Running Event Cameras in the Wild

Nicholas Leone

CMU-RI-TR-26-68

July 13, 2026

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Wenshan Wnag, *chair*
Deva Ramanan
Jay Karhade

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

Dedicated to my grandfather, Yasuo Yoshida, who passed away on December 9th 2024.

Abstract

Event cameras are high-speed, high-dynamic-range sensors whose microsecond temporal resolution and 120+ dB dynamic range make them essential for agile perception and SLAM systems operating in challenging lighting conditions. Despite this potential, reliable deployment of event cameras remains elusive due to two practical barriers: the hardware setup process requires careful lens selection, calibration, and bias parameter tuning that is poorly documented compared to conventional cameras; and existing event datasets are too small and insufficiently diverse to train generalizable perception models.

This thesis addresses both barriers in the context of deploying event cameras on the TartanStar multi-modal sensor payload, an existing platform developed by collaborators at CMU; neither TartanStar nor TartanAir-V2 [39] are contributions of this thesis. First, we document a systematic deployment pipeline for the Prophesee EVK4, covering lens and filter selection, intrinsic calibration using both the Metavision SDK [12] and E2Calib [29], and a characterization of the key bias parameters and their application-specific tuning. Second, we find that state-of-the-art optical flow models trained on existing real-world event datasets fail to generalize to new environments, motivating the need for more diverse training data. We propose that synthetic event data can serve as an effective pre-training substrate and demonstrate this using TartanAir-V2's native event camera modality, which renders high-fidelity event streams across 50+ diverse environments without requiring external simulation tools. Training E-RAFT on this synthetic data followed by fine-tuning on real data improves optical flow performance over fine-tuning on real data alone on both the DSEC [19] and MVSEC [44] benchmarks, confirming that TartanAir-V2's event data can bridge the real-world generalization gap. Together, these contributions lay the foundation for reliable deployment of event-based perception systems.

Acknowledgments

I would like to thank my advisor Wenshan Wang for being a great advisor throughout my time at CMU. She was always there whenever I needed help with my research. I would also like to thank my thesis commitee members, Jay Karhade and Deva Ramanan. Jay has been a great help on creating the narrative for my thesis, and stuck by me despite me at times not being the most responsive. I'm greatful for Deva, as no matter the circumstances, he has always been excited and interested in my work (even on days when I'm not enthusiastic about my own work).

I would like to thank all the collaborators for TartanStar, including but not limited to Parv Maheshwari, Ray Huang, and Tianshu Huang. Without everyone's help, the event camera integration would not have gone smoothly.

I would also like thank Srivatsa Grama Satyanarayana and Kritan Bhandari for being alongside me as we worked on training and evaluating on TartanAir-V2. As someone who had no background in learning-based approaches, Sri helped me on using and evaluating models on TartanAir-V2 and real event data. Kritan, though joined recently, has already been a tremendous help with our work on TartanAir-V2 and with the event cameras on TartanStar.

I am grateful to my colleagues in AirLab. Collaborating with those whose work intersected with mine has been a great experience, and greatly expanded my mind to other fields. For the folks of the (unofficial) dirtlab, though you all are distracting at times, conversing with you all has been a lot of fun and helped keep my mind off of work.

I want to thank my colleagues at Sandia National Laboratories who supported me throughout ever since I joined in May 2023.

I want to give a shoutout to my friends to Erik, Matthew, and Abhinav from Georgia Tech. Despite the huge time differences, we all managed to stay in touch. Whenever I was stressed, I can always confide in you all for great advice.

I also want to shoutout to my friends from Florida (KSF). It is very rare to have a huge and close-knit friend group that spans all the way back from elementary school. Between the hard and the great times, we all managed to be there for each other. It is something I shall never take for

granted.

Last but not least, I want to thank my family for their support. Without my parents and sister, I wouldn't be where I am today.

Funding

This work has been supported by Sandia National Laboratories Critical Skill Recruitment Program (CSRP) abd Graduate Fellowships for STEM Deployment (GFSD).

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

1.1.1 The Promise of Event Cameras for Agile Perception

Agile perception systems – autonomous vehicles, drones, and robotic manipulators – demand sensors that can track fast motion, operate across high-contrast lighting, and respond with minimal latency. Frame-based cameras, which have driven decades of computer vision research, impose fundamental constraints that become acute in these settings: they suffer from motion blur at high speeds, limited dynamic range in scenes with both bright and dark regions, and a latency that scales with the frame period rather than with the speed of motion.

Event cameras address all three constraints simultaneously. Rather than capturing full frames at fixed intervals, each pixel in an event camera independently monitors its local light intensity and fires an event asynchronously whenever the log-luminance changes by more than a preset threshold. This produces a sparse, continuous stream of events, each carrying a pixel coordinate, a microsecond timestamp, and a polarity. The result is a sensor with microsecond temporal resolution, a dynamic range exceeding 120 dB, and no motion blur by construction – properties that make event cameras well-suited for visual odometry, optical flow, and SLAM in fast-moving or poorly lit environments [35].

1.1.2 Two Barriers to Reliable Deployment

Despite these advantages, event cameras have not yet achieved reliable deployment in real robotic systems. Two practical barriers account for most of this gap.

The first barrier is hardware complexity. Preparing an event camera for deployment requires expertise in lens and filter selection, intrinsic calibration, and bias parameter tuning that is not required for standard cameras. Unlike frame-based cameras, event cameras expose several low-level parameters that control which brightness changes trigger events and at what rate. Getting these parameters right for a given environment(s) is essential for a clean event stream, yet the documentation for this process is sparse compared to the mature toolchains for RGB camera setup.

The second barrier is data scarcity. Learning-based perception algorithms require large, diverse training sets. The two most widely used real-world event datasets (for optical flow) – MVSEC [44] (3,000 frames) and DSEC [20] (8,000 frames) – are far smaller and less diverse than their frame-based counterparts. We find that state-of-the-art optical flow models trained on these datasets fail to generalize to new recording environments, which limits the practical utility of event-based perception.

1.2 Contributions

This thesis makes two primary contributions toward reliable deployment of event-based perception systems:

1. **A hardware deployment pipeline for the Prophesee EVK4** on the TartanStar payload. We document the full process of preparing an event camera for deployment, including lens selection (mount compatibility, image circle, pixel resolution, focal length, and aperture), optical filter selection and mounting, intrinsic calibration using two complementary methods (the Metavision SDK [12] and E2Calib [29]), and a characterization of the five key bias parameters with application-specific tuning guidance. **This thesis does not contribute the payload itself, only the event camera integration and deployment pipeline**
2. **A demonstration that TartanAir-V2 [39]'s native event data is an effective pre-training substrate for optical flow estimation.** TartanAir-

V2 includes a native event camera modality that renders event streams across 50+ diverse environments at 1,000 Hz with varying contrast thresholds, without requiring external event simulators. **TartanAir-V2 (including its event modality) is not a contribution of this thesis.** We show that pre-training E-RAFT on TartanAir-V2's synthetic event data followed by fine-tuning on real data outperforms fine-tuning on real data alone, on both the DSEC and MVSEC benchmarks.

1.3 Thesis Organization

Chapter 2 provides background on event camera operating principles, their benefits and shortcomings, and related work on event-based sensing, calibration, and learning. Chapter 3 describes the hardware deployment pipeline. Chapter 4 presents TartanAir-Event and its generation methodology. Chapter 5 covers learning-based representations for event data. Chapter 6 reports experimental results on optical flow estimation. Chapter 7 concludes with limitations and future directions.

Chapter 2

Background

2.1 What is an Event?

An event camera produces a stream of events $\mathcal{E} = \{e_i\}$ where each event

$$e_i = (x_i,\ y_i,\ t_i,\ p_i)$$

encodes the pixel location (x, y), timestamp t (in microseconds), and polarity $p \in \{-1, +1\}$. An event is triggered at pixel (x, y) when the change in log luminance L satisfies:

$$\Delta L(x, y, t) = \log I(x, y, t) - \log I(x, y, t - \Delta t)\ \geq\ \pm C \tag{2.1}$$

where C is the contrast threshold, a configurable bias parameter. Events with $p = +1$ indicate a brightness increase (ON events); $p = -1$ indicates a decrease (OFF events) [35].

2.2 How EVK4 Outputs Events

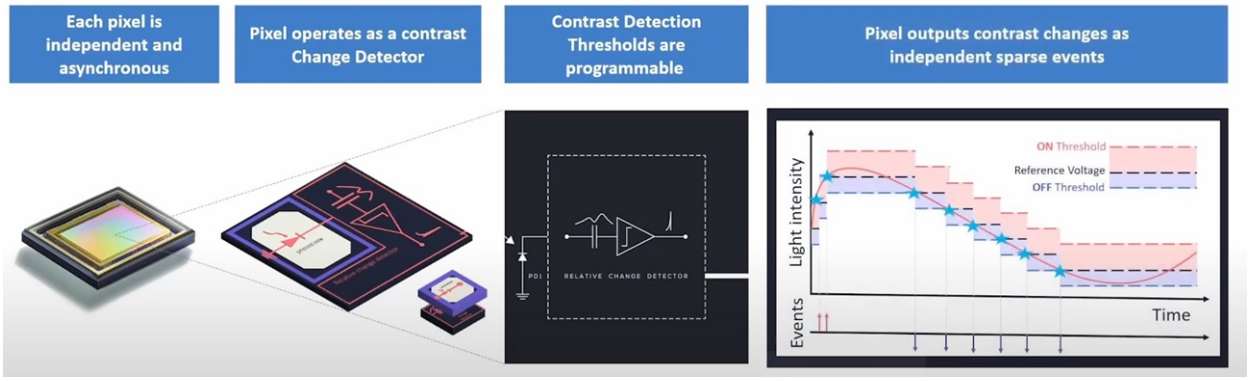


Figure 2.1: EVK4 event generation circuit. Photodiode current is log-converted to voltage, which is then compared against ON and OFF contrast thresholds to trigger events [8].

When the light hits the photo diode, the diode generates a current. The current is transformed into a voltage by the logarithmic converter. The voltage is then compared to contrast sensitivity thresholds. If the voltage surpasses crosses a threshold, an event is triggered and the relative voltage is updated to the currently measured voltage. If the voltage crosses either the ON or OFF threshold, a positive or negative polarity event is triggered respectively [8].

2.3 EVK4 Readout Architecture

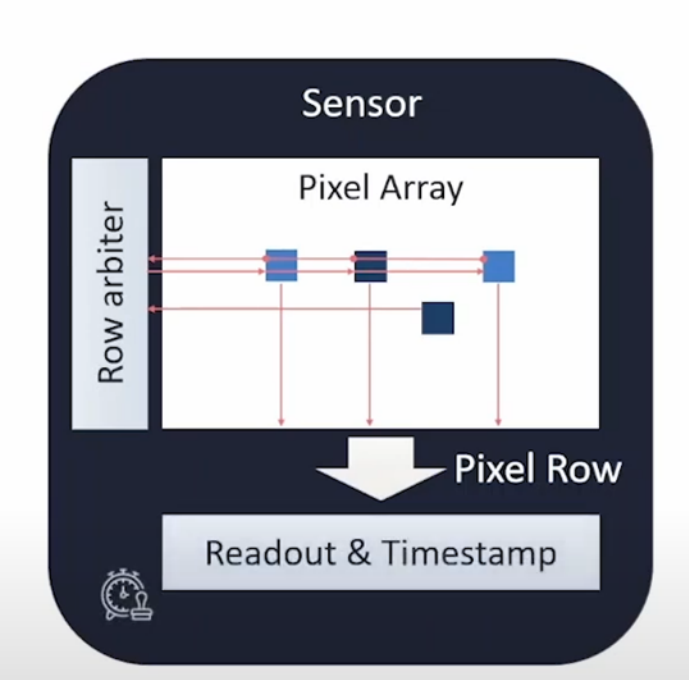
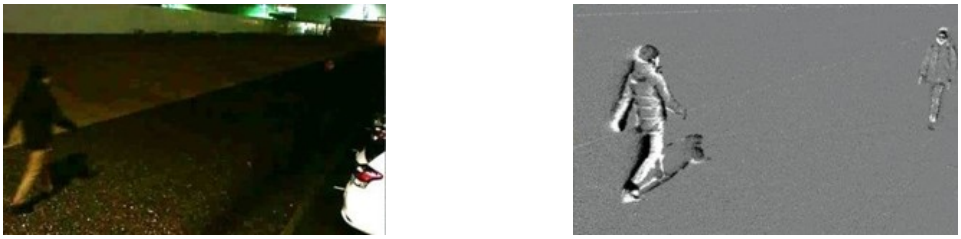


Figure 2.2: EVK4 row-arbiter architecture. Pixels that detect a threshold crossing send requests to the arbiter, which reads out and timestamps the requesting row on a first-come-first-served basis [8].

