Monocular Event Camera Odometry Using Deep Learning

Srivatsa Grama Satyanarayana

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2024

Committee:

Abhishek Gupta

Sawyer Fuller

Santosh Devasia

Program Authorized to Offer Degree:
Mechanical Engineering

University of Washington

Abstract

Monocular Event Camera Odometry Using Deep Learning

Srivatsa Grama Satyanarayana

Chair of the Supervisory Committee:
Abhishek Gupta
Paul G. Allen School of Computer Science and Engineering

Event cameras provide a number of advantages when compared to traditional cameras like high temporal resolution, low latency, and dynamic range. This makes event cameras a promising alternative to frame-based systems. Despite this event cameras face unique challenges in performing odometry due to their sparse and asynchronous data output. This thesis addresses these challenges by presenting a novel monocular event camera odometry framework that leverages deep learning techniques. The proposed approach, TartanEVO, integrates optical flow prediction from an event camera with a pose estimation network to produce incremental motion estimates. This thesis also introduces the Tartanair-v2 - Event Camera, the largest public event camera dataset for odometry and SLAM; this dataset features large and diverse scenes with challenging viewpoints, varying lighting, and diverse motion patterns. Extensive evaluations demonstrate the robustness of TartanEVO in diverse environments where traditional odometry algorithms fail. In low lighting and rapid motion, our method even outperforms frame-based algorithms. This work highlights the potential of event cameras and deep learning in advancing robust odometry systems.

Acknowledgments

This work would not have been possible without the unwavering support and guidance of Wenshan Wang from AirLab, CMU. Her expertise and mentorship were instrumental in shaping the direction of this research. This project required extensive computational resources, generously provided by AirLab, CMU, without which the dataset generation and experiments would have been impossible.

This research originated during the RACER project, and I am grateful to Prof. Byron Boots for providing the essential equipment that enabled the initial stages of this work. I would also like to thank Xiangyun Meng, Alex Spitzer, and Nathan Hatch for providing guidance during my time at RACER.

I am also indebted to my advisor, Prof. Abhishek Gupta, and committee members Prof. Sawyer Fuller, and Prof. Santosh Devasia. Their insightful feedback, constructive criticism, and encouragement were vital in refining and bringing this work to fruition. Their collective expertise and patience were truly invaluable throughout the process.

Contents

1 Introduction **1**

1.1 Background and Motivation 1

1.1.1 What is Visual Odometry and Why Do We Care? . . . 1

1.1.2 Algorithms used in Visual Odometry and its limitations. 2

1.1.3 How does an event camera work, and what advantages does it provide? 2

1.1.4 Algorithms used for event camera odometry and its limitations 3

1.1.5 How is deep learning filling the gap to improve the Visual Odometry? 4

1.2 Current Challenges with Event Camera Odometry 4

1.3 Proposed Solution 5

1.4 Contributions of the Thesis 5

2 Methodology **7**

2.1 Dataset 7

2.1.1 Rendering Engine 7

2.1.2 3D environment setup and Trajectory Sampling 7

2.1.3 Event Data Simulator 7

2.1.4 Event Camera Model 8

2.1.5 Parameters for event generation 8

2.2 TartanEVO 9

2.2.1 Event Representation 9

2.2.2 Flow Network 10

2.2.3 Pose Network 10

2.2.4 Metrics 11

3 Results **13**
3.1 Dataset . . . 13
3.1.1 Event Distribution . . . 15
3.1.2 Flow Distribution . . . 16
3.2 Flow Network . . . 16
3.2.1 Inference on DSEC and MVSEC . . . 16
3.3 TartanEVO . . . 17
3.3.1 Inference on TartanAir-v2 . . . 18
3.3.2 Inference on VECTor . . . 18
3.3.3 Inference on MVSEC . . . 19

4 Conclusion **22**
4.1 Summary of Contributions . . . 22
4.2 Future Work . . . 22
4.2.1 Transformer based Networks . . . 22
4.2.2 Optimizer with FlowNet . . . 23

