

Fast Feature Field (F^3): A Predictive Representation of Events

Richeek Das, Kostas Daniilidis, Pratik Chaudhari

University of Pennsylvania

Email: richeek@seas.upenn.edu, kostas@seas.upenn.edu, pratikac@seas.upenn.edu

September 28, 2025

Abstract

This paper develops a mathematical argument and algorithms for building representations of data from event-based cameras, that we call Fast Feature Field (F^3). We learn this representation by predicting future events from past events and show that it preserves scene structure and motion information. F^3 exploits the sparsity of event data and is robust to noise and variations in event rates. It can be computed efficiently using ideas from multi-resolution hash encoding and deep sets—achieving 120 Hz at HD and 440 Hz at VGA resolutions. F^3 represents events within a contiguous spatiotemporal volume as a multi-channel image, enabling a range of downstream tasks. We obtain state-of-the-art performance on optical flow estimation, semantic segmentation, and monocular metric depth estimation, on data from three robotic platforms (a car, a quadruped robot and a flying platform), across different lighting conditions (daytime, nighttime), environments (indoors, outdoors, urban, as well as off-road) and dynamic vision sensors (resolutions and event rates). Our implementations can predict these tasks at 25–75 Hz at HD resolution. All code is available at <https://www.seas.upenn.edu/~richeek/f3>.

1 Introduction

A spot of daylight 1% brighter than the background delivers $\sim 10^9$ photons to the retina, whereas the faintest visible star delivers fewer than ten. The same retinal circuit processes both types of signals, compressing them up to ~ 10 million times with only a 10-fold decrease in sensitivity [1]. The mammalian retina operates under a strict time constraint. It must capture, process and transmit an image to the brain in ~ 100 ms. It employs a repertoire of ~ 80 distinct neuronal types arranged across three layers to meet this constraint. Rods and cones transduce light into graded voltages in low-light and daylight settings, respectively. These signals are relayed to about ~ 10 bipolar cell types that perform computations in the analog domain (for example, distinguishing bright from dark spots). Bipolar outputs feed into ~ 20 types of ganglion cells that detect local patterns such as luminance contrast, motion along different directions and color. At each stage, approximately ~ 40 types of “interneurons”—horizontal and amacrine cells—carve out irrelevant parts of the stimulus by inhibition. Both the spot of daylight and the light from a faint star in our example above are compressed down to a single spike [2, 3].

In some of the earliest work on neuromorphic perception, Mead and Mahowald sought to emulate this marvelous information processing of the biological retina in silicon [4]. Event-based cameras exemplify the state of the art today: they offer a dynamic range comparable to the retina (80–120 dB vs. ~ 140 dB, respectively), much higher temporal resolution (less than 100 μ s vs. 10–30 ms) because they operate asynchronously, and similar precision (~ 1 ns). Their spatial acuity is lower (1 MP vs. ~ 600 MP) but remarkable nonetheless. Event

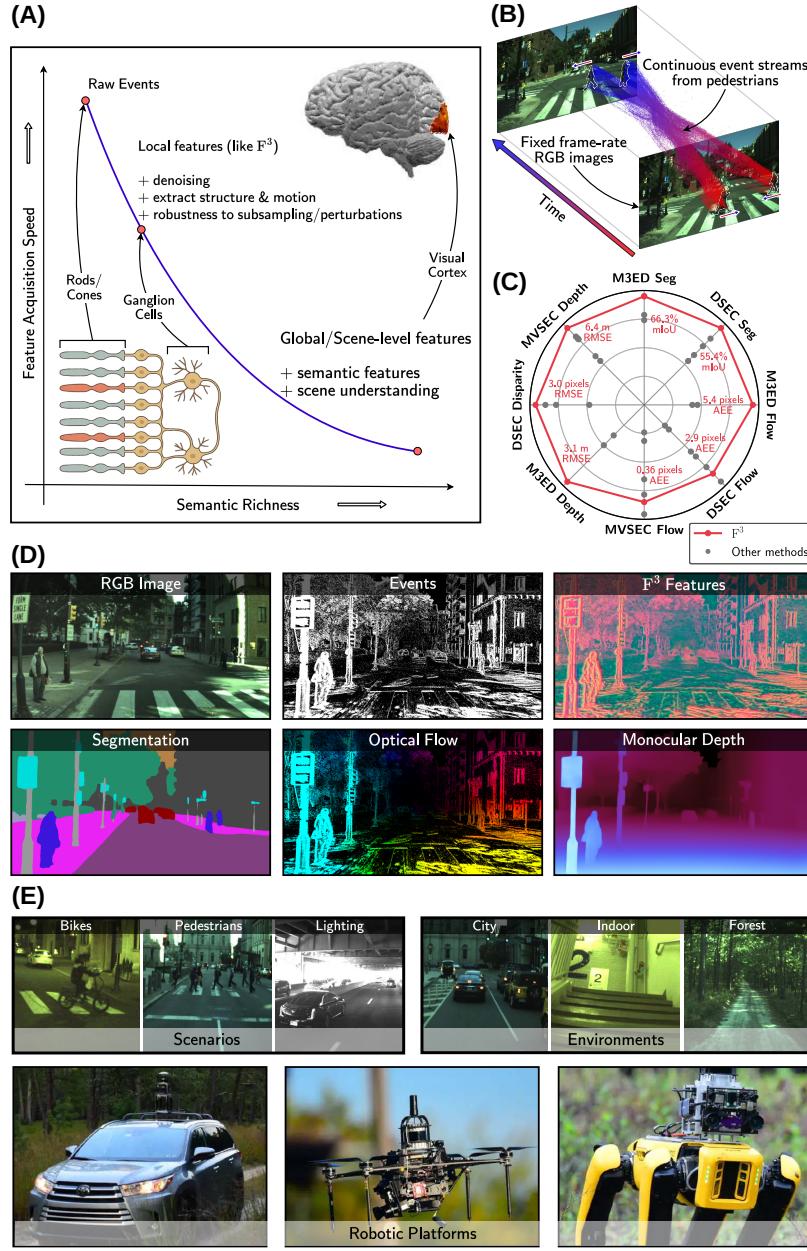


Figure 1: Overview. (A) The retina implements a complex repertoire of ~ 80 types of neurons arranged in three layers. Event cameras emulate rods and cones in the first layer, which transduce light. These measurements are highly local, redundant, and noisy, but fast. Successive steps of information processing in the retina produce more global, less redundant and less noisy features, culminating in more semantic features in the visual cortex. F^3 resembles features of the ganglion cells in the retina—fast, informative of structure and motion in the scene and robust to noise. (B) Unlike standard RGB cameras, event cameras are essentially asynchronous sensors that provide observations from the scene without temporal aliasing. (C) F^3 is a general representation for event-data that is effectively a multi-channel image. It can be incorporated into any standard computer vision algorithm and architecture. F^3 -based approaches significantly outperform existing methods (in gray) on tasks like semantic segmentation, optical flow estimation and monocular depth estimation. (D) A typical urban driving scene with an RGB image, events from the past few milliseconds (displayed as an image), F^3 features (three components) that capture salient aspects of the scene such as edges, moving objects, and road signs, segmentation masks, optical flow due to ego motion (colors denote direction), and monocular depth estimate.

Figure 1: (continued) (E) We study F^3 -based representations in urban driving scenarios, forested environments, and indoor navigation, across different lighting conditions (images with a high dynamic range, daytime, night-time), and across three different robotic platforms (a stereo event camera mounted atop a car, a flying platform and a quadruped robot).

cameras are very attractive to roboticists. They can be much more effective than standard RGB cameras that we use today. These sensors have the potential to enable robots to operate in different lighting conditions (harsh daylight as well as night-time, indoors and outdoors), at very high speeds (a quadrotor rotates at $\sim 700^\circ \text{ s}^{-1}$ when it flips but event cameras do not suffer from motion blur), and require little power and weight (they weigh only as much as a standard camera).

Event cameras, however, only emulate rods and cones in the retina. Photons are transduced into graded voltages and at each pixel, the camera registers whether the (log) intensity of light increased or decreased by a certain threshold since the last such “event”. The hallmark of information processing in the retina is the neural circuitry beyond these elementary units. Consequently, the output from an event camera is nowhere near as succinct as that of the biological retina. A single camera on a typical robot produces $\sim 50\text{M}$ raw—unprocessed—events per second, while retinal ganglion cells produce features at $\sim 0.5 \text{ Hz}$ [5]. This high throughput of an event camera puts severe constraints on the latency of downstream algorithms that must process this information, and drives up their power consumption. Circuits in the retina average their inputs and use negative feedback to work around thermal noise that activates light-sensitive proteins or synaptic noise that interferes with the signalling between neurons. Processing events also requires that we average over the readout and refractory noise to improve the signal-to-noise ratio, but the sensor itself does not do this. As a consequence, raw events from an event camera are extremely noisy, introducing severe hurdles for downstream algorithms that use the data.

In short, taking forward the intellectual program of Mead and Mahowald requires us to develop effective representations for event-based cameras and efficient algorithms to compute such representations. This is one of the most significant problems in the field today. While many existing approaches are specialized to various settings, no general principles exist for representation learning on data from event-based cameras.

Contributions of this paper

- We develop a mathematical argument for learning a predictive representation of event data. This representation is a statistic of past events sufficient to predict future events. We prove that such a representation retains information about the structure and motion in the scene.
- We instantiate this argument to develop a new representation for events called Fast Feature Field (F^3). F^3 achieves low-latency computation by exploiting the sparsity of event data using a multi-resolution hash encoder and permutation-invariant architecture. We give a mathematical argument for how the focal loss makes F^3 robust to noise and event rates.
- F^3 represents events in a contiguous spatiotemporal volume as a multi-channel image. This makes it possible to use F^3 in any standard computer vision algorithm or neural architecture built for RGB data. We show that F^3 can be the foundation for a variety of robotic perception tasks. We show state-of-the-art performance on data from three platforms (driving, quadruped locomotion and a flying platform) and four tasks (supervised semantic segmentation, unsupervised optical flow, unsupervised stereo matching, and supervised monocular metric depth estimation).
- F^3 enables real-time event-based robot perception. Our implementation can compute F^3 at 120 Hz and 440 Hz at HD and VGA resolutions, respectively, and can predict different downstream tasks at 25–75 Hz at HD resolution. These HD inference rates are roughly 2–5 times faster than the current state-of-the-art event-based methods.
- F^3 enables robust event-based robot perception. F^3 -based approaches work robustly without additional training across data from different robotic platforms, lightning and environmental conditions (daytime

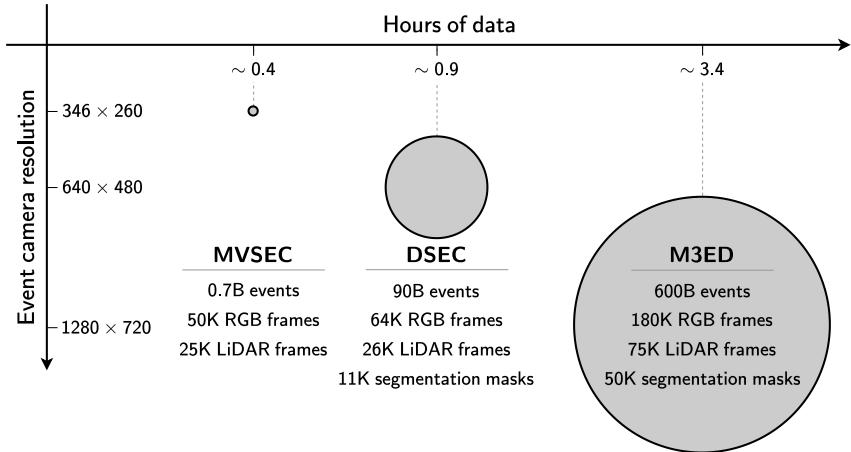


Figure 2: Overview of the datasets used in this paper. This work focuses on data collected from different robotic platforms (human-driven car, a flying platform, and a quadruped robot), (ii) different event cameras and sensors (VGA and HD resolutions in addition to a smaller 346×260 sensor), and (iii) experimental settings (urban and off-road environments, both indoor and outdoor locomotion tasks, different lighting conditions).

vs. night-time, indoors vs. outdoors, urban vs. off-road) and dynamic vision sensors (with different resolutions and event rates).

2 Results

Robotic platforms, sensors, experimental settings and tasks Fig. 2 summarizes the data used in this work. MVSEC [6], which was recorded in Philadelphia, only has about 0.4 hours of urban driving data. Still, it has an impressive diversity in terms of lighting conditions (both daytime and nighttime), with cars and motorcycles in the field of view. It contains data from a 346×260 resolution DAVIS346 event camera and depth data from LiDAR scans, as well as optical flow computed using the depth data. DSEC [7], which was recorded in Zurich, Switzerland, is a slightly larger dataset with about 0.9 hours of data from urban areas, mountains, and lakes. Events are recorded from a 640×480 resolution Prophesee Gen 3.1 VGA event camera under diverse illumination conditions. In addition to this, we also have data from a Velodyne LiDAR and stereo FLIR RGB cameras. M3ED [8] is a large dataset recorded in Philadelphia. It contains about 3.4 hours of event data in different conditions, like daytime, nighttime, sudden illumination changes in tunnels, etc. M3ED contains data from three robotic platforms, a car, a quadruped robot and a flying platform, in indoor and outdoor scenes. Pedestrians, cars, bikes and dense foliage are in the field of view in outdoor scenes, while indoor scenes contain a large diversity of objects of different shapes and sizes. M3ED has a time-synchronized sensor suite that includes a stereo setup with 1280×720 resolution Prophesee EVK4 HD cameras, a 360 degree field of view Ouster LiDAR, and stereo RGB cameras. Sec. S.3 provides more details of these datasets.

2.1 Predictive representations of events are informative of structure and motion

Consider a fixed Lambertian scene defined in terms of its shape and reflectance, under constant illumination. The pixels in an event camera respond to changes in the logarithmic intensity of light incident upon them. An event $e(t, u) \in \{+1, -1\}$ is “triggered” at a time $t \in \mathbb{Z}_+$ and pixel $u = (u_1, u_2) \in \Omega \subset \mathbb{Z}^2$ when this

log-intensity increases or decreases, respectively, by some pre-defined threshold.¹ We set $e(t, u) = 0$ for pixels where events are not triggered. We will ignore the sign and set $e(t, u) \in \{1, 0\}$ to denote the presence or absence of an event.² Given viewpoints along a camera trajectory $x \equiv (x(t))_{t \in \mathbb{Z}_+}$ with each $x(t) \in \text{SE}(3)$, let ξ be a sufficient statistic of the scene for creating the events, i.e.,

$$e(t, u) = \pi(\xi(t, u; x); \nu) \quad (1)$$

for some function π and noise ν . The event, at any time t and pixel u , is thus a random variable determined from the (past) camera trajectory x and the statistic $\xi(t, u; x)$. For example, Marr's primal sketch (edges, discontinuities in depth, etc.) [9] of a scene can render any future event. The statistic $\xi(t, u; x)$ is the projection of the primal sketch upon the image plane, followed by differentiation in time, with π being the acquisition mechanism of an event camera, e.g., thresholding of the log-intensity. We will design an estimator $\hat{\xi}$ using events e without knowledge of the camera trajectory x . The dependence on x is omitted for clarity.

Ideal denoising by identifying orthonormal bases that lead to sparse representations For a fixed $t \in \mathbb{Z}_+$ and $s \in [0, t]$, let events be $\mathbb{R} \ni e(s, u) = \xi(s, u) + \nu$ where $\nu \sim N(0, 1)$ is assumed to be unit-variance for clarity of the mathematical expressions. This simplistic model will be useful to develop our technical argument. We will comment on generalizations in Sec. 3. The acquisition mechanism π adds Gaussian noise to the statistic $\xi(t, u) \in \mathbb{R}$ and therefore estimating the statistic $\hat{\xi}$ is a denoising problem. This denoising problem can be solved by minimizing the risk

$$R(\hat{\xi}, \xi) = \left\| \hat{\xi} - \xi \right\|^2 \triangleq \sum_{s=0}^t \sum_{u \in \Omega} \left(\hat{\xi}(s, u) - \xi(s, u) \right)^2, \quad (2)$$

on average over noise ν where $\|\cdot\|$ denotes the ℓ_2 norm. We will focus on estimating coefficients $(\xi_{\mathcal{B}})_i = \langle \xi, \varphi_i \rangle$ of ξ projected upon the basis $\mathcal{B} = \{\varphi_1, \dots, \varphi_n\}$ with each function φ_i supported on $\mathbb{Z}_+ \times \Omega$. For example, \mathcal{B} could be the Fourier or wavelet bases. When the basis is orthonormal due to Parseval's identity, we have $R(\hat{\xi}, \xi) = \left\| \hat{\xi}_{\mathcal{B}} - \xi_{\mathcal{B}} \right\|^2 \triangleq \sum_{i=1}^n \left((\hat{\xi}_{\mathcal{B}})_i - (\xi_{\mathcal{B}})_i \right)^2$. In general, minimizing this risk is hard. But diagonal denoisers, i.e., when the estimate $(\hat{\xi}_{\mathcal{B}})_i$ only depends on the input projected on the i^{th} basis element $(e_{\mathcal{B}})_i$, are an important special case where the minimal risk is $R(\xi, \mathcal{B}) = \sum_i \min((\xi_{\mathcal{B}})_i^2, 1)$ [10]. If the hard threshold non-linearity is $\sigma_{\tau}(v) = v \mathbf{1}_{\{|v| > \tau\}}$ for $v, \tau \in \mathbb{R}$, then the estimator

$$(\hat{\xi}_{\mathcal{B}})_i = \sigma_{\sqrt{2 \log n}}((e_{\mathcal{B}})_i), \quad \forall i \in \{1, \dots, n\} \quad (3)$$

has $R(\hat{\xi}, \xi) \leq (2 \log n + 2.4)(1 + R(\xi, \mathcal{B}))$, which is sub-optimal only by a $\log n$ factor. Diagonal denoisers are nearly optimal if the signal is sparse in \mathcal{B} [11]. It is challenging to identify a basis in which real-world data are sparse. But one could search over a library \mathcal{L} , e.g., consisting of Fourier basis elements with support on different frequency bands, and select

$$\hat{\mathcal{B}} = \underset{\mathcal{B} \in \mathcal{L}}{\operatorname{argmin}} \sum_{i=1}^n \min((e_{\mathcal{B}})_i^2, \Lambda_n) \quad (4)$$

where $\Lambda_n = \lambda^2(1 + \sqrt{2 \log M_n})^2$, $\lambda > 8$ and M_n is the number of basis elements in the library \mathcal{L} . The

¹In this paper, we will work exclusively in discretized time, assuming that we have discretized event timestamps using a small interval, say, 1 ms. Multiple events at the same pixel within 1 ms of each other are treated as identical.

