Event-LiDAR: 3D Eventification for Efficient Point Cloud Processing

Masatoshi Murakami[1†], Eisho Tsuji[1†], and Ken Sakurada[1,2]

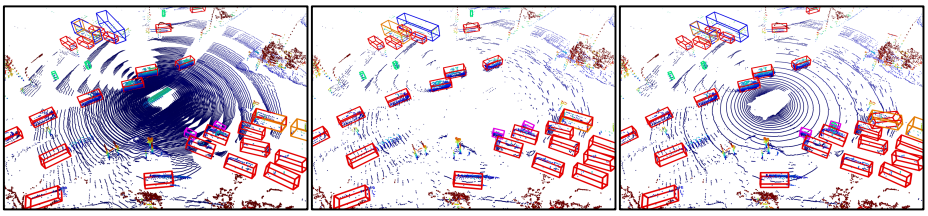
[1] Graduate School of Informatics, Kyoto University, Kyoto, Japan
[2] D3 Center, The University of Osaka, Osaka, Japan
[†]Equal contribution.

Abstract. We propose Event-LiDAR[*], a 3D eventification framework that converts conventional LiDAR scans into temporally sparse representations for efficient point cloud processing. Unlike dense multi-scan processing, naive frame differencing, or correspondence-based residuals, 3D event extraction is formulated as a temporal estimation problem under sparse, viewpoint-dependent observations. Short-term geometric evolution across consecutive scans is approximated with a first-order geometric model, and deviations unexplained by this model are treated as events. This yields compact, information-preserving representations that suppress redundancy while retaining changes unpredictable by the first-order model. Event-LiDAR is applied to LiDAR-based 3D object detection with existing backbones, and an event-aware network design is further introduced to reallocate modeling capacity toward the input stage for sparse inputs. On nuScenes, under the 1F+9T setting with a 72% point reduction, Event-LiDAR maintains accuracy comparable to full-scan baselines while achieving 23% faster end-to-end inference on PTv3, driven in part by a 32% speedup in its feature extraction backbone. The gains generalize across backbones, with a 9% end-to-end speedup on CenterPoint under the same setting. Training is likewise accelerated by 17% under the same setting. Event-LiDAR thus serves as a low-latency, architecture-agnostic geometric preprocessor inspired by event-based sensing, providing a practical front-end for efficient 3D point cloud perception.

1 Introduction

Modern 3D perception systems continuously acquire high-resolution LiDAR data to capture dynamic environments. While dense scans provide rich geometric information, they also contain substantial temporal redundancy: large portions of consecutive frames remain geometrically consistent, yet the full point cloud is repeatedly processed, transmitted, and stored. As sensing frequency and resolution increase, this redundancy amplifies computational load and inflates communication and storage costs in practical deployments.

[*] The source code will be made publicly available at `https://github.com/scsrp/event-lidar`



(a) 10F (b) 10T (c) 1F+9T

Fig. 1: Overview of Event-LiDAR. F denotes full-scan input without eventification, and T denotes temporally eventified scans. The framework converts dense LiDAR scans (10F) into sparse event representations by extracting geometric deviations unexplained by a local first-order geometric model. The eventified representations (10T and 1F+9T) emphasize structural changes unpredictable by the model while effectively suppressing redundancy.

A natural remedy is to represent only geometrically meaningful temporal variation. This change-based principle has long been exploited in video compression (e.g., MPEG), where sequences are coded via prediction and residuals rather than storing near-identical frames, albeit at non-trivial encoding cost. More recently, related ideas have appeared in vision models via token pruning for video Transformers [8], and in event cameras [12] that encode intensity changes instead of full frames. However, these mechanisms rely on dense grids or hardware-level derivatives and do not directly extend to sparse, irregular 3D point clouds; moreover, many compression pipelines introduce additional encoding/decoding overhead that is undesirable for low-latency perception.

We propose *Event-LiDAR*, a low-latency geometric preprocessor inspired by event-based sensing that extends change-based representations to LiDAR through lightweight processing (Fig. 1). In LiDAR, naive differencing is unreliable because point clouds, though densely representable in the image domain, are sparse in 3D, viewpoint-dependent, and only partially observed at each frame; ego-motion continuously alters the visible set and point sampling varies across scans. Consequently, straightforward differencing, whether via voxel occupancy comparison, depth-grid projection and differentiation, or nearest-neighbor matching, often yields spurious activations that are dominated by discretization artifacts, visibility changes, and sparsity-induced mismatches rather than genuine structural variation.

To address this, we cast 3D event extraction as a temporal estimation problem under sparse and viewpoint-dependent observations. Short-term geometric evolution across consecutive scans is approximated with a first-order geometric model, and deviations unexplained by this model are treated as events. By suppressing variations attributable to ego-motion-induced viewpoint changes and sampling effects, Event-LiDAR retains structural changes unpredictable by a local first-order geometric model while removing redundant observations, resulting in compact event representations.

Event-LiDAR is broadly applicable and can be incorporated into diverse 3D perception pipelines. To demonstrate this, we apply the proposed eventification framework to 3D object detection and present one example event-aware network that reallocates modeling capacity toward the input stage. This shows that sparse event representations can be processed effectively without modifying the core backbone architecture. Extensive experiments on large-scale benchmarks confirm that Event-LiDAR preserves detection accuracy comparable to full-scan baselines while substantially reducing processed points and overall computational cost.

Our main contributions are summarized as follows:

- We propose Event-LiDAR, a low-latency, architecture-agnostic 3D eventification front-end that converts conventional LiDAR scans into temporally sparse representations for efficient point cloud processing.
- We cast 3D event extraction as a temporal estimation problem under sparse, viewpoint-dependent observations, enabling geometrically consistent extraction beyond naive differencing and correspondence-based residuals with millisecond-scale, label-free preprocessing.
- We apply Event-LiDAR to LiDAR-based 3D object detection with existing backbones, and further introduce an event-aware network design that reallocates modeling capacity toward the input stage for sparse inputs.
- We validate the framework on large-scale benchmarks, showing accuracy comparable to full-scan baselines while substantially reducing processed points and improving both inference and training efficiency.

2 Related Work

We first review event cameras, which motivate representing visual streams through changes rather than dense frames. We then discuss compression methods that exploit temporal redundancy (e.g., predictive coding in MPEG and point cloud compression) and clarify how their objectives and computational pipelines differ from low-latency eventification for perception. Next, we summarize LiDAR-based moving object segmentation approaches and discuss their differences from 3D eventification under sparse, viewpoint-dependent observations. Finally, we review efficient LiDAR-based 3D detection methods and position our approach among sparsity- and temporal-differencing-based acceleration strategies.