List of Figures

3.1 Example images from frame-based and event-based camera in tartanair-v2 dataset 14
3.2 Event data analysis of TartanAir-v2 with different datasets . . 15
3.3 Flow analysis of TartanAir-v2 with different datasets 17
3.4 Example trajectories generated by TartanEVO on TartanAir-v2 Environments 21

List of Tables

2.1 Evaluation of event based flow models on the DSEC-Flow dataset. The data is adapted from E-RAFT [9]. 10

3.1 Comparison of event camera dataset with optical flow 15
3.2 Flow network inference on the DSEC dataset 18
3.3 Flow network inference on the MVSEC dataset 18
3.4 Comparison of methods on the TartanAir-v2:CountryHouse environment. 19
3.5 Comparison of methods on the TartanAir-v2:ShoreCaves environment. 19
3.6 Comparison of methods on the TartanAir-v2:MiddleEast environment. 20
3.7 Comparison of methods on VECTor trajectories. Data other than TartanVO and TartanEVO are adapted from [2][13]. . . . 20
3.8 Comparison of methods on MVSEC indoor flying trajectories. Data other than TartanVO and TartanEVO are adapted from [2] [13]. 20

Chapter 1

Introduction

1.1 Background and Motivation

1.1.1 What is Visual Odometry and Why Do We Care?

Odometry is a process used to estimate the position of a body using sensor information. Odometry is used in autonomous robots to estimate the distance traveled by the robot. Typically, an autonomous robot uses wheel encoders, IMUs, GPS, Cameras, LiDARs, or a combination of these sensors to perform odometry. A system that uses the visual information for odometry is called Visual odometry (VO). A VO system defines the inconsistency in the observed features in different frames as a loss function and then minimizes that loss function by non-linear optimization.

A VO system or inclusion of visual information into odometry provides several advantages. A VO system dramatically increases the accuracy of odometry or state estimation by reducing the drift. The VO systems provide low relative position errors ranging from 0.1% to 2% [3]. The VO system is also not affected by wheen slip, which hugely affects the odometry based on wheel encoders or engine/rotor vibration, which affects the IMU. The VO is also a relatively inexpensive sensor when compared to sensors like LiDAR, making them a more feasible option. All these advantages make the inclusion of the VO system, either independently or integrated with other sensors, an excellent choice for odometry.

1.1.2 Algorithms used in Visual Odometry and its limitations.

Researchers have developed several methods that use visual information to perform odometry. Some of the popular algorithms include PTAM [12], Direct Sparse Odometry (DSO) [4], Large-Scale Direct Monocular SLAM (LSD-SLAM) [5], Semi-Direct Visual Odometry (SVO) [6], and ORB-SLAM [18]. These algorithms demonstrate incredible accuracy in feature-rich scenes with decent lighting conditions.

These methods have a front-end for feature extraction and a back-end optimizer for reducing the loss functions. The front end gets an image stream as input and then extracts sparse or dense features from the scene. For example, ORB-SLAM tracks sparse features across multiple frames. DSO and LSD-SLAM utilize dense geometric features and try to minimize the photometric loss across multiple frames.

The back end performs iterative non-linear optimization to reduce the loss function, typically some form of reprojection error, created with the features obtained from the front end. This process reduces the error and improves the estimated camera poses and the reconstruction of the environment.

The effectiveness of these methods heavily relies on extracting high-quality visual features from the camera and the front end. Under challenging conditions like low illumination, sudden lighting change, and rapid movement, the quality of the features reduces drastically. This hinders the ability of the features to be tracked over multiple frames. The initialization process for VO algorithms is not trivial; therefore, once the features are lost, the ability of the VO system to recover from this situation is also fairly low. These disadvantages limit the application of vision odometry in crucial systems; instead, a combination of LiDAR and IMU is used [1].[34].

