases across all speeds, but with distinct decay profiles: fast motion produces more events and is more tolerant to high thresholds, while slow motion is more sensitive to threshold increase. This indicates that a global threshold cannot provide consistent sensing quality across motion scales.

We further measure the preservation of pattern structure using information completeness. For each velocity and threshold, we denoise the raw event stream using the denoising method [42], and treat the remaining events as valid events. Taking the valid events at $C = 0$ as the reference, information completeness is defined as

$$\mathrm{IC}(v, C) = \frac{N_{\mathrm{valid}}(v, C)}{N_{\mathrm{valid}}(v, 0)}, \tag{1}$$

where $N_{\mathrm{valid}}(v, C)$ denotes the number of denoised valid event pixels retained within the integration window associated with velocity v and threshold C. As shown in Fig. 2(c), completeness remains high at low thresholds, where mainly background activity and redundant weak responses are suppressed, but drops rapidly once high thresholds begin to remove valid pattern responses. The onset and severity of this drop differ across velocities: slow motion loses valid pattern responses earlier, while fast motion can tolerate a higher threshold before comparable degradation occurs. These results demonstrate that the operating threshold should be selected according to local motion speed, rather than being fixed globally for the entire scene.

Motivated by this observation, we replace the globally fixed threshold with a spatially adaptive threshold field. Let $L(\mathbf{x}, t)$ denote the latent log-intensity at pixel $\mathbf{x}$ and time t, and let $t_{\mathbf{x}}^{-}$ denote the timestamp of the most recent event at the same pixel. The ideal multi-threshold event camera is defined as

$$\left| L(\mathbf{x}, t) - L(\mathbf{x}, t_{\mathbf{x}}^{-}) \right| \geq C(\mathbf{x}, t), \qquad C(\mathbf{x}, t) = f(v(\mathbf{x}, t)) \,. \tag{2}$$

Here, $C(\mathbf{x}, t)$ denotes the adaptive operating threshold, and $v(\mathbf{x}, t)$ denotes the local motion speed. The mapping f assigns an operating threshold according to the local motion. This formulation allows static, slow, and fast components to be sensed under different threshold levels within the same scene.

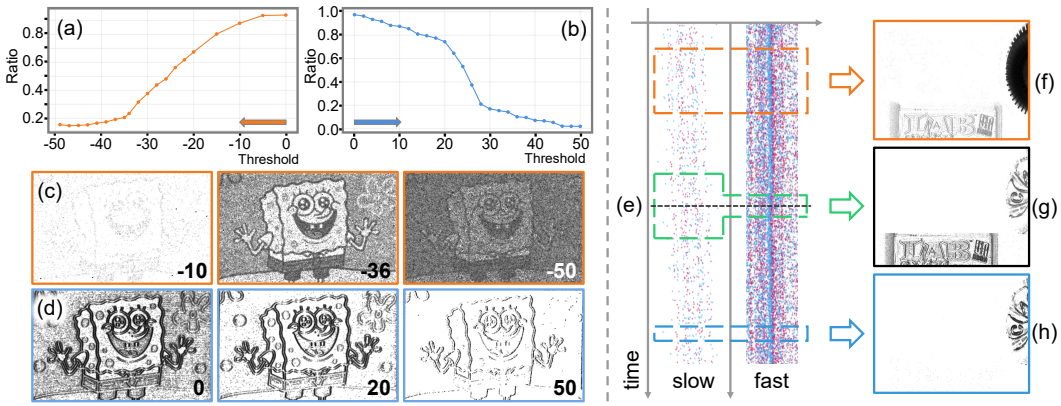


Figure 3: Scene-adaptive threshold estimation and heterogeneous-window fusion. (a) & (c) For static content, lowering the threshold progressively activates more scene structures, but overly low thresholds introduce dense noise. (b) & (d) For dynamic content, increasing the setting suppresses isolated responses, whereas overly insensitive settings can remove spatially supported motion responses. (e)∼(h) A scene with a fast-rotating fan and a slowly moving box illustrates the limitations of a single temporal window: wide integration (f) preserves slow structures but overlaps fast motion, whereas narrow integration (h) preserves fast motion but misses slow structures. Heterogeneous windows (g) adapt the integration time to each velocity scale and preserve both components.

3.2 Scene-adaptive Threshold Estimation

Efficient MVSI requires estimating a threshold that is compatible with each scene component. We treat static and dynamic regions separately, since the two regimes arise from distinct triggering mechanisms: static activity is dominated by noise-driven responses [13], whereas dynamic activity is induced by motion-dependent brightness changes [8].

Static threshold estimation. For static content, lowering the threshold initially activates previously silent structure, but further reduction increasingly amplifies noise and redundant triggering. To detect this transition, we progressively decrease the threshold and track the *fraction of incremental events contributed by newly activated pixels*,

$$r_{\text{static}}(C) = \Delta N_{\text{new}}(C)/\Delta N_{\text{all}}(C), \tag{3}$$

where $\Delta N_{\text{new}}(C)$ denotes the number of additional events produced by pixels that become active only after decreasing the threshold to C, and $\Delta N_{\text{all}}(C)$ denotes the total increase in event count. As $r_{\text{static}}(C)$ approaches zero, further threshold reduction no longer introduces meaningful spatial support and instead mainly increases redundant noise activity. We therefore select the first stable saturation point of this curve as the static operating threshold. In practice, it is the first threshold at which $r_{\text{static}}(C)$ falls to 20%. Fig. 3(a) shows it saturates near -36 (ON bias), and the integrated event frames in Fig. 3(c) exhibit the corresponding transition from under-activation to over-activation.

Dynamic threshold estimation. For dynamic content, event generation is primarily driven by motion-dependent brightness changes. Increasing the threshold suppresses isolated events—most of which are triggered by noise and exhibit spatial isolation—earlier than it suppresses motion-driven events. To exploit this behavior, we progressively increase the threshold and, at each threshold setting, compute the *fraction of spatially isolated deactivated pixels*,

$$r_{\text{dyn}}(C) = N_{\text{iso}}(C)/N_{\text{deact}}(C), \tag{4}$$

where $N_{\text{iso}}(C)$ denotes the number of isolated pixels that disappear after increasing the threshold to C, and $N_{\text{deact}}(C)$ denotes the total number of deactivated pixels. A high $r_{\text{dyn}}(C)$ indicates predominantly isolated responses, while a sharp decline suggests the loss of spatially supported responses. In practice, We select the threshold preceding the first drop exceeding 0.15 in $r_{\text{dyn}}(C)$ between consecutive settings as the dynamic operating threshold.

Fig. 3(b) shows an inflection near 20. The corresponding qualitative reconstruction (Fig. 3(d)) supports this selection in the measured scene; the criteria in Eqs. (3)–(4) are empirical change-point rules, not global optimality guarantees.