²All techniques and theory developed in this paper can be adapted to incorporate events with polarity, at the cost of a larger memory and computational footprint. One of our baseline approaches, named "event frames", does incorporate polarity.

hard-threshold estimator in this basis $(\hat{\xi}_{\mathcal{B}})_i = \sigma_{\sqrt{\Lambda_n}}((e_{\mathcal{B}})_i)$, for $i \leq n$ satisfies

$$R(\hat{\xi}, \xi) \leq (1 - 8/\lambda)^{-1} \Lambda_n \min_{\mathcal{B} \in \mathcal{L}} R(\xi, \mathcal{B}),$$

with probability greater than $(1 - e/M_n)$ where e is Euler's number [12]. Usually, the ideal risk $\min_{\mathcal{B} \in \mathcal{L}} R(\xi, \mathcal{B})$ grows as a polynomial of n . For example, in a library of wavelet packets, the number of basis elements is $M_n \sim n \log_2 n$. In such a library, this strategy is only sub-optimal by a $\Lambda_n \sim \log n$ factor. Similar results hold for more general bases, e.g., Riesz basis or frames [11, Chapter 11]. This discussion suggests that we can search over a library to (i) identify the basis in which input data are sparse, and (ii) denoise the data using hard thresholding in this basis to obtain a representation.

Ideal denoising and spatiotemporal prediction of events using a library of bases We will next specialize the above argument to signals like event camera data, which have spatiotemporal dynamics. The key idea is to simultaneously identify a basis to denoise events and the dynamics of the sufficient statistic ξ . Let $\xi^-(s, u) \in \mathbb{R}$ denote ξ restricted to $s \in [t - \Delta t, t]$. Similarly, let $\xi^+(s, u)$ denote the restriction to $s \in [t, t + \Delta t]$. The notations e^+ and e^- are analogous. Suppose

$$\xi^+ = A\xi^- + \zeta, \quad (5)$$

where $\zeta \sim N(0, I_{m \times m})$ is Gaussian noise with $m = \Delta t \times |\Omega|$. The linear map $A : \mathbb{R}^m \rightarrow \mathbb{R}^m$ transforms past statistics to future ones, up to noise ζ . Under two technical assumptions on the nature of this map, namely that the operator norm of A is bounded and that it does lead to a drastic decrease in sparsity when it acts upon a sparse signal, we can prove the following theorem under these assumptions.

Theorem 1. The dynamics and the denoising basis

$$\hat{A}, \hat{\mathcal{B}} = \underset{A, \mathcal{B} \in \mathcal{L}}{\operatorname{argmin}} \left\{ \|e^+ - A\hat{\xi}^-\|_2^2 + \Lambda_n \|\hat{\xi}_{\mathcal{B}}^-\|_0 \right\} \quad (6)$$

where $(\hat{\xi}_{\mathcal{B}})_i = \sigma_{\sqrt{\Lambda_n}}((e_{\mathcal{B}})_i)$ for $i \leq n$ and $\hat{\xi}^-$ and $\hat{\xi}_{\mathcal{B}}^-$ denotes the same signal represented in the cardinal and \mathcal{B} bases respectively, satisfy

$$\|\xi^+ - \hat{A}\hat{\xi}^-\|_2^2 \leq c_1 R_{\text{dynamics}}(\xi, \mathcal{L}) + c_2 \Lambda_n R_{\text{denoising}}(\xi^-, \mathcal{L}), \quad (7)$$

with probability at least $1 - ce/M_n$. Here c is a constant greater than 10. Constants c_1 and c_2 only depend on the assumed properties of the dynamics operator A . The quantities $R_{\text{dynamics}}(\xi, \mathcal{L})$ and $R_{\text{denoising}}(\xi^-, \mathcal{L})$ are oracle risks that depend upon (i) the ideal denoising basis for past events e^- , (ii) the coordinates i where $(e_{\mathcal{B}})_i$ is large, and (iii) the true future statistic ξ^+ .

Sec. S.1 provides a proof. We can interpret Theorem 1 as separating the structure $(\hat{\xi}^-)$ and motion (\hat{A}) in the scene. In Eq. (6), the first term measures the discrepancy between the future events e^+ and the hard threshold version of past events $\hat{\xi}_{\mathcal{B}}^-$, after transforming them using the operator A , and transforming back to the cardinal basis. The second term in the objective encourages $\hat{\xi}_{\mathcal{B}}^-$ to be sparse. This theorem suggests a broad principle we can implement for learning representations of events. If we identify a basis in which past events are sparse, and use the denoised past events to predict future events, then we can recover the true future statistic ξ^+ . Doing so incurs a risk that is off from the oracle risk by a constant factor c_1 , and the ideal denoising risk by a logarithmic factor $c_2 \Lambda_n \sim \log n$.

2.2 A fast neural architecture for learning representations of event data

Theorem 1 suggests training a representation $\hat{\xi}^-$ of past events e^- to predict future events e^+ . Fig. 3 demonstrates a simple instantiation of Eq. (6) using a U-Net architecture [13]. The encoder implements a basis $\hat{\mathcal{B}}$ and projects past events e^- to obtain $\hat{\xi}^-$ after thresholding. The decoder uses this representation to predict future events and implements the dynamics operator \hat{A} .³ When this representation is used to predict optical flow (Sec. 2.4 provides details), it obtains an average end-point error (AEE) of 7.71 pixels and a mean angular error (MAE) of 17.63° on M3ED HD event data, providing inference at 50 Hz. We will next develop two key ideas that improve upon this instantiation to obtain F^3 . An F^3 -based optical flow predictor has an AEE of 5.35 and MAE of 12.58° , 75 Hz predictions on HD event data and about $10 \times$ smaller training time.

Exploiting the sparsity of event data The above U-Net based instantiation does not exploit the sparsity of event data. For an HD event camera, there are about 850 million voxels/sec at a time-discretization of 1 ms. About 95% of these voxels are zero in typical environments. Architectures like the U-Net discussed above would process all voxels. We need an architecture that operates only upon the triggered events and does not process empty voxels. Such an architecture should be invariant to the order of occurrence of events within a spatiotemporal region. It should also be able to address the fact that the number of events within a spatiotemporal region is variable. These desiderata entail a set-based architecture for representing events.

Consider the timestamps $i^-(u) = \{(s : s \in [t - \Delta t, t], e(s, u) = 1\}$ of events in the past Δt time interval at a pixel $u \in \Omega$. The cardinality $|i^-(u)|$ is different at different pixels u . Let $\Omega_u \subset \mathbb{Z}^2$ denote a neighborhood of the pixel u . A representation $\hat{\xi}(t, u)$ of the set $i^-(u)$ is invariant to permutations of its elements if and only if it can be decomposed as

$$\begin{aligned} \mathbb{R}^n &\ni \bar{\varphi}(t, u) = \sum_{s \in i^-(u)} \varphi(s, u) \\ \mathbb{R}^p &\ni \hat{\xi}(t, u) = \rho [\bar{\varphi}(t, \cdot)](u) \end{aligned} \tag{8}$$

for functions $\varphi : \mathbb{Z}_+ \times \Omega \rightarrow \mathbb{R}^n$ and $\rho : \Omega_u \times \mathbb{R}^n \rightarrow \Omega_u \times \mathbb{R}^p$ that can be universal approximators such as neural networks [14]. We will next discuss how to appropriately choose the functions φ and ρ for event data.

The key is to think of $\varphi(s, u)$ as the “feature” of an individual event. These features are the projections of event coordinates $(s, u) \in \mathbb{Z}_+ \times \Omega$ into a learned basis. While any reasonable basis can perform this

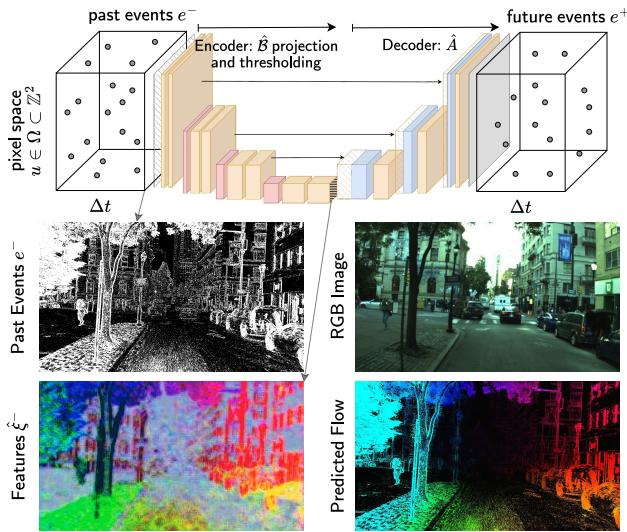


Figure 3: A simple instantiation of Theorem 1. A U-Net-based architecture learns to predict future events e^+ from past events e^- (top left image). The output of the encoder, i.e., the bottleneck layer, is the representation $\hat{\xi}^-$, while the decoder implements the dynamics operator \hat{A} . Features learned by this architecture (bottom left) retain global information across the frame and can be used to compute optical flow (bottom right) using the techniques developed in Sec. 2.4. This architecture works reasonably well for the task, thereby providing a computational validation of Theorem 1.

³We make two practically motivated modifications to Eq. (6): (i) a weighted mean-squared error to address the imbalance in the number of event and non-event voxels, and (ii) an ℓ_1 norm in place of ℓ_0 to make the optimization tractable.

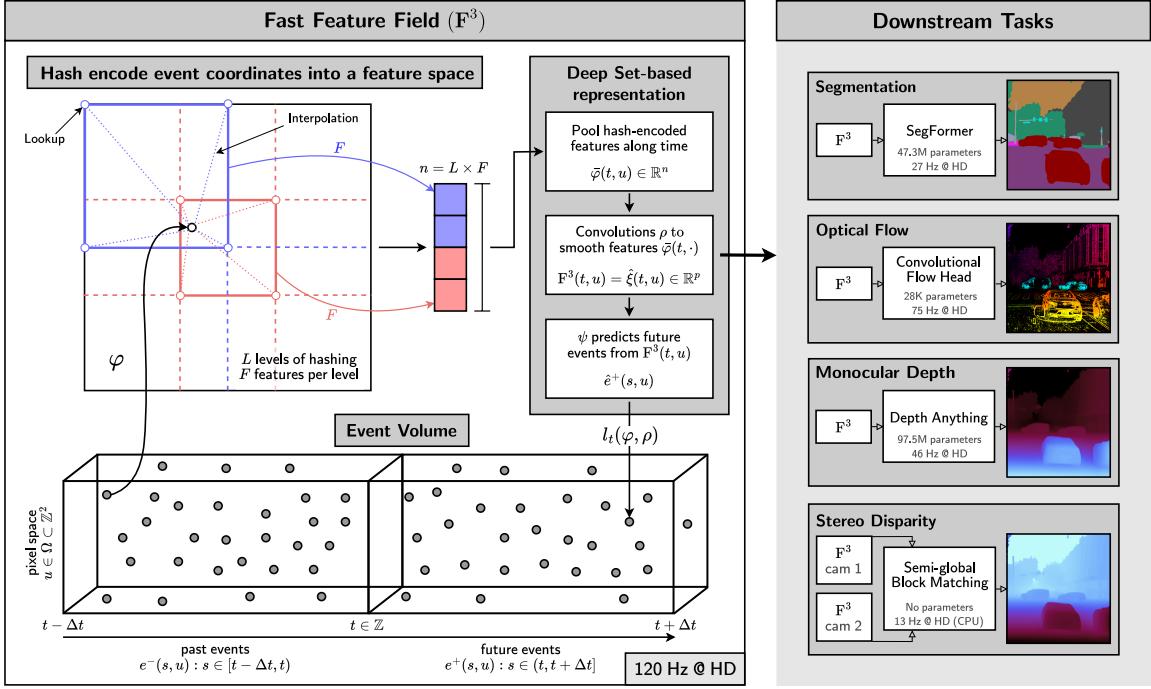


Figure 4: An overview of the neural architecture for Fast Feature Field F^3 . Time and pixel coordinates of past events $e^-(s, u)$ between times $s \in [t - \Delta t, t]$ and pixels $u \in \Omega$ are encoded using a hash encoder into a feature space, pooled across time, and smoothed in space to obtain the feature $F^3(t, u)$ at a pixel u and time t . F^3 can be computed very quickly because the hash encoder does not encode coordinates with no events. F^3 is trained to predict future events $e^+(s, u)$ using a linear layer (ψ) using a class-weighted focal loss. F^3 essentially encodes the event volume into a multi-channel image. Supervised tasks such as semantic segmentation and monocular depth estimation can be easily learned using F^3 . Unsupervised tasks such as optical flow or stereo disparity prediction can also be performed by matching these features across time or stereo cameras, respectively.

projection, in view of our downstream tasks, picking one that gives multi-scale spatiotemporal features tuned to the data is useful. As shown in Fig. 4, we use a multi-resolution structure which maintains L resolution levels and assigns indices to the corners by hashing their integer coordinates. Given an event coordinate (s, u) , the function φ interpolates F -dimensional features corresponding to the corners at each scale to result in an $n = LF$ dimensional feature $\varphi(s, u)$. Such bases are popular in the literature on neural rendering fields for encoding 3D space. Our implementation is identical to that of [15], except that one of our coordinates is time. A hash encoder brings in a number of benefits that were missing in the simplistic U-Net approach discussed above: (i) fast training and inference due to integer indices in the hash encoder, (ii) cheaper forward pass using lookup operations compared to, say, a basis projection implemented via matrix multiplication in a multi-layer perceptron (MLP), (iii) cheaper backward pass compared to that of an MLP because only features that were accessed in the forward pass are updated. We will pick ρ to be a convolutional neural network that smooths the encoding at each pixel $\sum_{s \in i-(u)} \varphi(s, u)$ in the spatial domain. This performs local marginalization of translation nuisances [16].

Fast feature field is defined as

$$\mathbb{R}^p \ni F^3(t, u) \equiv \hat{\xi}(t, u).$$

Fitting the multi-resolution hash encoder is equivalent to finding an optimal basis [12] from data in Eq. (6) and Eq. (4) and therefore we think of $F^3(t, u)$ as the denoised version of raw data from an event camera. For all

experiments in this paper, unless otherwise noted, we choose $\Delta t = 20$ ms, an encoder φ with $L = 4$ levels, a hash table of size 2^{19} and 2-dimensional features per entry, a receptive field of $\Omega_u = 37 \times 37$ with $p = 32$ output channels for the convolutional network ρ in Eq. (8). Let us note that this architecture can also handle events with real-valued timestamps $t \in \mathbb{R}_+$.

Fast feature field $F^3(t, \cdot)$ in Eq. (8) can be thought of as a p -channel “image” that retains information about correlations in the event stream across the time interval $[t - \Delta t, t]$ and within a spatial neighborhood $\Omega \ni u$. Any algorithm or neural architecture designed for RGB images, i.e., $p = 3$, can be adapted using F^3 to work with event data. As baseline approaches, we can consider two naive variants:

$$\begin{aligned} \text{Event voxel grid: } \mathbb{R} &\ni V^3(t, s, u) = e(s, u) \quad \text{for } s \in [t - \Delta t, t], \text{ and} \\ \text{Event frames: } \mathbb{R} &\ni I^3(t, r, u) = \mathbf{1}_{\{|i_r^-(u)| \geq 1\}} \quad \text{for } r \in \{+1, -1\}, \end{aligned} \tag{9}$$

where $i_r^-(u) = \{(s : s \in [t - \Delta t, t], e(s, u) = r)\}$ for polarity $r = \{+1, -1\}$. The first, which we call “event voxel grid” represents past events within a time window $[t - \Delta t, t]$ in the cardinal basis. This is similar to voxel grid representations used in the event perception literature [17]. The second, which we call “event frames” records whether or not there was an event at a pixel u within the time window [18]. Our event voxel grids V^3 retain temporal information but ignore polarity, while our event frames I^3 retain polarity information but ignore time. This is why we expect both baseline representations to be useful for downstream tasks pertaining to structure and motion.

Training objectives to build robustness to noise and event rates Next, we instantiate the objective in Eq. (6) to train a predictor of future events from past events. Noise and extreme imbalance between events and non-events make it difficult to predict future events. We therefore need a more robust objective than the squared error in Eq. (6) to ensure invariance to these nuisances. For events that are triggered between times $[t, t + \Delta t]$ with coordinates in $\cup_{u \in \Omega} i^+(u)$, we use a variant of the cross-entropy loss called the focal loss [19] given by

$$\ell_t(\varphi, \rho, \psi) = - \sum_{s \in [t, t + \Delta t]} \sum_{u \in \Omega} \alpha e(1 - \hat{e})^\gamma \log \hat{e} + (1 - \alpha)(1 - e)\hat{e}^\gamma \log(1 - \hat{e}), \tag{10}$$

where

$$e \equiv e(s, u), \quad \hat{e} \equiv \psi(s, u; F^3(t, u)),$$

and $\alpha = |\cup_{u \in \Omega} i^+(u)| / (\Delta t |\Omega|)$ is the fraction of voxels containing events. The dynamics \hat{A} in Theorem 1 and Eq. (6) is represented by the function $\psi : \mathbb{Z}_+ \times \Omega \times \mathbb{R}^p \rightarrow [0, 1]$ in this objective. It predicts a future event at coordinate (s, u) using the representation of the past events $F^3(t, u)$. Given events between times $t \in [0, N\Delta t]$, the objective for training F^3

$$\ell(\varphi, \rho, \psi) = \sum_{i=1}^{N-1} \ell_{i\Delta t}(\varphi, \rho, \psi)$$

for a hyper-parameter Δt . We forgo the regularization term $\Lambda_n \|\hat{\xi}_B^-\|_0$ in Eq. (6) and replace it with weight-decay during training. We use a fully-connected linear layer for the dynamics ψ in Eq. (10) for all experiments.