1.1.3 How does an event camera work, and what advantages does it provide?

Event cameras are biologically inspired cameras that take in visual information and generate sparse and asynchronous data called events. Similar to traditional frame-based cameras, they have photoreceptors that capture the intensity of each pixel, but instead of capturing the absolute values of the intensity, the event camera registers an event when the log intensity of the pixel increases above a certain threshold. If the log intensity increases, it is

a positive event, and if it reduces, it is a negative event; this is often called the polarity. In the event camera, an event contains the pixel location that observed the change in brightness, its polarity, and the timestamp [22].

Event cameras address many of the challenges faced by traditional cameras. As there is no explicit shutter rate, each pixel generates an event as soon as the pixel's brightness changes; this reduces the latency and greatly improves the temporal resolution. Because of this high temporal resolution, minimal motion blur is observed in the event cameras. The event cameras also offer a high dynamic range and this enables effective operation in environments with extreme lighting conditions, with sharp features even during rapid movement. These features of the event camera address most of the challenges traditional frame-based cameras face while performing odometry under challenging conditions.

However, despite their superior hardware capabilities, event cameras present their own set of challenges. They produce sparse and asynchronous data, which do not work well with the existing VO algorithms, which were predominantly developed for dense, constant frame rate image streams of traditional frame-based cameras. Researchers have yet to extensively study or standardize event cameras, leaving a gap in robust algorithms designed to process their unique output effectively.

1.1.4 Algorithms used for event camera odometry and its limitations

Most existing algorithms for event camera-based odometry extract geometric information from event data similar to frame-based cameras. Geometry-based methods proposed by Rebecq et al. [20] and Kueng et al. [14] minimize the reprojection error to estimate motion. However, these methods only perform well on small-scale scenes. They are also highly sensitive to parameters and require bootstrapping, which can limit their practicality.

Several studies have explored integrating event cameras with other sensors to enhance performance. For instance, [21] utilizes IMU data to synthesize a motion-corrected event image before extracting features from the event data. These sparse features are then tracked over multiple frames to perform odometry. Similarly, [28] adopts a comparable approach but also incorporates features from a frame-based camera. Although these methods perform reasonably well, they require configuring a large number of param-

eters. Additionally, the algorithms become susceptible to noise from other sensors, reducing the overall robustness of the system.

1.1.5 How is deep learning filling the gap to improve the Visual Odometry?

The use of deep learning algorithms for various parts of visual odometry stack have addressed the limitations of traditional methods face, especially under challenging conditions like rapid exposure changes, motion blur, low lighting, and complex scenes. The End-to-end deep learning algorithms have introduced robustness and adaptability to odometry tasks for frame-based systems.

Algorithms such as DeepVO [29] employ a combination of convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to process sequential images and estimate incremental motion between them. Methods like UnDeepVO [15] and D3VO [33] integrate deep networks to predict depth maps from monocular images and then perform odometry using depth data.

Additionally, algorithms like TartanVO [30], DPVO [27], and DROID-SLAM [26] leverage deep networks to establish dense or sparse correspondences between frames. These correspondences are then used to predict accurate pose estimation and mapping, especially in environments where traditional feature extraction and matching may fail.

1.2 Current Challenges with Event Camera Odometry

Event cameras offer promising advantages for odometry tasks but the current state of the art algorithms face the following problems:

- **Lack of robustness:** Most odometry algorithms for event cameras are based on algorithms used in traditional frame-based cameras. These algorithms extract features or geometry from the event camera stream, which leads to the following problems. First, the extraction of features or geometry from an event camera is not straightforward because of the sparse nature of the data. Second, these methods require cumbersome initialization processes. Therefore, when the features are lost, the initialization process needs to be redone, which takes a long time.