Within the Prophesee EVK4 sensor, pixels that detect a threshold crossing send a request to the *row arbiter*—a circuit that manages concurrent read requests from multiple pixels at a first-come-first-serve basis [8]. The arbiter reads the requesting pixel and its row, timestamping all detected pixels in the row during readout. Readout is passive, meaning the sensor only records data whenever a pixel sends a request . The Prophesee EVK4 implements this architecture and supports configurable bias parameters that directly control the event generation characteristics of each pixel .

2.4 Event Camera Benefits



(a) Normal RGB image captured at night (b) Same scene captured by event camera

Figure 2.3: Same scene, different cameras. Event cameras can capture more features in darker environments due to its high dynamic range [41]

Compared to frame-based cameras, event cameras offer several compelling advantages:

High Temporal Resolution Events are timestamped with microsecond precision, enabling tracking of phenomena far faster than any frame-based camera can capture [35]. Low motion blur follows directly from this asynchronous readout.

High dynamic range. The EVK4 achieves greater than 120 dB of dynamic range [11], compared to approximately 71 dB for a typical stereo camera such as the ZED X [1].

Low Power Consumption The EVK4 draws 50–70 mW typically and less than 100 mW at peak [11], compared to 101–148 mW for the ZED X [1].

Controllable Data Bandwidth Unlike standard RGB cameras where their bandwidth is static, event cameras' bandwidth is dependent on the customization of its biases and its environment. Depending on the camera's biases, event cameras can consistently have lower bandwidths than RGB cameras.

2.5 Shortcomings of Event Cameras

Despite their advantages, event cameras have inherent limitations that explain why they have not displaced frame-based cameras:

- **No Information in Static Environments** Because events are only triggered by brightness change, a perfectly static scene produces no output. Global illumination changes can cause spurious event floods.
- **Asynchronous and Sparse Output** The event stream is incompatible with conventional frame-based processing pipelines and requires new representations for use with standard deep learning architectures.

Figure 2.4: RGB, thermal, and event camera outputs in a static environment. Unlike RGB and thermal cameras, event cameras produce no output when there is no motion or brightness change in the scene.

Chapter 3

Hardware Deployment Pipeline

Preparing an event camera for deployment on TartanStar involved more setup than a conventional camera. The pipeline consists of five stages: (1) lens selection, (2) optical filter selection and mounting, (3) aperture and focus adjustment, (4) intrinsic calibration, and (5) bias parameter tuning. This chapter covers each stage for the Prophesee EVK4.

Figure 3.1: TartanStar Payload. Consists of Stereo RGB, Stereo Thermal, Radar, LIDAR, IMU, GPS, and Stereo Event

3.1 Lens Selection

Selecting the correct lens requires matching five factors: mount compatibility, image circle and sensor format, pixel size compatibility, field of view and focal length, and aperture. Each is addressed below.

3.1.1 Camera Mounts

The two mount standards that are compatible with the EVK4 are C mount and CS mount [10]. Both share the same $1'' \times 32$ TPI thread, but differ in back focus distance (BFD)—the distance from the lens flange to the image sensor [26]:

- **C mount:** BFD = 17.526 mm
- **CS mount:** BFD = 12.526 mm

The 5 mm difference is the only physical distinction. Table 3.1 summarizes compatibility between lens and camera mount types.

Table 3.1: C/CS mount compatibility matrix. A C-mount lens on a CS-mount body requires a 5 mm spacer ring.

Lens Type	On C Mount Camera	On CS Mount Camera
C Mount Lens	✓ Works perfectly	✓ Works with 5 mm ring
CS Mount Lens	× Won't focus	✓ Works perfectly

The EVK4 uses a CS mount body with a standard $1'' \times 32$ TPI thread. If a C mount lens is selected, a 5 mm ring is required to achieve correct back focus on the EVK4 (included with purchase). Most C mount lens have 4 mm of threading, in which one can use the 5 mm ring that can be screwed into the lens and onto the EVK4. However, with the Basler lens C125-0418-5M F4MM, which had 8 mm threading, a 5 mm spacer was required as shown in Figure 3.2. The spacer was placed onto the threading and then the lens was screwed into the EVK4.

Figure 3.2: Lens thread length comparison. The standard 4 mm threaded C-mount lens uses the included 5 mm ring adapter, while the 8 mm threaded Basler lens requires an additional 5 mm spacer before mounting onto the EVK4's CS-mount body.

3.1.2 Image Circle and Sensor Format

The lens *image circle* is the diameter of the projected image it produces. The image circle must be equal to or larger than the diagonal of the image sensor; otherwise, vignetting occurs—dark corners or edges appear in the captured image.

The EVK4 uses a 1/2.5" sensor format with a diagonal of 7.3 mm. Any lens selected must support the 1/2.5" image format to avoid vignetting. This means that a lens that supports the 1/2" image format would also be suitable for the EVK4.

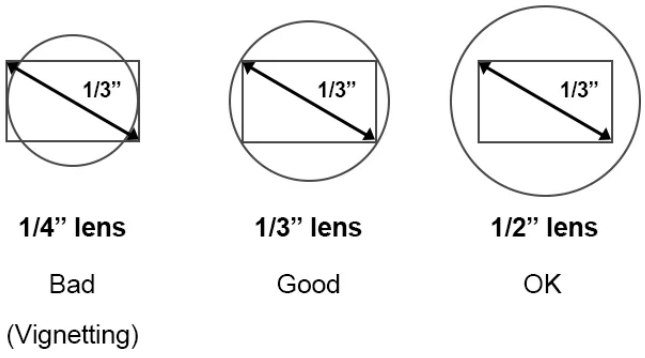


Figure 3.3: Image circle and sensor format. A lens whose image circle is smaller than the sensor diagonal produces vignetting — dark corners in the captured image [13].

3.1.3 Pixel Size Compatibility

Lens resolution must be matched to sensor pixel size and resolution. A lens with insufficient resolving power cannot take advantage of the sensor's spatial resolution; an over-specified lens adds cost without benefit. A way to measure and compare resolutions is by using the line pair metric. The line pair metric is the number of pairs of black and white lines per millimeter the sensor can observe without any aliasing [3].

The line pairs per millimeter metric is calculated in the following equation:

$$\text{Resolution required} = \frac{1}{2 \times \text{pixel pitch (mm)}} \quad [\text{lp/mm}] \tag{3.1}$$

For the EVK4, which uses the IMX 636 sensor with a 4.8 µm pixel size [11]:

$$\text{Resolution required} = \frac{1}{2 \times 0.0048} \approx 104 \text{ lp/mm}$$

A suitable lens must resolve at least 104 lp/mm. In our case, the Basler C125-0418-5M-P (4 mm, designed for 2.2 µm pixels) resolves 230 lp/mm, which satisfies this requirement with margin [15].

3.1.4 Field of View and Focal Length

Field of view (FOV) is determined by the sensor size and the lens focal length according to:

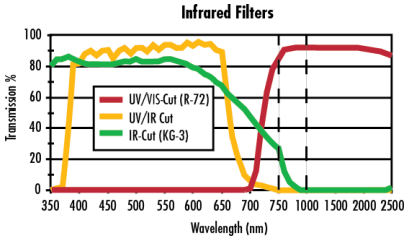
$$\text{FOV} = 2 \times \arctan\left(\frac{\text{sensor dimension}}{2 \times \text{focal length}}\right) \tag{3.2}$$

Shorter focal lengths produce wider fields of view; longer focal lengths produce narrower views. The original lens (Soyo SFA0820-5M lens) that came with the EVK4 had a focal length of 8 mm, which gave us an FOV (in 1/2.5" format) of 38.2° Horizontal and 29.11° Vertical [37]. The new lens (Basler C125-0418-5M-P) gave us an FOV (in 1/2.5" format) of 76.0° Horizontal and 58.2° Vertical [15].

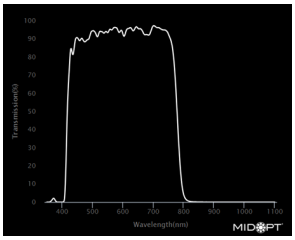
3.2 Optical Filter Selection and Mounting

Optical filters are mounted in front of the lens to selectively pass or block wavelengths of light. For event cameras, filter selection significantly affects what the sensor responds to.

3.2.1 Why Filters Matter for Event Cameras



(a) Edmund Optics filter: 20% transmission at 750 nm (green transmission curve) [4].



(b) Midwest Optical SP785: 50% transmission at 785 nm [5].

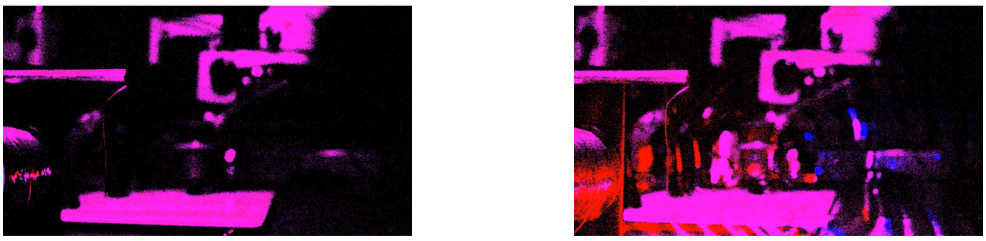
Figure 3.4: Filter transmission curves. Filter 1 (left) passes too much visible red light and IR; Filter 2 (right) provides better rejection of the Ouster OS2's 865 nm LiDAR emission.

Without an IR-cut filter, event cameras respond to near-infrared (NIR) light as well as visible light. Sources of NIR such as LIDAR and motion capture cameras add unwanted events, adding noise to events generated by visible light. An IR-cut filter removes wavelengths above approximately 650–700 nm, restricting the camera to the visible spectrum. For the TartanStar payload, the Ouster OS2's light, which emitted at 865 nm, needed to be filtered out [30].

The first filter had the following transmission curve displayed in green in Figure 3.4. The transmission is at 20% at 750 nm [4]. However, it was hard to tell when the transmission reaches 0%. The filter's documentation did not specify the specific wavelength at which the transmission is at 0%. In actuality, the first filter did not completely remove the Ouster's light. If there is an object close to the EVK4, the Ouster's light is reflected back to the sensor and can be observed.