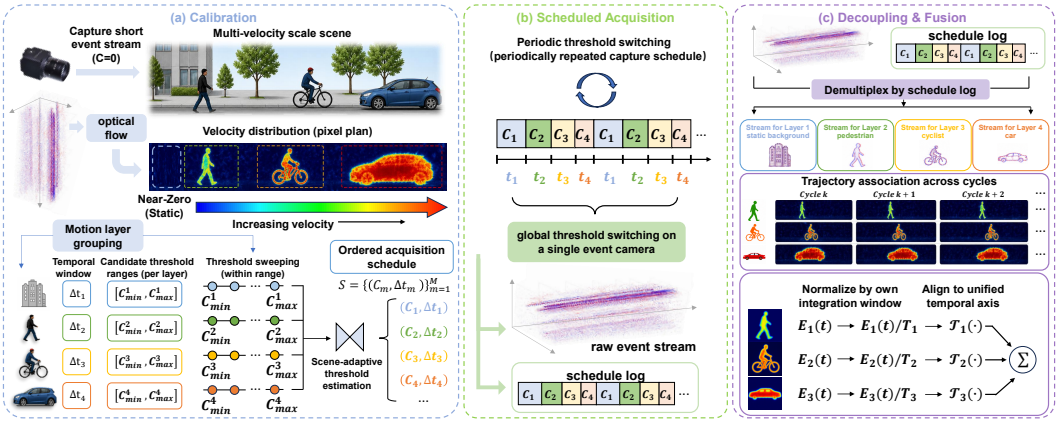


Figure 4: Scheduled acquisition overview. (a) **Calibration:** A short default-bias stream yields coarse event flow; flow-magnitude histogram modes with spatial support define a variable number M of motion layers (four are illustrative). Each layer receives a candidate threshold range and temporal window, yielding an ordered schedule $\mathcal{S} = \{(C_m, \Delta t_m)\}_{m=1}^{M}$. (b) **Scheduled acquisition:** a single event camera periodically switches among the layer-specific thresholds according to $\mathcal{S}$, producing a continuous raw event stream together with a schedule log. (c) **Decoupling and fusion:** the raw stream is demultiplexed into layer-specific streams using the schedule log, moving components are associated across acquisition cycles, and the resulting heterogeneous streams are temporally aligned and fused into a unified event representation for downstream reconstruction.

3.3 Multi-threshold Temporal Fusion

Multi-threshold event streams cannot be directly integrated, even if each velocity-scale component is captured at its corresponding operating threshold. Fast-motion streams are temporally dense and require short windows to avoid temporal overlap, while slow or quasi-static streams are temporally sparse and require longer accumulation to form complete structures. As shown in Fig. 3(e)–(h), a single wide window recovers slow structures but overlaps fast motion, whereas a single narrow window preserves fast motion but loses slow structures. Thus, a single global window is inherently mismatched to multi-velocity scenes.

We address this with heterogeneous-window fusion. Suppose the scene is decomposed into N motion layers, since different layers are accumulated over different temporal extents, as illustrated by the heterogeneous windows in Fig. 3(e), their raw event counts are not comparable. We therefore normalize each layer by its window width to obtain comparable temporal event densities. Then, the streams are aligned by their first timestamps so that all layers span the fusion interval synchronously:

$$E_{\text{fusion}}(t) = \sum_{i=1}^{N} \mathcal{T}_i(E_i(t)/T_i)\,, \tag{5}$$

where $E_{\text{fusion}}(t)$ is the fused event stream, $E_i(t)$ is the accumulated event representation of the i-th motion layer, and T_i is its integration width. The normalization term $E_i(t)/T_i$ corresponds to the density equalization needed when combining the different temporal supports in Fig. 3(e). The operator $\mathcal{T}_i(\cdot)$ shifts the local timestamp origin of each window to the common fusion time t, preventing events from different windows from falling into inconsistent temporal bins. This produces a unified representation that preserves fast-motion sharpness while maintaining sufficient support for slow and quasi-static structures, as reflected by the heterogeneous-window result in Fig. 3(g).

3.4 Scheduled Multi-threshold Acquisition

Since current commercial event cameras use a global threshold configuration and cannot adjust thresholds locally across the image plane, they cannot be deployed in multi-velocity scale scenes for capturing. To address this limitation, we propose a scheduled acquisition scheme that implements multi-threshold imaging on a standard event camera, the Prophesee EVK4, without requiring new sensor hardware. Instead of requiring simultaneous pixel-wise threshold control, we first estimate the

motion layers present in the scene, assign each layer an operating threshold, and then execute these thresholds sequentially within a periodic capture schedule. An overview of the acquisition pipeline is shown in Fig. 4.

Calibration. We first record a short event stream at the default threshold setting, $C = 0$, and estimate a coarse image-plane velocity distribution using a lightweight, custom-implemented optical-flow estimator. The resulting motion field is grouped into M motion layers, each associated with a dominant scene component, such as the static background, a slow-moving foreground object, or a fast-moving foreground object. For each dynamic motion layer, we set the integration window according to its estimated image-plane speed v (pixels/s), approximately as the time for a one-pixel displacement, $\Delta t \approx 1/v$. The static background is treated as a separate layer with a fixed 10 ms integration window per acquisition cycle. We then assign an empirical candidate threshold range based on the static and dynamic response calibrations in Appendix A.1, and sequentially sweep the thresholds within this range to record calibration streams.

These streams are processed by the scene-adaptive threshold estimation algorithm described in Sec. 3.2 to determine the operating threshold for each motion layer. Since different motion scales also require different temporal windows, each layer is assigned a threshold–window pair. This yields an ordered acquisition schedule,

$$\mathcal{S} = \{(C_m, \Delta t_m)\}_{m=1}^{M},$$

where C_m denotes the operating threshold for the m-th motion layer and Δt_m denotes its corresponding temporal window.

Scheduled acquisition. During acquisition, the camera globally switches among the thresholds in the schedule. Each threshold C_m is held for its assigned temporal window Δt_m, and the complete schedule is repeated periodically throughout the recording. This produces a continuous raw event stream together with a log that records the threshold value active during each temporal window.

Decoupling and fusion. We first partition the raw stream into capture cycles and threshold intervals using the schedule log. Within each cycle, we progressively subtract higher-threshold activation masks from lower-threshold ones, yielding approximately disjoint spatial supports for motion-layer separation. For each motion layer, events from adjacent acquisition cycles are further analyzed to estimate their trajectories across cycles. This allows us to associate threshold-specific events with the corresponding moving components despite the time-multiplexed acquisition. The resulting event subsets capture different scene components under their assigned operating thresholds and temporal windows. Finally, since the streams associated with different motion layers exhibit heterogeneous spatiotemporal characteristics, we apply the proposed multi-threshold temporal fusion algorithm to align and normalize them into a unified event representation for downstream reconstruction.

3.5 Network and Implementation Detail

To evaluate the proposed imaging model for multi-velocity scene reconstruction, we adopt a recurrent reconstruction network inspired by [36]. The network receives two representations per output interval: an accumulated 2D event frame for static or quasi-static structures and a fixed-bin voxel grid that preserves the temporal ordering of motion-induced events. They are concatenated and passed to a compact U-Net/ConvLSTM decoder [43, 44]. The network receives neither threshold values nor schedule identifiers. It is trained with PyTorch/AdamW [45], an initial learning rate of 10^{-4}, spatial/temporal augmentation, and frame-wise MSE loss.

Training uses the publicly available E-StaDyn dataset [36], following a similar event-frame/voxel-grid preprocessing pipeline. For evaluation, 15 Blender scenes with static backgrounds and differently moving foregrounds are displayed and acquired using a Prophesee EVK4 under fixed low, fixed high, and scheduled protocols. See Appendix B.7 for data and baseline details.

4 Experiments

4.1 Comparison of Synthetic Data

Since this work focuses on event-based video reconstruction in scenes with static backgrounds and dynamic foreground, we evaluate the proposed method on our in-house benchmark against two representative baselines: E2VID [29] and URSEE [36]. These baselines cover two relevant