2.1 Event cameras

Event cameras are bio-inspired sensors that asynchronously report local brightness changes [5,12]. Each pixel emits a positive or negative event when the logarithmic intensity change exceeds a threshold, producing sparse streams that concentrate on edges and motion. This sensing paradigm enables microsecond temporal resolution and low-latency processing by avoiding redundant frame transmission.

Event-based vision has been applied to object detection [1, 14, 17, 36, 40], semantic segmentation [2,17,33], localization and mapping [22,28,34], and human

pose estimation [18, 20, 46]. While these works demonstrate robustness under fast motion and high dynamic range, they also require dedicated processing pipelines tailored to asynchronous event streams.

Our work is conceptually related to video-to-event conversion, such as VID2E [13] using the ESIM simulator [27]. Whereas VID2E generates events from thresholded pixel-wise intensity changes, we target 3D sensing and derive sparse 3D events from inter-scan geometric variation under ego-motion and viewpoint-dependent sampling.

2.2 Data compression

Reducing sequential data by exploiting temporal redundancy is central to predictive coding in video compression, including MPEG. For 3D data, MPEG Point Cloud Compression (PCC) standardizes codecs such as G-PCC for static point clouds and V-PCC for dynamic ones [16, 29]. These codecs typically transform point clouds into alternative representations (e.g., octrees or 2D projections) and then encode residual information [21, 44]. Recent neural methods further improve rate-distortion performance by learning compact latent representations [26, 31].

However, compression pipelines are optimized for rate-distortion efficiency and generally incur non-trivial encoding/decoding cost. Moreover, the resulting bitstreams or latent codes are not directly aligned with sensor coordinates for immediate geometric processing. A lightweight alternative is to compute depth differences between consecutive RGB-D frames [37], but this suffers from range-dependent discretization errors and remains primarily oriented toward compression rather than low-latency perception.

In contrast, Event-LiDAR performs low-latency eventification directly in sensor coordinates without encoding or domain transformation. Although it is not designed to match dedicated codecs in compression ratio, it is computationally lightweight and preserves geometric structure for direct use in downstream perception tasks.

2.3 LiDAR-based moving object segmentation

Moving object segmentation (MOS) aims to separate dynamic objects from the static environment by exploiting temporal geometric inconsistency. MOS is related to eventification in that both analyze inter-scan variation, but MOS assigns motion labels, whereas our goal is to generate sparse event representations for redundancy reduction.

Model-based MOS approaches evaluate geometric consistency between consecutive scans using estimated sensor poses and flag violations of the static-world assumption as moving [3, 7, 10, 15, 24, 25, 38, 43]. In practice, partial observations and viewpoint-dependent visibility changes create correspondence failures even in static scenes, leading to erroneous dynamic labels and redundant activations if directly used for event generation.

Learning-based MOS approaches include image-based and point-based formulations. Image-based methods compute frame-to-frame residual images from

projected depth maps [6], which suffer from range-dependent discretization errors, while recent point-based methods employ 4D sparse convolutions or BEV-based temporal aggregation [23, 35, 45]. While effective for segmentation, these approaches rely on learned feature extraction and are not intended as lightweight, training-free preprocessing.

In contrast, Event-LiDAR does not aim to classify motion. It casts event extraction as a temporal estimation problem under sparse and viewpoint-dependent observations and analytically computes geometric deviations unexplained by a local first-order geometric model, enabling low-latency and geometrically consistent sparsification even when correspondences are incomplete.

2.4 Efficient LiDAR-based 3D object detection

LiDAR-based 3D detection pipelines typically comprise point encoding, feature extraction, and detection. To reduce computation, prior work suppresses redundant activations through adaptive voxelization [41], density-aware feature extraction [19], or feature-importance-based voxel selection [30]. These approaches primarily operate in the feature domain and often introduce learnable modules.

FSD++ [11] reduces computation by constructing residual point sets across frames with ego-motion information, retaining points without close correspondences in the previous scan to lower processed points and voxel activations. However, correspondence absence does not necessarily indicate true structural change in LiDAR sensing. In practice, viewpoint-dependent visibility changes (e.g., newly observed regions) and sparsity-induced mismatches can also yield residual points even when the underlying geometry is static.

Event-LiDAR differs in formulation and objective. Rather than using correspondence absence as a proxy for change, we cast event extraction as a temporal estimation problem under sparse, viewpoint-dependent observations, identifying events as geometric deviations unexplained by a local first-order geometric model. This design suppresses variations caused by ego-motion-induced viewpoint changes and sampling effects, yields low-latency eventification in sensor coordinates, and serves as a lightweight, architecture-agnostic front-end applicable to diverse downstream tasks.

3 3D Eventification

Event-LiDAR converts dense LiDAR scans into temporally sparse *eventified* point clouds by extracting geometrically meaningful inter-scan variations with low latency. Rather than relying on naive frame differencing (e.g., voxel occupancy or point-to-point matching), we formulate 3D event extraction as a *temporal estimation problem* under sparse and viewpoint-dependent observations. Specifically, we approximate short-term geometric evolution using a local first-order geometric model built from the previous scan, and identify deviations from this model as events. This design suppresses predictable variations caused by ego-motion-induced viewpoint changes and sampling effects, while retaining

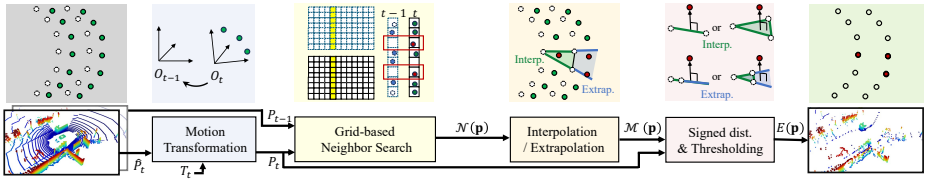


Fig. 2: Flowchart of temporal differentiation.

structural changes unpredictable by this model that are useful for downstream perception. Finally, we describe how eventified point clouds are integrated into an object detection pipeline as a representative application.

3.1 Temporal differentiation

Motivation and overview. A straightforward way to eventify range data is to compute differences between consecutive depth (range) images [6]. However, for LiDAR, coarse angular sampling and quantization yield discretization artifacts whose magnitude increases with range and can exceed the measurement precision, producing spurious activations. To obtain stable 3D events under sparse sampling, we estimate temporal variation in 3D by (i) compensating ego-motion, (ii) constructing a local first-order geometric model from the previous scan, and (iii) measuring signed deviations of current points from the predicted local geometry (Fig. 2).