- **Lack of Large-Scale Event Camera Datasets:** Another significant barrier to the penetration of deep learning algorithms for event camera odometry is the limited availability of large-scale, diverse datasets that are focussed on odometry. The success of deep learning algorithms in frame-based learning tasks heavily relies on datasets like TartanAir [31] for training robust models. In the case of event cameras, this lack of large-scale public datasets restricts the development of deep-learning event-based odometry solutions.

1.3 Proposed Solution

Despite the advancements in visual odometry algorithms, the traditional camera-based methods remain constrained by the hardware limitations of frame-based cameras, especially under challenging conditions like rapid motion or extreme lighting. To overcome these limitations, it is essential to develop odometry methods based on event cameras that can match or surpass the performance of traditional systems.

This thesis proposes an event camera-based odometry framework inspired by architectures like TartanVO [30]. This approach involves predicting optical flow from event camera data and utilizing a pose estimation network combined with intrinsic parameter layers to predict incremental camera motion. By leveraging the high temporal resolution, low latency, and high dynamic range of event cameras, this method aims to provide robust and accurate motion estimation even in environments where traditional cameras struggle.

1.4 Contributions of the Thesis

This thesis makes the following key contributions:

- **Contribution 1:** Development of a large-scale event camera dataset specifically designed for odometry tasks, addressing the scarcity of event-based data for training and evaluation.

- **Contribution 2:** A pre-trained optical flow network that generalizes effectively across the MVSEC [36] and DSEC [8] datasets, enhancing cross-dataset robustness.

- **Contribution 3:** Design and implementation of a novel framework for performing odometry with event cameras, integrating deep learning techniques to exploit the advantages of event-based sensing.

Chapter 2

Methodology

2.1 Dataset

2.1.1 Rendering Engine

The Unreal Engine was selected to be used to maintain consistency with TartanAir-v2 dataset and also because it offers a highly customizable and realistic virtual world for sensor simulation. Unreal Engine is particularly well-suited for this task due to its advanced rendering capabilities, including photorealistic lighting, high-dynamic-range scenes, and precise control over environmental variables such as weather and time of day. This ensures that the simulated camera data closely mirrors real-world conditions.

2.1.2 3D environment setup and Trajectory Sampling

The TartanAir-v2 dataset is collected on environments that include diverse terrains, lighting conditions, and dynamic objects. The event camera sensor will follow the exact trajectories that the cameras in the TartanAir-v2 dataset followed. This alignment allows us to reuse the existing high-quality ground truth data provided by TartanAir-v2, such as camera poses, depth maps, and optical flow.

2.1.3 Event Data Simulator

Several simulators are available for generating event data, the most notable being vid2e [7], V2E [10], and DVS-Voltmeter [17]. Among them, DVS-

Voltmeter is the most accurate, as it incorporates circuit properties into the stochastic process. However, DVS-Voltmeter runs on the CPU and is relatively slow. On the other hand, V2E and vid2e run on the GPU, offering significantly faster performance. Notably, networks trained using vid2e-generated data outperform those trained with V2E for tasks such as image reconstruction and segmentation [17].

2.1.4 Event Camera Model

The event data simulator vid2e [7] uses the following event camera model. An event $e_k = (t_k, x_k, y_k, p_k)$ is triggered based on the changes in the log intensity of light that is being observed at a particular pixel.

$$L(\mathbf{u}_k, t_k) - L(\mathbf{u}_k, t_{k-1}) \geq p_k C. \tag{2.1}$$

Here, $\mathbf{u} = (x, y)$, $L(u, t) = ln(I_t(u))$, is the log of the brightness at a particular pixel u at time t. t_{k-1} is the timestamp at the last event generated. $p_k \in \{-1, +1\}$ specifies the increase or decrease in the brightness, also called the polarity of the event. $\pm C$ is the threshold above which an event is produced. Equation 2.1 describes the event generation model of an ideal event camera [7].