Sec. S.2 discusses why the focal loss helps predict events. First, we show that when the scene is described by non-overlapping surfaces, it is necessary to reweigh the binary cross-entropy loss using α to prevent trivial features which predict $\hat{e}(s, u) = 0$ for all s, u . This necessity stems from the inherent stochasticity and sparsity of events. Second, we show that minimizing the focal loss with a non-zero value of the parameter γ leads to a maximum-entropy predictor for $\hat{e}(s, u)$ subject to predicting events $e(s, u)$. This regularization is important because it prevents F^3 from overfitting to noisy event data.

2.3 Supervised semantic segmentation

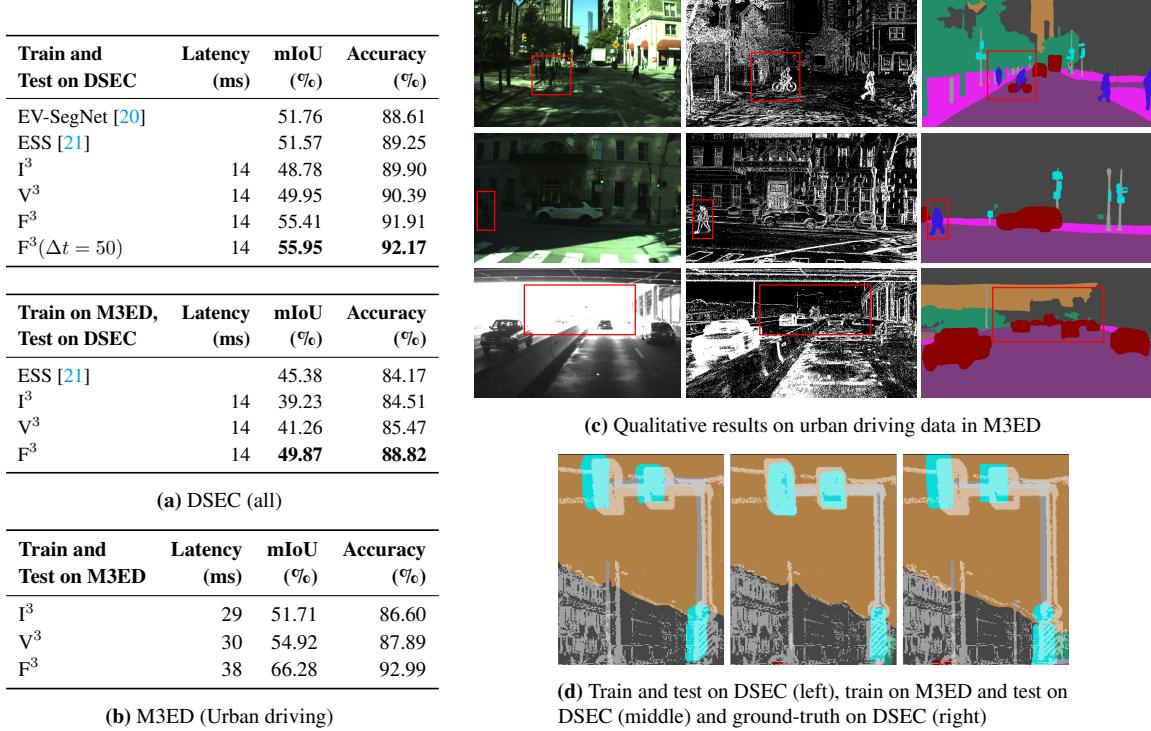


Figure 5: Supervised semantic segmentation task for driving data in daytime settings from DSEC and M3ED. (a) F^3 event-based semantic segmentation compares favorably to existing approaches on DSEC driving data in terms of mean intersection-over-union (mIOU) and accuracy. An F^3 -based network trained on M3ED data and tested on DSEC data performs much worse (bottom table), but, as Fig. 5d shows, this is due to misalignment between ground-truth segmentation masks and event data in DSEC. (b) An F^3 -based network trained and tested on HD event data from M3ED daylight driving sequences achieves a higher mIOU (66.28 vs. 55.95 earlier). (c) Qualitative results demonstrate the diversity of data. Objects that blend heavily into shadows or lie in highly saturated regions in the RGB image (left column) are prominently visible in event data (middle column). Objects of different sizes (cars, pedestrians, sidewalks, traffic lights) can be accurately segmented using event data. (d) The image on the right shows that ground-truth segmentation masks computed from RGB images are misaligned with event data in DSEC. An F^3 -based network trained on M3ED (where this misalignment is absent) can predict accurately (middle image). But this is not reflected in Fig. 5a (bottom) because the ground-truth itself is misaligned. An F^3 -based network trained on DSEC looks much better in terms of metrics in Fig. 5a (top) but predicts misaligned segmentation masks in reality (left image).

We first show the plug-and-play nature of pre-trained F^3 using semantic segmentation tasks. We can use any architecture that takes a pre-trained F^3 as input and is trained to predict ground-truth annotations or pseudo-labels. As an example, consider a SegFormer B3 architecture [22] that is pretrained on the Cityscapes [23] dataset. This network takes RGB images as input, but it is easy to adapt it to take F^3 with p channels as input by tiling and copying the weights of the first convolutional layer. This same procedure can be used to modify a pre-trained network to use V^3 or I^3 representations. We then fine-tune the modified SegFormer architecture on ground-truth annotations using the cross-entropy loss to obtain an F^3 -based semantic segmentation model.

Fig. 5c shows a qualitative evaluation of our approach. Figs. 5a and 5b show the accuracy and mean intersection-over-union (mIOU) across 11 object categories on M3ED and DSEC, respectively. The F^3 -based network achieves high mIOUs, 66.28% on M3ED and 55.95% on DSEC. This is better than existing event-based semantic segmentation approaches, e.g., [21, 20]. It is also better than baseline features like V^3 and I^3 that

were motivated in this paper. In some cases, the improvement is more than 11%. When the F^3 -based network is trained on M3ED data and evaluated on DSEC data in Fig. 5a (bottom), the mIOU and accuracy are still substantially high. This shows that F^3 can generalize to new datasets with different sensors (Prophesee EVK4 vs. Gen 3.1 for M3ED and DSEC, respectively), resolutions (HD vs. VGA) and scenery (urban vs. urban and outdoors). The specific choices in our development of F^3 , namely (i) a representation that is ideal for denoising and motion estimation in Theorem 1, and (ii) spatial convolution in Eq. (8) to build insensitivity to translation nuisances, are quite useful for this task. Baseline approaches based on V^3 and I^3 , which do not have these properties, perform poorly on the test data even if their mIOUs on the training data (not shown here) are comparable to that of F^3 . Observe that in both Figs. 5a and 5b voxel grids V^3 , which retain temporal information, are more effective for segmentation than event frames I^3 .

It is difficult to obtain ground-truth human annotations for event data because of its size and unusual nature. It is therefore common in the literature to transfer annotations from RGB images. For example, M3ED uses a recent state-of-the-art network called InternImage [24] to pseudo-label RGB images while DSEC uses an older network [25]. In such cases, it is important to synchronize timestamps between the RGB camera and the event camera. This synchronization typically requires specialized hardware. We verify that timestamps are accurately synchronized in M3ED, but we observe a noticeable misalignment in the timestamps of the RGB camera and the event camera in DSEC. See Fig. 5d. Pseudo-labeled segmentation masks are supported on a slightly different set of pixels than the events (left). Therefore, the F^3 -based network trained on M3ED obtains a slightly poor mIOU/accuracy on DSEC, see Fig. 5a (bottom). When F^3 is trained on DSEC in Fig. 5a (top), the mIOU is much higher—by more than 6%. However, as Fig. 5d (right) shows, this improvement could be spurious, because the predictions are spatially misaligned with the events. This phenomenon is also evident in the relative performance of ESS [21] in Fig. 5a.

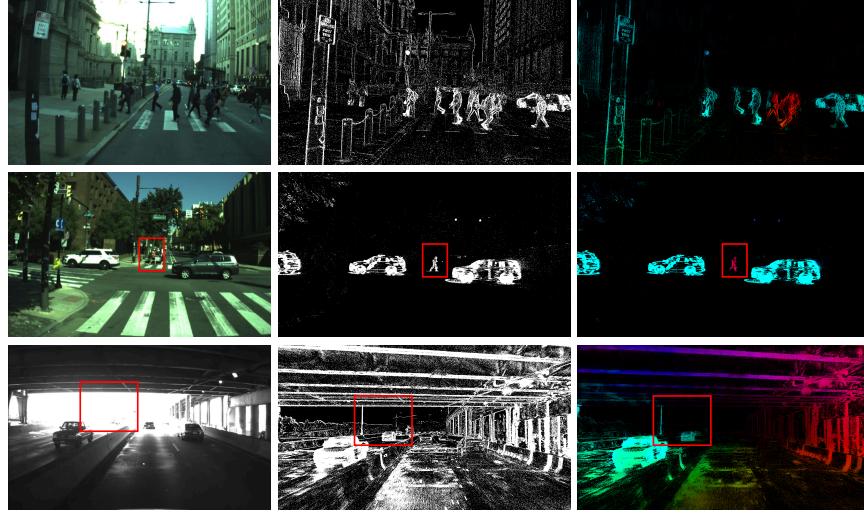
2.4 Unsupervised optical flow estimation

We next evaluate whether F^3 encodes spatiotemporal information in the scene that arises from the motion of underlying edges and textures. Unsupervised optical flow estimation is a good downstream task for this purpose. Optical flow estimation for RGB images is usually performed by enforcing brightness consistency across successive frames [34]. This technique is difficult to use for event data due to sparsity and noise [26]. Existing works, therefore, use deblurring losses on raw events [35, 36], supervision [29], or RGB-based frame-based matching [37] to work around this issue. We instead develop an unsupervised procedure for estimating optical flow from event data that enforces brightness constancy on our p -channel F^3 .

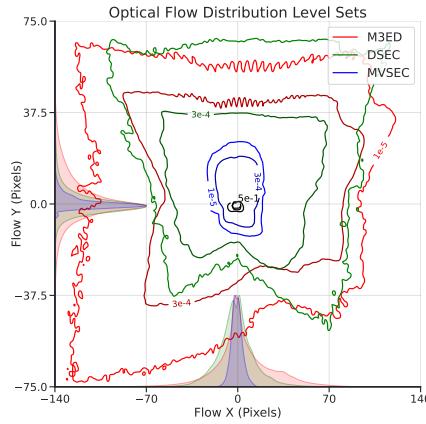
Suppose we have a neural network $\psi : \mathbb{R}^p \times \Omega \rightarrow \mathbb{R}^2 \times \Omega$ that takes $F^3(t, \cdot)$ as the input and predicts the optical flow $v(t, u) \in \mathbb{R}^2$ at time t for all pixels $u \in \Omega$. We first compute Gaussian pyramids $F_\sigma^3(t, u)$, $F_\sigma^3(t + \Delta t, u)$ and $v_\sigma(t, u)$ for multiple spatial scales $\sigma = 1, 2, \dots$ by smoothing and bilinear down-sampling by a factor of two at each scale. This provides the flow prediction network with a global context and helps with sparse structures in the scene, such as edges that can undergo large spatial displacements. We fit ψ to enforce brightness consistency at each scale, along with some regularization using an objective

$$\ell_{t, \Omega}(\psi) = \frac{1}{|\Omega|} \sum_{\sigma} 2^{2\sigma} \sum_{u \in \Omega} \ell_c \left(\frac{1}{\sqrt{Z}} F_\sigma^3(t, u) - F_\sigma^3(t + \Delta t, u + v_\sigma(t, u)) \right) + \lambda R(\Delta u), \quad (11)$$

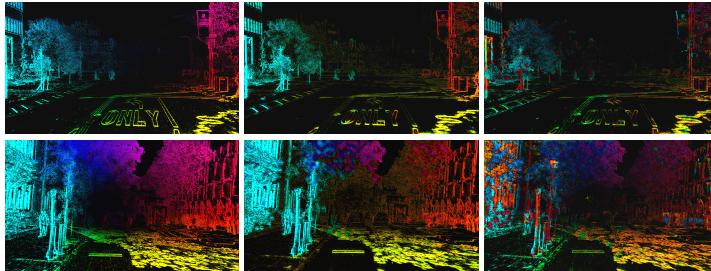
where the denominator $Z \in \mathbb{R}^n$ with the i^{th} element $Z_i = \sum_u (F_\sigma^3(t, u))_i^2 + (F_\sigma^3(t + \Delta t, u))_i^2$ performs normalization. The square root and division are interpreted element-wise. The Charbonnier loss $\ell_c(x) = n^{-1} \sum_c (x_c^2 + \epsilon^2)^\beta$ helps with outlier rejection [38]. Local approaches such as this, including ones for RGB images, are susceptible to the aperture problem and cannot recover optical flow in spatial regions without



(a) Qualitative results on daytime urban driving data in M3ED



(b) Optical flow for different sensors and environments are quite different, especially M3ED and DSEC



(c) Optical flow estimates based on F^3 (left), V^3 (middle) and I^3 (right).

Method	Latency (ms)	Pixel error			
		45 Hz ground-truth		11.25 Hz ground-truth	
		3PE (%)	AEE	3PE (%)	AEE
MultiCM [26]		0.21	0.28	5.68	1.05
EV-FlowNet [27]	4.1	0.00	0.32	9.70	1.30
EV-MGRFlowNet [28]		0.02	0.28	6.22	1.10
$I^3(\Delta t = 50)$	1.95	5.14	1.27	34.09	3.10
$V^3(\Delta t = 50)$	1.88	1.90	0.97	34.91	3.09
$F^3(\Delta t = 50)$	2.5	0.23	0.36	10.83	1.50
E-RAFT* [29]	48.75	1.70	0.24	1.12	0.72

(d) MVSEC (outdoor_day1)

Method	Latency (ms)	Pixel error			Angular error
		3PE (%)	AEE	AAE (deg)	
MultiCM [26]	10^4	30.86	3.47	13.98	
VSA-SM [30]		16.83	2.22	8.86	
TamingCM [31]		17.77	2.33	10.56	
MotionPriorCM [32]	7.27	15.21	3.20	8.53	
$I^3(\Delta t = 50)$	2.89	73.19	9.59	43.90	
$V^3(\Delta t = 50)$	2.72	64.23	8.09	40.67	
$F^3(\Delta t = 50)$	3.92	27.93	2.92	9.15	
E-RAFT* [29]	52.45	2.68	0.79	2.85	

(e) DSEC (10 Hz ground-truth)

Train & Test dataset	Method	Latency (ms)	Pixel error			Angular error
			3PE (%)	AEE	AAE (deg)	
Car Daytime	I^3	7.45	78.64	12.21	41.77	
	V^3	7.59	82.01	10.99	40.18	
	F^3	14	52.05	5.35	12.58	
Spot	V^3	7.59	95.55	22.83	31.63	
	F^3	14	69.95	9.65	15.10	
Falcon	V^3	7.59	91.02	11.83	31.59	
	F^3	14	64.51	7.43	14.16	

(f) M3ED (10 Hz ground-truth)

Figure 6: (a) Quantitative results on different types of urban driving scenes indicate that F^3 -based optical flow estimates (right column) are accurate despite (i) multiple pedestrians moving in different directions in the same region (top row); (ii) very fast-moving objects (cars in middle row) versus slower-moving objects (running pedestrian in middle row); (iii) and highly saturated regions in the RGB data. (b) Due to differences in sensor resolutions, the typical motion patterns of different robotic platforms, the magnitude of ground-truth optical flow can vary enormously. A contour at 0.5 indicates that flow is within that region for 50% of the data. For low-resolution data in MVSEC, the flow is larger along the Y-axis, but not

Figure 6: (continued) more than about 30 pixels. For higher resolution MVSEC and M3ED data, the magnitude of flow is much larger overall, but with skewed probability distributions. **(c)** A qualitative comparison of F^3 -based optical flow against our baselines such as V^3 and I^3 that corroborates the quantitative metrics in Figs. 6d to 6f. F^3 -based flow is spatially consistent, especially in terms of the flow direction, and identifies detailed structures. All three approaches use the same brightness consistency loss and regularization to compute the flow. The fact that F^3 -based flow is significantly better indicates that F^3 encodes motion (flow) and structure (which is required for accurate matching across time) more faithfully. **(d-f)** Quantitative results on daytime urban driving data from MVSEC, DSEC and M3ED. Optical flow estimates are evaluated against the ground-truth computed using LiDAR-based depth (see Sec. 4). Depending upon the data, this ground-truth is available at different frequencies (45 Hz and 11.25 Hz for MVSEC, 10 Hz for DSEC/M3ED). 3PE refers to the fraction of pixels with absolute flow error larger than 3; AEE refers to the average endpoint error (in pixels); AAE refers to the average angular error as suggested in [33]. As is common in the literature, on MVSEC and M3ED, we evaluate the flow estimate only on pixels containing events; for DSEC, flow is evaluated at all pixels. F^3 -based optical flow is (i) very fast, e.g., 14 ms latency even on HD data in M3ED, and less than 4 ms latency on VGA and lower resolutions; (ii) performs comparably to more specialized existing techniques across different metrics; (iii) can be trained robustly across different platforms in M3ED, and also generalizes across illumination conditions without re-training (see Fig. 8b).

features. These issues are often tackled using regularization of the form

$$R(v) = \frac{1}{|\Omega|} \sum_{\sigma} 2^{2\sigma} \sum_{u \in \Omega} \sum_{u' \in \Omega_u} \ell_c(v_{\sigma}(t, u') - v_{\sigma}(t, u))$$

to smooth the predictions [39].