Ego-motion compensation. Let $\hat{P}_t$ denote the current scan in its native sensor frame, and let $\hat{\mathbf{p}} \in \hat{P}_t$ be a point in it. Using a rigid transformation $T_t \in SE(3)$, we warp the scan into the coordinate frame of the previous scan:

$$P_t = T_t\,\hat{P}_t, \qquad \mathbf{p} = T_t\,\hat{\mathbf{p}}, \tag{1}$$

so that P_t is the current scan expressed in the coordinate frame of the previous scan. This backward warping enables a consistent comparison against the local geometric model built from P_{t-1} and helps reduce artifacts caused by viewpoint-dependent visibility changes. Because LiDAR scans are captured at short intervals (≤ 0.1 s), T_t can be obtained from wheel odometry, IMU, or standard temporal fusion; for articulated systems, it can be derived from kinematics.

Local first-order geometric model. For a warped current point $\mathbf{p} \in P_t$, we retrieve a neighborhood $\mathcal{N}(\mathbf{p}) \subset P_{t-1}$. To support low-latency queries, we index P_{t-1} using a grid structure constructed in the equirectangular projection domain and retrieve neighbors in constant time on average. From $\mathcal{N}(\mathbf{p})$, we build a locally first-order approximated geometry $\mathcal{M}(\mathbf{p}) \subset \mathbb{R}^3$, representing a locally linear/planar structure. The line and plane models are constructed directly from two and three neighbor points, respectively, avoiding fitting overhead; degenerate configurations fall back as plane→line→point. We then predict the local surface

implied by the previous scan as the orthogonal projection of $\mathbf{p}$ onto $\mathcal{M}(\mathbf{p})$,

$$\Pi(\mathbf{p}) = \arg\min_{\mathbf{x}\in\mathcal{M}(\mathbf{p})} \|\mathbf{p}-\mathbf{x}\|_2, \tag{2}$$

which is an interpolation when $\Pi(\mathbf{p})$ falls within the region spanned by its neighbors $\mathcal{N}(\mathbf{p})$, and an extrapolation otherwise.

Signed deviation. Using this projection, we compute the temporal geometric deviation as a signed distance:

$$\Delta d(\mathbf{p}) = \phi(\mathbf{p})\,\|\mathbf{p}-\Pi(\mathbf{p})\|_2, \tag{3}$$

where $\phi(\mathbf{p}) \in \{-1,+1\}$ is a sign-determining function based on sensor ranges, applied uniformly to both line and plane models: $\phi(\mathbf{p}) = \text{sign}(\|\mathbf{p}\|_2 - \|\Pi(\mathbf{p})\|_2)$.

Extrapolation for newly observed regions. A key difficulty in inter-scan processing is that viewpoint-dependent visibility changes create regions with few or no correspondences in P_{t-1}. In such cases, point-to-point residual criteria tend to label most points as changes, preventing effective redundancy reduction. Inspired by the goal of low-latency pruning, our method treats low-frequency structures (e.g., ground and walls) as predictable from the previous scan via a first-order model. When neighbor support is insufficient for interpolation, we extrapolate $\mathcal{M}(\mathbf{p})$ using available scanline/local context so that newly observed regions are not trivially marked as events; only points that deviate substantially from the predicted local geometry are retained.

Thresholding. An event is triggered when the deviation exceeds a range-aware threshold. Let $r = \|\mathbf{p}\|_2$ and $r_{\min} = \min(\|\mathbf{p}\|_2, \|\Pi(\mathbf{p})\|_2)$. We define

$$E(\mathbf{p}) = \begin{cases} 1, & \text{if } |\Delta d(\mathbf{p})| > \tau(r_{\min}), \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

where $E(\mathbf{p})$ indicates whether $\mathbf{p}$ exhibits a significant deviation from the predicted local structure in the previous scan. For points with $E(\mathbf{p}) = 1$, we emit $\Delta d(\mathbf{p})$ as the event magnitude. We model the threshold as

$$\tau(r_{\min}) = g(\tau_{\min},\, \tau_{\text{sensor}}(r_{\min}),\, \tau_{\text{model}}(r_{\min}))\,, \tag{5}$$

where $\tau_{\min}$ is the minimum detectable deviation, τ_{sensor} and τ_{model} account for range-dependent measurement and modeling errors, and $g(\cdot)$ combines them into a single range-aware threshold. In this work, we instantiate τ as a range-independent constant τ_0 ($\tau(r_{\min}) \equiv \tau_0$), set per dataset and mode to match a target point reduction ratio for consistent evaluation.

3.2 Event-based 3D object detection

The proposed eventification serves as a lightweight front-end for 3D perception tasks. In this paper, we evaluate it on LiDAR-based 3D object detection as a representative application. Each point is represented by a feature vector

$$\mathbf{f} = (x, y, z, r, t_{\text{offset}}, \Delta),$$

where (x, y, z) are coordinates, r is reflection intensity, and t_{offset} is the time offset from the reference frame. The last component Δ is the event magnitude produced by temporal differentiation (i.e., the signed deviation Δd). For Waymo Open Dataset [32], the LiDAR pulse elongation channel is also commonly used and can be included as an additional input feature.

Conventional multi-scan detection aligns the past $N-1$ scans to the reference frame via ego-motion and concatenates them as input. Event-LiDAR instead converts each scan into a sparse eventified representation, reducing redundancy prior to network processing. We consider the following temporal input modes:

- **F**: full-scan input without eventification.
- **T**: temporally eventified input produced by the proposed temporal estimation and thresholding.

For multi-scan settings, these modes can be combined across time, e.g., NF or 1F+$(N-1)$T, where the most recent frame remains full (F) and the remaining scans are replaced by events (T). We analyze the trade-off between accuracy and efficiency in Section 4.

3.3 Event-aware network design

Eventification changes the input distribution: eventified point clouds are sparser and concentrate information on temporally informative variations. To better exploit such inputs, we explore an *event-aware* network design that reallocates model capacity toward the input stage. This design stays within the same backbone family but adjusts practical configurations such as per-stage channel widths and stage depth.

The event-aware variant is constructed to have a parameter budget comparable to the original network. By allocating more capacity to early stages, the network can better utilize sparse eventified inputs, while later stages are correspondingly reduced to maintain overall model complexity. This reallocation may increase early-stage computation, but its runtime impact depends on input sparsity and is therefore expected to be smaller for eventified inputs than for dense multi-scan inputs. We empirically evaluate these accuracy-efficiency trade-offs in Section 4.