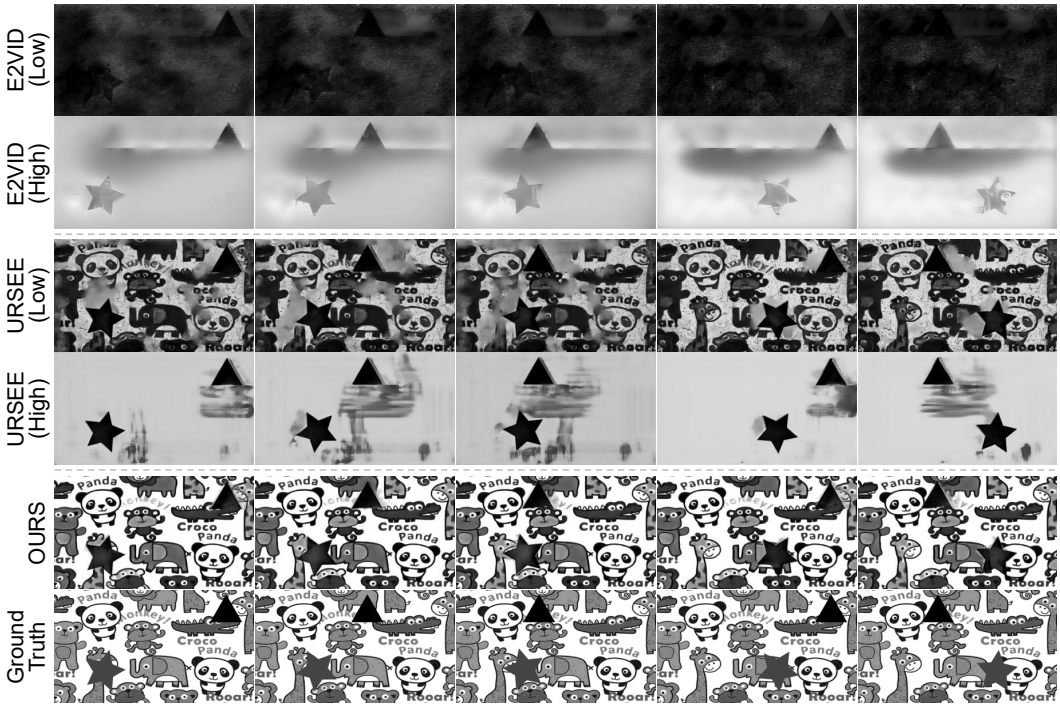


Figure 5: Qualitative comparison on the local synthetic test set. Fixed-threshold event streams are captured at low threshold ($C = -35$) and high threshold ($C = 50$), and reconstructed using E2VID [29] and URSEE [36]. Compared with fixed-threshold baselines, our scheduled multi-threshold acquisition better preserves both static background details and moving objects across different motion scales.

Table 1: Quantitative comparison of our method with mainstream approaches for event-based dynamic video reconstruction on our local test dataset. Arrows ↑ (↓) indicate that higher (lower) values are preferable. Best performance is highlighted in **bold**.

	E2VID (High)	E2VID (Low)	URSEE (High)	URSEE (Low)	Ours
PSNR ↑	9.21	3.70	9.49	8.78	**20.42**
SSIM ↑	0.48	0.06	0.47	0.35	**0.92**
LPIPS ↓	0.88	1.00	0.86	0.56	**0.08**
Data Volume (GB) ↓	**7.49**	13.33	**7.49**	13.33	8.01

reconstruction settings: generic event-to-video intensity reconstruction and event-based reconstruction with explicit static-scene modeling. We report PSNR, SSIM, and LPIPS to measure reconstruction fidelity, and additionally compare the event data volume to assess sensing efficiency.

Figure 5 and Table 1 show the qualitative and quantitative results. E2VID [29] is unable to recover static background structures regardless of the threshold, while URSEE [36] recovers them only in the low-threshold setting. The quantitative results further reveal the limitation of fixed-threshold acquisition: high thresholds reduce event volume but lose critical scene information, whereas low thresholds preserve more content but introduce substantial noise and redundancy. By assigning suitable thresholds to different dynamic components, our method achieves the best reconstruction quality while using only 60% of the event volume required by the low-threshold setting.

4.2 Comparison of Real-captured Data

To evaluate the proposed method on real data, we capture an outdoor multi-velocity scene with a Prophesee EVK4 event camera. The scene contains a static background and a person slowly walking forward while rapidly shaking a handheld airplane model, producing static, slow, and fast motion components. We record the scene with three acquisition protocols: a fixed low threshold ($C = -35$),

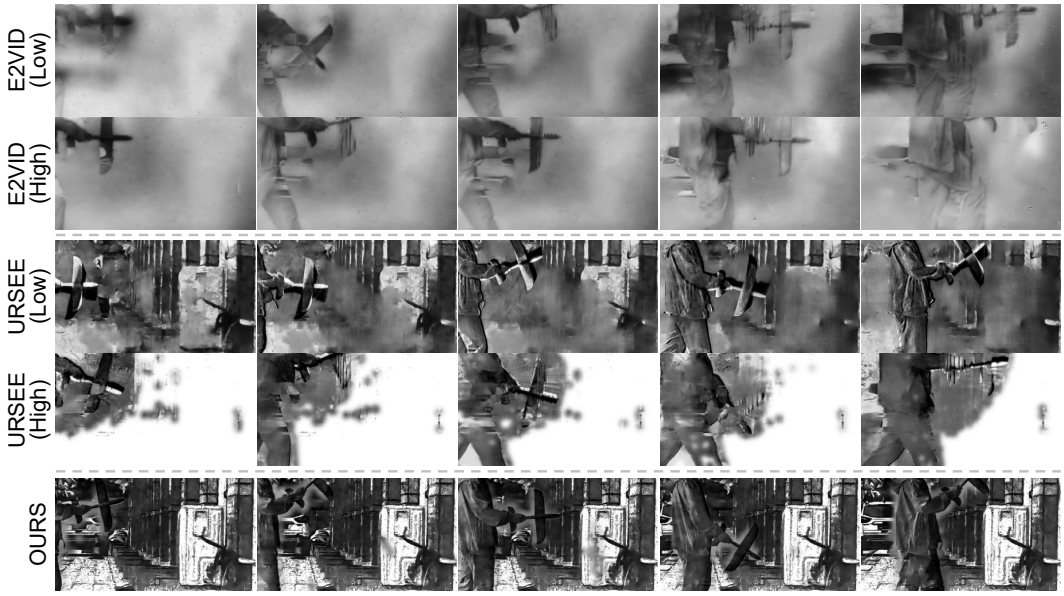


Figure 6: Qualitative comparison on a real outdoor multi-velocity scene containing a static background, a person slowly walking forward, and a rapidly shaken handheld airplane model. The scene is recorded under three acquisition protocols: a fixed low threshold ($C = -35$), a fixed high threshold ($C = 50$), and the proposed scheduled multi-threshold acquisition. Rows show E2VID [29] and URSEE [36] reconstructed from the low- and high-threshold streams, followed by our method reconstructed from the scheduled multi-threshold stream. Since the outdoor motion is manually repeated, the sequences are not strictly frame-aligned; this figure is intended for qualitative real-data comparison. All results are reconstructed at the same output interval.

a fixed high threshold ($C = 50$), and our scheduled multi-threshold acquisition. The fixed-threshold streams are reconstructed using E2VID [29] and URSEE [36].

Since the outdoor sequence involves manual motion, we repeat the same action as consistently as possible, but the walking trajectory and airplane shaking cannot be exactly reproduced across recordings. Thus, Fig. 6 serves as a qualitative real-data comparison rather than a strictly frame-aligned benchmark. Results show that fixed-threshold sensing either loses static structures or introduces redundancy and artifacts, whereas our method better preserves both the static background and fast-moving object. A controlled comparison with repeatable indoor motion is provided in Appendix A.2. Besides, our framework also supports uniform-velocity scenes, as shown in Appendix A.3.

4.3 Component-wise Ablation Studies

Table 2: Component ablations under matched reconstruction and evaluation settings. Full results and variant definitions are given in Appendix B.1.