Fig. 6a provides a qualitative understanding of our approach for optical flow estimation. A numerical comparison is shown in Figs. 6d to 6f on MVSEC, DSEC and M3ED data, respectively. Ground-truth optical flow was estimated for this data by masking out independently moving objects using LiDAR. The F^3 -based unsupervised approach is comparable to state of the art methods, particularly in terms of the average end-point error (AEE). Unsupervised approaches, including ours, do not compare favorably to supervised approaches like E-RAFT [29] because the latter predicts dense flow even at pixels without any events. In MVSEC (Fig. 6d), where the flow is evaluated only at pixels with events, E-RAFT performs about as well as unsupervised approaches. Unsupervised approaches have much smaller latency than supervised approaches—which is especially important for an application like optical flow. Representations such as voxel grids V^3 and frames I^3 work quite poorly for this task. This is because, unlike F^3 , they are not sufficiently denoised and thereby difficult to match across time. As one would expect, the V^3 -based approach is better than the one with I^3 , because the former retains temporal information in the events, while the latter only retains their polarity.

Different unsupervised flow estimation methods can be characterized in terms of (i) the representation of event data (e.g., F^3), (ii) the operation used to warp features (which is linear in Eq. (11)), and (iii) the loss used to compare warped features (Eq. (11)). MultiCM [26] uses a simple representation (raw events) but a sophisticated loss (contrast maximization). MotionPriorCM [32] uses a simple representation (voxel grid), but a spline-based warp to address situations when the brightness constancy assumption is invalid, and a contrast maximization-based loss. This is useful for situations like DSEC and M3ED, where ground-truth optical flow is computed only at 10 Hz. As noted in [26], 20% of the ground-truth flow has more than 22 pixel displacement in DSEC. VSA-SM [30] uses a more sophisticated representation, a multi-scale encoding of time-surfaces [40, 41], and a piece-wise linear warp. Our innovation in the context of optical flow focuses fundamentally on the representation (F^3). And the fact that we can predict optical flow faithfully with a small convolutional network with $\sim 28,000$ parameters, a linear warp, and a brightness consistency loss, suggests that the representation F^3 encodes local motion faithfully. For example, although MotionPriorCM has a smaller 3PE, the average endpoint error (AEE) for F^3 is better. An important benefit of our approach is that F^3 -based optical flow estimation is about 2.5 times faster than existing approaches, see Fig. 6.

2.5 Monocular depth estimation

For camera trajectories with no changes in roll, pitch, or translation along the yaw axis, optical flow at a sufficiently large number of pixels on the image plane can provide depth up to a global scale [34, Section 5.4.4]. Monocular relative depth estimation from events is therefore a well-posed problem, unlike RGB-based monocular depth estimation. Estimating metric monocular depth, however, is still ill-posed because the global scale depends on translational velocity.

To demonstrate that F^3 retains fine-grained information about motion and structure in the scene, we develop a procedure to estimate metric monocular depth. We adapt a pre-trained relative depth network (Depth Anything V2 Base model [46] with 97.5 M parameters) by tiling and copying weights in the first layer to take p -channel F^3 as input instead of RGB images. One could directly fine-tune this network on depth computed from LiDAR but this is non-trivial for two reasons: (i) for many event camera datasets, the amount of LiDAR data is relatively small (see Fig. 2), and (ii) LiDAR depth measurements are very sparse when re-projected onto high resolution event camera frames leading to inaccurate object boundaries and blurry predictions, see Fig. 7b. A good alternative is to (i) first fine-tune a F^3 -based network to predict dense pseudo-labeled disparity computed from RGB images (e.g., using a pre-trained DepthAnything V2 Large model), and then (ii) fine-tune to predict dense metric disparity computed from LiDAR (assuming known focal length and baseline).⁴

Suppose we have a neural network $\psi : \mathbb{R}^p \times \Omega \rightarrow \mathbb{R} \times \Omega$ that takes $F^3(t, \cdot)$ as input and predicts the disparity $d(t, u) \in \mathbb{R}_+$ at time t for all pixels $u \in \Omega$. Let $d_{\text{pseudo}}^*(t, u)$ denote the pseudo-labeled disparity. Let $d_{\text{metric}}^*(t, u)$ denote metric disparity computed from LiDAR, which is only available on a subset of the pixels, say $\Omega' \subset \Omega$. We will work with normalized disparity $\tilde{d}(t, u)$ obtained by subtracting the median disparity from $d(t, u)$ and dividing by the average deviation from the median across Ω [47]. The objective used to fit ψ in the first stage is

$$\ell_t(\psi) = \frac{1}{|\Omega|} \sum_u \left| \tilde{d}(t, u) - \tilde{d}_{\text{pseudo}}^*(t, u) \right| + \lambda R(\tilde{d}, \tilde{d}_{\text{pseudo}}^*). \quad (12)$$

The first term computes the discrepancy between the prediction $\tilde{d}(t, u)$ and the pseudo-labeled disparity $\tilde{d}_{\text{pseudo}}^*(t, u)$. The regularizer

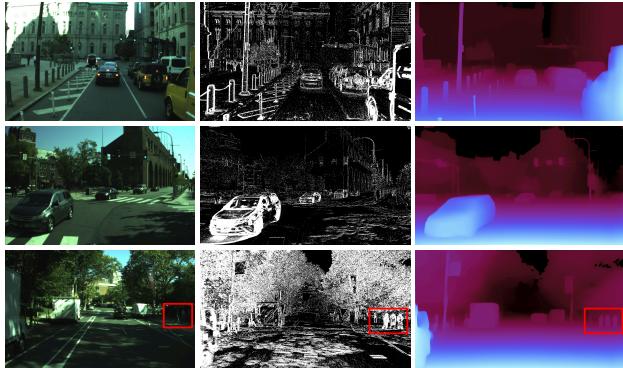
$$R(d, d') = \frac{1}{|\Omega|} \sum_{\sigma} 2^{2\sigma} \sum_u \left\| \nabla_u d_{\sigma}(t, u) - \nabla_u d'_{\sigma}(t, u) \right\|_1. \quad (13)$$

encourages accurate object boundaries by matching the gradient of the predictions with the gradient of the pseudo-labeled disparity from RGB images. Averaging across spatial scales σ (computed by down-sampling by a factor of two, without smoothing) encourages the network to build global context. Suppose the predicted disparity after the first stage is $d_{\text{stage-1}}(t, u)$. This is expected to predict object boundaries accurately, but may have an incorrect scale. The objective minimized in the second stage is

$$\ell_t(\psi) = \frac{1}{|\Omega'|} \sum_u \left(\log \frac{d(t, u)}{d_{\text{metric}}^*(t, u)} \right)^2 - \frac{1}{2|\Omega'|^2} \left(\sum_u \log \frac{d(t, u)}{d_{\text{metric}}^*(t, u)} \right)^2 + \lambda R(\tilde{d}, \tilde{d}_{\text{stage-1}}) \quad (14)$$

The first two terms are the scale-invariant ‘‘SiLog’’ loss between the predicted and metric disparity [48]. The third term is again a regularizer that encourages sharp object boundaries as predicted by $\tilde{d}_{\text{stage-1}}$. We could have used $R(\tilde{d}, \tilde{d}_{\text{pseudo}})$ as the regularizer in the second stage, but this can result in temporal aliasing. In practice on a robotic platform, RGB images and LiDAR data are often acquired at slightly different time instants. The estimate $d_{\text{stage-1}}(t, u)$ can be queried at the exact instant t that matches the LiDAR timestamp, as opposed to

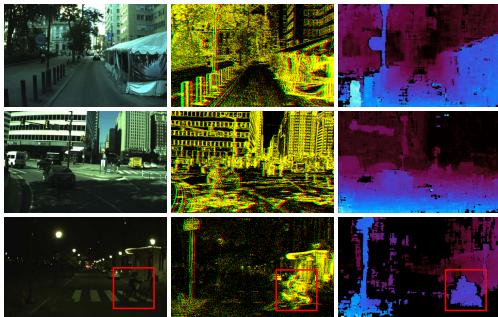
⁴For datasets such as MVSEC with low-resolution event cameras, LiDAR measurements are sufficiently dense when re-projected on the pixel-space and we forgo the first stage.



(a) Qualitative results on daytime urban driving in M3ED



(b) Sharp object boundaries (right) with gradient-based regularization



(c) Stereo disparity (unsupervised)

Method	Relative	RMSE	Absolute error below		
	ℓ_1 error	(m)	10 m	20 m	30 m
MDDE [42]	0.47	9.89	2.62	3.23	3.80
DTL [43]	0.43	-	2.31	3.01	3.59
EReFormer [44]	0.29	-	1.41	2.21	2.79
SSL [45]	-	-	2.44	3.12	3.79
$I^3(\Delta t = 50)$	0.34	7.27	1.71	2.62	3.11
$V^3(\Delta t = 50)$	0.32	7.17	1.57	2.45	2.91
$F^3(\Delta t = 50)$	0.31	6.29	1.51	2.43	2.76

(d) MVSEC (outdoor_day1 and outdoor_night1)

Train & Test	Method	Latency (ms)	Relative	RMSE	Pixels (%) with δ below		
			ℓ_1 error	(m)	1.25	1.25 ²	1.25 ³
Car Daytime	I^3	12.94	0.26	7.84	0.69	0.84	0.94
	V^3	13.51	0.17	5.88	0.77	0.92	0.98
	F^3	21.66	0.09	3.15	0.91	0.99	1.00
Spot	I^3	12.94	0.18	3.04	0.77	0.91	0.95
	V^3	13.51	0.18	3.09	0.78	0.90	0.92
	F^3	21.66	0.16	2.80	0.79	0.93	0.98
Falcon	I^3	12.94	0.21	4.17	0.68	0.87	0.95
	V^3	13.51	0.19	3.53	0.75	0.93	0.98
	F^3	21.66	0.17	3.44	0.72	0.93	0.99

(e) M3ED

Method	Gradient regularization	1PE	2PE	MAE	RMSE
		(%)	(%)		
SSL [45]		84.92	70.47	4.95	6.27
I^3		65.25	40.94	2.47	3.59
V^3	✓	62.44	37.59	2.21	3.26
$F^3(\bar{d}_{\text{stage-1}} \text{ on M3ED})$		60.60	34.91	2.07	2.99
I^3		64.33	40.48	2.52	3.71
V^3	✗	62.84	37.9	2.25	3.27
$F^3(\bar{d}_{\text{stage-1}} \text{ on M3ED})$		61.06	35.94	2.12	3.07
F^3		59.38	34.28	2.07	3.03

(f) DSEC (disparity)

Figure 7: (a) Qualitative demonstration on daytime urban driving scenarios in M3ED suggests that F^3 -based monocular depth estimates (color indicates magnitude) (i) accurately predict depth discontinuities around objects of different sizes at different distances from the camera in the scene, e.g., multiple pedestrians in the bottom row, traffic lights in the middle row and bollards in the top row; (ii) predictions are accurate even in extreme far-field regions (sky); (iii) depth prediction is robust to spurious events (shadow of the car in middle row); (iv) depth at pixels corresponding to the trees varies smoothly even if the events exhibit high-frequency texture. (b) Gradient-based regularization in Eq. (13) encourages accurate object boundaries by matching the gradient of the predictions with the gradient of the pseudo-labeled disparity from RGB images. (c) Stereo disparity (right column) computed using F^3 features derived from left and right stereo event cameras in M3ED (overlaid on top of each other in the middle column); color indicates magnitude, with black indicating low-confidence regions where matching was unreliable. In the third row, the road sign and the bicycle are barely visible in the RGB image at night but yield confident disparity estimates. (d, e, f) Quantitative evaluation across different data using standard metrics in the literature. In Fig. 7d, relative ℓ_1 error is the average (over pixels) of $|d/d^* - 1|$ where d and d^* are the predicted and true metric depth at a pixel respectively; RMSE standards for root mean-squared error in meters; average absolute error $|d - d^*|$ is reported for pixels with ground-truth depth below 10m, 20m and 30 m. In Fig. 7e because the scenes are quite diverse, we report the fraction of pixels (%) for which $\delta = \max(d/d^*, d^*/d)$ is below different thresholds. For Fig. 7f we evaluate the disparity; 1PE and 2PE stand for 1- or 2- pixel error and MAE stands for the mean-absolute error. Across a broad diversity of environments and platforms, F^3 -based monocular depth estimates are more accurate than existing approaches (for MVSEC they are comparable to EReFormer) which often use extra synthetic training data (MDDE) or RGB information (DTL). On high-resolution M3ED data, the RMSE of F^3 -based depth estimation is about 3.5 m, suggesting that these estimates can be fruitfully used for navigation tasks. In Fig. 7f, we noticed that while gradient-based regularization led to slightly worse metrics, the actual predictions in Fig. 7b were much more accurate. This is because in DSEC, depth estimates are evaluated only on pixels with LiDAR measurements.

$\tilde{d}_{\text{pseudo}}$, which is constrained to the timestamp of the RGB image.

Fig. 7 shows qualitative and quantitative monocular metric depth estimation results on M3ED, DSEC and MVSEC data. Although there is no prior work on this task using the former two datasets, there are some existing studies on stereo disparity using DSEC. Across a variety of these experimental settings, F^3 -based metric depth estimates are quite accurate, often better than the best existing approaches such as EReFormer [44]. The root mean square error (RMSE) is about 3 m on M3ED, and about 6 m on MVSEC. On MVSEC data, our approach performs essentially comparably to approaches like RAM Net [49] and EvT⁺ [50] that use both RGB images and events in daylight conditions (not shown in Fig. 7d). Training F^3 on DSEC (last row) does not significantly improve predictions compared to the version trained on M3ED, which suggests that F^3 features can robustly predict depth. Our F^3 -based approach retrieves sufficient information about object boundaries even from only event data.

For this task, a V^3 -based network is much more effective than an I^3 -based network. But similar to Sec. 2.3, the V^3 -based approach does not generalize as well as the F^3 -based approach, even if the two perform similarly on the training data (not shown here). This can be attributed to the robustness of F^3 to nuisances in the event data. From Fig. 7, we see that the naive voxel-grid-based representation of events V^3 is already comparable to state-of-the-art approaches like EReFormer, and better than others like MDDE and DTL.

2.6 F^3 -based approaches work robustly across robotic platforms, lighting conditions, dynamic vision sensors, and event rates

Robotic platforms To demonstrate that F^3 is effective for different types of robotic platforms, we demonstrate optical flow and depth estimation tasks on data collected from a quadruped robot named Spot from Boston Dynamics and a custom-built flying platform named Falcon 450 [51]). We use the same training strategy and hyperparameters as those for urban daytime driving sequences in Secs. 2.3 to 2.5. Figs. 6f and 7e show that the F^3 -based approach is equally capable for other platforms as for a car driving in urban settings. In Fig. 6f, for optical flow we have 52.1 3PE and 5.4 AEE for Car Daytime compared to 64–69 3PE and 7.4–9.7 AEE for Spot and Falcon. In Fig. 7e, for depth estimation, we have 0.09 absolute relative error and 3.15 RMSE for Car Daytime compared to 0.16–0.17 absolute relative error and 2.80–3.44 RMSE for Spot and Falcon. Optical flow and depth estimation tasks for urban driving are very different from those for quadruped robots or flying platforms. Periodic gaits in quadrupeds and attitude changes in the flying platform lead to a large optical flow in the entire frame. Distance of objects in the scene from the platform is much smaller for quadrupeds, a bit larger for the flying platform, and largest for urban driving. Therefore, it is remarkable that our approach for pre-training F^3 on events and predicting these downstream tasks is effective across these diverse settings.

Fig. 8a gives a qualitative understanding of optical flow and depth estimates when an F^3 -based network trained on daytime urban driving data is evaluated on indoor data from Spot and outdoor data from Falcon. This is a challenging evaluation because these scenes contain objects of very different sizes and categories viewed from quite different viewpoints. Such zero-shot transfer is also challenging due to spurious events on the floor/asphalt/bookshelf and reflective patterns (grid on the barrel). Note the detailed structure of the pipes, cars, and pedestrians in the depth estimates, and variations in the flow across the field of view, e.g., the Spot robot is traveling forwards in the top row but turning left in the second row, Falcon is going left in the top row and right in the bottom row.