2.1.5 Parameters for event generation

In the simulation pipeline the following four key parameters to generate events need to be tuned:

1. Image Sampling Frequency: The rate at which the simulator samples images to generate event data. Higher the image sampling frequency the more accurate and time consuming the simulation becomes. For the TartanAir-v2 the sampling frequency of 1KHz will be maintained. For image reconstruction tasks, having a sampling frequcny of 1KHz produces decent results [19]. For reference, Gehrig et al. sampled the Caltech101 dataset at 530 Hz, and the blinkflow dataset [16] have not published the exact sampling frequency, it seems to have been sampled at 30Hz when using V2E.

2. Positive Contrast Threshold: The minimum increase in brightness required to trigger a positive event in the event camera. Stoffregen et al.

[23] investigated their impact on performance across different datasets. They discovered that datasets like IJRR performed better with a contrast threshold of 0.2, whereas MVSEC [36] showed optimal results with thresholds close to 1.0. This indicates that to develop a model with good generalization capabilities, it is beneficial to collect data using a range of contrast thresholds from 0.15 to 1.0.

3. Negative Contrast Threshold: The minimum decrease in brightness required to trigger a negative event. Additionally, it's important to note that in real event cameras, the negative contrast threshold does not necessarily equal the positive contrast threshold. To reflect this characteristic, the negative contrast threshold intentionally offset from the positive contrast threshold by a small value in our simulations.

4. Refractory Period: Event camera pixels have a refractory period where the pixel does not fire immediately after an event. This value is selected to be 1 ms for the event generation, aligning with values commonly used in other papers.

2.2 TartanEVO

The method, TartanEVO, introduced in this thesis is an event-based visual odometry system adapted from the existing TartanVO [30] framework. TartanVO is a deep learning-based odometry system designed for traditional frame-based cameras, consisting of two main components: a flow network that computes dense pixel-wise correspondences (optical flow) between consecutive frames, and a pose network that estimates the relative camera motion based on these correspondences.

2.2.1 Event Representation

Event cameras produce data as a stream of events, each represented by a tuple $(t_k,\ x_k,\ y_k,\ p_k)$. t_k indicates the timestamp of the event, $(x_k,\ y_k)$ indicates the pixel location of the event. $p_k = 1$, indicates a positive event and $p_k = 0$ or -1 indicates a negative event.

To make this event data compatible with optical flow networks, we aggregate the events into a sequence of volumetric voxel grids, where the height and width of the grid is $V_k \in \mathbb{R}^{H \times W \times C}$ of the sensor, and C is the number

of temporal bins. Specifically, we discretize the time dimension into fifteen bins, effectively dividing the events into fifteen temporal slices. This voxel grid representation preserves temporal information by organizing events into these discrete time intervals.

2.2.2 Flow Network

The following two networks were considered for optical flow estimation from event camera data: EV-FlowNet, based on the FlowNet architecture, and E-RAFT, derived from the RAFT architecture. These algorithms are among the few that have open-sourced their code and architectures for event-based optical flow estimation. The RAFT architecture has been proven to generalize well and perform exceptionally on standard benchmarks for optical flow. Correspondingly, E-RAFT exhibits superior performance compared to EV-FlowNet. Table 2.1 presents a comparison between the two networks when trained and tested on the EV-FlowNet dataset. Notably, E-RAFT provides dense optical flow estimations, whereas EV-FlowNet does not, making E-RAFT more suitable our TartanEVO.

Table 2.1: Evaluation of event based flow models on the DSEC-Flow dataset. The data is adapted from E-RAFT [9].

Methods	AEE	1px	3px	5px
EV-FlowNet[35]	2.32	44.6	81.40	-
E-RAFT[9]	0.79	87.5	97.3	-

2.2.3 Pose Network

TartanEVO retains the pose network from TartanVO due to its demonstrated effectiveness in estimating six degrees of freedom (6-DoF) camera poses. This network takes in the dense correspondences provided by the flow network to compute the relative motion between frames. To optimize performance, the pose network undergoes a two-stage fine tuning process. Initially, the pretrained network is fine-tuned on the ground truth optical flow from the TartanAir-v2 dataset to adapt to the specific motion patterns and features of the dataset. In the second stage, the flow network is frozen, allowing the pose network to refine its learning while maintaining the same quality of the dense correspondences.