Configuration	PSNR ↑	SSIM ↑	LPIPS ↓
Naive Low+High fusion	10.30	0.48	0.52
Fixed uniform thresholds	15.45	0.60	0.39
Without layer grouping	17.16	0.78	0.25
Blockwise acquisition	18.82	0.87	0.15
Fixed temporal window	17.73	0.81	0.23
No temporal alignment	18.44	0.85	0.18
No duration normalization	16.71	0.75	0.29
Full method	**20.42**	**0.92**	**0.08**

To identify which stages contribute to the final reconstruction, we ablate threshold assignment, motion-layer grouping, acquisition scheduling, temporal support, and fusion under matched settings

(Table 2). Naively merging low- and high-threshold events yields 10.30 dB, whereas using fixed, scene-independent multiple thresholds improves this to 15.45 dB but remains below adaptive acquisition (20.42 dB). This distinction shows that the benefit is not explained by access to multiple thresholds alone. Removing grouping drops PSNR to 17.16 dB, consistent with duplicated contours and imbalanced responses across layers. Replacing periodic acquisition with a single block per threshold (18.82 dB), or using one temporal window (17.73 dB), degrades the recovery of heterogeneous motion. Finally, omitting temporal alignment (18.44 dB) or duration normalization (16.71 dB) reduces quality, indicating that both temporal consistency and comparable event rates matter. The full method achieves the highest PSNR/SSIM and lowest LPIPS among these variants. Appendix B.1 reports all settings, including assignment-error tests.

4.4 Oracle Fixed-Threshold Baseline

To provide a more comprehensive comparison with fixed-threshold baselines, rather than considering only two predefined low and high settings, we evaluate ten fixed sensor settings, $C \in \{-35, -30, -20, -10, 0, 10, 20, 30, 40, 50\}$, on the same controlled multi-velocity scene with the same reconstruction network and evaluation protocol. We retrospectively choose the best fixed setting, giving this baseline oracle access to the test-scene scores. The optimum is $C = -30$, with 10.42 dB PSNR, 0.52 SSIM, and 0.30 LPIPS; our scheduled acquisition obtains 20.42 dB, 0.92, and 0.08, respectively. Thus, the PSNR advantage remains 10.00 dB even against the best tested single threshold. Lowering the fixed setting beyond -30 does not improve reconstruction in this scene, while higher settings progressively suppress weak or slow responses. This supports a *scene-specific* noise–information trade-off that no tested fixed global threshold resolves. The complete sweep is in Appendix B.2.

4.5 Hardware Impact of Redundant Events

Reducing event volume matters not only for storage or computation: excessive activity can exhaust finite sensor readout bandwidth and remove informative events. To test this mechanism, we record the same rapidly rotating fan in a controlled display-based setup at a fixed bias setting $C = -30$, using the same reconstruction pipeline. With low-amplitude background activity suppressed, the measured event rate remains below saturation and fan-region PSNR reaches 17.33 dB. When additional background activity drives readout into saturation, row-aligned gaps appear in the recorded events, blade texture is lost, and PSNR falls to 9.12 dB (an 8.21 dB decrease). By controlling background event activity while keeping the fan motion and sensor threshold fixed, we demonstrate that background-induced readout saturation leads to event loss in the fast-moving fan region, directly revealing the impact of background load on high-speed motion capture. Appendix B.3 provides the experimental description.

5 Conclusion

We presented a multi-threshold event imaging model for multi-velocity-scale imaging, where static, slow, and fast components are sensed with motion-compatible thresholds instead of a single global configuration. By estimating threshold responses across motion regimes and integrating events with velocity-adaptive temporal windows, our method preserves heterogeneous scene components in a unified representation. We further introduced a scheduled acquisition pipeline that realizes the model on a standard commercial event camera without requiring new hardware. Experiments on synthetic and real scenes demonstrate improved preservation of both static structures and fast dynamics with reduced redundant event generation.

Limitation. Our current implementation only approximates pixel- or region-wise control. Abrupt speed changes, occlusions, or new objects can invalidate motion-layer schedules within a cycle and cause transient artifacts. Low-threshold calibration errors, switching/settling delays, and finite readout bandwidth constrain performance. Since thresholds are switched globally and sequentially, the method assumes stable motion-layer structure across acquisition cycles and may be less suitable for abrupt or short-lived dynamics. Faster updates and spatially programmable thresholds are future directions (Appendix B.4–B.6).

Acknowledgments and Disclosure of Funding

This work was supported by the National Key R&D Program of China (Grant No. 2025YFA1016700), the Beijing Natural Science Foundation (Grant No. L233024), the Beijing High Innovation Plan (Contract No. 202604841443), JST CRONOS (JPMJCS25K6), and the JST-Mirai Program (JPMJMI23G1), Japan.

References

[1] Hilary Buxton. Learning and understanding dynamic scene activity: A review. *Image and Vision Computing*, 21(1):125–136, 2003.

[2] Konstantinos G Derpanis, Matthieu Lecce, Kostas Daniilidis, and Richard P Wildes. Dynamic scene understanding: The role of orientation features in space and time in scene classification. In *Proc. of Computer Vision and Pattern Recognition*, pages 1306–1313, 2012.

[3] Tae Hyun Kim, Byeongjoo Ahn, and Kyoung Mu Lee. Dynamic scene deblurring. In *Proc. of International Conference on Computer Vision*, pages 3160–3167, 2013.

[4] Nima Khademi Kalantari and Ravi Ramamoorthi. Deep high dynamic range imaging of dynamic scenes. *ACM Transactions on Graphics*, 36(4):144:1–144:12, 2017.

[5] Cenek Albl, Zuzana Kukelova, Viktor Larsson, Michal Polic, Tomas Pajdla, and Konrad Schindler. From two rolling shutters to one global shutter. In *Proc. of Computer Vision and Pattern Recognition*, pages 2505–2513, 2020.

[6] Bryan Peterson. *Understanding Exposure: How to Shoot Great Photographs with Any Camera*. 2016.

[7] Bharatesh Chakravarthi, Aayush Atul Verma, Kostas Daniilidis, Cornelia Fermuller, and Yezhou Yang. Recent event camera innovations: A survey. In *Computer Vision – ECCV 2024 Workshops*, volume 15646 of *Lecture Notes in Computer Science*, pages 342–376. Springer, 2025.

[8] Patrick Lichtsteiner, Christoph Posch, and Tobi Delbruck. A 128×128 120 dB 15 μs latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-State Circuits*, 43(2), 2008.

[9] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J Davison, Jörg Conradt, Kostas Daniilidis, et al. Event-based vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1):154–180, 2022.

[10] Qiyao Gao, Xiaoyang Sun, Zhitao Yu, and Xu Chen. Understanding and controlling the sensitivity of event cameras in responding to static objects. In *Proc. of IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 783–786, 2023.

[11] Ziwei Wang, Yonhon Ng, Pieter van Goor, and Robert Mahony. Event camera calibration of per-pixel biased contrast threshold. *arXiv preprint arXiv:2012.09378*, 2020.

[12] Henri Rebecq, Daniel Gehrig, and Davide Scaramuzza. ESIM: An open event camera simulator. In *Conference on Robot Learning*, pages 969–982, 2018.

[13] Ruiming Cao, Dekel Galor, Amit Kohli, Jacob L Yates, and Laura Waller. Noise2Image: Noise-enabled static scene recovery for event cameras. *Optica*, 12(1):46–55, 2025.

[14] Brian McReynolds, Rui Graca, and Tobi Delbruck. Experimental methods to predict dynamic vision sensor event camera performance. *Optical Engineering*, 61(7):074103–074103, 2022.

[15] Abbas El Gamal and Helmy Eltoukhy. CMOS image sensors. *IEEE Circuits and Devices Magazine*, 21(3):6–20, 2005.

[16] Ch Hu-Guo, J Baudot, G Bertolone, A Besson, AS Brogna, C Colledani, G Claus, R De Masi, Y Degerli, A Dorokhov, et al. CMOS pixel sensor development: A fast read-out architecture with integrated zero suppression. *Journal of Instrumentation*, 4(04):P04012, 2009.

[17] Zhisheng Lu, Juncheng Li, Hong Liu, Chaoyan Huang, Linlin Zhang, and Tieyong Zeng. Transformer for single image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 457–466, 2022.

[18] Sicheng Gao, Xuhui Liu, Bohan Zeng, Sheng Xu, Yanjing Li, Xiaoyan Luo, Jianzhuang Liu, Xiantong Zhen, and Baochang Zhang. Implicit diffusion models for continuous super-resolution. In *Proc. of Computer Vision and Pattern Recognition*, pages 10021–10030, 2023.