The second filter has the following transmission curve. Although the transmission reaches 50% at 785 nm, the filter's documentation states that the transmission at 865

nm is between 0.03% and 0.04% [5]. Unlike the first filter, the second filter eliminates the Ouster's light, no matter how close an object is to the EVK4.



(a) Static scene with IR Cut Filter (b) Static scene with no IR Cut Filter.

Figure 3.5: Visualization of the EVK4 with (left) and without (right) the IR cut filter. In the right image, the ring artifacts are the Ouster's light reflecting back into the EVK4. Note: Red are negative events, blue are positive events, and purple consist of both events.

3.2.2 Filter Thread Sizes and Mounting

Optical filters are threaded onto the front of the lens. The filter thread diameter must match the lens front element thread. A common pairing for the 4 mm Basler lens used in this work is an M29 $\times$ 0.5 lens thread with an M46 $\times$ 0.75 filter adapter, as shown in Figure 3.6. Step-up rings allow a larger filter to be used on a smaller lens thread (e.g., M29$\rightarrow$M46); always verify the filter thread diameter of the specific lens before purchasing filters.

3.3 Lens Focusing

Focusing must be done manually by observing the live event stream against a dynamic, high-contrast target.

3.3.1 Choosing a Focus Target

A good focus target for an event camera is:

- **Dynamic or blinking** — to generate a continuous event stream.
- **Feature-rich** — so that sharpness is visible at many points in the image.

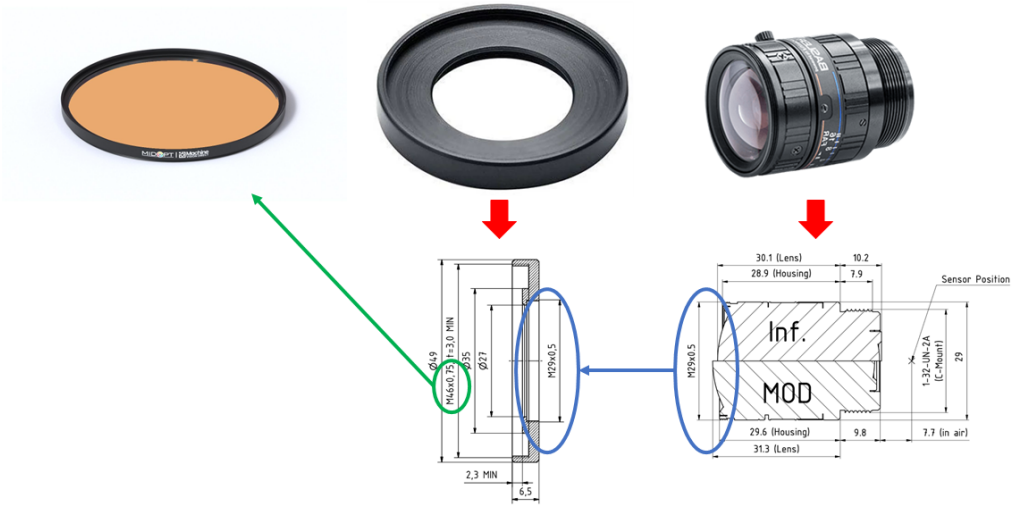


Figure 3.6: Complete lens and filter assembly on the EVK4. From left to right: the EVK4 body, the Basler 4 mm C-mount lens with 5 mm spacer, the step-up adapter (M29 to M46), and the M46 IR-Cut filter.

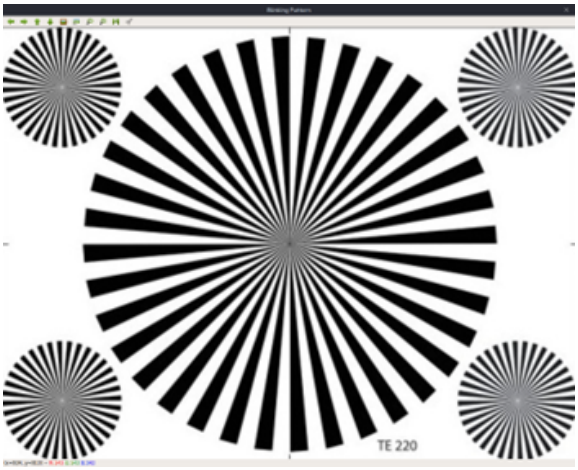


Figure 3.7: Focus target recommended by Prophesee for manual focusing of the EVK4.

- **High contrast** — to maximize the signal-to-noise ratio of events generated at feature edges.

A blinking LED checkerboard or a monitor displaying a high-contrast flickering

pattern are suitable targets.

3.3.2 Aperture, Focus, and Depth of Field

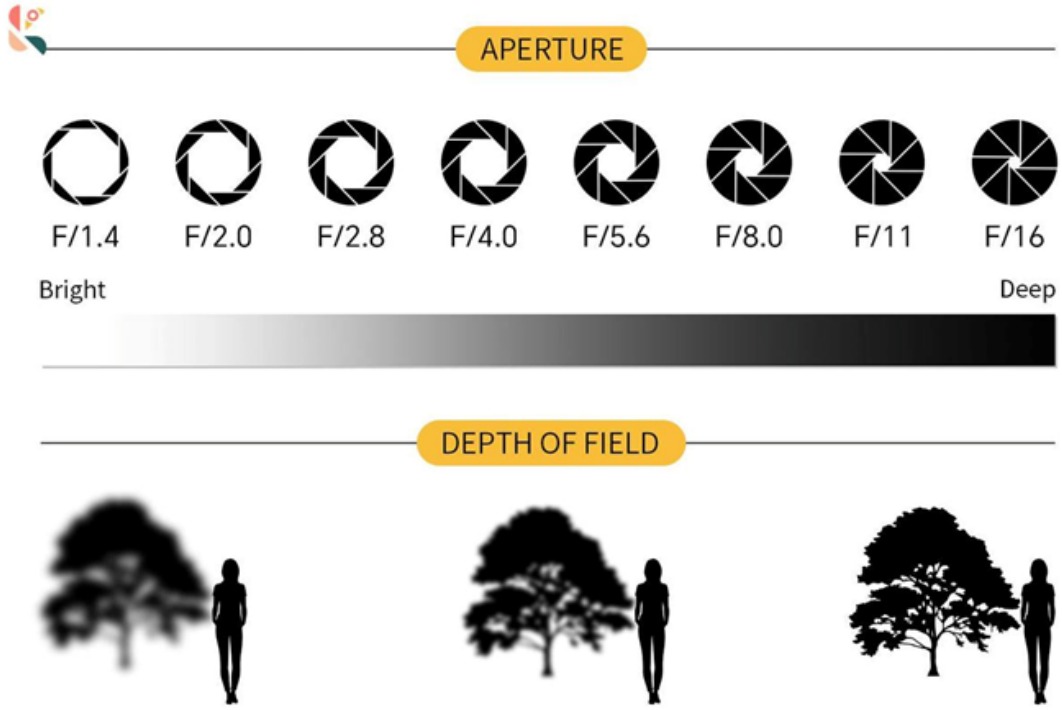


Figure 3.8: Increasing the aperture (lower f-number) admits more light but narrows the depth of field [6].

A camera lens typically has two rings. One controls the aperture (expressed as an f-number), which controls the amount of light reaching the sensor [6]. Another ring controls the focus, which adjusts the location at which the lens focuses on an object. The distance between the edge of the lens to the object that appears sharp is called the working distance [2]. Depth of field (DoF) is the range of distances over which objects appear in sharp focus.

Low f-number (e.g., f/1.4). Large aperture: shallow depth of field, excellent low-light performance. Preferred for low-light environments, subject isolation, or fast pixel response.

High f-number (e.g., f/8). Small aperture: deep depth of field, entire scene in focus. Good for wide-range 3D scenes, and outdoor imaging with controlled exposure.

For event cameras specifically, a larger aperture (lower f-number) admits more light into each pixel, which reduces background noise and improves pixel response time—both of which directly affect the quality of the event stream [7]. However, a smaller aperture (higher f-number) can help increase the depth of field, allowing a wider range in which objects appear in focus. It also helps with reducing the number of events due to limiting the amount of light entering the lens.

3.4 Intrinsic Calibration

Camera calibration estimates the intrinsic parameters that define the mapping from 3D world coordinates to 2D image pixels. For a pinhole camera model, these are encapsulated in the intrinsic matrix $\mathbf{K}$:

$$\mathbf{K} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3.3}$$

where f_x, f_y are focal lengths in pixels, (c_x, c_y) is the principal point (optical center), and additional coefficients (k_1, k_2, p_1, p_2) model lens distortion. The procedure for RGB cameras is well-established: collect images of a checkerboard pattern, detect corners, and solve for the parameters that minimize *reprojection error*—the mean pixel distance between projected 3D corners and detected 2D corners.

Event cameras do not natively produce frames, so two methods for generating frame-like inputs for standard calibration toolboxes were evaluated.

3.4.1 Metavision SDK Method

The Metavision SDK provides a native calibration pipeline that detects a blinking checkerboard pattern directly from the event stream without requiring frame reconstruction [12]. The pipeline:

1. Gathers and slices the event stream from a blinking checkerboard.
2. Detects checkerboard corners natively from events.
3. Estimates intrinsics and distortion coefficients.

The SDK provides real-time visual feedback of detected corners, making it the fastest and most user-friendly option. Its limitation is that the outputs cannot be fed directly into Kalibr for multi-camera extrinsic calibration. When given enough viewing angles (around 100 different views), the reprojection error of the calculated intrinsics are less than 1 pixel, typically between 0.6 and 0.8.

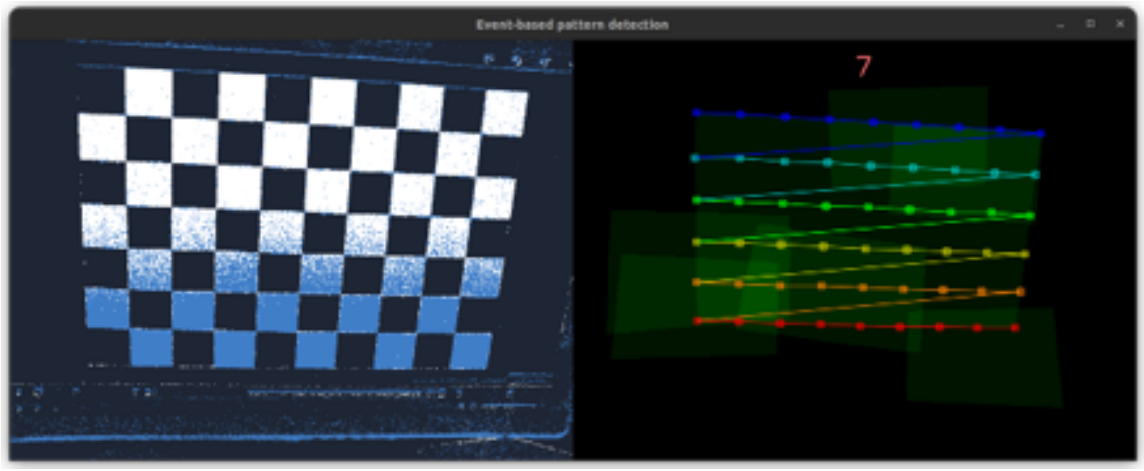
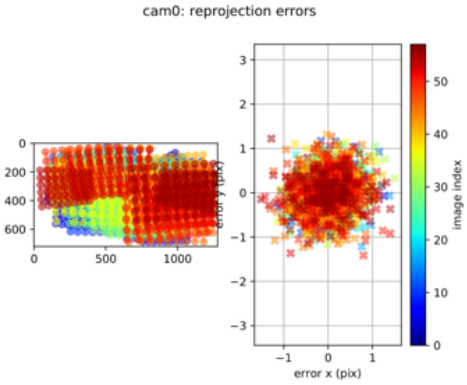
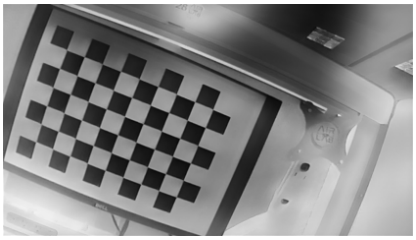


Figure 3.9: The green circles indicate detected corners by Metavision's Calibration Software [12].