Lighting and environmental conditions The large dynamic range of event cameras is one of the main attractions for roboticists. We therefore evaluate our event representation across large illumination changes. An F^3 trained to predict optical flow on daytime urban driving data discussed in Fig. 6f can predict very well even on nighttime driving sequences without any additional training. In terms of pixel error, we get 59.4 3PE and 6.5 AEE in Fig. 8b compared to the original 52.1 3PE and 5.4 AEE in Fig. 6f. The angular error is similar

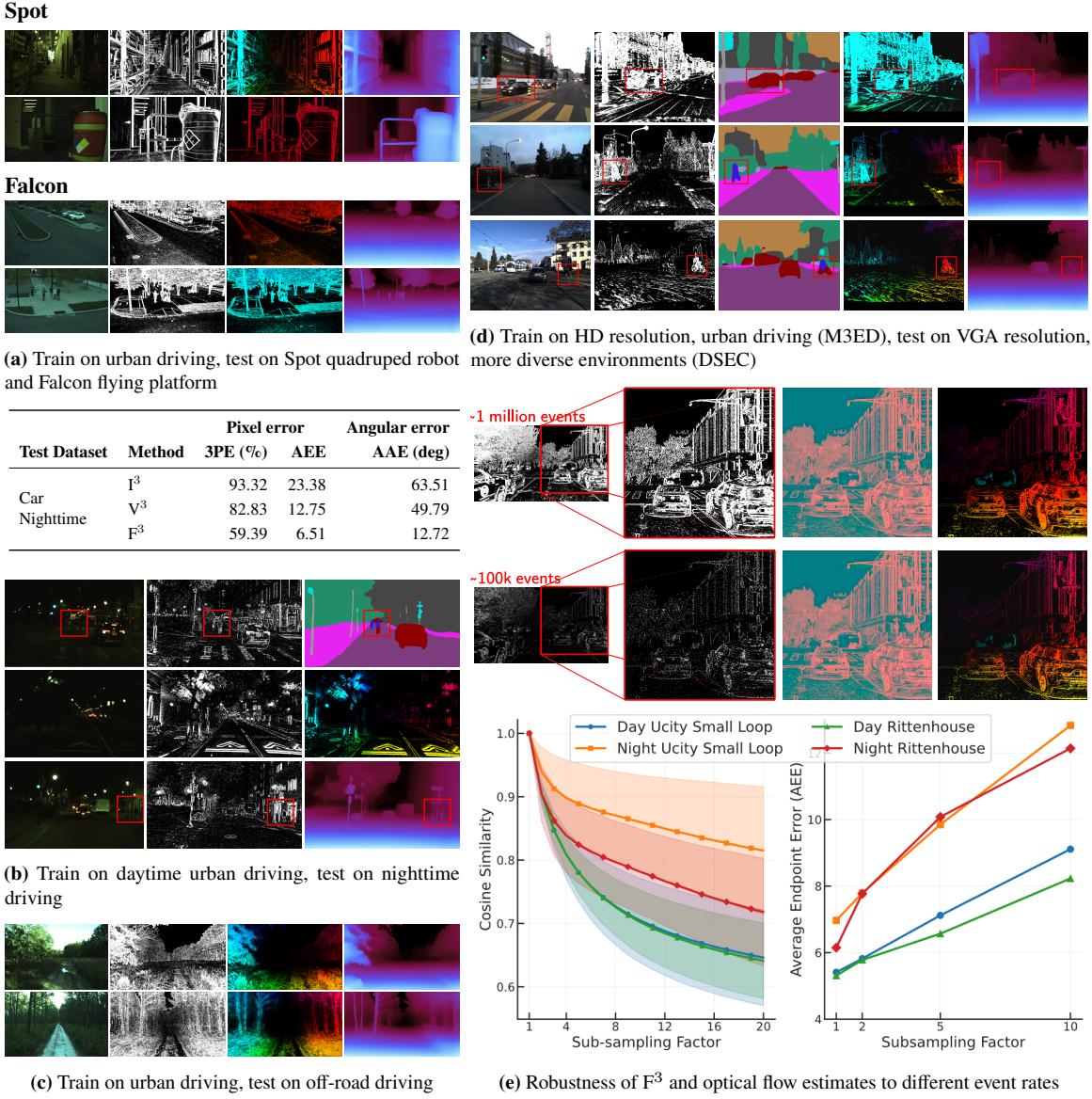


Figure 8: Robustness across robotic platforms, lighting conditions, dynamic vision sensors and event rates **(a)** RGB (first column), events (second), F³-based networks for optical flow (third column) and depth estimation (fourth column) trained on daytime urban driving data continue to work robustly (zero-shot) on event data from a quadruped robot (top) and a flying platform (bottom) even if the typical objects in the environment and viewpoints are quite different. **(b)** F³-based approaches for segmentation (top row), optical flow (middle row) and depth estimation (bottom row) trained on daytime urban driving data work robustly for nighttime driving. There are very large differences in the typical event rates across these settings. The table shows that a quantitative evaluation of optical flow estimation is comparable to daytime results in Fig. 6f. **(c,d)** F³-based approaches trained on HD resolution urban driving data in M3ED work robustly and effectively, without any additional training, on off-road driving data **(c)** where the scene statistics are quite different, as also on a different dataset (DSEC, **d**) which consists of driving in urban areas, mountains, and lakes and VGA resolution data. **(e)** The top panel shows F³ (top three PCA components, in the third column) can capture the essential structure of the scene even if the event rate is only 10% of what was available during training (see second column). The estimated optical flow in the fourth column is near-identical. For different sub-sampling rates, the bottom left and right panels show the cosine similarity of F³ and the average endpoint error (AEE) on the test set (without any sub-sampling), respectively, in daytime and nighttime driving scenarios.

(12.7 versus 12.6). This demonstrates that F^3 -based predictors are robust to lighting changes without any additional re-training or fine-tuning.

Fig. 8b provides a qualitative understanding of the performance of F^3 for segmentation (top row), optical flow (middle) and depth estimation (bottom) tasks. The RGB images (left column) are extremely dark and unrecognizable. However, the event camera depicts a remarkable amount of structure in the scene (middle column). Our F^3 -based approach predicts perfectly even in these challenging situations, e.g., notice the segmentation of the bicycle (top middle), the optical flow on the tree and road signs (middle) and the sharp boundaries of the lamp-poles (bottom). It is interesting to observe that F^3 rejects a large amount of noise due to street lamps, that is quite common in nighttime event data in the bottom row.

Fig. 8c demonstrates how F^3 -based networks can generalize to new environments (trained on urban driving and evaluated in forested areas). A large number of events are triggered due to heavy vegetation here (first column) and it is difficult for the human eye to discern structure from the event frame (second column). A number of practical issues also crop up in these cases, e.g., saturation of the bandwidth between the event camera and the robotic platform due to extreme event rates. Despite these issues, optical flow (third column) and depth estimates (fourth column) are quite accurate. Notice the sharp boundaries at the branches and tree trunks for both flow and depth, accuracy of the depth estimates at pixels corresponding to the sky, and smooth depth gradient on the forest floor.

Dynamic vision sensors and event rates In the prequel, we saw a number of examples of how F^3 -based approaches are effective across different dynamic vision sensors (across MVSEC, DSEC and M3ED) using the same training objectives and hyperparameters. In fact, F^3 -based networks can predict accurately on data from different event sensors without any additional training or fine-tuning. Fig. 8d provides some qualitative examples of networks trained for segmentation, optical flow and depth estimation on M3ED daytime driving data collected from a larger Prophesee EVK4 1280×720 resolution sensor evaluated on data from DSEC (a smaller 640×480 resolution sensor). Note the heavily saturated regions in the event frame due to fast-moving objects (top row), objects in the foreground blurring into the background (middle row), and objects that are impossible to discern in the RGB images (bottom row). Accurate performance in these scenarios demonstrates the remarkable robustness of F^3 -based approaches. These examples hint that pooling data across different robotic platforms and environments would further improve the robustness of F^3 -based approaches.

Beyond differences in properties of the sensors, working with event data is also difficult because different types of motion and lighting conditions can lead to much fewer/larger events than typical rates. A good representation of event data should be insensitive to sensors and handle different event rates. Fig. 8e shows that F^3 features are very similar even after significantly sub-sampling the events in a scene. Even with only 25% of the original events, F^3 features retain a cosine similarity of more than 0.8 to those obtained from the original event stream. We show qualitative examples of event frames, their sub-sampled counterparts, and the PCA of F^3 features for both. Although the event frames with 90% of the events dropped look significantly less informative than the non-sub-sampled frames, the PCA of their F^3 features look very similar. This suggests that we can aggressively sub-sample the event stream before computing F^3 features or running the downstream predictors, without losing performance significantly. A quantitative analysis of the average endpoint error (AEE) on the optical flow M3ED test set for different sub-sampling rates is shown in the bottom right panel. The AEE degrades gracefully as the event rate is reduced. Even with only 50% of the original events, the AEE degrades less than 10%.

3 Discussion

Implications for building effective representations of event data A representation is a statistic of the data that is sufficient for the task. A good representation is minimal in the sense that it discards information that is irrelevant for the task [52]. These broad principles for representation learning have been specialized to different kinds of data, e.g., to argue that convolutional and pooling layers build insensitivity to local group transformations of images [16], and to build neural architectures that model different kinds of symmetries [53, 54]. Foundation models for RGB images, such as I-JEPA [55], DiNO [56], or MAEs [57], are a counter-thread to this line of work; they are highly general representations learned from data by forcing the features to be insensitive to transformations of the data. Today, state-of-the-art approaches across computer vision belong to one of these two paradigms. This paper is a step towards similar broad principles and algorithms for representations of event-camera data. It specializes the key idea behind foundation models for spatiotemporal data like V-JEPA [58], Video MAEs [59]—predicting future data from past data—to event data. Our mathematical argument is also specialized to event data. It suggests that the ideal representation for such noisy spatiotemporal data is obtained by building a data-dependent basis for projecting the data, and then modeling the dynamics of this representation over time.

Event data is asynchronous, noisy, sparse and needs to be processed with a low latency in spite of its high volume. This is a very challenging design space and different algorithms and implementations work around this in different ways. Time-surfaces [40, 41, 60, 61], voxel-grids [62, 63], and event frames modify events to conform to dense, grid/frame-based inputs required by deep networks. Our baseline features V^3 and I^3 are designed to emulate them. These are simplistic representations of event data. Some, such as voxel grids—which have become a de facto choice for tasks ranging from video reconstruction to monocular depth prediction [64]—are fast to construct, but leave a large memory footprint. They do not exploit the sparsity of events, and as our experiments in this paper show, they are more susceptible to noise. Others, such as time surfaces, use filter banks that are either crafted by hand or use simple bases, which still do not exploit the structure in event data or explicitly handle noise. Histograms [65] or multi-channel images encoding polarity and recent timestamps [66] sacrifice the precise timing of events and cannot correctly resolve overlapping structures and motion. Recent work on asynchronous architectures treats individual events as nodes in a spatiotemporal graph [67]. There is also work on asynchronously-computed event features that are updated recursively and can be queried at any point in time [68], which bears some resemblance to F^3 . Some other lines of work are exploring learned [69] or optimally structured [70] feature representations, indicating a trend away from hand-crafted data structures.

The architecture for F^3 is tailored from the ground up to the unique properties of event data. The hash-encoder φ in Eq. (8) can be computed asynchronously and quickly for every event (even without a GPU). Averaging across time and the focal loss in the training objective makes the learned basis in the hash-encoder resilient to noise. The F^3 architecture exploits sparsity because it only executes on events.⁵ F^3 is a multi-channel image representing all events in a given spatiotemporal volume. We should emphasize three properties. First, $F^3(t, u)$ can be queried at any time t . It is a continuous-time representation, which makes it very useful for tasks such as optical flow and depth estimation (and related ones such as estimating the time-to-collision). Although we did not implement this here, F^3 can be updated incrementally after each event with some bookkeeping. Second, a large majority of computer vision algorithms and architectures are designed for three-channel images. In contrast to many existing event representations, F^3 can be plugged into any of these existing approaches. As we showed in this paper, in many cases, deep networks trained on RGB data can be fine-tuned to use events with F^3 . Third, we chose the hyperparameters of F^3 (receptive field of the convolutions) to be more local to focus on geometric vision tasks such as optical flow estimation, and to be

⁵Our current implementation uses dense 2D convolutional layers. This is because dense convolutions in PyTorch are slightly faster than sparse convolutions; we expect this to change very soon.

able to compute the features quickly. The F^3 architecture would work as-is for building more global features (e.g., with more convolutional layers and predicting larger future patches). This would lead to features with richer semantics like the ones in the visual cortex.

Implications for robotics Vision is ideal for building real-time perception for robots that need to operate in different environments. It is low-power and lightweight, has a large field of view, and can be used for a broad range of tasks with learning-based techniques. Event cameras are asynchronous sensors and work across very different lighting conditions. So, they significantly expand these capabilities of operating in low-light and low-latency settings. However, there are enormous engineering challenges in processing event data that make it difficult to build real-time perception systems on SWaP-constrained platforms (it is difficult to even store event data in real-time [8]).

Although there is a large body of work on event-based odometry and simultaneous location and mapping (SLAM) [71–73]—the Results section discussed many approaches specific to segmentation, optical flow and monocular depth estimation—a majority of these methods cannot provide real-time perception. There have been some demonstrations of real-time event-based perception in robotics for obstacle avoidance [74], navigation [75], landing [76], or object tracking and grasping [77–80]. These approaches do showcase the remarkable utility of event-based sensing, but they are specific to these demonstrations and have not yet been shown to generalize to different robots, tasks, or environments. This paper makes progress along two key directions in this context.

First, F^3 is a general representation for event data. They obtain state-of-the-art performance on a variety of tasks, but more importantly, the same pre-trained F^3 representation is used to perform different tasks in different environmental settings (daylight vs. night-time, indoors vs. outdoors, urban vs. off-road), and across different robot platforms (car, a quadruped robot and a flying platform). Tasks in these environments have quite different requirements (a large number of small/big objects in urban driving, vs. small objects from high up on the flying platform, vs. rapid change of view on a quadruped robot). This paper is a step towards real-time and generalizable event perception for different robot embodiments.

Second, both CPUs and GPUs are ill-suited to handling low-latency, high-throughput and sparse event data. F^3 exploits the sparsity of event data and can be implemented extremely efficiently. We showed that we can compute F^3 at 120 Hz and 440 Hz on HD and VGA resolutions, respectively, using a desktop GPU (NVIDIA RTX 4090). This efficiency of F^3 leads to downstream task predictions at 25–75 Hz at HD resolution. Our work brings event camera processing computationally at par with the latest computer vision approaches in robotics. It shows that event processing can be both fast and effective on commodity hardware. Neuromorphic computing [81] or ASICs that perform computation directly at the pixel [82] can further improve the practical performance and energy usage of F^3 .

Finally, the approach presented here benefited enormously from stereo RGB, LiDAR, internal measurement unit (IMU), and event camera data captured using multi-sensor platforms across different types of robots. Such cross-modal supervision is promising because annotating event data for supervision on downstream tasks is challenging, both due to its volume and unusual nature. Transferring annotations from existing data is easy and effective, provided appropriate care is taken to synchronize timestamps and accurate extrinsic calibration. This paper demonstrates some remarkable outcomes that could be “game changers” in robotics, e.g., dense depth estimates or semantic segmentation at frame rates much faster than LiDAR or RGB cameras in Fig. 7 or recognizing essentially invisible objects (in RGB data) at night-time in Fig. 8 without any training in such scenarios.

4 Materials and Methods

Details of the F^3 architecture

As discussed in Sec. 2.2 and Fig. 4, we use a φ that can learn multi-scale spatiotemporal features of individual events. The L scales consist of independent grids, with resolutions chosen as a geometric progression from $R_{\min} \in \mathbb{R}^3$ to $R_{\max} \in \mathbb{R}^3$. For instance, when Ω corresponds to an HD pixel space and $\Delta t = 20$, we select $R_{\max} = (8, 8, 1)$ and $R_{\min} = (180, 320, 8)$. Similar to the 2D illustration of φ in Fig. 4, for each level, the feature of an event coordinate (s, u) is a trilinear interpolation of the features of the 8 closest vertices enclosing it at the resolution R_l of a level l . These F -dimensional interpolated features are thus independent at each level. Concatenating feature vectors across levels gives $\varphi(s, u) \in \mathbb{R}^n$ with $n \equiv LF$. For coarse resolutions, the number of vertices, $\prod_{i=1}^3 (R_{li} + 1)$, is usually smaller than the size of the hash table, say, T , which ensures that each vertex has a unique feature vector. For finer resolutions, the number of grid points at a level can exceed T . In this case, we can use a hash function $h : \mathbb{Z}_+ \times \Omega \rightarrow \{1, \dots, T\}$ as suggested in [15] to index into the hash table. There is no need to address hash collisions explicitly because the training process disambiguates them.

Based on the discussion around Eq. (8), we want ρ to be a universal approximator acting on $\bar{\varphi}(t, \cdot)$ to get the desired representation $\hat{\xi}(t, u)$. To make this implementation efficient and parallelizable, we assume that future events at a coordinate (s, u) with $s \in [t, t + \Delta t)$ only depend on spatiotemporally nearby past events, i.e., from the neighborhood $[t - \Delta t, t) \times \Omega_u$. This forces the representation $\hat{\xi}(t, \cdot)$ to extract the local structure and motion in the scene to predict future events—encouraging generalization. Ideally, the smallest radius for which the neighborhood Ω_u is sufficiently predictive of a future spatiotemporal location is the optical flow at Δv for the time horizon $2\Delta t$. This quantity is bounded for most natural scenes. We choose Ω_u to be a neighborhood of size 37×37 for all experiments. We use ConvNeXt V2 [83] blocks with 7×7 kernels and 32 channels to parameterize $\rho : \Omega_u \times \mathbb{R}^n \rightarrow \Omega_u \times \mathbb{R}^p$. Unlike the original ConvNeXt V2 architecture, we use these blocks without any down-sampling operations, resulting in the final $\hat{\xi}(t, \cdot)$ to retain the spatial domain Ω . Our largest model for ρ requires only $\sim 38K$ trainable parameters.

A robust and fast training procedure for F^3

We optimize the objective presented in Eq. (10) to fit φ and ρ . We discussed how the focal loss and a small receptive field of ρ prevent overfitting and encourage generalization for noisy event data. To ensure further robustness of the training to noise and obtain better generalization, we randomly sub-sample input events e^- during training with target events e^+ remaining unchanged. This amounts to using dropout in the input layer of the F^3 featurizer. Evidence supporting the effectiveness of this idea can be found in Fig. 8e.

Note that summing up the hash encodings of event coordinates along time in Eq. (8) to form $\bar{\varphi}(t, \cdot)$ does not alter the spatial sparsity of events. These features are perfect candidates to be processed by sub-manifold sparse convolutions. With this idea, we reimplemented ConvNeXt V2 blocks in ρ to have sparse convolutions using MinkowskiEngine [84] and TorchSparse++ [85], with the latter proving more efficient. Although these sparse formulations are faster than their standard counterparts, in small networks like F^3 , fused dense kernels on structured, fixed-size inputs outweigh these benefits. We therefore implemented our approach to be amicable to early-stage implementations of compilation techniques in PyTorch [86]. Although these compiled kernels are dense, our implementation takes ~ 8.4 ms to process $\sim 200K$ HD events during inference. In comparison, our TorchSparse++ implementation requires ~ 23 ms. As sparse kernels become better supported, or in scenes where the number of events is very small, e.g., extreme low-light conditions, we expect the sparse convolution-based implementation to be faster. All experiments in the paper use compiled versions of dense convolutions.

All training and inference statistics are measured on a single Nvidia RTX 4090 GPU, unless stated otherwise.

In practice, we train F^3 using a distributed setup across two GPUs in half-precision. All downstream tasks are trained on a single GPU using full precision.