[19] Xin Huang, Qi Zhang, Ying Feng, Hongdong Li, Xuan Wang, and Qing Wang. HDR-NeRF: High dynamic range neural radiance fields. In *Proc. of Computer Vision and Pattern Recognition*, pages 18398–18408, 2022.

[20] Minggui Teng, Chu Zhou, Hanyue Lou, and Boxin Shi. NEST: Neural event stack for event-based image enhancement. In *Proc. of European Conference on Computer Vision*, pages 660–676, 2022.

[21] Cheng Zhang, Shaolin Su, Yu Zhu, Qingsen Yan, Jinqiu Sun, and Yanning Zhang. Exploring and evaluating image restoration potential in dynamic scenes. In *Proc. of Computer Vision and Pattern Recognition*, pages 2067–2076, 2022.

[22] Zhihang Zhong, Xiao Sun, Zhirong Wu, Yinqiang Zheng, Stephen Lin, and Imari Sato. Animation from blur: Multi-modal blur decomposition with motion guidance. In *Proc. of European Conference on Computer Vision*, pages 599–615, 2022.

[23] Chen Haoyu, Teng Minggui, Shi Boxin, Wang YIzhou, and Huang Tiejun. Learning to deblur and generate high frame rate video with an event camera. *arXiv preprint arXiv:2003.00847*, 2020.

[24] Xiang Zhang and Lei Yu. Unifying motion deblurring and frame interpolation with events. In *Proc. of Computer Vision and Pattern Recognition*, pages 17765–17774, 2022.

[25] Timo Stoffregen, Cedric Scheerlinck, Davide Scaramuzza, Tom Drummond, Nick Barnes, Lindsay Kleeman, and Robert Mahony. Reducing the sim-to-real gap for event cameras. In *Proc. of European Conference on Computer Vision*, 2020.

[26] Yi Ma, Peiqi Duan, Yuchen Hong, Chu Zhou, Yu Zhang, Jimmy S. Ren, and Boxin Shi. Color4E: Event demosaicing for full-color event guided image deblurring. In *Proc. of ACM MM*, 2024.

[27] Chu Zhou, Minggui Teng, Jin Han, Jinxiu Liang, Chao Xu, Gang Cao, and Boxin Shi. Deblurring low-light images with events. *International Journal of Computer Vision*, 131(5):1284–1298, 2023.

[28] Peiqi Duan, Boyu Li, Yixin Yang, Hanyue Lou, Minggui Teng, Xinyu Zhou, Yi Ma, and Boxin Shi. Eventaid: Benchmarking event-aided image/video enhancement algorithms with real-captured hybrid dataset. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(8):6959–6973, 2025.

[29] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. High speed and high dynamic range video with an event camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(6):1964–1980, 2021.

[30] Cedric Scheerlinck, Henri Rebecq, Daniel Gehrig, Nick Barnes, Robert Mahony, and Davide Scaramuzza. Fast image reconstruction with an event camera. In *Proc. of Winter Conference on Applications of Computer Vision*, pages 156–163, 2020.

[31] Lin Wang, S. Mohammad Mostafavi I., Yo-Sung Ho, and Kuk-Jin Yoon. Event-based high dynamic range image and very high frame rate video generation using conditional generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10081–10090, 2019.

[32] Jin Han, Chu Zhou, Peiqi Duan, Yehui Tang, Chang Xu, Chao Xu, Tiejun Huang, and Boxin Shi. Neuromorphic camera guided HDR imaging. In *Proc. of Computer Vision and Pattern Recognition*, pages 1730–1739, 2020.

[33] Jin Han, Yixin Yang, Peiqi Duan, Chu Zhou, Lei Ma, Chao Xu, Tiejun Huang, Imari Sato, and Boxin Shi. Hybrid HDR imaging fusing neuromorphic and conventional images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(7):8553–8565, 2023.

[34] Liyuan Pan, Richard Hartley, Cedric Scheerlinck, Miaomiao Liu, Xin Yu, and Yuchao Dai. High frame rate video reconstruction based on an event camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(5):2519–2533, 2022.

[35] Jin Han, Yixin Yang, Chu Zhou, Chao Xu, and Boxin Shi. EvIntSR-Net: Event guided multiple latent frames reconstruction and super-resolution. In *Proc. of International Conference on Computer Vision*, pages 4882–4891, 2021.

[36] Qiyao Gao, Peiqi Duan, Hanyue Lou, Minggui Teng, Ziqi Cai, Xu Chen, and Boxin Shi. Unified reconstruction of static and dynamic scenes from events. In *Proc. of Computer Vision and Pattern Recognition*, 2025.

[37] Tobias Fischer and Michael Milford. How many events do you need? event-based visual place recognition using sparse but varying pixels. *IEEE Robotics and Automation Letters*, 7(4):12275–12282, 2022.

[38] Alexander Kugele, Thomas Pfeil, Michael Pfeiffer, and Elisabetta Chicca. How many events make an object? improving single-frame object detection on the 1 mpx dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 3913–3922, 2023.

[39] Urbano Miguel Nunes, Ryad Benosman, and Sio-Hoi Ieng. Adaptive global decay process for event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9771–9780, 2023.

[40] Damien Joubert, Hubert Konik, and Frederic Chausse. Convolutional neural network for detection and classification with event-based data. In *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP)*, 2019.

[41] Prophesee. Event signal processing: Event trail filter (stc/trail). Metavision SDK Documentation. Accessed 2026.

[42] Shasha Guo and Tobi Delbruck. Low cost and latency event camera background activity denoising. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):785–795, 2023.

[43] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Proc. of International Conference on Medical Image Computing and Computer Assisted Intervention*, pages 234–241, 2015.

[44] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Proc. of Advances in Neural Information Processing Systems*, 28, 2015.

[45] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

A Additional Experiments

A.1 Event Response Calibration for Multi-Velocity Scenes

The main paper uses scene-adaptive threshold estimation to assign motion-compatible operating thresholds. Here we provide the full calibration study that supports this design choice. The goal is to characterize how the event response of a Prophesee EVK4 sensor varies with illumination, scene appearance, motion speed, and contrast threshold. This calibration is necessary because an optimal threshold should preserve the complete scene structure while minimizing redundant events, and this optimum changes across static, slow, and fast components.

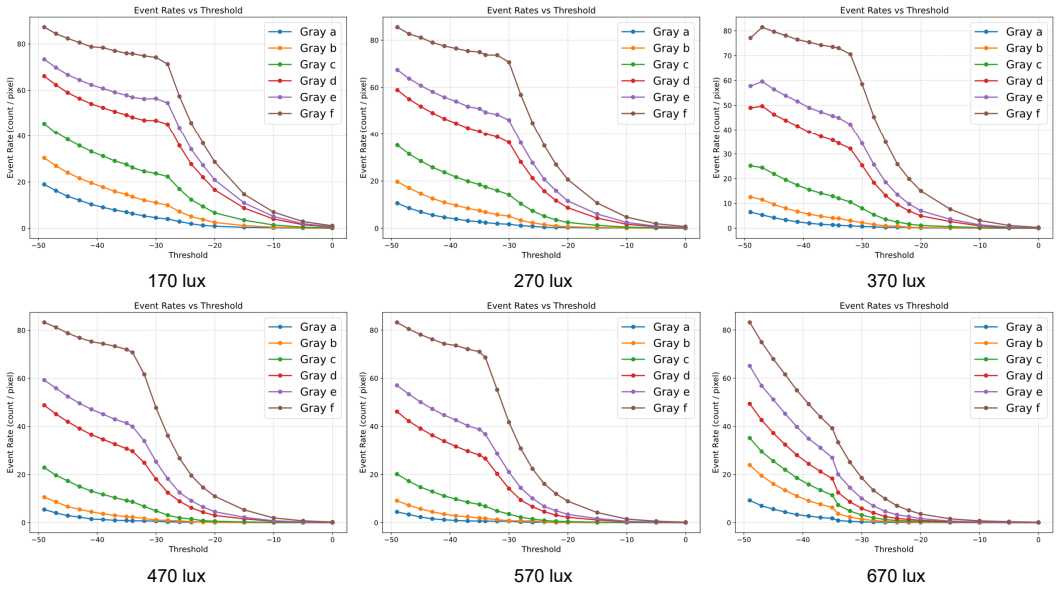


Figure 7: Static pixel responses of a Prophesee EVK4 sensor under different illumination levels (170–670 lux). The six printed grayscale patches (a–f) correspond to 8-bit grayscale values of 255, 204, 153, 102, 51, and 0, respectively.