3.4.2 E2Calib Method



(a) Kalibr Calibration Results



(b) Computer Screen with Checkerboard Pattern

Figure 3.10: Kalibr calibration results from a recording with a checkerboard pattern on a computer screen

E2Calib [29] converts raw event data into frames using E2VID [32], a learned event-to-video reconstruction network. The pipeline:

1. Records a static checkerboard as the camera (or board) is moved.
2. Converts the raw event recording or ROS bag to HDF5 format.
3. Reconstructs grayscale frames from events using E2VID.
4. Converts the frames to a ROS bag and calibrates using Kalibr [16].

This method is compatible with multi-camera extrinsic calibration via Kalibr triggers, which is a significant advantage in multi-sensor rigs. However, the file conversions are time consuming and there are no real-time feedback of corner detections during recording. From our experience, the Kalibr approach tends to produce lower reprojection errors than the Metavision approach, with reprojection error around 0.545 pixels ($[\pm 0.395,\ \pm 0.375]$ pixels).

3.4.3 Method Comparison and Selection

Both methods yield sub-pixel reprojection errors and comparable intrinsic estimates. Table 3.2 summarizes the key tradeoffs.

Table 3.2: Comparison of event camera intrinsic calibration methods. Both achieve sub-pixel reprojection error; the Metavision SDK is faster while E2Calib supports multi-sensor extrinsic calibration.

	Metavision SDK	**E2Calib**
Real-time feedback	✓Yes	× No
File conversions	None	Multiple
Extrinsic calibration	× No	✓Yes (Kalibr)

The Metavision SDK method is preferred for single-camera setups due to its speed and real-time feedback, whereas E2Calib is preferable when calibrating for extrinsics with other sensors.

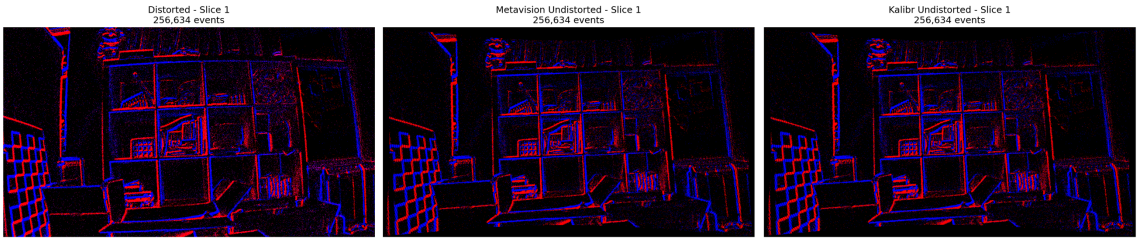


Figure 3.11: Intrinsic calibration results for the EVK4. Both the E2Calib pipeline and the Metavision SDK method achieve sub-pixel reprojection error.

3.5 Bias Parameter Tuning

Bias parameters are hardware-level settings in the EVK4 that control how events are generated at the pixel level. Getting these right is essential for a clean, informative event stream. The section below are based off of the tutorial provided by Prophesee [9].

3.5.1 Key Pixel Characteristics

Three pixel-level characteristics are most relevant to bias tuning:

Contrast sensitivity. The minimum luminance change a pixel can detect, set by the contrast threshold C in Equation 2.1.

Background Rate. The average number of events generated per pixel per second under static lighting conditions, measured in Hz/pixel. Background rate consists of photonic noise (noise from the environment) and electronic noise (noise from pixel). Even with no motion, photonic and electronic noise cause some pixels to fire spuriously, setting the noise floor of the camera.

Pixel response time and latency. The time taken for a pixel to respond to a brightness change exceeding the threshold. Response times are characterized as a probability distribution; pixel latency is the mode of this distribution. *Jitter* is the interquartile range of the probability distribution, measuring the precision of the response times of pixels.

All three of these characteristics impact the overall event rate of the EVK4, which is often measured as mega events per second.

3.5.2 `Bias_Diff_ON` and `Bias_Diff_OFF`

These biases set the contrast threshold C for ON (positive) and OFF (negative) events, respectively. Increasing them raises the threshold:

- $\uparrow$ `Bias_Diff_ON/OFF` $\rightarrow \downarrow$ Pixel Sensitivity, $\downarrow$ Background Rate, $\uparrow$ Pixel Latency, $\downarrow$ Event Rate

The key tradeoff is sensitivity vs. noise: lower thresholds capture finer motion but at the cost of higher background noise.

3.5.3 `Bias_Diff_LPF` (Low-Pass Filter)

This bias controls the cutoff frequency for high frequency signals, acting as a low-pass filter:

- ↑ `Bias_Diff_LPF` → ↓ Pixel Latency, ↑ Background Rate, ↑ Event Rate, ↓ Noise.

Increasing this bias decreases pixel latency at the cost of increasing high frequency noise. Decreasing this bias does help smooth the signal however.

3.5.4 `Bias_Diff_HPF` (High-Pass Filter)

This bias controls the cutoff below which slow brightness changes are suppressed:

- ↑ `Bias_Diff_HPF` → ↓ Noise, ↓ Event Rate

Decreasing this bias preserves slow and fine motion. Unlike the other biases, increasing the high pass filter does not impact the pixel latency.

3.5.5 `Bias_refr` (Refractory Period)

The refractory period is the minimum time that must elapse before a pixel can fire again after generating an event.

- ↑ `Bias_refr` → ↑ Noise, ↑ Event Rate

Increasing this bias actually reduces the refractory period. This improves the pixel availability at the cost of an increased event rate, specifically an increase when experiencing a large contrast change.

3.5.6 Regions of Interest and Non-Interest

The EVK4 supports spatially selective activation:

- **Region of Interest (ROI):** Active pixels that generate events normally.
- **Region of Non-Interest (RONI):** Inactive pixels that are suppressed.

ROI/RONI allow the operator to focus the event stream on a specific image region, reducing power consumption, read-out load, and data rate without adjusting the bias parameters globally.

3.5.7 Example 1: Eye Tracking

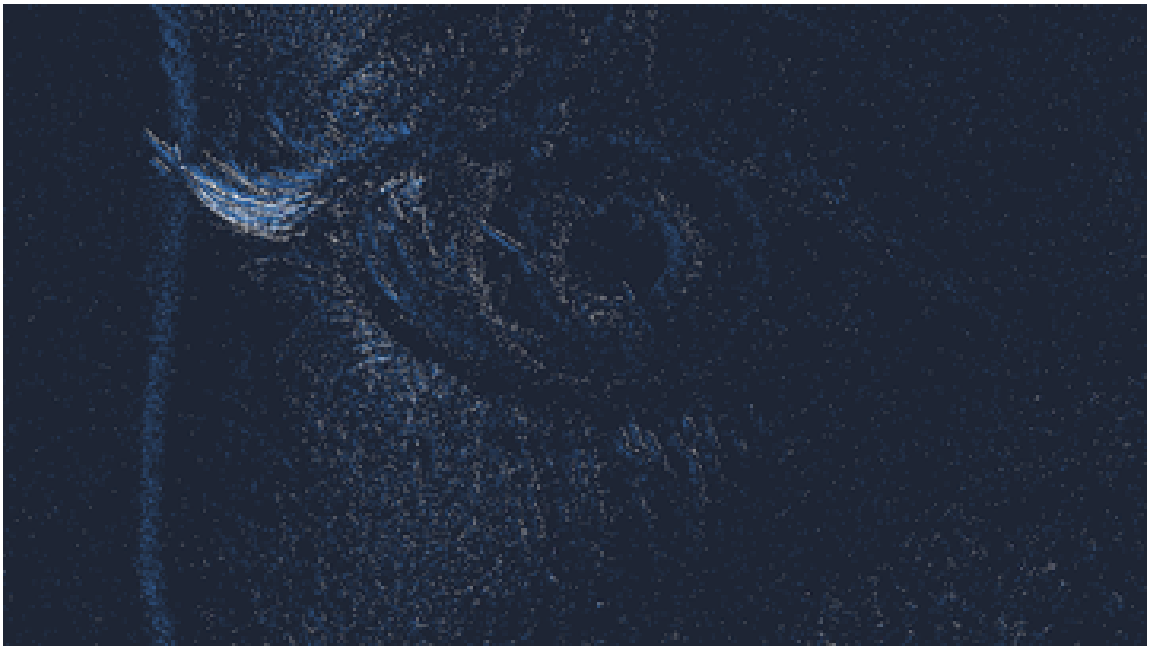


Figure 3.12: Eye captured by EVK4 [8].

For eye tracking, we would want to capture slower motions, so we would decrease the contrast (both on and off) and highpass biases. To only track the iris of the eye, we would use the ROI mask to only capture the eye and ignore rest of the face.

Table 3.3 summarizes recommended configurations for the eye tracking example

Table 3.3: Recommended bias configurations for eye tracking: high sensitivity, slow-motion capture, and tight ROI around the iris.

Parameter	**Eye Tracking**
`Bias_Diff_ON/OFF`	Decrease (more sensitive)
`Bias_Diff_HPF`	Decrease (capture slow motion)
ROI	Tight (iris region only)

Figure 3.13: Street recording of EVK4 [9].

3.5.8 Example 2: Traffic

When using an event camera in an outdoor, urban environment with traffic, the event rate would be high due to the dynamic environment. To decrease the event rate, we would increase the contrast biases. We would also increase the refractory period (decrease `Bias_refr`) to reduce the number of events generated in large contrast change, such as when entering and leaving a tunnel. To reduce the events generated by street and car lights, we would increase the low pass filter bias.

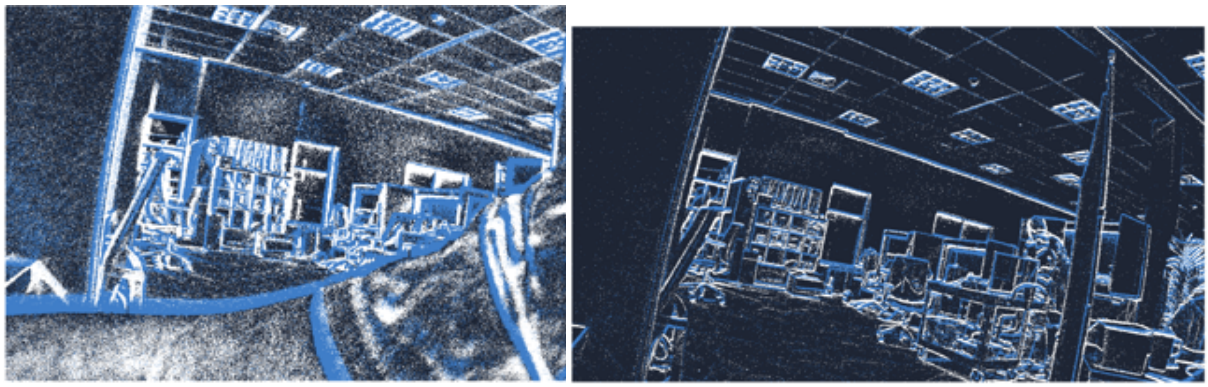
Table 3.4 summarizes recommended configurations for the urban driving scenario.