4 Experiments

This section evaluates the proposed 3D eventification method on 3D object detection in terms of computational efficiency and detection accuracy. Additional experimental results and analyses are provided in the supplementary material.

4.1 Datasets and configurations

Datasets. We conduct experiments on nuScenes [4] and Waymo Open Dataset [32]. nuScenes provides 32-channel LiDAR scans at 20 Hz and is evaluated under a 10-scan concatenation setting with ego-motion compensation (N=10). Waymo

provides denser LiDAR measurements at 10 Hz and up to 64 vertical channels. For Waymo, we adopt a short temporal aggregation setting with N=3 consecutive scans. All experiments are conducted under multi-scan configurations.

Detection models. We adopt two detection frameworks: CenterPoint [9, 42] and voxel-based PTv3 [39]. The architectures are kept fixed across all settings and only the input point clouds are modified by eventification, except that PTv3 uses the event-aware (EA) variant (Section 3.3) as the default in Tables 3 and 4. Its effect is isolated in Table 5. Inference time is measured on a GeForce RTX 3090 by summing the runtime of all processing stages.

Differentiation modes. We evaluate temporal differentiation (T) described in Section 3. The full-scan (F) setting serves as the baseline without differentiation. For multi-scan inputs, we additionally evaluate hybrid configurations such as 1F+9T for nuScenes (N=10) and 1F+2T for Waymo (N=3), which combine one full scan with temporally differentiated scans.

Additional baselines and parameter settings. For comparison, we evaluate three eventification baselines: uniform random sampling (R, 20%), height thresholding (H), and range image-based depth differencing (I). For nuScenes, height thresholding uses a fixed 0.30 m threshold in the global frame to remove ground points, and range image differencing uses 0.452 m. In range image differencing, points without a depth-map correspondence are treated as events in 10I and excluded in 10I', we report both variants. For our temporal differentiation, the constant threshold τ_0 (Section 3.1) is tuned on the training split for an average event rate of about 20%, giving $\tau_0 = 0.072$ m (nuScenes) and 0.069 m (Waymo). Hybrid configurations use the same τ_0 as their T-only variants.

Event-extraction front-ends for comparison. To isolate the effect of the eventification stage, we replace only our temporal differentiation module with three alternatives, keeping the rest of the pipeline identical. Mapless-MOS (M) [43][4] flags points violating geometric consistency with a motion-compensated reference, FSD++ (RPP) (V) [11] retains points without close correspondences via voxel-based residual probing, and M-detector (D) [38] extracts moving points using its official implementation. These inputs are denoted M, V, and D, analogous to T for ours. Since the MOS-oriented settings of these methods retain too few points to train the detector (almost no static objects), we set the reduction ratio to about 80% for the T-only configurations, and report both the MOS-oriented and 80%-reduction settings for the hybrid configurations.

Training. For both datasets, CenterPoint and PTv3 are trained with a total batch size of 16 and an initial learning rate of 3×10^{-4}, for 20 epochs on nuScenes and 24 on Waymo. For the voxel-based PTv3 variant, we match the epoch count to PTv3 (pillar) and adopt the remaining hyperparameters from CenterPoint. To reduce computational cost, each epoch uses 25% of the full iteration schedule on nuScenes and 20% on Waymo, run on one GPU for nuScenes and two for Waymo. Training settings are kept identical across all input variants within each dataset-model pair.

[4] No public code is available; we use our own re-implementation.

Table 1: Retained ratio of eventified points on nuScenes (in %) for dynamic objects, static objects, and background. Ratios are computed using ground-truth bounding boxes and velocity information.

Category	R	H(0.3m)	H(80%)	I	I'	T_{point}	T_{line}	T_{plane}
Dynamic	20.1	83.4	8.0	33.6	27.7	36.5	51.4	53.4
Static	20.0	82.5	7.5	23.3	8.7	16.6	28.4	28.1
Background	20.0	41.7	21.3	19.8	20.7	20.4	19.6	19.2

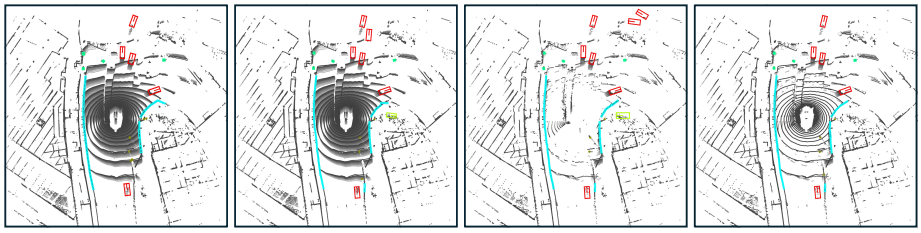
Table 2: Comparison of distance computation types (current point vs. past primitive) using CenterPoint on the nuScenes [4] and Waymo Open datasets [32]. nuScenes uses 10T and 1F+9T; Waymo uses 3T and 1F+2T. Pt. Red. (%) and Speedup are computed relative to the full-scan baseline for each dataset. Time is reported in milliseconds (ms). Metrics are mAP/NDS (nuScenes) and L2 mAP/L2 mAPH (Waymo).

Dataset	# Scan	Distance	Pt. Red.↑	Time ↓	Speedup ↑	mAP ↑	NDS/mAPH ↑
nuScenes	10T	Point	79.4	**33.7**	**1.16**×	55.8	63.5
		Line	79.9	34.9	1.12×	**58.7**	**65.1**
		Plane	80.4	<u>34.7</u>	<u>1.13</u>×	<u>57.5</u>	<u>64.5</u>
	1F+9T	Point	71.5	**34.7**	**1.13**×	59.0	65.2
		Line	71.9	35.9	1.09×	**59.5**	<u>65.5</u>
		Plane	72.3	<u>35.8</u>	<u>1.09</u>×	<u>59.3</u>	**65.7**
Waymo	3T	Point	79.2	31.7	1.24×	62.3	60.9
		Line	80.4	**31.6**	**1.25**×	**66.5**	**64.9**
		Plane	80.1	<u>31.6</u>	<u>1.24</u>×	<u>65.8</u>	<u>64.3</u>
	1F+2T	Point	52.8	**34.9**	**1.13**×	66.9	65.3
		Line	53.5	35.6	1.10×	<u>67.1</u>	<u>65.6</u>
		Plane	53.4	<u>35.4</u>	<u>1.11</u>×	**67.4**	**65.8**

4.2 Results

Distance type and point retention analysis. Here, T_{point}, T_{line}, and T_{plane} denote temporal eventification using point-, line-, and plane-based distance computation, respectively. Table 1 reports the retained ratio of eventified points for dynamic objects, static objects, and background on nuScenes. Compared with random sampling, temporal differentiation retains far more dynamic points while keeping a similar background ratio. T_{point} retains fewer object points, consistent with its lower detection accuracy. Range image differencing (I) shows a weaker bias toward dynamic points and yields lower detection accuracy (see Tables 3 and 4). Height filtering with a 0.300 m threshold retains many object points but provides limited reduction, while H(80%) uses a 2.197 m threshold and degrades accuracy by removing many object points. These trends suggest that the proposed local first-order geometric model used for temporal eventification helps prioritize motion-informative regions compared with naive baselines.