2.2.4 Metrics

We will be using the following metrics to keep track of the performance.

Flow Networks

1. **Average End point Error: AEE**: Measures the average Euclidean distance between the estimated optical flow vectors and the ground truth flow vectors across all valid pixels in the image. The lower this value, the better the prediction. This metric reported by a lot of the optical flow measurement methods[9] [25] [32].

$$\mathrm{AEE_{ij}} := \frac{\sum_{\mathrm{i,j}} \sqrt{\left(\mathrm{u_{ij}^{est}} - \mathrm{u_{ij}^{gt}}\right)^2 + \left(\mathrm{v_{ij}^{est}} - \mathrm{v_{ij}^{gt}}\right)^2}}{\mathrm{N}} \tag{2.2}$$

2. **5px**: Percentage of valid pixels where Optical flow is off by less than 5 pixels. The Higher this value, the better the prediction.

3. **3px**: Percentage of valid pixels where Optical flow is off by less than 3 pixels. The higher this value, the better the prediction.

4. **1px**: Percentage of valid pixels where Optical flow is off by less than 1 pixel. The higher this value, the better the prediction.

Pose Networks

1. **Mean Position Error [MPE]**: Evaluates the local consistency of the estimated trajectory with respect to the ground truth [2]. This metric is similar to the RPE error mentioned in the TUM RGB-D benchmark [24] but it is further normalized by the distance. The equation are partially adapted from TUM RGB-D benchmark[24]. First, the relative transformations $\mathbf{E}_{1:n}$ are calculated between the poses i and $i + \Delta$ for both the estimated trajectory $\mathbf{P}_{1:n}$ and the ground truth trajectory $\mathbf{Q}_{1:n}$.

$$\mathbf{E}_i := (\mathbf{Q}_i^{-1}\mathbf{Q}_{i+\Delta})^{-1}(\mathbf{P}_i^{-1}\mathbf{P}_{i+\Delta}) \tag{2.3}$$

   The translational component of the error transformation $\mathbf{E}_{1:n}$ is extracted and this is normalized by the distance traveled in the ground truth:

$$\hat{\mathbf{E}}_i := \frac{\sum_{i=1}^{n} \text{trans}(\mathbf{E}_\text{i})}{\text{trans}(\mathbf{Q}_\text{i}^{-1}\mathbf{Q}_{\text{i}+\Delta})} \tag{2.4}$$

MPE, measured in [%/m] is the mean of the normalized error $\hat{\mathbf{E}}_\mathbf{i}$. The lower this value, the better the prediction.

$$\text{MPE} = \overline{\hat{\mathbf{E}}}_{1:\text{n}} \tag{2.5}$$

2. **Absolute Trajectory Error [ATE]**: Evaluates the global consistency of the estimated trajectory with respect to the ground truth. This metric and the equation are adapted from TUM RGB-D benchmark [24]. This metric is First, the estimated trajectory $\mathbf{P}_{1:n}$ is aligned with the ground truth trajectory $\mathbf{Q}_{1:n}$ using a rigid transform $\mathbf{S}$, which is calculated using the Horn's method. Then for each time step i, relative transformation $\mathbf{F}_i$ is calculated which transforms the $\mathbf{P}_i$ to $\mathbf{Q}_i$.

$$\mathbf{F}_i := \mathbf{Q}_i^{-1}\mathbf{S}\mathbf{P}_i \tag{2.6}$$

The ATE, measured in [m], is then quantified by computing the Root Mean Squared Error (RMSE) of the translational components of $\mathbf{F}_i$ over all timestamps. The lower this value, the better the prediction.