Table 3.4: Recommended bias configurations for an urban driving scenario: higher contrast thresholds and LPF to reduce event rate and suppress street-light flicker.

Parameter	**Traffic (Noise Reduction)**
`Bias_Diff_ON/OFF`	Increase (less noise)
`Bias_Diff_LPF`	Increase
`Bias_refr`	Decrease

3.5.9 Tuning for Multiple Environments

A practical question for deployment is whether to re-tune biases between environments. M3ED [14], a multi robot and multi environment dataset, argues against per-environment tuning, on the grounds that biases affect event generation even in static regions, and that consistent bias settings improve reproducibility across sequences. Initially, for the TartanStar Payload, there needed to be separate biases for outdoor day and night environments. When recording the EVK4 in the outdoors, the event rate would be so high that we exceed the USB 3.0 bandwidth and fail to capture all the events. However, using the same biases at night would reduce the event rate to the point that the cameras wouldn't capture any distinct features. Opening up the lens' aperture would let in more light, allowing the event camera to capture more features in darker environments. Also, instead of having both event cameras plugged into the same USB hub, one camera would instead be plugged directly into the NVIDIA Orin running the payload. The other camera would be left plugged into the hub. This significantly reduced the number of dropped events and allowed the cameras to run at higher event rates. These two changes may allow the cameras to use one set of biases instead of multiple.



(a) Default Biases (b) Increased `Bias_Diff_ON/OFF`

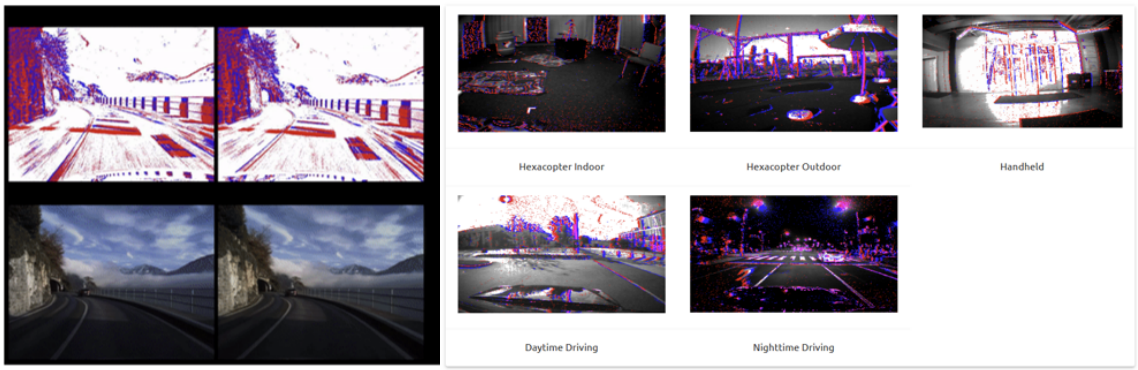
Figure 3.14: Comparing event cameras with default and increased `Bias_Diff_ON/OFF` biases. The image with increased biases has less background noise.

Chapter 4

Datasets and Simulation

4.1 Existing Event Camera Datasets

Several event camera datasets have been published for different purposes, such as benchmarking optical flow, object detection, human pose tracking, and SLAM. For this thesis, the main focus is on datasets that benchmark on optical flow. The most widely used datasets for event-based optical flow and visual odometry are MVSEC [44] and DSEC [20]. DSEC is a driving-based stereo event and frame-based (RGB) dataset that also include LiDAR, GPS, IMU, and optical flow ground truth. MVSEC is also a stereo event and frame-based (gray) camera dataset with LiDAR, GPS, and IMU, but also has motion capture for indoor environments. Unlike DSEC that only has driving-based trajectories with a car, MVSEC has a variety of non-car based trajectories, such as hexacopter, handheld, and motorcycle. However, there are limitations with the two datasets. Most of DSEC and MVSEC trajectories feature straight lines, and most rotations are limited to the vertical axis. Also, the size of the two datasets are relatively small, with DSEC only having around 8,000 frames and MVSEC with about 3,000 frames. The lack of variety in motions and the small size of the two datasets limit the generalization of event-based optical flow models.



(a) DSEC (b) MVSEC

Figure 4.1: Visualizations of DSEC and MVSEC

4.2 Dataset Limitations and the Case for Simulation

Three factors limit existing real-world event camera datasets:

- Some datasets were recorded with older, lower-resolution event cameras.
- Event cameras are expensive and not yet widely available, limiting dataset scale and diversity.
- It is difficult to capture a large variety of environments with a physical camera rig.

Simulation offers a path to large-scale, diverse, densely annotated event data by generating synthetic events from pre-existing datasets or from photorealistic rendered scenes, where ground truth is available by construction.

4.3 Event Simulators

Several simulators have been developed to generate synthetic event data off of videos or rendered scenes in simulated environments. The first of which is ESIM [33] and VID2E [18]. ESIM is the first modern event simulator, which operates by being integrated with a rendering engine. This close integration allowed ESIM to adaptively sample frames and generate events based on either change in brightness or pixel

intensity. VID2E is the first event simulator that can take in any RGB video and generate events. Instead of being integrated with a rendering engine, VID2E uses the interpolation model SuperSloMo [23] to adaptively interpolate frames based on pixel displacement. To account for different event cameras, the trigger thresholds are samples from a uniform thresholds between a minimum and maximum trigger threshold.

Another event simulator called V2E [22] uses a similar approach to event generation as VID2E but also simulates other characteristics of event cameras such as hot pixels, leak noise events, and shot noise. According to [22], hot pixels are events that fire off at a high frequency, even without any input. Leak noise events are low frequency events that are set off due to small currents triggering the change detector reset switch. Shot noise is noise caused by photon counting. The effect of shot noise is exacerbated in dark environments.

The most recent event camera simulator is DVS-Voltmeter [28]. Unlike ESIM/VID2E and V2E, DVS-Voltmeter is a physics-driven simulator, based off of the circuit of the first event camera. This allows DVS-Voltmeter to model event generation and noise better than the previous simulators and also generate timestamps more accurately for each event. ESIM/VID2E and V2E linearly interpolate between frames to assign timestamps after generating events. However, DVS-Voltmeter is the slowest to run, as it lacks a GPU implementation.

In the DVS-Voltmeter paper [28], semgentation and intensity recreation models trained off DVS-Voltmeter generated events out-performed models trained on VID2E and V2E data. However, the performance differences between models trained on VID2E and V2E data were similar, and in some cases VID2E models outperformed V2E models.

4.3.1 Simulated Event Datasets

In the Event Camera Data Dense Pretraining paper [40], a synthetic event dataset called E-Tartan was created by using EMA-VFI [24] to interpolate sequences from TartanAir at a higher frequency and then used V2E to generate events. BlinkFlow [27], another synthetic event-based dataset, is generated from a custom Blender-based rendering suite called BlinkSim. BlinkSim offers the user to either sample renders

based on evenly spaced timestamps or based on maximum pixel displacement or brightness change between frames. BlinkSim also provides the user to generate events using ESIM, V2E, or DVS-Voltmeter. BlinkFlow also consist of sequences from TartanAir. Similar to E-Tartan, BlinkFlow interpolated the sequences and then generated events using one of three event simulators. However, BlinkFlow only uses 20 sequences from TartanAir, as they were the few sequences that did not suffer from interpolation artifacts.

4.3.2 Limitations of Generating Events off of Videos

The main issue of generating events off of other datasets is that most image sequences are sampled at low frequencies, such as 10 - 30 Hz. Trying to interpolating these image sequences to higher frequencies will risk interpolation artifacts that cause erroneous events to be generated. Three interpolation models that are often used in conjunction with V2E and VID2E are EMA-VFI [24], FILM [34], and SuperSlomo [23]. To demonstrate the limitations of SOTA interpolation models, a sequence from TartanGround [31] was interpolated with 32 frames in between the original frames. The output of all three models are shown in 4.2

(a) EMA-VFI (b) FILM (c) SuperSlomo

Figure 4.2: Visualization of artifacts from SOTA interpolation models

The best way to generate synthetic events with event simulators is to use them alongside rendering engines, as they provides greater control over sampling frequencies compared to recorded sequences.

4.4 TartanAir-V2 as a Training Data Source

TartanAir-V2 [39] is a large scale synthetic multi-modal dataset that expands the original TartanAir [38] dataset with more environments and additional sensor modalities, including event cameras. TartanAir-V2, including its event camera modality, is not a contribution of this thesis; it is only a source of synthetic training data for this thesis.

4.4.1 Synthetic Event Generation with No Interpolation

Unlike prior work that applied external event simulators (VID2E, V2E) to low-frame-rate RGB video, TartanAir-V2 generated events with ESIM on 1,000 Hz sampled images. This avoids the interpolation artifacts that degrade event quality when upsampling video from 10 Hz to approximately 300 Hz as seen in the previous section, and produces cleaner, more realistic events that better match the characteristics of physical sensors.

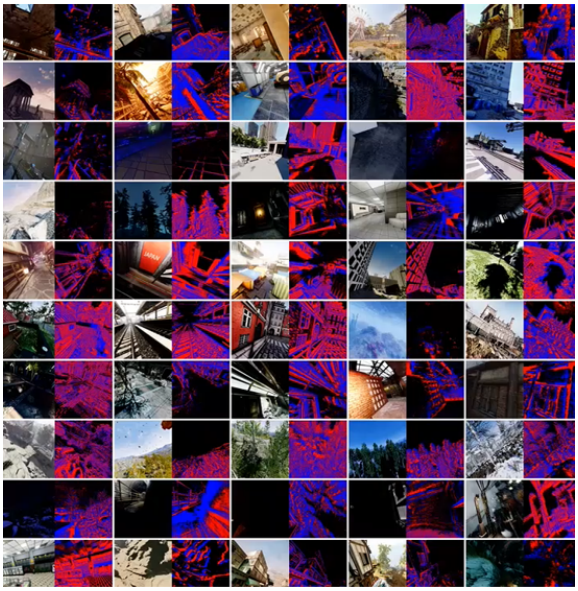


Figure 4.3: Events from different trajectories in TartanAir-V2

The TartanAir-V2 event data used for training in this thesis contains over 700,000 reference frames at 640×640 resolution across 65 simulated environments rendered

in Unreal Engine [39]. Events are generated with contrast thresholds varied across $C \in [0.15, 1.0]$ to improve model robustness to sensor-specific threshold settings. Dense optical flow ground truth is available for all RGB frames. Table 4.1 places this data in context against existing real-world benchmarks.

Table 4.1: Comparison of event camera datasets. MVSEC and DSEC are real-world benchmarks used for evaluation. TartanAir-V2 (Sim) is the synthetic training source used in this work; its event modality is a feature of the pre-existing TartanAir-V2 simulator.

Dataset	**Motion**	**Frames**	**Resolution**	**Flow GT**	**Real/Sim**
MVSEC	Drone, Motorcycle	3,000	260×346	Sparse	Real
DSEC	Car	8,000	640×480	Sparse	Real
TartanAir-V2 (Sim)	Random	700,000+	640×640	Dense	Sim

Chapter 5

Learning from Events

5.1 The Representation Problem

Standard learning-based architectures expect dense, grid-structured input tensors. Raw event streams are asynchronous, sparse, and variable-length, making them incompatible with this expectation. Several representations have been developed to bridge this gap, each with different tradeoffs between temporal fidelity, computational cost, and compatibility with existing architectures. This section covers some of the common representations used for event data in visualization and learning-based applications.