Table 2 compares T_{point}, T_{line}, and T_{plane} on nuScenes and Waymo. T_{point} yields a larger speedup but lower detection accuracy on both datasets. T_{plane} shows mixed behavior. On nuScenes, its speed is comparable to T_{line}, while its accuracy is slightly lower in the T-only setting. On Waymo, T_{plane} achieves



(a) GT (b) 10F (c) 10T (d) 1F+9T

Fig. 3: Qualitative comparison of input point clouds and detection results for GT, 10F, 10T, and 1F+9T.

comparable speed and matches or surpasses T_{line} in accuracy for the hybrid input, with timing differences within run-to-run variation. Considering these trends together with the point-retention characteristics on nuScenes, we use T_{line} as the default choice for subsequent experiments and report T_{plane} results in the supplementary material.

Qualitative examples. Fig. 3 visualizes the input point clouds and detection outputs for GT, 10F, 10T, and 1F+9T. Black points denote LiDAR returns, where lighter points correspond to older scans and darker points to more recent ones. In the full-scan multi-scan input (10F), accumulating multiple scans increases redundancy and expands the occupied regions in the discretized representation. In contrast, the proposed eventification (10T and 1F+9T) suppresses redundant points from past scans while preserving geometry relevant to object detection, resulting in a more compact input with similar detection outputs.

Processing time of eventification. The proposed eventification method enables highly efficient, low-latency preprocessing on a single thread of an Intel Xeon w5-3435X (3.1 GHz). With a SIMD-vectorized line-based approximation, the average processing time is 0.340 ± 0.049 ms per frame on the nuScenes dataset, evaluated over all frames from the 150 scenes in the validation split, and 1.644 ± 0.107 ms per frame on the Waymo Open dataset, evaluated over all frames from the 202 scenes in the validation split. For reference, M-detector classifies moving points and is substantially slower, taking 26.772 ± 13.903 / 78.947 ± 104.259 ms per frame on nuScenes/Waymo (16 threads) on the same machine; such cost largely offsets the inference-time benefit, whereas our millisecond-scale preprocessing preserves it.

Accuracy of object detection. Tables 3 and 4 summarize the multi-scan detection results. Uniform random sampling improves efficiency but causes a clear accuracy drop, whereas the proposed temporal eventification preserves accuracy much better at substantial point reduction, indicating that the selected points are more relevant for 3D detection than naive subsampling. It also outperforms the image-based baselines (10I, 10I') and the other front-ends (Mapless-MOS [43], FSD++ (RPP) [11], and M-detector [38]) at matched budgets, showing that retaining points unpredictable by a local first-order geometric model is more

Table 3: Point reduction (Pt. Red.), inference time, speedup, and detection accuracy on the nuScenes dataset [4]. Pt. Red. and speedup are relative to the full-scan baseline, and time is in milliseconds (ms). Metrics are mAP and NDS, and Δ indicates the use of the temporal residual feature. For PTv3, all rows use the event-aware (EA) architecture, whose effect is isolated in Table 5.

Network	Method	Δ	# Scan	Pt. Red. ↑	Time ↓	Speedup ↑	mAP ↑	NDS ↑
CenterPoint	Full-scan (baseline)		10F	0.0	39.1	1.00×	59.6	65.9
	Random		10R	80.0	**34.3**	**1.14**×	49.9	59.5
	Height Filtering		10H	54.4	**34.3**	**1.14**×	53.8	62.5
	Residual Image		10 I	79.7	34.8	1.12×	55.8	63.4
	Residual Image	✓	10 I	79.7	34.8	1.12×	55.5	63.4
	Residual Image		10I'	79.4	**34.3**	**1.14**×	42.1	53.7
	Residual Image	✓	10I'	79.4	34.4	1.14×	42.4	52.4
	3D Event (ours)		10T	79.9	34.8	1.12×	57.9	65.3
	3D Event (ours)	✓	10T	79.9	34.9	1.12×	58.7	65.1
	3D Event (ours)		1F+9T	71.9	35.9	1.09×	**59.5**	**66.1**
	3D Event (ours)	✓	1F+9T	71.9	35.9	1.09×	**59.5**	65.5
PTv3	Full-scan (baseline)		10F	0.0	71.2	1.00×	63.5	68.5
	Random		10R	80.0	53.3	1.34×	53.2	61.5
	Height Filtering		10H	54.4	54.5	1.31×	56.6	64.5
	Residual Image		10 I	79.7	53.7	1.33×	58.4	65.0
	Residual Image	✓	10 I	79.7	53.5	1.33×	59.7	65.7
	Residual Image		10I'	79.4	53.7	1.33×	46.4	56.4
	Residual Image	✓	10I'	79.4	53.9	1.32×	47.5	57.5
	Mapless-MOS [43]		10M	79.7	51.1	1.39×	51.4	59.8
	FSD++ (RPP) [11]		10V	79.4	58.1	1.23×	54.9	62.5
	M-detector [38]		10D	79.1	**46.6**	**1.53**×	34.2	49.8
	3D Event (ours)		10T	79.9	53.5	1.33×	60.9	66.6
	3D Event (ours)	✓	10T	79.9	53.4	1.33×	61.8	67.3
	Mapless-MOS [43]		1F+9M	71.7	55.3	1.29×	59.4	64.9
	Mapless-MOS [43]		1F+9M	87.0	48.6	1.46×	56.7	62.4
	FSD++ (RPP) [11]		1F+9V	71.4	61.8	1.15×	61.8	67.2
	M-detector [38]		1F+9D	71.2	52.4	1.36×	57.1	63.6
	M-detector [38]		1F+9D	88.2	47.6	1.50×	52.8	60.4
	3D Event (ours)		1F+9T	71.9	57.2	1.25×	62.9	67.8
	3D Event (ours)	✓	1F+9T	71.9	57.8	1.23×	**63.1**	**68.1**

Table 4: Point reduction (Pt. Red.), inference time, speedup, and detection accuracy on the Waymo Open dataset [32]. Pt. Red. and speedup are relative to the full-scan baseline, and time is in milliseconds (ms). Metrics are L2 mAP and L2 mAPH, and Δ indicates the use of the temporal residual feature.