$$\text{ATE} = \text{RMSE}(\mathbf{F}_{1:\text{n}}) := \left(\frac{1}{\text{n}} \sum_{\text{i}=1}^{\text{n}} \|\text{trans}(\mathbf{F}_\text{i})\|^2 \right)^{1/2} \tag{2.7}$$

Chapter 3

Results

3.1 Dataset

There are very few event camera datasets with optical flow ground truth. Among them, DSEC and MVSEC are real-world datasets that provide optical flow derived from LiDAR SLAM. However, in both MVSEC and DSEC, the camera predominantly moves in a straight line with very limited rotation, and any rotations are mainly along the vertical axis. This lack of diverse motion patterns severely limits the usefulness of these datasets for training purposes.

BlinkFlow offers a decent number of frames and motion patterns, but its data is collected at around 30 Hz, which does not accurately mimic the high temporal resolution of event cameras. This discrepancy in sampling frequency means that BlinkFlow may not capture the fine-grained temporal information essential for training event-based models effectively.

In contrast, TartanAir-v2, presented in this thesis, is currently the largest publicly available event camera dataset. It contains significantly more frames than previous datasets—by several orders of magnitude—and features random motion patterns that provide a wide diversity of data. The dataset is collected from over 50 different environments within TartanAir-v2, encompassing a wide variety of lighting conditions. This extensive variability in scenes and conditions makes TartanAir-v2 a highly valuable resource for training and evaluating event-based odometry algorithms. Example images of the image and event stream can be observed in the figure 3.1 Table 3.1 provides a comparison of all the datasets.