We measure the event triggering rate, defined as the number of events per pixel within a unit time interval, under illumination levels from 170 to 670 Lux and motion speeds ranging from static scenes to 1000 pixels/s. Following the different triggering behaviors of static and dynamic events, we calibrate static responses mainly under negative threshold settings and dynamic responses under positive threshold settings. The complete calibration results are provided in both the static and dynamic response plots.

The static calibration (Fig. 7) shows that lowering the threshold monotonically increases the event rate across grayscale levels and improves the visibility of weak static structures. However, overly low thresholds produce excessive responses, which can dominate the normalized representation and suppress mid-level grayscale details. Thus, static imaging requires a threshold that is sufficiently low to activate weak scene structures but not so low that the response becomes noise- or saturation-dominated.

The dynamic calibration (Fig. 8) shows that, for a fixed motion speed, the event rate decreases as the threshold increases, and darker regions are more easily suppressed under high thresholds. For a fixed grayscale level, low-speed motion is more sensitive to threshold increase than medium- and high-speed motion, since fewer events can be accumulated within the same temporal support. These observations confirm that different motion regimes require different operating thresholds: slow or weakly changing components benefit from lower thresholds to maintain sufficient event support, whereas fast components can tolerate higher thresholds to suppress redundancy. This full calibration study supports the scene-adaptive threshold estimation and scheduled multi-threshold acquisition strategy used in the main paper.

A.2 Controlled Indoor Real-Data Evaluation

To complement the outdoor real-data comparison in the main paper, we further construct a controlled indoor multi-velocity scene with repeatable motion. The scene contains a static background, a slowly translating foreground component, and a rapidly rotating fan, providing simultaneous static, slow-motion, and fast-motion contents under real event-camera acquisition. Unlike the outdoor sequence, where manual motion cannot be exactly repeated, this setup allows us to reproduce similar motion patterns across different acquisition protocols and enables a more controlled qualitative comparison.

We record the scene using a Prophesee EVK4 event camera under three protocols: a fixed low threshold ($C = -35$), a fixed high threshold ($C = 50$), and the proposed scheduled multi-threshold

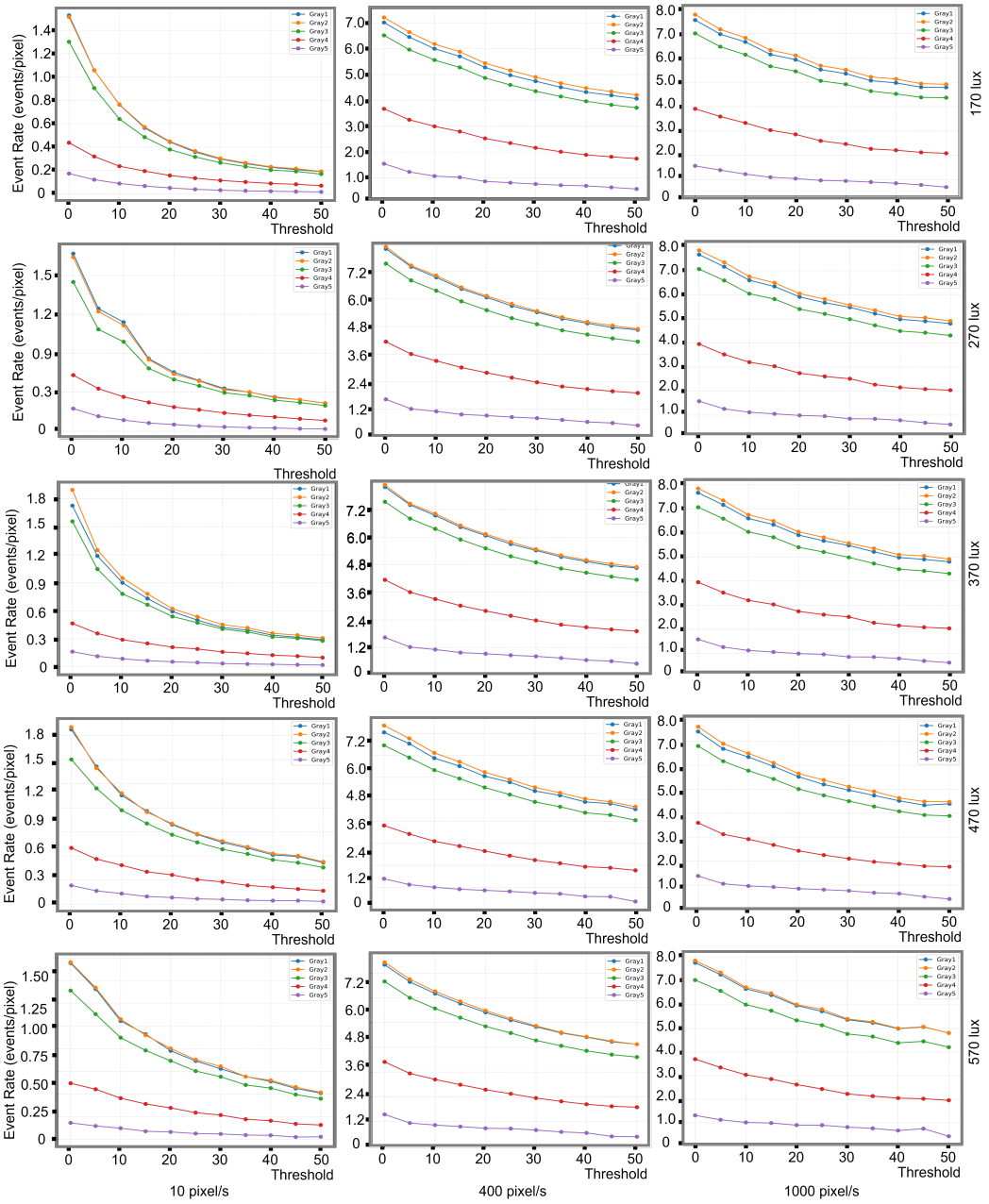


Figure 8: Dynamic pixel response of a Prophesee EVK4 sensor across a illumination range (170–570 Lux) and motion speeds from 10 pixel/s to 1000 pixel/s.

acquisition. For the fixed-threshold streams, we reconstruct videos using E2VID [29] and URSEE [36]. All methods are evaluated at the same output interval.

As shown in Fig. 9, fixed-threshold sensing exhibits clear limitations in this controlled multi-velocity setting. E2VID struggles to recover static structures under both low and high thresholds, since purely dynamic reconstruction cannot reliably exploit weak static responses. URSEE recovers the static background from the low-threshold stream, but the same low threshold also introduces redundant events and artifacts in fast-motion regions. When using the high-threshold stream, redundant triggering is suppressed, but static and slow components become incomplete or missing. These results confirm that neither a single low threshold nor a single high threshold can simultaneously satisfy the sensing requirements of static, slow, and fast components.