Implementation details of downstream tasks

Semantic segmentation As discussed in Sec. 2.3, we adapt a SegFormer B3 model pretrained on Cityscapes to process p -channel F^3 features. This lightweight model with ~ 47.3 M parameters is quite small by modern standards. Training and testing for this task are performed directly on the standard train/test splits of each of the three datasets. However, when evaluating cross-dataset transfer from M3ED to DSEC, we make some important considerations. Pseudo-labels for event data in M3ED derived from RGB images often contain unannotated pixels due to warping under homography. Since these artifacts appear both in the train and test splits, they do not impact evaluation within M3ED. In contrast, DSEC segmentation labels have a different pattern of unannotated regions. To mitigate this mismatch and support full-frame predictions, we apply random 600×800 crops to M3ED HD frames and labels during training. Further, DSEC is equipped with a VGA event camera, unlike the HD one in M3ED. To test the transfer of models trained on M3ED to DSEC, we center the VGA events in the HD frame before passing them to the network.

Optical flow estimation In Sec. 2.4, we described how a small neural network ψ can be used to compute the optical flow $v(t, u)$ from event features $F^3(t, \cdot)$ without supervision. The feature extractor F^3 is kept frozen during this process, while ψ is trained by minimizing the objective in Eq. (11). Here, the number of scale-space levels σ controls the receptive field of the photometric loss. It determines how far apart two matching structures can be in pixel space of $F^3(t, \cdot)$ and $F^3(t + \Delta t, \cdot)$ while still providing a gradient. In datasets M3ED, DSEC, and MVSEC, we set the number of σ levels to 5, 4, and 2, respectively. This choice is based on the event camera resolutions and the typical amount of motion in the scene. We fix the smoothness regularization parameter in Eq. (11) to $\lambda = 10^{-3}$ for all experiments. The flow prediction function $\psi : \mathbb{R}^p \times \Omega \rightarrow \mathbb{R}^2 \times \Omega$, is parameterized using four ConvNeXt V2 blocks with 9×9 kernels and p -channels. This simple architecture has ~ 28 K trainable parameters with a receptive field of 33×33 . We use the same architecture across all datasets and robotic platforms to focus the discussion on the temporal matching ability and the motion information content of F^3 . To enable fair comparisons with baselines using V^3 and I^3 , we project both of them to p channels before passing them to ψ . For V^3 , this is done using a linear layer followed by layer normalization [87] applied to the Δt channels. For I^3 , the same projection is applied to the two polarity channels.

Monocular depth estimation In Sec. 2.5, we described how a neural network ψ —a DepthAnything V2 Base architecture—takes $F^3(t, \cdot)$ as input and predicts the disparity $d(t, u) \in \mathbb{R}_+$ at time t for all pixels $u \in \Omega$. The training follows a two-stage procedure. The first stage focuses on learning accurate object boundaries, albeit with incorrect scale. The second stage retains this boundary information and aims to recover the correct metric depth scale. In both cases, a gradient matching regularizer Eq. (13) is used to encourage sharp object boundaries in the predicted disparity maps. The number of spatial scales σ controls the size of the neighborhood over which boundaries are enforced. We use 4 scales σ and set the regularization weight λ to 0.3 for both objectives Eqs. (12) and (14) across all experiments. During training, we restrict valid pixels to those with disparity less than 384. We also apply random spatial cropping of the input events and target disparity maps to 518×518 . This cropping serves as a robust data augmentation policy and also improves training efficiency.

Acknowledgments

RD and PC were supported by grants from the National Science Foundation (IIS-2145164, CCF-2212519), IoT4Ag ERC under NSF Grant EEC-1941529 and DSO National Laboratories, Singapore. We are grateful for

the discussions with Daniel Gehrig on event representations.

References

- [1] Bart G Borghuis, Peter Sterling, and Robert G Smith. Loss of sensitivity in an analog neural circuit. *Journal of Neuroscience*, 29(10):3045–3058, 2009.
- [2] Peter Sterling and Simon Laughlin. *Principles of Neural Design*. 2015.
- [3] William Bialek. *Biophysics: Searching for Principles*. 2012.
- [4] Carver A Mead and Misha A Mahowald. A silicon model of early visual processing. *Neural networks*, 1(1):91–97, 1988.
- [5] Kristin Koch, Judith McLean, Ronen Segev, Michael A Freed, Michael J Berry II, Vijay Balasubramanian, and Peter Sterling. How much the eye tells the brain. *Current Biology*, 16(14):1428–1434, 2006.
- [6] Alex Zihao Zhu, Dinesh Thakur, Tolga Özslan, Bernd Pfommer, Vijay Kumar, and Kostas Daniilidis. The multivehicle stereo event camera dataset: An event camera dataset for 3d perception. *IEEE Robotics and Automation Letters*, 3(3):2032–2039, 2018.
- [7] Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza. DSEC: A Stereo Event Camera Dataset for Driving Scenarios. *IEEE Robotics and Automation Letters*, 2021.
- [8] Kenneth Chaney, Fernando Cladera, Ziyun Wang, Anthony Bisulco, M Ani Hsieh, Christopher Korpela, Vijay Kumar, Camillo J Taylor, and Kostas Daniilidis. M3ed: Multi-robot, multi-sensor, multi-environment event dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4016–4023, 2023.
- [9] David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. 2010.
- [10] David L Donoho and Iain M Johnstone. Ideal spatial adaptation by wavelet shrinkage. *biometrika*, 81(3):425–455, 1994.
- [11] Stephane Mallat. *A Wavelet Tour of Signal Processing: The Sparse Way*. 2008.
- [12] David L Donoho, Iain M Johnstone, et al. Ideal denoising in an orthonormal basis chosen from a library of bases. *Comptes rendus de l'Académie des sciences. Série I, Mathématique*, 319(12):1317–1322, 1994.
- [13] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015.
- [14] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in Neural Information Processing Systems*, pages 3391–3401, 2017.
- [15] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022.
- [16] Joan Bruna and S. Mallat. Invariant Scattering Convolution Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1872–1886, August 2013.

- [17] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised event-based learning of optical flow, depth, and egomotion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 989–997, 2019.
- [18] Ana I Maqueda, Antonio Loquercio, Guillermo Gallego, Narciso Garcia, and Davide Scaramuzza. Event-based vision meets deep learning on steering prediction for self-driving cars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5419–5427, 2018.
- [19] Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip Torr, and Puneet Dokania. Calibrating deep neural networks using focal loss. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15288–15299. Curran Associates, Inc., 2020.
- [20] Iñigo Alonso and Ana C Murillo. Ev-segnet: Semantic segmentation for event-based cameras. In *IEEE International Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.
- [21] Zhaoning Sun, Nico Messikommer, Daniel Gehrig, and Davide Scaramuzza. Ess: Learning event-based semantic segmentation from still images. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIV*, page 341–357, Berlin, Heidelberg, 2022. Springer-Verlag.
- [22] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. In *Neural Information Processing Systems (NeurIPS)*, 2021.
- [23] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [24] Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, et al. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14408–14419, 2023.
- [25] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical multi-scale attention for semantic segmentation. *CoRR*, abs/2005.10821, 2020.
- [26] Shintaro Shiba, Yannick Klose, Yoshimitsu Aoki, and Guillermo Gallego. Secrets of event-based optical flow, depth and ego-motion estimation by contrast maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
- [27] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised event-based learning of optical flow, depth, and egomotion. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 989–997, 2018.
- [28] Hao Zhuang, Zheng Fang, Xinjie Huang, Kuanxu Hou, Delei Kong, and Chenming Hu. Ev-mgrflownet: Motion-guided recurrent network for unsupervised event-based optical flow with hybrid motion-compensation loss. *IEEE Transactions on Instrumentation and Measurement*, 73:1–15, 2024.
- [29] Mathias Gehrig, Mario Millhäusler, Daniel Gehrig, and Davide Scaramuzza. E-raft: Dense optical flow from event cameras. In *International Conference on 3D Vision (3DV)*, 2021.
- [30] Hongzhi You, Yijun Cao, Wei Yuan, Fanjun Wang, Ning Qiao, and Yongjie Li. Vector-symbolic architecture for event-based optical flow, 2025.

- [31] Federico Paredes-Vallés, Kirk Y. W. Scheper, Christophe De Wagter, and Guido C. H. E. de Croon. Taming contrast maximization for learning sequential, low-latency, event-based optical flow. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9695–9705, October 2023.
- [32] Friedhelm Hamann, Ziyun Wang, Ioannis Asmanis, Kenneth Chaney, Guillermo Gallego, and Kostas Daniilidis. Motion-prior contrast maximization for dense continuous-time motion estimation. In *European Conference on Computer Vision (ECCV)*, pages 18–37, 2024.
- [33] Simon Baker, Daniel Scharstein, James P Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International journal of computer vision*, 92(1):1–31, 2011.
- [34] Yi Ma, Stefano Soatto, Jana Košecká, and Shankar Sastry. *An Invitation to 3-d Vision: From Images to Geometric Models*, volume 26. 2004.
- [35] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Live demonstration: Unsupervised event-based learning of optical flow, depth and egomotion. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1694–1694, 2019.
- [36] Shintaro Shiba, Yoshimitsu Aoki, and Guillermo Gallego. Event collapse in contrast maximization frameworks. *Sensors*, 22(14):1–20, 2022.
- [37] Alex Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Ev-flownet: Self-supervised optical flow estimation for event-based cameras. In *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.
- [38] Deqing Sun, Stefan Roth, and Michael J. Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *Int. J. Comput. Vision*, 106(2):115–137, January 2014.
- [39] Deqing Sun, Stefan Roth, and Michael J Black. Secrets of optical flow estimation and their principles. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2432–2439. IEEE, 2010.
- [40] Amos Sironi, Manuele Brambilla, Nicolas Bourdis, Xavier Lagorce, and Ryad Benosman. Hats: Histograms of averaged time surfaces for robust event-based object classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1731–1740, 2018.
- [41] Xavier Lagorce, Garrick Orchard, Francesco Galluppi, Bertram E Shi, and Ryad B Benosman. HOTS: A hierarchy of event-based time-surfaces for pattern recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(7):1346–1359, 2016.
- [42] Daniel Gehrig Javier Hidalgo-Carrio and Davide Scaramuzza. Learning monocular dense depth from events. *IEEE International Conference on 3D Vision.(3DV)*, 2020.
- [43] Lin Wang, Yujeong Chae, and Kuk-Jin Yoon. Dual transfer learning for event-based end-task prediction via pluggable event to image translation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2135–2145, 2021.
- [44] Xu Liu, Jianing Li, Jinqiao Shi, Xiaopeng Fan, Yonghong Tian, and Debin Zhao. Event-based monocular depth estimation with recurrent transformers. *IEEE Transactions on Circuits and Systems for Video Technology*, 34(8):7417–7429, 2024.
- [45] Jesse J Hagenaars, Yilun Wu, Federico Paredes-Vallés, Stein Stroobants, and Guido CHE de Croon. On-device self-supervised learning of low-latency monocular depth from only events. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 17114–17123, 2025.

- [46] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *arXiv:2406.09414*, 2024.
- [47] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3), 2022.
- [48] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS’14, page 2366–2374, Cambridge, MA, USA, 2014. MIT Press.
- [49] Daniel Gehrig, Michelle Rüegg, Mathias Gehrig, Javier Hidalgo-Carrio, and Davide Scaramuzza. Combining events and frames using recurrent asynchronous multimodal networks for monocular depth prediction. *IEEE Robotic and Automation Letters. (RA-L)*, 2021.
- [50] Alberto Sabater, Luis Montesano, and Ana C. Murillo. Event Transformer⁺. A Multi-Purpose Solution for Efficient Event Data Processing . *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 45(12):16013–16020, December 2023.
- [51] Kartik Mohta, Michael Watterson, Yash Mulgaonkar, Sikang Liu, Chao Qu, Anurag Makineni, Kelsey Saulnier, Ke Sun, Alex Zhu, Jeffrey Delmerico, et al. Fast, autonomous flight in GPS-denied and cluttered environments. *Journal of Field Robotics*, 35(1):101–120, 2018.
- [52] Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. In *Proc. of the 37-Th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.
- [53] Taco Cohen. *Equivariant Convolutional Networks*. PhD thesis, University of Amsterdam (UVA), 2021.
- [54] Yinshuang Xu, Jiahui Lei, and Kostas Daniilidis. SE(3) equivariant convolution and transformer in ray space. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 2463–2510, 2023.
- [55] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. *arXiv preprint arXiv:2301.08243*, 2023.
- [56] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.
- [57] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.
- [58] Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.
- [59] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022.

- [60] Shifan Zhu, Zhipeng Tang, Michael Yang, Erik Learned-Miller, and Donghyun Kim. Event camera-based visual odometry for dynamic motion tracking of a legged robot using adaptive time surface. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3475–3482. IEEE, 2023.
- [61] Anthony Bisulco, Vijay Kumar, and Kostas Daniilidis. Ev-ttc: Event-based time to collision under low light conditions. *IEEE Robotics and Automation Letters*, 2025.
- [62] 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, 2019.
- [63] Mohammad Mostafavi, Lin Wang, and Kuk-Jin Yoon. Learning to reconstruct hdr images from events, with applications to depth and flow prediction. *International Journal of Computer Vision*, 129(4):900–920, 2021.
- [64] Yeongwoo Nam, Mohammad Mostafavi, Kuk-Jin Yoon, and Jonghyun Choi. Stereo depth from events cameras: Concentrate and focus on the future. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6114–6123, 2022.
- [65] Min Liu and Tobi Delbruck. Adaptive time-slice block-matching optical flow algorithm for dynamic vision sensors. *BMVC*, 2018.
- [66] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised event-based optical flow using motion compensation. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018.
- [67] Simon Schaefer, Daniel Gehrig, and Davide Scaramuzza. Aegnn: Asynchronous event-based graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12371–12381, 2022.
- [68] Cedric Scheerlinck, Nick Barnes, and Robert Mahony. Asynchronous spatial image convolutions for event cameras. *IEEE Robotics and Automation Letters*, 4(2):816–822, 2019.
- [69] Ze Huang, Li Sun, Cheng Zhao, Song Li, and Songzhi Su. Eventpoint: Self-supervised interest point detection and description for event-based camera. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 5396–5405, 2023.
- [70] Nikola Zubić, Daniel Gehrig, Mathias Gehrig, and Davide Scaramuzza. From chaos comes order: Ordering event representations for object recognition and detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12846–12856, 2023.
- [71] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J. Davison, Jörg Conradt, Kostas Daniilidis, and Davide Scaramuzza. Event-based vision: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(1):154–180, January 2022.
- [72] Daniel Gehrig, Henri Rebecq, Guillermo Gallego, and Davide Scaramuzza. Ekl: Asynchronous photometric feature tracking using events and frames. *International Journal of Computer Vision*, 128(3):601–618, 2020.
- [73] Antoni Rosinol Vidal, Henri Rebecq, Timo Horstschaefner, and Davide Scaramuzza. Ultimate slam? combining events, images, and imu for robust visual slam in hdr and high-speed scenarios. *IEEE Robotics and Automation Letters*, 3(2):994–1001, 2018.

- [74] Davide Falanga, Kevin Kleber, and Davide Scaramuzza. Dynamic obstacle avoidance for quadrotors with event cameras. *Science Robotics*, 5(40):eaaz9712, 2020.
- [75] Davide Falanga, Kevin Kleber, Stefano Mintchev, Dario Floreano, and Davide Scaramuzza. The foldable drone: A morphing quadrotor that can squeeze and fly. *IEEE Robotics and Automation Letters*, 4(2):209–216, 2018.
- [76] Federico Paredes-Vallés, Jesse J Hagenaars, Julien Dupeyroux, Stein Stroobants, Yingfu Xu, and Guido CHE de Croon. Fully neuromorphic vision and control for autonomous drone flight. *Science Robotics*, 9(90):eadi0591, 2024.
- [77] Elias Mueggler, Basil Huber, and Davide Scaramuzza. Event-based, 6-dof pose tracking for high-speed maneuvers. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2761–2768. IEEE, 2014.
- [78] Ziyun Wang, Fernando Cladera, Anthony Bisulco, Daewon Lee, Camillo J Taylor, Kostas Daniilidis, M Ani Hsieh, Daniel D Lee, and Volkan Isler. Ev-catcher: High-speed object catching using low-latency event-based neural networks. *IEEE Robotics and Automation Letters*, 7(4):8737–8744, 2022.
- [79] Anton Mitrokhin, Cornelia Fermüller, Chethan Parameshwara, and Yiannis Aloimonos. Event-based moving object detection and tracking. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9. IEEE, 2018.
- [80] Nitin J Sanket, Chethan M Parameshwara, Chahat Deep Singh, Ashwin V Kuruttukulam, Cornelia Fermüller, Davide Scaramuzza, and Yiannis Aloimonos. Evdodgenet: Deep dynamic obstacle dodging with event cameras. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10651–10657. IEEE, 2020.
- [81] Mike Davies, Andreas Wild, Garrick Orchard, Yulia Sandamirskaya, Gabriel A Fonseca Guerra, Prasad Joshi, Philipp Plank, and Sumedh R Risbud. Advancing neuromorphic computing with loihi: A survey of results and outlook. *Proceedings of the IEEE*, 109(5):911–934, 2021.
- [82] Stephen J Carey, Alexey Lopich, David RW Barr, Bin Wang, and Piotr Dudek. A 100,000 fps vision sensor with embedded 535gops/w 256×256 simd processor array. In *2013 symposium on VLSI circuits*, pages C182–C183. IEEE, 2013.
- [83] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. *arXiv preprint arXiv:2301.00808*, 2023.
- [84] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019.
- [85] Haotian Tang, Shang Yang, Zhijian Liu, Ke Hong, Zhongming Yu, Xiuyu Li, Guohao Dai, Yu Wang, and Song Han. Torchsparsen++: Efficient training and inference framework for sparse convolution on gpus. In *Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO ’23, page 225–239, New York, NY, USA, 2023. Association for Computing Machinery.
- [86] Jason Ansel, Edward Yang, Horace He, and Natalia et al. Gimelshein. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, page 929–947, 2024.