5.2 Event Frame

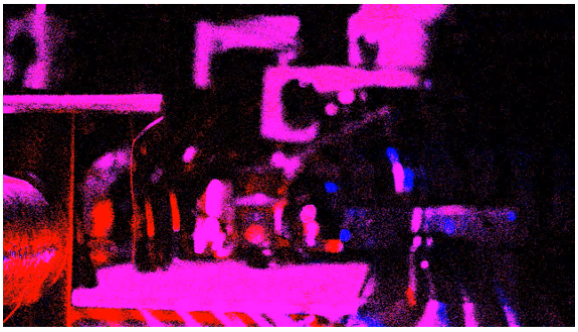


Figure 5.1: Events represented as an RGB image

The simplest representation encodes positive events into one channel and negative events into another (blue and red respectively) of a 2D image, accumulated over a fixed time window. This is easy to compute and visualize, but discards all temporal information within the window—events that occurred at the beginning and end of the window are indistinguishable.

5.3 Time Surface

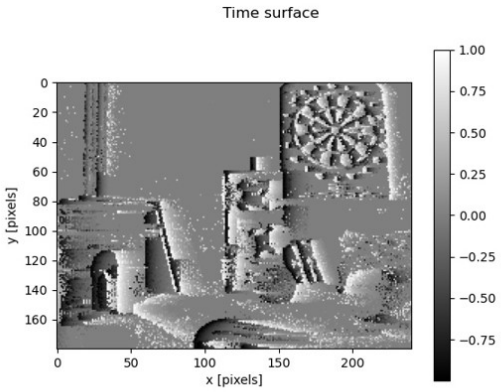


Figure 5.2: Events represented as an 2D Time Surface [36]

A time surface stores the timestamp of the most recent event at each pixel location, producing a 2D map that encodes the *recency* of motion across the image. The time

surface is defined as:

$$\mathcal{T}(x, y) = \exp\left(-\frac{t_{\text{ref}} - t_{\text{last}}(x, y)}{\tau}\right)$$

where τ is a decay constant and $t_{\text{last}}(x, y)$ is the timestamp of the most recent event at pixel (x, y). This preserves more temporal structure than the event frame, but still discards polarity information and is sensitive to noise.

The event frame and time surface are used for visualization purposes rather than for learning representations.

5.4 4D Voxel Grid

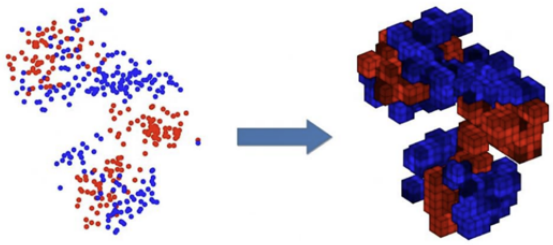


Figure 5.3: Events represented as a 4D Voxel [45]

The voxel grid accumulates events into a 3D tensor of shape $[T, H, W]$, where T is the number of time bins [17]. Each event is assigned to the nearest time bin based on its normalized timestamp. Events are aggregated by polarity and bin index, preserving both temporal structure and polarity information. This representation is compatible with standard learning architectures. However, the voxel grid representation removes the anachronistic nature of event cameras and forces event data to work with frame-based paradigms.

5.5 Spiked Neural Networks

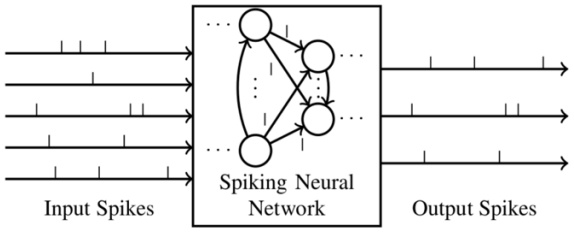


Figure 5.4: Events fed directly into an SNN

Spiked Neural Networks (SNNs) operate directly on the asynchronous event stream without requiring any dense representation. Each neuron communicates via discrete spikes, analogous to the events themselves, which offers low power consumption and avoids the temporal discretization of voxel-based representations. However, SNNs remain significantly harder to train than standard learning architectures: spikes are non-differentiable, limiting the applicability of standard gradient-based optimization [35]. SNN architectures are also less mature than the CNN and transformer architectures widely used in frame-based vision.

5.6 Event Tensors in State-of-the-Art Algorithms

State-of-the-art methods for event-based approaches, including E-RAFT [21] for optical flow use voxel-grid tensor representations as input to modified versions of established frame-based architectures. This demonstrates that well-designed tensor representations are sufficient to achieve competitive performance, without requiring fully asynchronous processing.

Chapter 6

Experiments

6.1 Training and Assessing E-RAFT on TartanAir-V2

Optical flow estimation is a natural benchmark for event cameras given their high temporal resolution. We evaluate E-RAFT [21], a state-of-the-art event-based optical flow model that takes event voxel grids as input. Three training configurations are compared:

- **TAV2**: E-RAFT trained solely on TartanAir-V2's native synthetic event data.
- **DSEC / MVSEC**: E-RAFT trained solely on the target real-world dataset.
- **TAV2 + DSEC / TAV2 + MVSEC**: E-RAFT pre-trained on TartanAir-V2 synthetic event data, then fine-tuned on the target real-world dataset.

6.1.1 TartanAir-V2 Training Procedures

E-RAFT is trained for five epochs with batch size 32, Adam optimizer, learning rate 10^{-4}, and a one-cycle learning rate schedule. Augmentations include uniform noise injection to narrow the sim-to-real gap, random dropout of 0–90% of the oldest events (based on their timestamps) in each slice, random cropping to 480×480, and horizontal and vertical flipping with probabilities of 50% and 10%, respectively. The uniform noise injection and random drop out was done to reduce the sim-to-real

gap of the event data, with the random drop out partially inspired by photometric augmentations in [25]. The train set consists of all environments and trajectories that have generated event data. The test set consists of the CountryHouse, MiddleEast, and ShoreCaves environments.

6.1.2 DSEC Training Procedures

Following [21], E-RAFT is trained for 40 epochs with Adam at 10^{-4}. Augmentations include random cropping to 288×384 and horizontal and vertical flipping. Fine-tuning runs for an additional 10 epochs at full resolution with learning rate 10^{-5}. DSEC has separate train and test sequences.

6.1.3 MVSEC Training Procedures

The following is also taken from [21]. E-RAFT is trained on the outdoor_day2 sequence for 10 epochs with Adam at 10^{-4}. Augmentations include random cropping to 256×256 and horizontal and vertical flipping. Fine-tuning runs for an additional 5 epochs at full resolution with learning rate 10^{-5}. To follow [21], we only train on Outdoor Day 2 sequence and evaluate on the Outdoor Day 1 sequence.

6.1.4 UFM Baseline

The baseline is **UFM** [42] (specifically UFM-980-Refine model), a non-event-based optical flow method. UFM will calculate optical flow off of 2D time surface frames of events with a decay constant (τ) set to 0.5. UFM is included to demonstrate the importance of event representation and event-based models.

6.1.5 Performance Metrics

Performance is reported using Average Endpoint Error (EPE, lower is better) and percentage of pixels with error within 1, 2, and 3 pixels (higher is better).

6.2 Evaluation on DSEC

Table 6.1 reports optical flow results on the DSEC test set. E-RAFT trained on TartanAir-V2 alone (TAV2) reduces EPE from 7.300 for the UFM baseline to 1.161, showing how simply compressing events into 2D representation does not result in high quality optical flow. E-RAFT trained and finetuned on DSEC further reduces EPE to 0.904. However, E-RAFT that is pre-trained on TartanAir-V2 and finetuned on DSEC has an EPE of 0.847. Although not a significant increase compared to the DSEC-Only model, the combined training of TartanAir-V2 and DSEC did improve the overall accuracy.

Table 6.1: E-RAFT optical flow results on the DSEC test set.

Method	**Training Data**	**EPE ↓**	**1px% ↑**	**2px% ↑**	**3px% ↑**
UFM-980-Refine	—	7.300	26.78	46.25	56.38
E-RAFT	TAV2	1.161	77.05	90.93	94.63
E-RAFT	DSEC	0.904	83.31	93.51	96.32
E-RAFT	TAV2 + DSEC	**0.847**	**84.79**	**95.39**	**96.84**

6.3 Evaluation on MVSEC (Outdoor Day 1)

Table 6.2 reports optical flow results on a small sequence of the MVSEC outdoor_day1 sequence. This sequence was chosen as it is the sequence used for evaluation in the original E-RAFT paper [21] E-RAFT trained on TartanAir-V2 (TAV2) achieves EPE of 1.555, outperforming the UFM baseline (1.839) despite no exposure to real event data. Training and fine-tuning on MVSEC reduces the EPE to 0.846. The model that is pre-trained on TartanAir-V2 and fine-tuned on MVSEC achieves the best result of 0.583 EPE and 99.24% of pixels within 3 pixels.

Table 6.2: E-RAFT optical flow results on the MVSEC outdoor_day1 sequence. Metrics as in Table 6.1.

Method	**Training Data**	**EPE ↓**	**1px% ↑**	**2px% ↑**	**3px% ↑**
UFM-980-Refine	—	1.839	47.12	66.28	78.17
E-RAFT	TAV2	1.555	51.48	73.57	84.36
E-RAFT	MVSEC	0.846	71.41	92.57	97.87
E-RAFT	TAV2 + MVSEC	**0.583**	**84.83**	**96.82**	**99.24**

6.4 Ablation: Sampling Rate (10 Hz Interpolated vs. 1,000 Hz Direct)

To assess the cost of interpolation artifacts, E-RAFT is trained on two variants of TartanAir-V2 and evaluated on the TartanAir-V2, DSEC, and MVSEC test sets:

- **10 Hz**: Events generated from RGB images subsampled at 10 Hz and interpolated to 1000 Hz using EMA-VFI [24].
- **1,000 Hz**: Events generated directly by using the 1,000 Hz images without interpolation.

Note: Only the first trajectory of each environment in TartanAir-V2 are used for training the two models.

Table 6.3 shows that 1,000 Hz direct generation consistently reduces EPE compared to 10 Hz interpolation, with the large gains on DSEC (1.328 vs. 1.863) and modest gains on TartanAir-V2 (2.040 vs. 2.540). In particular for DSEC, the 1,000 Hz model has 7.39% more pixels within the 1 pixel error than the 10 Hz model. The improvement on MVSEC is negligible (1.588 vs. 1.598). The small improvement maybe due to how small the outdoor day 1 sequence is compared to the other sequences in TartanAir-V2 and DSEC. MVSEC's smaller camera resolution could be another potential explantion for the lack of performance difference.

Table 6.3: Ablation comparing E-RAFT trained on 10 Hz interpolated events vs. 1,000 Hz direct events from TartanAir-V2. Evaluated on three test sets.

Test Set	**Sampling**	**EPE ↓**	**1px% ↑**	**2px% ↑**	**3px% ↑**
TartanAir-V2	10 Hz	2.540	55.01	76.10	84.77
	1,000 Hz	**2.040**	**51.84**	**75.19**	**84.28**
DSEC	10 Hz	1.863	63.27	83.25	89.43
	1,000 Hz	**1.328**	**70.66**	**88.08**	**93.84**
MVSEC	10 Hz	1.598	48.25	72.77	84.16
	1,000 Hz	**1.588**	**48.44**	**73.11**	**84.48**

6.5 Qualitative Results

We also ran E-RAFT models trained on DSEC and TartanAir-V2 + DSEC on an indoor walking sequence recorded on TartanStar. We also ran UFM on the RGB images to provide a reference optical flow to compare to. UFM's and E-RAFT's optical flow are not one to one due to the different FOVs of the respective cameras' lens.