Network	Method	Δ	# Scan	Pt. Red. ↑	Time ↓	Speedup ↑	L2 mAP ↑	L2 mAPH ↑
CenterPoint	Full-scan (baseline)		3F	0.0	39.4	1.00×	67.7	66.2
	Random		3R	80.0	31.9	1.23×	59.5	57.9
	3D Event (ours)		3T	80.4	**31.5**	**1.25**×	66.1	64.5
	3D Event (ours)	✓	3T	80.4	31.6	1.25×	66.5	64.9
	3D Event (ours)		1F+2T	53.5	35.6	1.11×	66.8	65.3
	3D Event (ours)	✓	1F+2T	53.5	35.6	1.10×	**67.1**	**65.6**
PTv3	Full-scan (baseline)		3F	0.0	87.9	1.00×	69.7	68.3
	Random		3R	80.0	57.6	1.53×	58.8	57.4
	3D Event (ours)		3T	80.4	**57.2**	**1.54**×	67.0	65.6
	3D Event (ours)	✓	3T	80.4	57.9	1.52×	67.0	65.7
	3D Event (ours)		1F+2T	53.5	72.9	1.21×	68.4	67.0
	3D Event (ours)	✓	1F+2T	53.5	73.1	1.20×	**68.8**	**67.5**

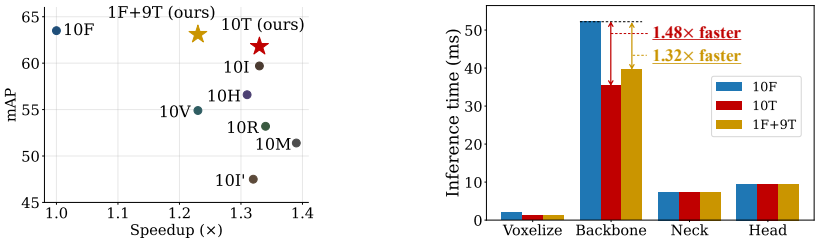


Fig. 4: Efficiency-accuracy analysis on nuScenes with PTv3. Left: speedup vs. detection accuracy (mAP) for different temporal sampling strategies; star markers denote our methods. Right: module-wise inference time breakdown, comparing temporal sampling strategies.

effective than image-domain differencing, absent-correspondence flagging, or motion classification.

Incorporating the temporal residual feature (Δ) provides complementary motion cues that help localize moving objects under aggressive point reduction, yielding a small but consistent improvement for PTv3 and gains in most CenterPoint settings. Furthermore, the hybrid configuration combining one full current scan with eventified past scans recovers accuracy toward the full-scan baseline while reducing redundancy: the full scan provides reliable spatial context, and eventified past scans supply motion-aware complementary information, yielding a favorable accuracy-efficiency trade-off.

Computational efficiency. Tables 3 and 4 report inference time and accuracy under multi-scan settings. Across both datasets and both backbones, uniform random sampling provides speedup but introduces a clear accuracy degradation, indicating an unfavorable efficiency-accuracy trade-off. In contrast, the proposed temporal eventification consistently preserves detection accuracy much better while still reducing inference time, achieving a more balanced trade-off between efficiency and performance (Fig. 4, left). The hybrid configuration that combines one full current scan with temporally differentiated past scans further improves accuracy toward the full-scan baseline while retaining most of the runtime benefit, suggesting that the full current scan stabilizes geometric evidence whereas temporal differences suppress redundant history.

For PTv3, module-wise profiling shows that the runtime reduction is dominated by the backbone stage (Fig. 4, right), and the proposed temporal eventification yields up to 48% speedup in the backbone computation compared with the full-scan input. Even under the hybrid 1F+9T setting, it still achieves a 32% backbone speedup. We observe a larger speedup for PTv3 than for CenterPoint in our setup, which is consistent with PTv3 having a heavier backbone computation where input sparsification provides greater benefit. At the same time, the achieved speedup depends on both the network design and the hardware characteristics, so the absolute ratios may vary across different implementations and devices. Overall, these results demonstrate that temporal eventification offers

Table 5: Effect of an event-aware (EA) network under full-scan (10F) and hybrid (1F+9T) inputs on nuScenes [4] with PTv3. The EA rows correspond to the PTv3 configuration in Table 3. #Param. is the number of network parameters, Pt. Red. is in %, and Time is in ms. ΔmAP/Δt and ΔNDS/Δt denote accuracy gain per added runtime when enabling EA.

# Scan	EA	#Param.	Pt. Red. ↑	Time ↓	mAP ↑	NDS ↑	ΔmAP/Δt ↑	ΔNDS/Δt ↑
10F		15.6M	0.0	61.6	62.3	67.7	N/A	N/A
10F	✓	15.7M	0.0	71.2	63.5	68.5	0.125	0.083
1F+9T		15.6M	71.9	52.3	62.0	67.1	N/A	N/A
1F+9T	✓	15.7M	71.9	57.8	63.1	68.1	**0.207**	**0.180**

Table 6: Ablation on reference frame vertical resolution for eventification on the Waymo Open dataset [32] under the 3T setting with Δ using CenterPoint. We vary the vertical resolution and report detection accuracy.

Vertical res.	64	32	16
L2 mAP ↑ / mAPH ↑	66.5 / 64.9	66.5 / 65.0	66.2 / 64.5

a practical way to accelerate multi-scan 3D detection without the large accuracy loss typically observed with naive subsampling.

Training time. We additionally evaluate the training-time efficiency of Event-LiDAR under the same multi-scan settings. We measure the end-to-end elapsed training time per epoch on an RTX PRO 6000 Blackwell Max-Q. On nuScenes, eventified inputs achieve up to 23% training-time speedup for 10T, and 17% for 1F+9T. On Waymo, eventified inputs achieve up to 36% training-time speedup for 3T and 14% for 1F+2T. We use pre-eventified point clouds saved to disk for both datasets; therefore, the reported speedups also include reduced data loading time. Detailed results are provided in the supplementary material.

Effect of reallocating model capacity. Table 5 reports results on nuScenes for an event-aware design that reallocates model capacity toward the input stage under 10F and 1F+9T. The event-aware model has a comparable parameter budget to the base model (15.7M vs. 15.6M) but increases early-stage computation, which leads to a moderate increase in end-to-end inference time. Notably, the accuracy gain per added runtime is larger for 1F+9T than for 10F, indicating that the hybrid input benefits more from emphasizing early-stage processing. Architectural details are provided in the supplementary material.