- [87] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. *arXiv:1607.06450 [cs, stat]*, July 2016.
- [88] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1321–1330. JMLR.org, 2017.
- [89] Chung Bai, Tao Xiao, Yajie Chen, Haoqian Wang, Fang Zhang, and Xiang Gao. Faster-LIO: Lightweight tightly coupled LiDAR-inertial odometry using parallel sparse incremental voxels. *IEEE Robotics and Automation Letters*, 7(2):4861–4868, 2022.
- [90] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, 2008.

Supplementary Material

S.1 Proof of Theorem 1

We assume that there exist constants c_1 and c_2 such that (i) the operator norm $\|A\|_2 \leq c_1$, and (ii) if $N_{\mathcal{L}}(e) = \min_{\mathcal{B} \in \mathcal{L}} \|e_{\mathcal{B}}\|_0$ is the minimal sparsity of the signal e in the library, then $N_{\mathcal{L}}(Ae) \leq c_2 N_{\mathcal{L}}(e)$, i.e., the operator A does not decrease sparsity of the signal by more than a factor c_2 .

Consider the objective in Eq. (6). The joint minimizers $\hat{A}, \hat{\mathcal{B}}$ of this objective can be used to recover the future statistic ξ^+ . In Theorem 1, we show that doing so incurs a risk that is additively off from the oracle dynamics risk by a constant factor, and the ideal denoising risk by a logarithmic factor. In this case, the denoising oracle has access to the best basis $\mathcal{B} \in \mathcal{L}$ for denoising the events, and the knowledge of which coordinates projected in this basis have energy larger than the noise variance. On the other hand, the risk for the dynamics oracle is the regression error on the denoised past statistics provided by the ideal denoising oracle with full knowledge of the future statistics ξ^+ . This can be thought of as an errors-in-variables regression problem, where ξ^- is a nuisance, which is only partially known to both the dynamics and denoising oracles.

Proving the aforementioned strong properties for the joint minimizers $\hat{A}, \hat{\mathcal{B}}$ is non-trivial. We break our proof idea into two major steps. We show that a two-step procedure, first estimating an ideal basis for denoising, followed by regression to find the dynamics operator, satisfies similar claims for recovering the future statistic ξ^+ with high probability. Second, we show that joint estimates of $\hat{A}, \hat{\mathcal{B}}$ that minimize Eq. (6) can recover ξ^+ at least as well as the two-step procedure, concluding our proof.

Two-step estimation of the dynamics operator and the denoising basis Consider the following denoising complexity functional

$$D(\xi, \tilde{\xi}) = \left\| \xi - \tilde{\xi} \right\|^2 + \Lambda_n N_{\mathcal{L}}(\tilde{\xi}). \quad (15)$$

To find the ideal basis for denoising, in this two-step procedure, we follow the strategy of [12]. Let the empirically denoised estimate $\tilde{\xi}^- = \operatorname{argmin}_{\tilde{\xi}} D(e^-, \tilde{\xi})$. Observe that this minimization is equivalent to applying the hard thresholding operator on a basis representation of e^- , given by $\tilde{\mathcal{B}} = \operatorname{argmin}_{\mathcal{B} \in \mathcal{L}} \sum_i \min((e_{\mathcal{B}}^-)^2, \Lambda_n)$. Specifically $(\tilde{\xi}_{\tilde{\mathcal{B}}}^-)_i = \mathbf{1}_{\{|(e_{\tilde{\mathcal{B}}}^-)_i| > \sqrt{\Lambda_n}\}} (e_{\tilde{\mathcal{B}}}^-)_i$.

From [12, Theorem 1], we know that this estimator $\tilde{\xi}^-$ incurs a risk that is only a logarithmic factor off the ideal risk, with high probability. We state it here for ease of reference. With probability at least $\pi_n = 1 - e/M_n$,

$$\left\| \xi^- - \tilde{\xi}^- \right\|^2 \leq D(\xi^-, \tilde{\xi}^-) \leq (1 - 8/\lambda)^{-1} \Lambda_n R_{\text{denoising}}(\xi^-, \mathcal{L}) \quad (16)$$

where, $R_{\text{denoising}}(\xi^-, \mathcal{L}) = \sum_i \min((\xi_{\mathcal{B}^*}^-)^2, 1)$ and $\mathcal{B}^* = \operatorname{argmin}_{\mathcal{B} \in \mathcal{L}} \sum_i \min((\xi_{\mathcal{B}}^-)^2, 1)$ is the ideal basis for denoising e^- . Similarly, the signal that attains this ideal denoising risk can be defined as ξ^{*-} , where $(\xi_{\mathcal{B}^*}^{*-})_i = \mathbf{1}_{\{|(\xi_{\mathcal{B}^*}^-)_i| > 1\}} (e_{\mathcal{B}^*}^-)_i$. Note that obtaining this ideal basis and estimate requires us to know which coordinates of ξ^- , when projected on a basis, are larger than 1. This information is unknown when we are finding our empirical estimate $\tilde{\xi}^-$.

Broadly, we are interested in performing regression using these denoised variables to recover ξ^+ . Theoretical interest is in comparing the dynamics risk incurred by using $\tilde{\xi}^-$ (empirical estimate) or ξ^{*-} (ideal estimate) as the regression predictors. Next, we introduce a theoretical denoising estimator $\xi^{0-} = \operatorname{argmin}_{\tilde{\xi}} D(\xi^-, \tilde{\xi})$ requiring knowledge of ξ^- . The purpose of this estimator is to serve as a bridge in the mathematical analysis to compare the properties of $\tilde{\xi}^-$ and ξ^{*-} . We define the empirical and theoretical dynamics operators as $\tilde{A} = \operatorname{argmin}_A \left\| e^+ - A\tilde{\xi}^- \right\|$ and $A^0 = \operatorname{argmin}_A \left\| \xi^+ - A\xi^{0-} \right\|$ respectively. We relate $\left\| \xi^+ - \tilde{A}\tilde{\xi}^- \right\|$

(empirical dynamics risk) and $\|\xi^+ - A^0 \xi^{0-}\|$ (theoretical dynamics risk) by the following steps,

$$\|\epsilon^+ - \tilde{A} \tilde{\xi}^-\|^2 = \|\xi^+ - \tilde{A} \tilde{\xi}^-\|^2 + \|\nu^+\|^2 + 2\langle \nu^+, \xi^+ - \tilde{A} \tilde{\xi}^- \rangle \leq \|e^+ - A^0 \tilde{\xi}^-\|^2$$

since, \tilde{A} is the empirical minimizer. Rearranging and combining, we get,

$$\|\xi^+ - \tilde{A} \tilde{\xi}^-\|^2 \leq \|\xi^+ - A^0 \tilde{\xi}^-\|^2 + 2\langle \nu^+, \tilde{A} \tilde{\xi}^- - A^0 \xi^{0-} \rangle + 2\langle \nu^+, A^0(\xi^{0-} - \tilde{\xi}^-) \rangle \quad (17)$$

We write $\|\xi^+ - A^0 \tilde{\xi}^-\|^2$ in terms of a more useful quantity $\|\xi^+ - A^0 \xi^{0-}\|^2$, the theoretical dynamics risk,

$$\begin{aligned} \|\xi^+ - A^0 \tilde{\xi}^-\|^2 &= \|\xi^+ - A^0 \xi^{0-}\|^2 + \|A^0(\xi^{0-} - \tilde{\xi}^-)\|^2 \\ &\quad + 2\langle A \xi^- - A^0 \xi^{0-}, A^0(\xi^{0-} - \tilde{\xi}^-) \rangle + 2\langle \zeta, A^0(\xi^{0-} - \tilde{\xi}^-) \rangle. \end{aligned} \quad (18)$$

Now, to upper bound the empirical dynamics risk, we need to obtain bounds for the quantities on the right-hand side of Eq. (17) and Eq. (18). We bound two quantities on the right-hand side of Eq. (18) as

$$\begin{aligned} \|A^0(\xi^{0-} - \tilde{\xi}^-)\|^2 &\leq \|A^0\|^2 (\|\xi^{0-} - \tilde{\xi}^-\|^2 + \|\tilde{\xi}^- - \xi^-\|^2) \\ &\leq 2c_1 \frac{\lambda - 4}{\lambda - 8} \Lambda_n R_{\text{denoising}}(\xi^-, \mathcal{L}) \end{aligned} \quad (19)$$

with probability at least π_n using Eq. (16), and

$$\langle A \xi^- - A^0 \xi^{0-}, A^0(\xi^{0-} - \tilde{\xi}^-) \rangle \leq 2\sqrt{c_1} \left(\|\xi^+ - A^0 \xi^{0-}\|^2 + D(\xi^-, \tilde{\xi}^-) \right). \quad (20)$$

However, the remaining terms are inner products of bounded quantities with the noise ν^+ and ζ . We can upper bound them by the following random variable,

$$W(k_1, k_2) = \sup \left\{ \langle \nu, \xi_1 - \xi_2 \rangle : \|\xi_1 - \xi_2\|^2 \leq k_1; \forall i \in \{1, 2\}, \Lambda_n N_{\mathcal{L}}(\xi_i) \leq k_2 \right\}$$

where $\nu \in N(0, I)$. It can be shown that with probability greater than π_n , we have $W(k_1, k_2) \leq 2\sqrt{k_1 k_2}/\lambda \leq (k_1 + k_2)/\lambda$ for all k_1 and k_2 . The proof for this statement follows from a similar argument as the one in [12, Lemma 2]. We use this concentration inequality to upper bound the inner products in Eq. (17) and Eq. (18). The following two inequalities hold with probability at least π_n ,

$$\begin{aligned} \langle \nu^+, \tilde{A} \tilde{\xi}^- - A^0 \xi^{0-} \rangle &\leq W \left(4\|\xi^+ - \tilde{A} \tilde{\xi}^-\|^2, c_2 D(\xi^-, \tilde{\xi}^-) \right) \\ &\leq \frac{1}{\lambda} \left(4\|\xi^+ - \tilde{A} \tilde{\xi}^-\|^2 + c_2 D(\xi^-, \tilde{\xi}^-) \right), \\ \langle \nu^+, A^0(\xi^{0-} - \tilde{\xi}^-) \rangle &\leq W \left(4c_1 D(\xi^-, \tilde{\xi}^-), c_2 D(\xi^-, \tilde{\xi}^-) \right) \\ &\leq \frac{4\sqrt{c_1 c_2}}{\lambda} D(\xi^-, \tilde{\xi}^-). \end{aligned} \quad (21)$$

The same inequality as presented in Eq. (21), holds for $\langle \zeta, A^0(\xi^{0-} - \tilde{\xi}^-) \rangle$. In addition, we also observe a simple relation between the theoretical (A^0, ξ^{0-}) and the ideal estimates (A^*, ξ^{*-}) . With probability greater

than π_n ,

$$\|\xi^+ - A^0 \xi^{0-}\|^2 \leq \|\xi^+ - A^* \xi^{*-}\|^2 + \frac{2c_1 \lambda}{\lambda - 8} \Lambda_n R_{\text{denoising}}(\xi^-, \mathcal{L}). \quad (22)$$

This ideal dynamics risk $R_{\text{dynamics}}(\xi, \mathcal{L}) = \|\xi^+ - A^* \xi^{*-}\|^2$. Rearranging and combining the inequalities Equations (17) to (22), we have with probability greater than $1 - c'e/M_n$, that

$$\|\xi^+ - \tilde{A} \tilde{\xi}^-\|^2 \leq c'_3 R_{\text{dynamics}}(\xi, \mathcal{L}) + c'_4 \Lambda_n R_{\text{denoising}}(\xi^-, \mathcal{L}) \quad (23)$$

where c'_3 and c'_4 are constants dependent on c_1, c_2 and λ . This concludes our analysis of the two-step estimation procedure, showing that our empirical risk is only a constant factor off the ideal dynamics risk and a logarithmic factor off the ideal denoising risk. Now, we extend this discussion to show that we can jointly estimate the dynamics operator A and the denoising basis \mathcal{B} , using a single optimization objective to achieve equivalent asymptotic guarantees.

Joint estimation of the dynamics operator and the denoising basis Consider the following new complexity functional that can be optimized to jointly recover A and \mathcal{B} ,

$$D(e^-, \xi^+, A, \mathcal{B}) = \|\xi^+ - A \hat{\xi}^-\|^2 + \Lambda_n \|\xi_{\mathcal{B}}^-\|_0 \quad (24)$$

where $(\hat{\xi}_{\mathcal{B}}^-)_i = \mathbf{1}_{\{|(e_{\mathcal{B}}^-)_i| > \sqrt{\Lambda_n}\}} (e_{\mathcal{B}}^-)_i$. We jointly optimize this complexity functional to get the new empirical estimates as $\hat{A}, \hat{\mathcal{B}} = \operatorname{argmin}_{A, \mathcal{B}} D(e^-, e^+, A, \mathcal{B})$. Similarly, we define the new theoretical estimates as $A', \mathcal{B}' = \operatorname{argmin}_{A, \mathcal{B}} D(e^-, \xi^+, A, \mathcal{B})$, serving as a means to compare between the joint empirical estimates $\hat{A}, \hat{\mathcal{B}}$ and the two-step estimates $\tilde{A}, \tilde{\mathcal{B}}$. We go through the following steps to relate $D(e^-, \xi^+, \hat{A}, \hat{\mathcal{B}})$ and the two-step empirical risk $\|\xi^+ - \tilde{A} \tilde{\xi}^-\|^2$, which we have shown to have certain desired properties in Eq. (23).

$$D(e^-, e^+, \hat{A}, \hat{\mathcal{B}}) = D(e^-, \xi^+, \hat{A}, \hat{\mathcal{B}}) + \|\nu^+\|^2 + 2 \langle \nu^+, \hat{A} \hat{\xi}^- \rangle \leq D(e^-, e^+, A', \mathcal{B}')$$

since $\hat{A}, \hat{\mathcal{B}}$ are the minimizers of $D(e^-, e^+, \cdot, \cdot)$. Rearranging and combining the terms, we get,

$$D(e^-, \xi^+, \hat{A}, \hat{\mathcal{B}}) \leq D(e^-, \xi^+, A', \mathcal{B}') + 2 \langle \nu^+, A' \xi'^- - \hat{A} \hat{\xi}^- \rangle.$$

Now notice that, $\langle \nu^+, A' \xi'^- - \hat{A} \hat{\xi}^- \rangle \leq W(4D(e^-, \xi^+, \hat{A}, \hat{\mathcal{B}}), c_2 D(e^-, \xi^+, \hat{A}, \hat{\mathcal{B}}))$. Combined with the fact that A', \mathcal{B}' are the minimizers of $D(e^-, \xi^+, \cdot, \cdot)$, we can further extend the above inequality to connect the two of our desired quantities,

$$D(e^-, \xi^+, \hat{A}, \hat{\mathcal{B}}) \leq \|\xi^+ - \tilde{A} \tilde{\xi}^-\|^2 + D(\xi, \tilde{\xi}^-) + 2W(4D(e^-, \xi^+, \hat{A}, \hat{\mathcal{B}}), c_2 D(e^-, \xi^+, \hat{A}, \hat{\mathcal{B}})).$$

Finally, combining this with the concentration inequality for $W(\cdot, \cdot)$, Eq. (16) and Eq. (23), we have with probability greater than $1 - c''e/M_n$,

$$\|\xi^+ - \hat{A} \hat{\xi}^-\|^2 \leq D(e^-, \xi^+, \hat{A}, \hat{\mathcal{B}}) \leq c_3 R_{\text{dynamics}}(\xi, \mathcal{L}) + c_4 \Lambda_n R_{\text{denoising}}(\xi^-, \mathcal{L}).$$

This concludes our proof of Theorem 1.

S.2 Details of the training objective in Eq. (10)

Eq. (10) is the binary cross-entropy loss when $\alpha = 0.5$ and $\gamma = 0$. Parametric regression to minimize this loss often leads to miscalibration and overfitting to the data [88], especially for event data which is sparse. This adversely affects generalization to new sequences, particularly under objectives such as ours in this paper which are un/self-supervised. During training, F^3 features can end up being trivial if input and output event spatio-temporal volumes are very similar. However, with $\gamma > 0$, minimizing the focal loss in Eq. (10) minimizes the upper bound of the entropy-regularized KL divergence [19]

$$\ell_t(\varphi, \rho, \psi) \geq \sum_{s \in [t, t + \Delta t]} \sum_{u \in \Omega} \text{KL}(p_e, p_{\hat{e}}) + H(p_e) - \gamma H(p_{\hat{e}})$$

where $p_e(s, u) = [e(s, u), 1 - e(s, u)]$ is a probability distribution corresponding to events $e(s, u)$ at time s and pixel u , and $p_{\hat{e}}(s, u) = [\hat{e}(s, u), 1 - \hat{e}(s, u)]$ is the probability distribution corresponding to the predicted events $\hat{e}(s, u) = \psi(s, u; F^3(t, u))$. The quantity KL denotes the Kullback-Leibler divergence and $H(\cdot)$ stands for the Shannon entropy.

In the above expression, the regularization using the coefficient γ encourages predictions \hat{e} with a large entropy, and thereby prevents the model from becoming overconfident. We set $\gamma = 2$ for all experiments in this work. Entropy regularization is important for generalization and preventing feature collapse. Selecting α as suggested in Eq. (10) helps address class imbalance and also addresses the stochasticity of event data.

The same underlying scene statistic ξ and the camera trajectory x can produce different event streams due to the noise described in Eq. (1). Solving this inverse problem to recover ξ is difficult, under these constraints of unknown scene statistics, dynamics, and noise. Suppose we have a scene ξ shown in Fig. S.1 that consists of a set of non-overlapping event-generating spatiotemporal surfaces C . Let $e_i(t, u)$ be the different realizations of events from the same scene, where i denotes the realization and $t \in \mathbb{Z}_+, u \in \Omega$. Let us denote the area of C by $a(C)$ and $N = \sum_{s, u} \mathbb{E}_i[e_i(s, u)]$ be the average number of events, with $\mu = N/a(C)$. Theorem 2 motivates the objective in Eq. (10) using this scene.