For the following images, the top right is the RGB frame of the left ZED X camera, the bottom left is the event frame, the top middle is UFM's optical flow, the bottom middle is the optical flow of E-RAFT trained on DSEC, and the bottom right is the optical flow of E-RAFT pretrained on TartanAir-V2 and finetuned on DSEC (TAV2 + DSEC). These two were specfic E-RAFT models were chosen as the DSEC dataset is larger than the MVSEC dataset.

Initially, the models were evaluated on 30 Hz, the frequency of the synchronization signal used to time sync all of TartanStar's modalities.

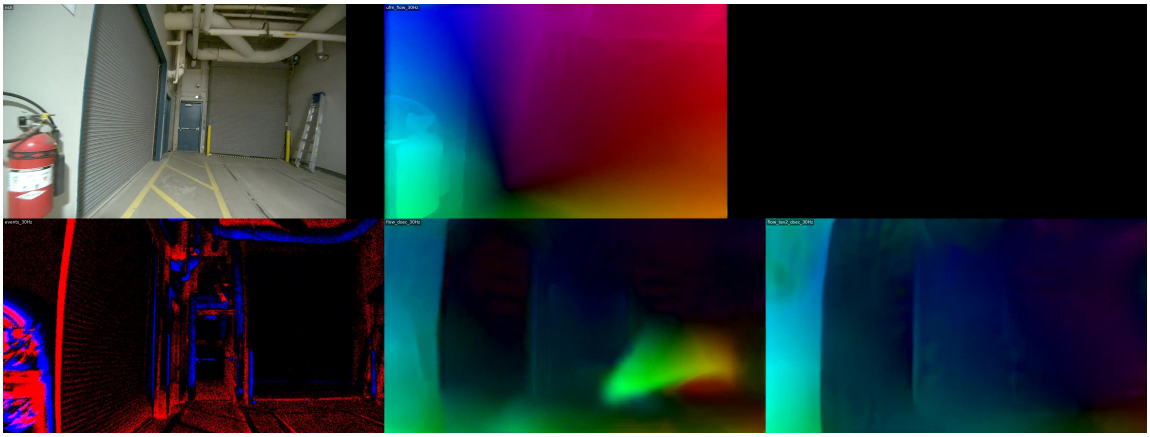


Figure 6.1: E-RAFT (DSEC) (in the bottom middle) has a green artifact that is not seen in either UFM or E-RAFT (TAV2 + DSEC) (top middle and bottom right respectively)

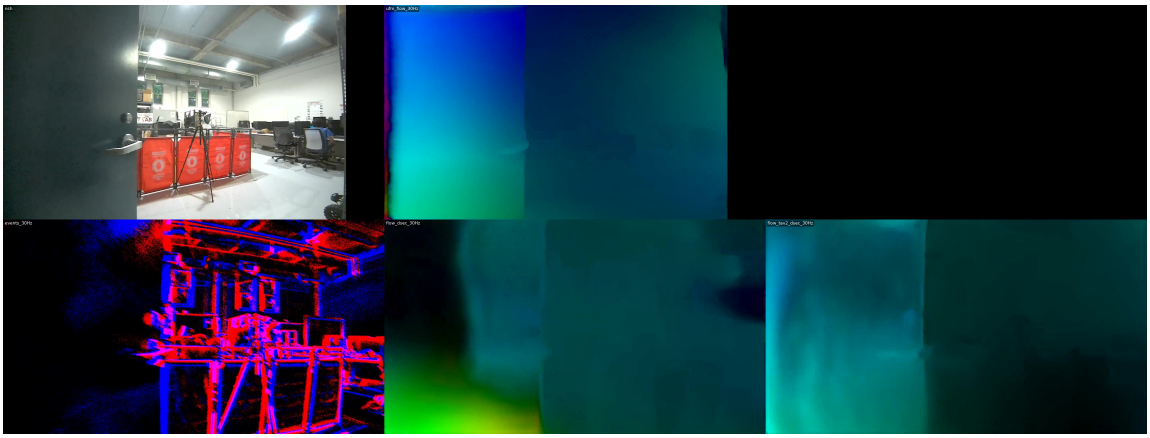


Figure 6.2: E-RAFT (TAV2 + DSEC) and UFM capture the door. E-RAFT (DSEC) fails to capture the door instead has another green artifact in the bottom left corner.

Despite the artifacts and sometimes failing to capture parts of the scene, E-RAFT (DSEC) performed somewhat similarly compared to E-RAFT (TAV2 + DSEC). However, the movements in the sequence are relatively small (e.g. walking forward and turning around). To simulate larger movements, we processed the indoor sequence at 5 Hz. The smaller sampling size meant that there would be large movements between each frame. After running the models on the 5 Hz sequence, we noticed a reduction in performance for the E-RAFT (DSEC) model:

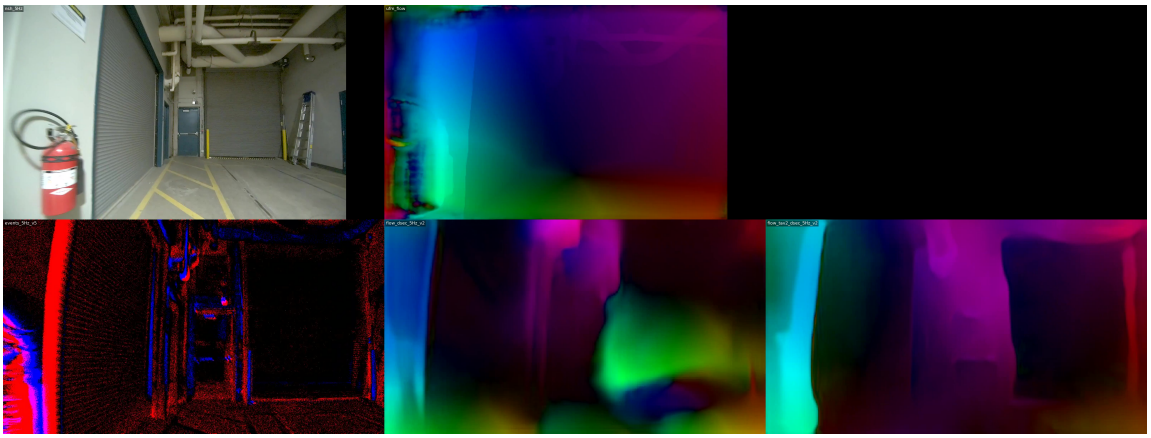


Figure 6.3: Similar to before, E-RAFT (DSEC) has the same green artifact. Also, E-RAFT (TAV2 + DSEC) and UFM have similar optical flow orientation and both also capture the fire hydrant.

Decreasing the sampling rate to 5 Hz displayed how E-RAFT (DSEC) fails to perform well under huge motions. E-RAFT (TAV2 + DSEC) manages to capture more features clearly in the scenes. Also, E-RAFT (TAV2 + DSEC) and UFM displayed similar optical flow orientations in the 5 Hz evaluation than the 30 Hz evaluation.

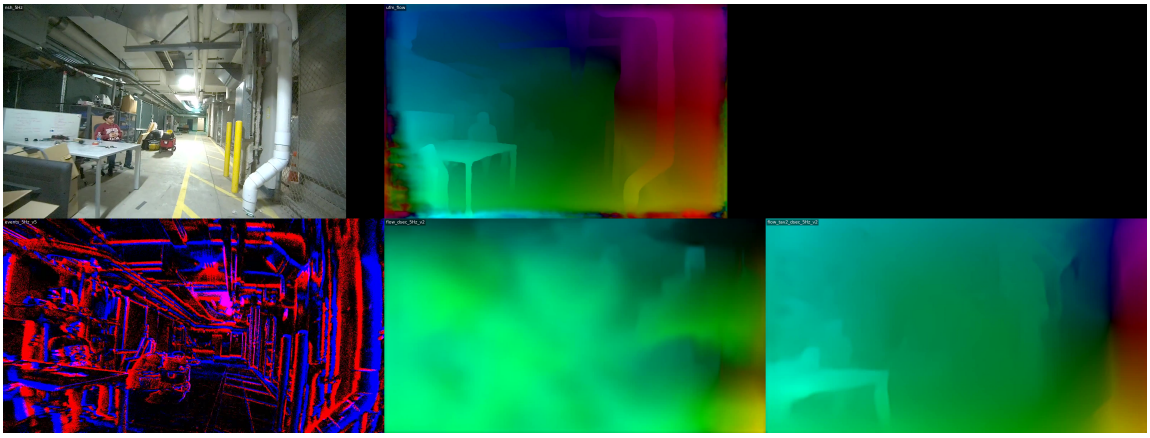


Figure 6.4: UFM captures the feature of the pipe on the right and the table and the person on the left. E-RAFT (TAV2 + DSEC) captures part of the table and person, but doesn't capture the pipe. E-RAFT (DSEC) fails to capture anything in the scene.

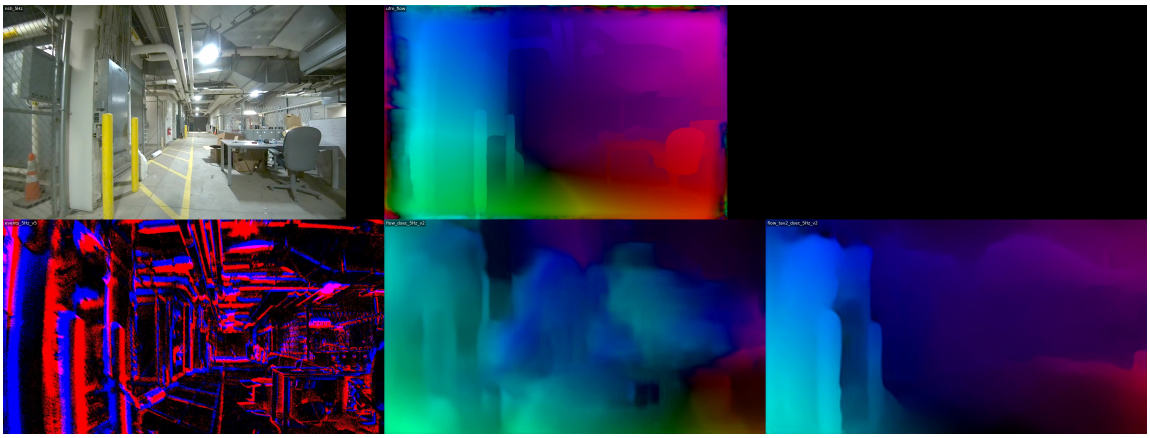


Figure 6.5: UFM captures the feature of the two poles on the left and the chair and table on the right. E-RAFT (TAV2 + DSEC) captures the two poles and part of the table on the right. The chair is mostly out of frame for the event cameras. E-RAFT (DSEC) attempts to capture the two poles but the features are too blurry with many artifacts.

6.6 Discussion

The results across Tables 6.1–6.3 support three conclusions.

First, designing models and learning representations are vital for performing optical flow and other perception tasks on event cameras. Simply converting events into a 2D representation and feeding into a non-event model will lead to poor performance. Second, TartanAir-V2 provides a useful pre-training data: E-RAFT trained on TartanAir-V2 and finetuned on DSEC and MVSEC outperforms the models only trained on those two datasets. This suggests that synthetic dataset adds complementary diversity, in particular with diversity in movements. As seen in the TartanStar evaluations, E-RAFT manages to handle larger motions better when pre-trained on TartanAirV2 than trained solely on DSEC. Third, direct 1,000 Hz simulation yields lower EPE than 10 Hz interpolation, confirming that interpolation artifacts degrade event quality and that high-rate direct rendering is preferred when feasible.