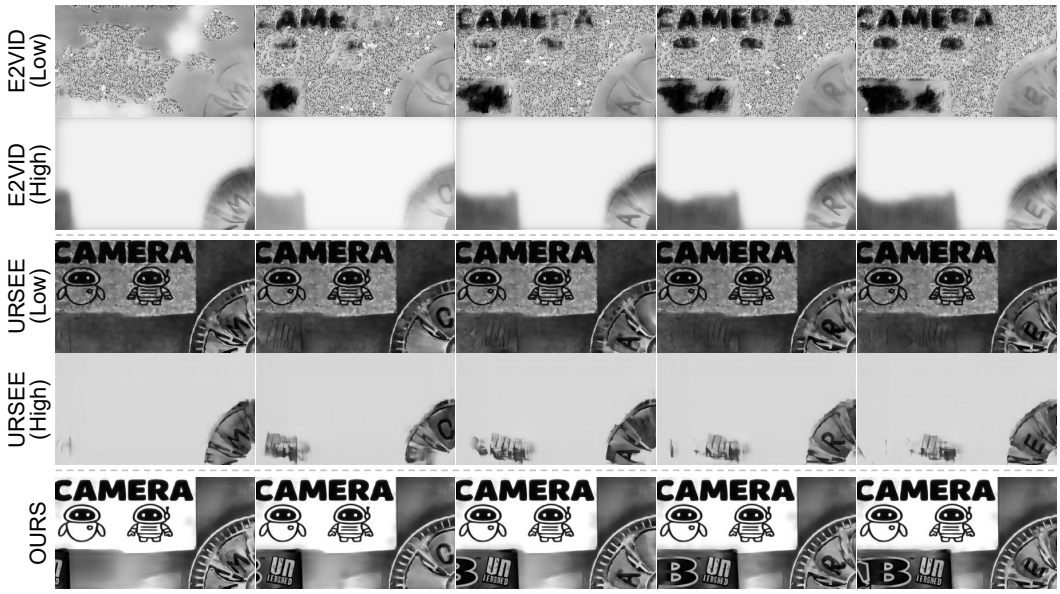


Figure 9: Controlled indoor real-data comparison on a repeatable multi-velocity scene containing a static background, slow translation, and fast rotation. The scene is captured using a fixed low threshold ($C = -35$), a fixed high threshold ($C = 50$), and the proposed scheduled multi-threshold acquisition. Rows show E2VID [29] and URSEE [36] reconstructed from the low- and high-threshold streams, followed by our method. The controlled motion enables a more consistent comparison across acquisition protocols. Our method better preserves static, slow, and fast components within a single reconstruction.

In contrast, our scheduled multi-threshold acquisition assigns motion-compatible thresholds and temporal windows to different scene components. The resulting reconstruction preserves the static background, maintains the slowly moving structure, and avoids severe overlap in the fast rotating fan. This controlled real-data experiment further verifies that the proposed acquisition strategy improves multi-velocity-scale reconstruction beyond the motion variations present in the outdoor qualitative results.

A.3 Evaluation on Uniform-Velocity Scenes

Although the proposed method is designed for multi-velocity-scale scenes, it is not restricted to cases where different scene components move at different speeds. When a scene contains a dominant motion scale, such as handheld camera motion or uniform scene translation, the proposed framework naturally reduces to a single-scale configuration: the motion-layer estimation assigns similar temporal supports to most components, and the fusion step behaves as normalized velocity-adaptive integration. This setting allows us to verify that the proposed method does not rely on artificially separated motion layers and remains applicable to standard event-camera scenarios.

We first present qualitative results on two selected scenes from the public M3ED dataset, where camera ego-motion induces approximately coherent image-plane motion. As shown in Fig. 10, our method produces stable reconstructions across different real-world scenes, preserving both large-scale structures and fine scene details under natural event-camera motion. These results indicate that the proposed representation remains effective beyond explicitly constructed multi-velocity scenes.

We further construct a controlled indoor uniform-velocity scene, where both foreground and background patterns move at the same speed, $v = 50$ pixels/s. This setting removes the advantage of assigning different thresholds to different velocity scales and directly tests whether the method can handle a single dominant motion regime. As shown in Fig. 11, our method consistently reconstructs both foreground and background structures across the sequence. Together, these results show that the proposed multi-threshold imaging framework generalizes to conventional uniform-velocity event sensing, while retaining its advantage in more challenging multi-velocity-scale scenes.



Figure 10: Offline reconstruction evaluation on uniform-velocity scenes from the public M3ED dataset.. The sequences are dominated by camera ego-motion and exhibit approximately coherent image-plane motion. Our method produces stable reconstructions across different real-world scenes, showing that the proposed framework remains applicable beyond explicitly constructed multi-velocity settings.



Figure 11: Controlled indoor evaluation on a uniform-velocity scene. Both foreground and background patterns move at the same speed, $v = 50$ pixels/s. The results show that our method can reconstruct scenes with a single dominant motion scale and does not rely on velocity separation between foreground and background.

B Additional Ablations and Robustness Analyses

B.1 Complete Component-wise Ablations

We compare ablated configurations under the same reconstruction and evaluation settings as the full pipeline. The tested variants respectively remove or modify scene-adaptive threshold selection, layer grouping, acquisition scheduling, temporal-window assignment, and the two fusion operations. Table 3 reports the complete numerical comparison.

Table 3: Complete component-wise ablations in the controlled multi-velocity setting. For PSNR and SSIM higher is better; for LPIPS lower is better.

Configuration	PSNR ↑	SSIM ↑	LPIPS ↓
Naive Low+High fusion	10.30	0.48	0.52
Fixed uniform thresholds	15.45	0.60	0.39
Without layer grouping	17.16	0.78	0.25
Assignment error: 20%	19.59	0.90	0.11
Assignment error: 40%	18.55	0.86	0.16
Assignment error: 60%	18.02	0.83	0.19
Blockwise acquisition	18.82	0.87	0.15
Fixed temporal window	17.73	0.81	0.23
Normalize, no alignment	18.44	0.85	0.18
Alignment, no normalization	16.71	0.75	0.29
No normalization or alignment	16.02	0.70	0.35
Full method	**20.42**	**0.92**	**0.08**

Threshold selection and layer grouping. Naive fusion of low- and high-threshold streams retains noise and overlapping responses, whereas a fixed scene-independent set of multiple thresholds remains less compatible with scene-specific motion and contrast. Removing motion-layer grouping can duplicate responses along shared physical edges and cause density imbalance. To assess the impact of layer-assignment errors, we randomly select 20%, 40%, or 60% of pixels and deliberately reassign them to different motion layers, while keeping the remaining assignments unchanged.

Acquisition schedule and temporal windows. Blockwise acquisition uses each threshold in one uninterrupted segment rather than revisiting thresholds periodically; slow responses may be missed while other thresholds are active. A single fixed temporal window cannot simultaneously ensure adequate support for slow structures and limited temporal overlap for fast motion.

Temporal fusion. Without temporal alignment, threshold-specific streams retain mismatched timestamp origins; without duration normalization, event counts accumulated over different windows remain biased toward longer observation periods. Removing both operations produces the largest loss among these fusion variants. These ablations establish component contributions in the tested controlled setting, not theoretical global optimality.

B.2 Fixed-Threshold Sweep with an Oracle Selection

To rule out an unfavorable choice of fixed-threshold baseline, we evaluated ten settings on the same real multi-velocity scene using the same reconstruction network and evaluation protocol. For a conservative comparison, we report the best-performing fixed setting among the ten candidates, selected based on its performance on the evaluated scene. Table 4 reports the results for all ten settings.

Table 4: Fixed-threshold sweep on our local test dataset. The best single threshold ($C = -30$) was selected after evaluation.

Sensor setting	PSNR ↑	SSIM ↑	LPIPS ↓
Fixed -35	10.38	0.51	0.33
Fixed -30 (best)	**10.42**	**0.52**	**0.30**
Fixed -20	10.18	0.49	0.31
Fixed -10	10.04	0.46	0.43
Fixed 0	9.94	0.44	0.49
Fixed $+10$	9.83	0.43	0.50
Fixed $+20$	9.78	0.42	0.51
Fixed $+30$	9.70	0.42	0.52
Fixed $+40$	9.69	0.42	0.52
Fixed $+50$	9.64	0.41	0.53
Full method	**20.42**	**0.92**	**0.08**

Lowering the fixed threshold beyond -30 increases event volume without improving reconstruction quality in this scene; increasing the threshold progressively removes weak and slow-moving structure. The full method improves PSNR by 10.00 dB over the oracle single-threshold baseline, within the scope of the tested scene and protocol.