Lemma 2. If $e_i(t, u)$ is a Bernoulli random variable with parameter μ that is independent and identically distributed for all $(t, u) \in C$ and zero when $(t, u) \notin C$. If $\hat{e} : \mathbb{Z}_+ \times \Omega \rightarrow \{1, 0\}$ minimizes

$$\ell(e, \hat{e}) = \sum_i \sum_{t, u} \mathbf{1}_{\{e_i(t, u) \neq \hat{e}(t, u)\}} (\alpha e_i(t, u) + (1 - \alpha)(1 - e_i(t, u))),$$

then $\hat{e}(t, u) = \mathbf{1}_{\{(t, u) \in C\}}$ when $\alpha > 1 - 2\mu$ and $\hat{e}(t, u) = 0$ otherwise.

Proof. Observe that

$$\ell = \sum_i \sum_{t, u \in C} \mathbf{1}_{\{e_i(t, u) \neq \hat{e}(t, u)\}} (\alpha e_i(t, u) + (1 - \alpha)(1 - e_i(t, u))) + \sum_{u, t \notin C} \mathbf{1}_{\{\hat{e}(u, t) \neq 0\}}.$$

The minimum is achieved when $\hat{e}(u, t)$ equals zero for all $t, u \notin C$. Now since $e_i(u, t)$ are independent and identically distributed, if we set $\hat{e} \equiv \hat{e}(t, u)$

$$\begin{aligned}\operatorname{argmin}_{\hat{e}} \ell &= \operatorname{argmin}_{\hat{e}} \sum_i \sum_{t, u \in C} (e_i(1 - \hat{e}) + (1 - e_i)\hat{e})(\alpha e_i + (1 - \alpha)(1 - e_i)) \\ &= \operatorname{argmin}_{\hat{e}} \sum_{u, t \in C} \hat{e} \mathbb{E}_i[1 - \alpha - 2e_i(1 - 2\alpha) - 4\alpha e_i^2] \\ &= \operatorname{argmin}_{\hat{e}} \hat{e} (1 - \alpha - 2\mu).\end{aligned}$$

This shows that for all (t, u) , the predicted events $\hat{e}(t, u) = 1$ when $\alpha > 1 - 2\mu$ while $\hat{e}(t, u) = 0$ when $\alpha \leq 1 - 2\mu$ or when $(t, u) \notin C$. \square

Intuitively, when events are very sparse and stochastic, the representation that seeks to predict future events will be trivial. It will predict all zeros if the weight α is not chosen carefully. In typical scenes, the sparsity of events is $\sim 95\%$. The above lemma suggests that we need $\alpha > 0.9$ to learn a non-trivial representation (here $\alpha = 0.5$ corresponds to giving equal weight to events or non-events).

S.3 Experimental details

This section provides details of the training procedures for F^3 and downstream tasks. It also discusses pre-processing and evaluation methodology on the different datasets used in our analysis. In some cases, e.g., optical flow estimation, we calculated ground-truth labels using geometric vision techniques. We will release all data publicly to aid reproducibility.

F^3

For all experiments while training F^3 across different datasets and robotic platforms, we use the AdamW optimizer with a learning rate schedule that decays linearly from 5×10^{-5} to 5×10^{-6} , a weight decay of 0.01, and a batch size of 8. Training is conducted for a total of 200 epochs.

For the M3ED dataset, when evaluating F^3 on each platform (car, spot and falcon), we train separate models on data from each platform using the training splits in Table S.1. Note that training F^3 only requires event data (not the other modalities). In total, this amounts to approximately 0.42, 0.48, and 0.3 hours of event recordings to train F^3 on car, spot and falcon, respectively. F^3 trained on DSEC uses the standard training split provided in the original dataset. This split includes 41 sequences, a mix of day and night driving scenarios, totaling approximately 0.7 hours of event data. For MVSEC, we train F^3 using only the event data from the “outdoor_day2” sequence, which is the standard training sequence used in prior works to perform downstream tasks. This sequence contains approximately 0.18 hours of event data.

Semantic segmentation

We keep the optimizer and hyper-parameters fixed across all methods and datasets during training. We use the AdamW optimizer with a learning rate schedule that decays linearly from 6×10^{-5} to 6×10^{-6} and a weight decay of 0.01. A batch size of 8 is used throughout. We train for a total of 200 epochs on M3ED and 100 epochs on DSEC.

Table S.1: Summary of M3ED sequences used in this work. This table lists all M3ED sequences used in our experiments, indicating the availability and usage of each modality for training and testing.

Robotic Platform	Train/Test Split	Sequence	Semantic Segmentation	Pseudo Depth	LiDAR Depth	Optical Flow
Car	Daytime Train	urban day penno big loop	✓	✓	✓	✓
		urban day penno small loop	✓	✓	✓	✓
		urban day ucity big loop	✓	✓		✓
		urban day city hall	✓	✓	✓	✓
	Daytime Test	urban day rittenhouse	✓		✓	✓
		urban day ucity small loop	✓		✓	✓
	Nighttime Test	urban night rittenhouse				✓
		urban night ucity small loop				✓
Spot	Train	indoor building loop		✓	✓	✓
		indoor stairs		✓	✓	✓
		indoor stairwell		✓	✓	✓
		outdoor day art plaza loop		✓	✓	✓
		outdoor day rocky steps		✓	✓	✓
		outdoor day skatepark 1		✓	✓	✓
		outdoor day skatepark 3		✓		✓
		outdoor day srt green loop		✓	✓	✓
		outdoor day srt under bridge 1		✓	✓	✓
		outdoor day penno building loop				✓
	Test	outdoor night penno building loop				✓
		outdoor night penno plaza lights		✓	✓	
		indoor obstacles		✓	✓	
		outdoor day penno short loop		✓	✓	
		outdoor day skatepark 2		✓	✓	
Falcon	Train	outdoor day srt under bridge 2		✓	✓	
		outdoor night penno short loop		✓	✓	
		indoor flight 2		✓	✓	✓
		indoor flight 3		✓	✓	✓
		outdoor day fast flight 2		✓	✓	✓
		outdoor day fast flight 3		✓		✓
		outdoor day penno cars		✓	✓	✓
		outdoor day penno parking 2		✓	✓	✓
	Test	outdoor day penno parking 3		✓		✓
		outdoor day penno trees			✓	✓

Evaluation on M3ED We train networks on pseudo-labeled segmentation masks obtained on RGB images corresponding to sequences listed under the “car daytime train” section in Table S.1. Every other event-label pair is skipped. RGB images and semantic labels for the “car urban day city big loop” sequence are not publicly available. So we used grayscale images in the dataset to generate our own pseudo-labels using InternImage [24] (which was also used in the original dataset). This gives a total 12,356 event-label pairs for training. As test data, we use sequences listed under “car daytime test” in Table S.1. We only evaluate on event-label pairs where 20 ms windows centered on the semantic label timestamps contain more than 200K events, this gives a total of 13,501 test samples.

Evaluation on DSEC We conduct two experiments: (1) testing cross-dataset transfer from M3ED to DSEC, and (2) evaluating directly on the DSEC train-test split. To evaluate transfer, we train models using the full “car daytime” semantic segmentation dataset of M3ED. All available car daytime train and test sequences in Table S.1 are used for training. These models are evaluated on DSEC and reported under “Not trained on DSEC” in Fig. 5a. For the “Trained on DSEC” section in the figure, we train models from scratch following the train-test split introduced in ESS [21].

Optical flow estimation

To learn ψ , we use the same optimizer and hyper-parameters across all methods and datasets during training. We use the AdamW optimizer with a fixed learning rate of 10^{-3} and a weight decay of 0.01. We used a batch size of 8, and train for 100 epochs on each dataset.

Evaluation on M3ED We use event sequences from the car, spot and falcon robotic platforms to train our unsupervised event-based optical flow prediction networks, as described in Sec. 2.4. The “Car Daytime,” “Spot,” and “Falcon” models shown in Fig. 6f are trained on their respective training sequences listed in Table S.1. Since we only use events to train these networks, this adds up to the same amount of data used to train the respective F^3 models, discussed above in the details of the training procedure for F^3 .

We need ground truth optical flow to evaluate these models, which is not publicly available for M3ED. However, most of the M3ED sequences have time-synchronized and reprojected LiDAR depth maps along with the camera poses computed from LiDAR odometry (Faster-LIO [89]). We adopt a similar strategy as that of MVSEC [6] to compute ground truth optical flow using these known quantities.

Suppose we would like to compute the per-pixel flow given the poses of the event camera at two time instances, t_0 and $t_0 + \Delta t$, specified as rotation matrices $R_t \in \text{SO}(3)$ and translation vectors $p_t \in \mathbb{R}^3$, along with the depth map $Z_{t_0} \in \mathbb{R}_+ \times \Omega$ at time t_0 . Under a linearity assumption, translational and angular velocities for the event camera frame, \dot{p}_{t_0} and ω_{t_0} respectively are

$$\begin{aligned}\dot{p}_{t_0} &= \frac{p_{t_0+\Delta t} - p_{t_0}}{\Delta t} \\ \hat{\omega}_{t_0} &= \frac{\log(R_{t_0}^\top R_{t_0+\Delta t})}{\Delta t},\end{aligned}$$

where $\log(\cdot)$ denotes the matrix logarithm and $\hat{\omega}$ is the skew-symmetric representation of the angular velocity $\omega \in \mathbb{R}^3$. To mitigate any abrupt changes in the velocities, we apply a 10-point moving average filter to \dot{p} and ω . Now, let $u = (u_1, u_2) \in \Omega$ denote the coordinates of the undistorted event camera pixel space and $Z = Z_{t_0}(u) \in \mathbb{R}_+$ be its corresponding depth in the scene at time t_0 . The image-plane motion field due to ego-motion

$$\dot{u} = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{u_1}{Z} & u_1 u_2 & -(1 + u_1^2) & u_2 \\ 0 & -\frac{1}{Z} & \frac{u_2}{Z} & 1 + u_2^2 & -u_1 u_2 & -u_1 \end{bmatrix} \begin{pmatrix} \dot{p}_{t_0} \\ \omega_{t_0} \end{pmatrix}$$

This formulation differentiates the perspective projection of 3D points in a rigid body motion. Hence, \dot{u} , although it captures the effect of egomotion on static objects, fails to account for independently moving objects in the scene. The pixels where independently moving objects are present are marked as invalid and not used for evaluation (such pixels haven't been computed before-hand in the original M3ED dataset). Optical flow is obtained by scaling the instantaneous motion field \dot{u} by the time interval Δt between the depth and pose measurements. Hence, $\dot{u}\Delta t$ is the 2D displacement field that represents the apparent motion of pixel (u_1, u_2) from time t_0 to $t_0 + \Delta t$. We use this quantity as the ground truth to evaluate our flow prediction models.

For evaluating on “Car Daytime” in Fig. 6f, we use the corresponding sequences listed in Table S.1. Since the LiDAR frequency in M3ED is 10Hz, we set Δt to 100 ms for calculating the ground truth flow. Depth maps provided in M3ED have been filtered for independently moving objects, but they are sometimes inaccurate. We manually go through the car daytime test sequences and remove all examples containing moving objects in the scene. We also filter out examples containing fewer than 100 valid flow measurement pixels or where 20 ms window centered around the optical flow start timestamp contains fewer than 200K events. This results in 5,774 test samples for a car driving in an urban and daytime setting.

To evaluate the robustness of our flow prediction framework under challenging illumination conditions, we test models trained on car daytime sequences on nighttime driving scenarios in Fig. 8b. We generate ground truth optical flow for the car nighttime test sequences listed in Table S.1 using the same procedure as above (without manual filtering because it's not easy to see objects in RGB images at nighttime). After filtering based on valid flow pixel count and event rates, there are 5,216 testing examples. Similarly, for results in Fig. 6f for spot and falcon, we evaluate the trained models on their respective test sequences listed in Table S.1. Unlike the car daytime sequences, these scenes contain few independently moving objects and we did not perform manual filtering. However, we do filter samples based on the number of valid flow pixels and event rates. This yields 4,195 and 2,057 test examples for the spot and falcon platforms, respectively.

Evaluation on DSEC Since our proposed optical flow method in Sec. 2.4 is unsupervised, we only use event data to train our models, ground truth optical flow obtained via LiDAR is not used. We use the train split suggested by the authors of DSEC dataset for training. The results reported in Fig. 6e are obtained using the evaluation procedure on the DSEC website dsec.ifi.uzh.ch/uzh/dsec-flow-optical-flow-benchmark.

Evaluation on MVSEC We train our unsupervised method on the “outdoor_day2” sequence and test on the 222.4s – 240.4s interval of the “outdoor_day1” sequence, as introduced in [37]. There are some specific conventions that are often used in the existing literature for evaluation, which we also follow. We evaluate our optical flow estimates at 45Hz and 11.25Hz intervals, denoted as “dt=1” and “dt=4” in the literature. We evaluate flow only on the top 193 rows, which ignores the hood of the car under the camera. Additionally, pixels with valid ground truth flow and at least one event in the flow duration are considered while evaluating the estimate. We report the results for our method and baselines in Fig. 6d.

Monocular depth estimation

Just like the training of F^3 , semantic segmentation and optical flow estimation, we use the same optimizer and hyper-parameters for all our pseudo and metric ψ disparity models. We use the AdamW optimizer with a fixed learning rate of 6×10^{-6} and a weight decay of 0.01. A batch size of 8 is used throughout. Both training stages—pseudo depth prediction and metric depth prediction—are run for 100 epochs.

Evaluation on M3ED To train the pseudo-depth model in stage-1, we generate monocular relative disparity maps from RGB images using the Depth Anything V2-Large [46] model. These maps are reprojected into the event camera frame. We use the pseudo-labeled sequences listed under car daytime, spot and falcon train splits in Table S.1. This results in approximately 45K, 22K, and 17K valid pairs of events and pseudo-labeled

disparity maps for training the car daytime, spot and falcon stage-1 models, respectively.

To train the second stage, fine-tuning on metric depth for retaining sharp object boundaries, we use LiDAR ground truth data present in the M3ED sequences in conjunction with relative depth estimates from stage-1 as discussed in Sec. 2.5. Ground truth LiDAR depth maps marked available in the car daytime, spot and falcon train splits in Table S.1 are used for training the second stage for that particular robotic platform. In M3ED, some sequences do not have LiDAR observations in Table S.1. Also note that we add some of the available nighttime sequences in this stage of the training for the spot and falcon platforms. Pseudo-labeled disparity maps generated from these nighttime sequences are not used in stage-1 training, since the RGB images are too dark to recover any meaningful monocular depth information. This provides us with a total of 5,223, 8,724, and 6,957 samples for training the second stage on car, spot and falcon, respectively.

We evaluate these car daytime, spot and falcon monocular metric depth (second-stage) models on their respective test sequences listed in Table S.1. For all sequences, we evaluate only on depth maps that contain at least 10 valid LiDAR points within a maximum depth of 80 meters, and where the 20 ms window centered on the depth timestamp contains more than 200K events. Additionally, we exclude 15% of the depth samples from the start and end of the falcon sequences, because depth measurements during drone takeoff and landing can be highly erratic. After applying these criteria, we obtain 5,387, 4,756, and 2,118 test examples for car, spot and falcon, respectively. For all metrics reported in Fig. 7e, evaluation is restricted to pixels with valid LiDAR points at depths less than 80 meters.

Evaluation on DSEC For gradient supervision in Fig. 7f, we use the same stage-1 model trained on car daytime sequences as described in the evaluation details for M3ED. In other words, we do not generate pseudo-labeled depth maps using monocular RGB images for DSEC—stage-1 is not retrained on DSEC pseudo-labels. For the second stage of training, or just training for metric disparity from scratch, we use the train split provided in the original DSEC dataset. We evaluate our methods on the disparity benchmark on the DSEC website dsec.ifi.uzh.ch/uzh/disparity-benchmark.

Evaluation on MVSEC As described in Sec. 2.5, MVSEC event camera resolution is quite low (346×260) and therefore we do not need to use the two-stage training approach that we used for M3ED and DSEC. We directly train on events and metric depth map pairs. Following prior works like [42], we use the “outdoor_day2” sequence in MVSEC for training. Two sequences “outdoor.day1” and “outdoor.night1” are used for testing. Unless mentioned otherwise, for all the metrics in Fig. 7d, we only evaluate the methods on pixels with ground truth depth less than 80 meters.

Stereo Depth

F^3 features are spatially consistent and encode meaningful information about the structure in the scene. This enables depth estimation (with correct scale) by matching F^3 features computed for a stereo pair of event cameras. We illustrate this idea under the “Stereo Disparity” section of downstream tasks in Fig. 4. To emphasize the spatial consistency of F^3 , we demonstrate that traditional RGB-based block matching methods can effectively match stereo F^3 features at a given time instant.

We extract events from the stereo camera pair up to time t and pass them to the trained featurizer, obtaining two $F^3(t, \cdot)$ feature maps—one per camera. These p -channel features are then rectified assuming known camera intrinsics and the relative extrinsic transformation between the stereo views. We reduce the dimensionality of these rectified feature maps by retaining only the top three principal components for each view. Disparity is then computed using the resulting 3-channel images using the OpenCV implementation of Semi-Global Block Matching (SGBM) [90]. This approach mirrors the one used for optical flow in Sec. 2.4, where we matched F^3 across time. Here, we instead perform matching across space, leveraging the same structural properties of the F^3 representation.

All three datasets discussed in this paper—M3ED, DSEC, and MVSEC—are equipped with stereo event cameras, making it possible to analyze this proposal further. We show qualitative results for the stereo-matching algorithm on M3ED sequences in Fig. 7c. This task is quite challenging given the diversity of data in M3ED and the use of a simple feature similarity-based matcher in our method. We show that we can match F^3 features under vastly different lighting conditions and scenes, even with the same choice of hyper-parameters in the SGBM algorithm. This demonstrates the spatial consistency of F^3 , under pixel-wise similarity metrics.