4.3 Ablation study

Robustness to reference frame resolution. Table 6 evaluates how the vertical resolution of the reference frame used for eventification affects detection accuracy on Waymo under the 3T setting with Δ using CenterPoint. Across the tested resolutions (64/32/16), detection accuracy remains essentially unchanged. Reducing the reference resolution from 64 to 32 shows no noticeable degradation

in L2 mAP or L2 mAPH. Further reducing it to 16 leads to a small drop, but the difference is minor. These results suggest that the proposed eventification is not sensitive to the reference frame vertical resolution within this range.

Sensitivity analysis on motion error. We evaluate the robustness of Event-LiDAR to ego-motion errors by injecting zero-mean Gaussian noise into the relative rotation and translation used for scan alignment, sweeping the noise over a range that extends well beyond drift levels typical of automotive-grade IMUs and odometry (see the supplementary material for the full sweep). Across this range, mAP and NDS remain essentially unchanged from the noise-free reference for both 10T and 1F+9T, with no abrupt degradation. While the point and non-zero voxel reduction ratios decrease gradually as the perturbation grows, detection accuracy is preserved throughout the tested range. Overall, these results suggest that Event-LiDAR is robust to realistic ego-motion estimation errors, maintaining both detection accuracy and sparsification behavior.

5 Conclusion

This paper presented Event-LiDAR, a 3D eventification framework that converts conventional LiDAR scans into temporally sparse representations by explicitly modeling inter-scan geometric variation. Rather than relying on direct scan differencing, we formulate event extraction as a temporal estimation problem under sparse, viewpoint-dependent observations: by approximating short-term geometric evolution with a local first-order geometric model, Event-LiDAR retains changes unpredictable by this model while suppressing redundant measurements in sensor space. Experiments on large-scale benchmarks show a favorable accuracy-efficiency trade-off for multi-scan 3D object detection: across datasets and backbones, the eventified representations substantially reduce processed points, accelerating inference and training while maintaining accuracy comparable to full-scan baselines. Being lightweight and architecture-agnostic, Event-LiDAR can be readily integrated into existing LiDAR perception pipelines as a front-end for efficient 3D processing, highlighting the potential of change-based representations for 3D sensing inspired by event-based sensing in vision. Future work includes a deeper analysis of failure cases, extending the framework to motion detection and to sequential propagation for single-scan inputs, and exploring broader applications such as tracking, mapping, and scene understanding.

Acknowledgments

This work was supported by JST BOOST Program Japan Grant Number JPMJBY24D3, JST PRESTO Grant Number JPMJPR22C4, and JSPS KAKENHI Grant Number JP25KJ1658. The authors are grateful to Jakgarin Hongfongfah for the insightful discussions and support that benefited this work in its early stage. The authors also thank Prof. Yuki Uranishi and Prof. Naoya Chiba of the University of Osaka for sharing their computational resources and for valuable discussions and comments.

References

1. Ahmed, S.H., Finkbeiner, J., Neftci, E.: Efficient Event-Based Object Detection: A Hybrid Neural Network with Spatial and Temporal Attention. In: CVPR. pp. 13970–13979 (2025)
2. Alonso, I., Murillo, A.C.: EV-SegNet: Semantic Segmentation for Event-based Cameras. In: CVPRW (2019)
3. Asvadi, A., Premebida, C., Peixoto, P., Nunes, U.: 3D Lidar-based static and moving obstacle detection in driving environments: An approach based on voxels and multi-region ground planes. Robotics and Autonomous Systems **83**, 299–311 (2016)
4. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuScenes: A multimodal dataset for autonomous driving. In: CVPR. pp. 11621–11631 (2020)
5. Chakravarthi, B., Verma, A.A., Daniilidis, K., Fermuller, C., Yang, Y.: Recent Event Camera Innovations: A Survey. In: ECCVW. pp. 342–376. Springer (2024)
6. Chen, X., Li, S., Mersch, B., Wiesmann, L., Gall, J., Behley, J., Stachniss, C.: Moving Object Segmentation in 3D LiDAR Data: A Learning-based Approach Exploiting Sequential Data. IEEE Robotics and Automation Letters (RA-L) **6**, 6529–6536 (2021)
7. Chen, X., Milioto, A., Palazzolo, E., Giguère, P., Behley, J., Stachniss, C.: SuMa++: Efficient LiDAR-based Semantic SLAM. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4530–4537 (2019)
8. Choudhury, R., Zhu, G., Liu, S., Niinuma, K., Kitani, K., Jeni, L.: Don't Look Twice: Faster Video Transformers with Run-Length Tokenization. In: NeurIPS (2024)
9. Contributors, M.: MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. `https://github.com/open-mmlab/mmdetection3d` (2020)
10. Dewan, A., Caselitz, T., Tipaldi, G.D., Burgard, W.: Motion-based detection and tracking in 3D LiDAR scans. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 4508–4513 (2016)
11. Fan, L., Yang, Y., Wang, F., Wang, N., Zhang, Z.: Super Sparse 3D Object Detection. IEEE TPAMI **45**(10), 12490–12505 (2023)
12. Gallego, G., Delbrück, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A.J., Conradt, J., Daniilidis, K., et al.: Event-based Vision: A Survey. TPAMI **44**(1), 154–180 (2022)
13. Gehrig, D., Gehrig, M., Hidalgo-Carrió, J., Scaramuzza, D.: Video to Events: Recycling Video Datasets for Event Cameras. In: CVPR (2020)
14. Gehrig, M., Scaramuzza, D.: Recurrent Vision Transformers for Object Detection with Event Cameras. In: CVPR (2023)
15. Gehrung, J., Hebel, M., Arens, M., Stilla, U.: AN APPROACH TO EXTRACT MOVING OBJECTS FROM MLS DATA USING A VOLUMETRIC BACKGROUND REPRESENTATION. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences **IV-1/W1**, 107–114 (2017)
16. Graziosi, D., Nakagami, O., Kuma, S., Zaghetto, A., Suzuki, T., Tabatabai, A.: An overview of ongoing point cloud compression standardization activities: Video-based (V-PCC) and geometry-based (G-PCC). APSIPA Transactions on Signal and Information Processing **9**, e13 (2020)