Chapter 7

Conclusion

This thesis addressed two practical barriers to event camera adoption: hardware deployment complexity and dataset scarcity. On the hardware side, a systematic deployment pipeline was presented covering lens selection and focusing, intrinsic calibration using both the E2Calib and Metavision SDK methods, and a detailed characterization of the five key bias parameters.

On the dataset and learning side, we demonstrated that small real datasets are not sufficient for training a model to be used on a robotics system. Using a large synthetic dataset such as TartanAir-V2 can enable the training of large, generalizable event-based perception models. We also showed that to create a synthetic event dataset, it is best to avoid interpolating pre-existing datasets and instead generate images on a rendering engine at a high sampling rate.

7.1 Limitations and Future Work

7.1.1 Explore Other Event Simulators and Models

The event simulation pipeline for TartanAirV2 relies on VID2E, which models only contrast threshold variation and does not capture the full noise characteristics of real event cameras (e.g., leakage, hot pixels). More realistic simulators such as V2E or DVS-Voltmeter could be explored in future and can help close the sim-to-real gap between synthetic and real event data. Closing the sim-to-real gap would remove the

need for event-based models to finetune on real world datasets.

To further explore TartanAirV2's capabilities, it is also worth to train other event models, in particular models such as STFlow [43], as they utilize both event and RGB modalities to perform perception-based tasks.

7.1.2 Process Optical Flow for TartanStar

One of the main reasons why DSEC and MVSEC are used is because they are one of the few real event datasets that provide ground truth optical flow. If TartanStar has a pipeline that can generate ground truth optical flow, then TartanStar can easily become the next real dataset for not only event research, but also for other modalities such as thermal cameras

Bibliography

[1] Zed x sensor stack specifications. œZEDXSensorStackSpecifications. StereoLabs.. 2.4

[2] The Anatomy of a Lens — edmundoptics.eu. https://www.edmundoptics.eu/knowledge-center/application-notes/imaging/the-anatomy-of-a-lens/, . 3.3.2

[3] Resolution — edmundoptics.com. https://www.edmundoptics.com/knowledge-center/application-notes/imaging/resolution/, . 3.1.3

[4] M25.5 x 0.5 Mounted UV/IR Cut-Off Filter — Edmund Optics — edmundoptics.com. https://www.edmundoptics.com/p/m255-x-05-mounted-uvir-cut-off-filter/10281/. 3.4a, 3.2.1

[5] MidOpt® SP785 Modified Near-IR Dichroic Block Shortpass Filter for Machine Vision 425-770nm — midopt.com. https://midopt.com/filters/sp785/. 3.4b, 3.2.1

[6] Understanding Focusing in Photography — katebackdrop.com. https://www.katebackdrop.com/blogs/photography-tutorials/focusing-in-photography. 3.8, 3.3.2

[7] Metavision training videos — focusing an event camera. . URL https://www.youtube.com/watch?v={Z}64{K}n29j{O}7w. 3.3.2

[8] Metavision training videos — introduction to event-based vision sensor. . URL www.youtube.com/watch?v=SPrdvhuAISk&t=15s. 2.1, 2.2, 2.2, 2.3, 3.12

[9] - YouTube — youtube.com. https://www.youtube.com/watch?v=7c7n5{D}{M}uat{Y}, . 3.5, 3.13

[10] EVK4 - HD — Metavision SDK Docs 5.3.1 documentation — docs.prophesee.ai. https://docs.prophesee.ai/stable/hw/evk/evk4.html, . 3.1.1

[11] IMX636 HD Sensor — Metavision SDK Docs 5.3.1 documentation — docs.prophesee.ai. https://docs.prophesee.ai/stable/hw/sensors/imx636.html#chapter-hw-sensors-imx636, . 2.4, 3.1.3

[12] Metavision Calibration Pipeline (C++) — Metavision SDK Docs 5.3.1 documentation — docs.prophesee.ai. https://docs.prophesee.ai/stable/samples/modules/calibration/calibration_pipeline.html, . (document), 1, 3.4.1, 3.9

[13] What Is a Lens Optical Format? — shop.visiondatum.com. https://shop.visiondatum.com/ar/blogs/blog/what-is-a-lens-optical-format. 3.3

[14] Kenneth Chaney et al. M3ED: Multi-robot, multi-sensor, multi-environment event dataset. In *CVPR Workshops*, 2023. 3.5.9

[15] Basler Lens C125-0418-5M F4MM. Basler Lens C125-0418-5M-P — Basler Product Documentation — docs.baslerweb.com. https://docs.baslerweb.com/c125-0418-5m-p. 3.1.3, 3.1.4

[16] Paul Furgale, Joern Rehder, and Roland Siegwart. Unified temporal and spatial calibration for multi-sensor systems. pages 1280–1286, 11 2013. doi: 10.1109/IROS.2013.6696514. 4

[17] Daniel Gehrig, Antonio Loquercio, Konstantinos G. Derpanis, and Davide Scaramuzza. End-to-end learning of representations for asynchronous event-based data, 2019. URL https://arxiv.org/abs/1904.08245. 5.4

[18] Daniel Gehrig, Mathias Gehrig, Javier Hidalgo-Carrió, and Davide Scaramuzza. Video to events: Recycling video datasets for event cameras, 2020. URL https://arxiv.org/abs/1912.03095. 4.3

[19] Mathias Gehrig. Dsec: A stereo event camera dataset for driving scenarios. arXiv.Org, 10 Mar. 2021, arxiv.org/abs/2103.06011. Accessed 23 Apr. 2026. (document)

[20] Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza. DSEC: A stereo event camera dataset for driving scenarios. *IEEE Robotics and Automation Letters*, 6(3):4947–4954, 2021. doi: 10.1109/LRA.2021.3068942. 1.1.2, 4.1

[21] 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. doi: 10.1109/3DV53792.2021.00016. 5.6, 6.1, 6.1.2, 6.1.3, 6.3

[22] Yuhuang Hu, Shih-Chii Liu, and Tobi Delbruck. v2e: From video frames to realistic dvs events, 2021. URL https://arxiv.org/abs/2006.07722. 4.3

[23] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation, 2018. URL https://arxiv.org/abs/1712.00080. 4.3, 4.3.2

[24] Zhen Jin et al. EMA-VFI: Event-guided multi-scale adaptive video frame interpolation. In *arXiv*, 2023. arXiv:2305.12001. 4.3.1, 4.3.2, 6.4

[25] Simon Klenk, Marvin Motzet, Lukas Koestler, and Daniel Cremers. Deep event visual odometry, 2023. URL https://arxiv.org/abs/2312.09800. 6.1.1

[26] Prabu Kumar. C-mount vs. CS-mount: Everything you should know about these lens types - e-con Systems — e-consystems.com. https://www.e-consystems.com/blog/camera/technology/c-mount-vs-cs-mount-everything-you-should-know-about-these-lens-types/. 3.1.1

[27] Yijin Li, Zhaoyang Huang, Shuo Chen, Xiaoyu Shi, Hongsheng Li, Hujun Bao, Zhaopeng Cui, and Guofeng Zhang. Blinkflow: A dataset to push the limits of event-based optical flow estimation, 2024. URL https://arxiv.org/abs/2303.07716. 4.3.1

[28] Songnan Lin. Dvs-voltmeter: Stochastic process-based event simulator for dynamic vision sensors. *Lecture Notes in Computer Science*, pages 578–593,. doi: 10.1007/978-3-031-20071-7_34. 4.3

[29] Manasi Muglikar, Mathias Gehrig, Daniel Gehrig, and Davide Scaramuzza. How to calibrate your event camera, 2021. URL https://arxiv.org/abs/2105.12362. (document), 1, 3.4.2

[30] Ouster. Os2 specs. https://data.ouster.io/downloads/datasheets/datasheet-rev7-v2p5-os2.pdf. 3.2.1

[31] Manthan Patel, Fan Yang, Yuheng Qiu, Cesar Cadena, Sebastian Scherer, Marco Hutter, and Wenshan Wang. Tartanground: A large-scale dataset for ground robot perception and navigation. *arXiv preprint arXiv:2505.10696*, 2025. 4.3.2

[32] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. High speed and high dynamic range video with an event camera, 2019. URL https://arxiv.org/abs/1906.07165. 3.4.2

[33] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. High speed and high dynamic range video with an event camera, 2019. URL https://arxiv.org/abs/1906.07165. 4.3

[34] Fitsum Reda, Janne Kontkanen, Eric Tabellion, Deqing Sun, Caroline Pantofaru, and Brian Curless. Film: Frame interpolation for large motion, 2022. URL https://arxiv.org/abs/2202.04901. 4.3.2

[35] Davide Scaramuzza. Tutorial on event-based cameras: Davide scaramuzza. rpg.ifi.uzh.ch/docs/scaramuzza/2019.07.11$_{S}caramuzza_{E}vent_{C}ameras_{T}utorial.pdf$. 1.1.1, 2.1, 2.4, 5.

[36] Bo Shao, Yingxun Wang, Zhihao Cai, and Jiang Zhao. Event camera visualization. In Liang Yan, Haibin Duan, and Yimin Deng, editors, *Advances in Guidance, Navigation*

and Control, pages 6023–6032, Singapore, 2023. Springer Nature Singapore. ISBN 978-981-19-6613-2. 5.2

[37] Ltd Soyo Security Co. Soyo sfa0820-5m. http://www.soyocctv.com/manager/upload/file/20211021/20211021144062346234.pdf. 3.1.4

[38] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. 2020. 4.4

[39] Wenshan Wang, Yaoyu Hu, Yorai Shaoul, Mohit Singh, Yuheng Qiu, Yuchen Zhang, Shibo Zhao, Srivatsa Grama Satyanarayana, Nicholas Leone, and Sebastian Scherer. Pip install tartanair: A dataset to push the general robot perception. Manuscript Under Review, 2026. (document), 2, 4.4, 4.4.1

[40] Yan Yang. Event camera data dense pre-training. arXiv.Org, 22 Sept. 2024, arxiv.org/abs/2311.11533. Accessed 21 Apr. 2026. 4.3.1

[41] Mubariz Zaffar. *Visual Place Recognition for Autonomous Robots*. PhD thesis, 09 2020. 2.3

[42] Yuchen Zhang, Nikhil Keetha, Chenwei Lyu, Bhuvan Jhamb, Yutian Chen, Yuheng Qiu, Jay Karhade, Shreyas Jha, Yaoyu Hu, Deva Ramanan, Sebastian Scherer, and Wenshan Wang. Ufm: A simple path towards unified dense correspondence with flow, 2026. URL https://arxiv.org/abs/2506.09278. 6.1.4

[43] Qianang Zhou, Junhui Hou, Meiyi Yang, Yongjian Deng, Youfu Li, and Junlin Xiong. Spatially-guided temporal aggregation for robust event-rgb optical flow estimation, 2025. URL https://arxiv.org/abs/2501.00838. 7.1.1

[44] Alex Zihao Zhu, Dinesh Thakur, Tolga Ozaslan, Bernd Pfrommer, Vijay Kumar, and Kostas Daniilidis. The multivehicle stereo event camera dataset: An event camera dataset for 3d perception. *IEEE Robotics and Automation Letters*, 3(4):2032–2039, 2018. doi: 10.1109/LRA.2018.2800793. (document), 1.1.2, 4.1

[45] Zhen Zhu, Tengteng Huang, Baoguang Shi, Miao Yu, Bofei Wang, and Xiang Bai. Progressive pose attention transfer for person image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 5.3