B.3 Readout Saturation from Redundant Background Events

We rendered two Blender videos featuring identical rapidly rotating fans but different static backgrounds: one blank and the other richly textured. Both videos were played on a monitor and recorded using a Prophesee EVK4 at a fixed threshold of $C = -30$, with the same reconstruction pipeline. The blank-background recording generated little background activity, preserving a complete fan contour in the temporally integrated event frames. In contrast, the textured-background recording exhibited substantially higher background event activity, accompanied by row-aligned event losses and incomplete fan structures, consistent with readout saturation.

Table 5: Fan-region reconstruction under the two controlled readout conditions at $C = -30$.

Readout condition	Fan-region PSNR ↑
No observed saturation	17.33 dB
Background-activity-induced saturation	9.12 dB

The 8.21 dB PSNR decrease, together with the row-aligned event losses, suggests that excessive background activity can exhaust the finite sensor readout bandwidth and degrade reconstruction quality. This observation is specific to the tested hardware configuration.

B.4 Sensitivity to Motion Layers and Switching Frequency

Number of motion layers. For the same controlled eight-object scene, we varied the number of motion layers from $M = 2$ to $M = 6$ while keeping other settings unchanged. Across that range, the largest reported relative variations in PSNR, SSIM, and LPIPS were 0.51%, 1.06%, and 1.18%, respectively. Thus, the results show limited variation within the tested scene and range, but we do not claim general insensitivity to M.

Switching frequency. The default ($1\times$) schedule cycles through four thresholds $[-30, 0, 15, 50]$ with respective dwell times of $[100, 50, 20, 10]$ ms, yielding a measured switching rate of 21.54 Hz (5.39 full cycles/s). This setting achieves 9.38 dB PSNR, 0.404 SSIM, and 0.530 LPIPS with 197,823 events. We tested $0.5\times$, $1\times$, $2\times$, and $4\times$ the default switching frequency. Table 6 reports the relative changes at $0.5\times$ and $4\times$. Faster switching can improve reconstruction quality at the cost of increased event volume and acquisition-bandwidth demand.

Table 6: Sensitivity to threshold-switching frequency, reported as relative changes from the default $1\times$ setting. A dash indicates an unavailable measurement.

Frequency	ΔPSNR	ΔSSIM	ΔLPIPS	Δevents
$0.5\times$	-0.23%	-0.27%	$+0.31\%$	–
$4\times$	$+8.40\%$	$+14.92\%$	-21.69%	$+120.18\%$

B.5 Sensitivity to Threshold Estimation and Layer Assignment

For low-threshold settings ($C < -30$), a ± 5-unit sensor-bias perturbation can change PSNR by more than 15% in the tested conditions. In the intermediate and high-threshold regime ($C > -10$), a five-unit perturbation changed PSNR/SSIM by less than 2.7%. A -10-unit bias perturbation increased event count by approximately 120%, whereas a $+10$-unit perturbation reduced it by 14.91%. These measurements indicate that calibration error can affect acquisition efficiency particularly strongly in sensitive low-threshold regimes.

We additionally test random motion-layer assignment errors of 20%, 40%, and 60%; the corresponding reconstruction scores are reported in Table 3. Their progressive degradation does not establish robustness against systematic layer-selection failures or spatially correlated errors.

B.6 Crowded Scenes and Abrupt Motion: Scope of the Tests

In an initial controlled stress test, the number of simultaneously moving objects increased from two to eight, with a reported 224.23% increase in event count. Object count is not interchangeable with the number of distinct velocity layers M. Practical scalability also depends on calibration cost, cycle duration, cross-cycle association, and readout bandwidth.

In qualitative real-data trials with rapid object insertion/removal and camera shaking, reconstruction fidelity and event efficiency temporarily declined, although strong motion responses generally remained visible. These observations do not demonstrate robustness to arbitrary nonstationary motion. The current schedule assumes approximately stable dominant motion layers within an acquisition cycle; abrupt scene changes may introduce re-estimation delay, threshold-settling effects, and incorrect associations.

B.7 Acquisition and Reconstruction Protocol

The capture pipeline first records a short default-bias event stream, estimates coarse flow magnitudes, and identifies dominant speed modes with spatial support. The retained modes determine the number of layers. Threshold-specific segments are obtained from the logged global bias schedule and approximately associated across adjacent cycles. Local timestamp coordinates are shifted to a common reconstruction reference, and event representations are normalized by their integration durations before network input. These operations depend on the locally stable motion approximation; abrupt changes can cause association errors.

Algorithm 1 Conceptual scheduled acquisition and reconstruction

Require: Event camera with programmable global bias; candidate bias ranges; output times
Record short stream at default bias; estimate coarse event flow
Group spatially supported flow-magnitude modes; treat near-zero mode as static
Set M to the number of retained modes
for $m = 1, \ldots, M$ **do**
    Choose a finite window Δt_m compatible with motion scale
    Calibrate/select bias C_m by the empirical response criteria
end for
$\mathcal{S} \leftarrow \{(C_m, \Delta t_m)\}_{m=1}^{M}$
while acquisition continues **do**
    for each $(C_m, \Delta t_m) \in \mathcal{S}$ **do**
        Apply C_m globally and record its active timestamps and events
    end for
    Demultiplex the raw stream into threshold-specific temporal segments
    Construct binary activation masks A_m for each threshold
    Order masks by increasing effective threshold
    for $m = M - 1, \ldots, 1$ **do**
        $P_m \leftarrow A_m \setminus \bigcup_{j=m+1}^{M} A_j$
    end for
    $P_M \leftarrow A_M$
    Assign threshold-specific events to layers using masks P_m
    Associate responses approximately across adjacent cycles using estimated motion modes
    Align segment timestamps to the reconstruction reference; divide by durations
    Construct static event frame and motion voxel grid; reconstruct output frame
end while

The reconstruction network takes two inputs: an accumulated event frame for static or quasi-static structures and a fixed-bin voxel grid for motion-induced events. Training uses the publicly available E-StaDyn dataset [36], following a similar preprocessing pipeline. E2VID uses its official pretrained checkpoint without fine-tuning, while URSEE and our network share the same training dataset and protocol. All fixed-threshold and scheduled acquisitions are evaluated at the same output interval.

Construction of the Blender test set. Inspired by E-StaDyn [36], we independently constructed 15 Blender scenes, each comprising a static, richly textured background and multiple moving foreground objects. Unlike E-StaDyn, which contains only one single moving foreground per scene, our test set introduces two or three independently objects moving at different speeds to simulate multi-velocity-scale motion. The Blender-rendered videos were played on a display and captured using a Prophesee EVK4 event camera under different threshold protocols, with display-flicker interference excluded through experimental checks. After acquisition, the event streams were segmented using the playback start and end timestamps, and their temporal alignment with the corresponding Blender-rendered ground-truth frames was verified.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The Abstract and Introduction section clearly show our contribution ans scope (Event based Multi-Velocity-Scale Imaging).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please see the Conclusion section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: This is not a theoretical paper (i.e., we do not proof something mathematically).

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the implementation details in section 3.4 and 3.5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: During the peer review period, we do not release the code and data.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide the dataset details in section 4.1, and provide the comparison details in section 4.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We report both quantitative and qualitative results but do not provide error bars, confidence intervals, or formal statistical significance tests.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: Our network was trained on a single NVIDIA GeForce RTX 4090 GPU. Detailed memory usage and execution times are not reported.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and confirm that our research complies with its requirements.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: There is no societal impact of the work performed.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have cite the used dataset in sectin 3.5, and the compared methods in section 4.1.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.