17. Hamaguchi, R., Furukawa, Y., Onishi, M., Sakurada, K.: Hierarchical Neural Memory Network for Low Latency Event Processing. CVPR (2023)
18. Hori, R., Isogawa, M., Mikami, D., Saito, H.: EventPointMesh: Human Mesh Recovery Solely From Event Point Clouds. TVCG (2024)
19. Hu, J.S., Kuai, T., Waslander, S.L.: Point Density-Aware Voxels for LiDAR 3D Object Detection. In: CVPR. pp. 8469–8478 (2022)
20. Ikeda, W., Hatano, M., Hara, R., Isogawa, M.: Event-based Egocentric Human Pose Estimation in Dynamic Environment. In: ICIP (2025)
21. Kammerl, J., Blodow, N., Rusu, R.B., Gedikli, S., Beetz, M., Steinbach, E.: Real-time compression of point cloud streams. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 778–785. IEEE (2012)
22. Kim, H., Leutenegger, S., Davison, A.J.: Real-Time 3D Reconstruction and 6-DoF Tracking with an Event Camera. In: ECCV. pp. 349–364. Springer (2016)
23. Mersch, B., Chen, X., Vizzo, I., Nunes, L., Behley, J., Stachniss, C.: Receding Moving Object Segmentation in 3D LiDAR Data Using Sparse 4D Convolutions. IEEE Robotics and Automation Letters (RA-L) **7**(3), 7503–7510 (2022)
24. Pagad, S., Agarwal, D., Narayanan, S., Rangan, K., Kim, H., Yalla, G.: Robust Method for Removing Dynamic Objects from Point Clouds. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 10765–10771 (2020)
25. Pfreundschuh, P., Hendrikx, H.F., Reijgwart, V., Dubé, R., Siegwart, R., Cramariuc, A.: Dynamic Object Aware LiDAR SLAM based on Automatic Generation of Training Data. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 11641–11647 (2021)
26. Quach, M., Pang, J., Tian, D., Valenzise, G., Dufaux, F.: Survey on Deep Learning-Based Point Cloud Compression. Frontiers in Signal Processing **2**, 846972 (2022)
27. Rebecq, H., Gehrig, D., Scaramuzza, D.: ESIM: an Open Event Camera Simulator. Conference on Robot Learning (CoRL) (2018)
28. Rebecq, H., Horstschaefer, T., Gallego, G., Scaramuzza, D.: EVO: A Geometric Approach to Event-Based 6-DOF Parallel Tracking and Mapping in Real Time. IEEE Robotics and Automation Letters (RA-L) **2**(2), 593–600 (2017)
29. Schwarz, S., Preda, M., Baroncini, V., Budagavi, M., Cesar, P., Chou, P.A., Cohen, R.A., Krivokuća, M., Lasserre, S., Li, Z., et al.: Emerging MPEG Standards for Point Cloud Compression. IEEE Journal on Emerging and Selected Topics in Circuits and Systems **9**(1), 133–148 (2018)
30. Shrout, O., Ben-Shabat, Y., Tal, A.: GraVoS: Voxel Selection for 3D Point-Cloud Detection. In: CVPR. pp. 21684–21693 (2023)
31. Stathoulopoulos, N., Kanellakis, C., Nikolakopoulos, G.: Have We Scene It All? Scene Graph-Aware Deep Point Cloud Compression. IEEE Robotics and Automation Letters (RA-L) **10**(12), 12477–12484 (2025)
32. Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al.: Scalability in Perception for Autonomous Driving: Waymo Open Dataset. In: CVPR. pp. 2446–2454 (2020)
33. Sun, Z., Messikommer, N., Gehrig, D., Scaramuzza, D.: ESS: Learning Event-based Semantic Segmentation from Still Images. In: ECCV. pp. 341–357. Springer (2022)
34. Vidal, A.R., Rebecq, H., Horstschaefer, T., Scaramuzza, D.: Ultimate SLAM? Combining events, images, and IMU for robust visual SLAM in HDR and high-speed scenarios. IEEE Robotics and Automation Letters (RA-L) **3**(2), 994–1001 (2018)
35. Wang, N., Shi, C., Guo, R., Lu, H., Zheng, Z., Chen, X.: InsMOS: Instance-Aware Moving Object Segmentation in LiDAR Data. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 7598–7605. IEEE (2023)

36. Wang, X., Jin, Y., Wu, W., Zhang, W., Zhu, L., Jiang, B., Tian, Y.: Object Detection using Event Camera: A MoE Heat Conduction based Detector and A New Benchmark Dataset. In: CVPR. pp. 29321–29330 (2025)
37. Wang, X., Şekercioğlu, Y.A., Drummond, T., Natalizio, E., Fantoni, I., Frémont, V.: Fast Depth Video Compression for Mobile RGB-D Sensors. IEEE Transactions on Circuits and Systems for Video Technology **26**(4), 673–686 (2016)
38. Wu, H., Li, Y., Xu, W., Kong, F., Zhang, F.: Moving event detection from LiDAR point streams. Nature Communications **15**(1), 345 (2024)
39. Wu, X., Jiang, L., Wang, P.S., Liu, Z., Liu, X., Qiao, Y., Ouyang, W., He, T., Zhao, H.: Point Transformer V3: Simpler, Faster, Stronger. In: CVPR (2024)
40. Wu, Z., Gehrig, M., Lyu, Q., Liu, X., Gilitschenski, I.: LEOD: Label-Efficient Object Detection for Event Cameras. In: CVPR (2024)
41. Yang, H., Liang, D., Zhang, D., Liu, Z., Zou, Z., Jiang, X., Zhu, Y.: AVS-Net: Point Sampling with Adaptive Voxel Size for 3D Scene Understanding. Neurocomputing p. 130533 (2025)
42. Yin, T., Zhou, X., Krahenbuhl, P.: Center-based 3D Object Detection and Tracking. In: CVPR. pp. 11784–11793 (2021)
43. Yoon, D., Tang, T., Barfoot, T.: Mapless Online Detection of Dynamic Objects in 3D Lidar. In: 2019 16th Conference on Computer and Robot Vision (CRV). pp. 113–120 (2019)
44. You, K., Chen, T., Ding, D., Asif, M.S., Ma, Z.: RENO: Real-Time Neural Compression for 3D LiDAR Point Clouds. In: CVPR. pp. 22172–22181 (2025)
45. Zhou, B., Xie, J., Pan, Y., Wu, J., Lu, C.: MotionBEV: Attention-Aware Online LiDAR Moving Object Segmentation with Bird's Eye View based Appearance and Motion Features. IEEE Robotics and Automation Letters (RA-L) **8**(12), 8074–8081 (2023)
46. Zou, S., Guo, C., Zuo, X., Wang, S., Wang, P., Hu, X., Chen, S., Gong, M., Cheng, L.: EventHPE: Event-based 3D Human Pose and Shape Estimation. In: ICCV. pp. 10996–11005 (2021)