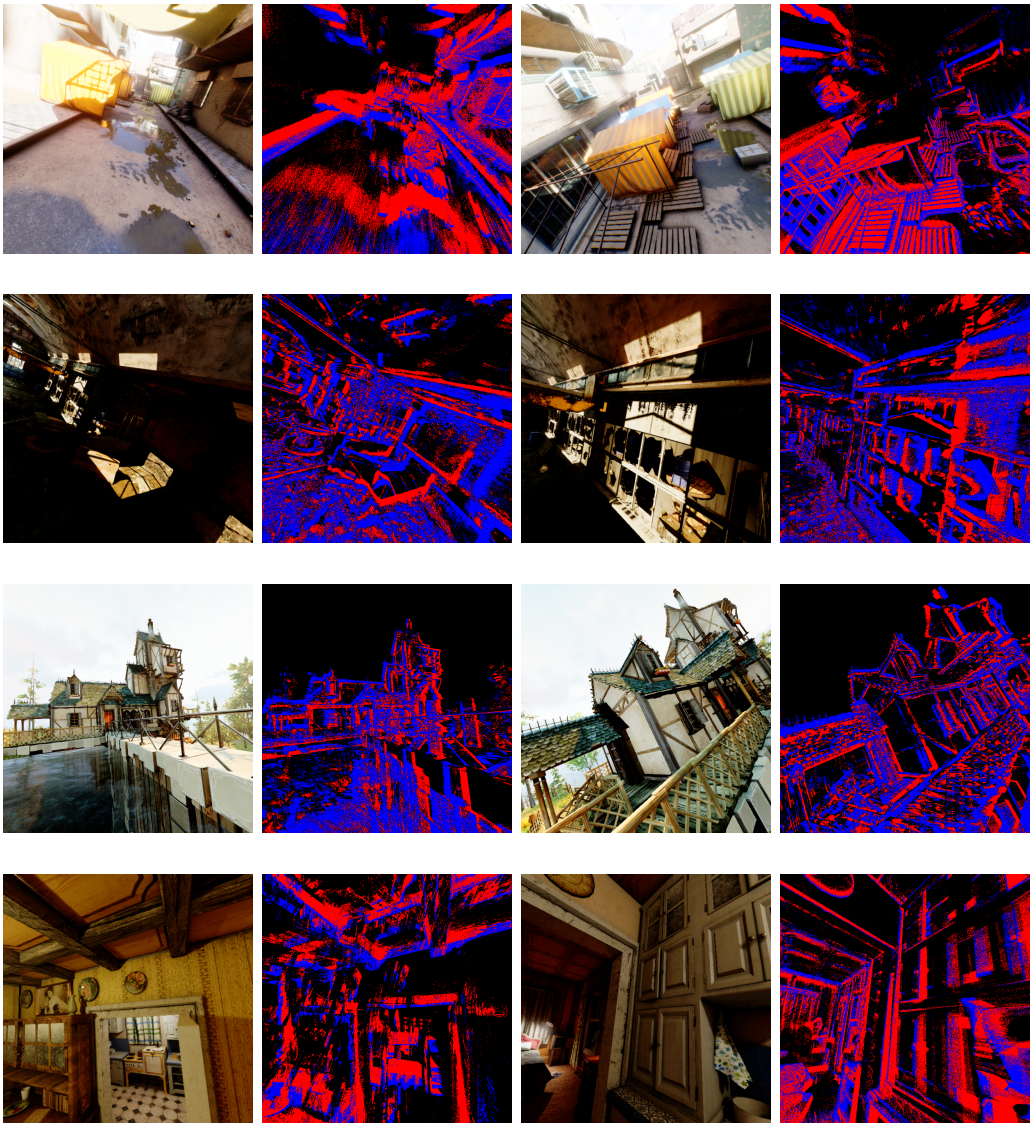


Figure 3.1: Example images from frame-based and event-based camera in tartanair-v2 dataset

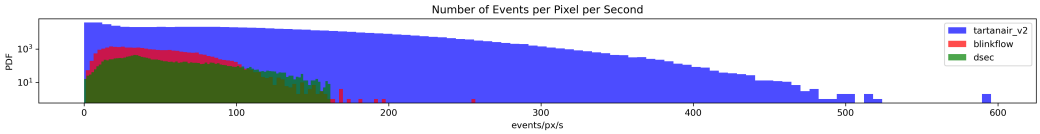


Figure 3.2: Event data analysis of TartanAir-v2 with different datasets

Table 3.1: Comparison of event camera dataset with optical flow

Dataset	**Motion Pattern**	**Reference Frames**	**Resolution**	**Optical Flow**
MVSEC	Drone, Motorcycle	3k	260x346	Sparse
DSEC	Car	8k	640x480	Sparse
BlinkFlow	Random	33k+	640x480	Dense
TartanAir-v2 (Ours)	Random	700k+	640x640	Dense

3.1.1 Event Distribution

Diversity in event data is crucial for developing robust event-based vision systems. Event data varies significantly based on camera specifications, lighting conditions, and the motion dynamics of the scene. Different event cameras have varying levels of sensitivity and contrast thresholds, which directly affect the number of events they produce per unit time. A more sensitive event camera, or one with a lower contrast threshold, will generate a higher number of events, especially in dynamic scenes with rapid brightness changes.

Environmental and lighting conditions play a significant role in event generation. In scenes with higher contrast or pronounced lighting variations, more events are produced because the camera detects more significant changes in brightness. Additionally, increased motion—whether from the camera itself or moving objects within the scene—results in a higher frequency of events, as each change in pixel intensity triggers an event.

To account for these factors, the number of events per pixel per second becomes an important metric. This metric provides a quantitative measure of event density, allowing us to assess whether our dataset fully captures the variety of scenes and conditions encountered in real-world applications. By ensuring a wide range of event densities, we aim to include scenarios

ranging from low-motion, low-contrast environments to highly dynamic, high-contrast scenes.

To validate that our dataset encompasses the full spectrum of possible scenarios, we analyze and compare this metric across different datasets. This comparison helps us ensure that our dataset is comprehensive and representative of real-life event camera data. Figure 3.2 presents the distribution of the number of events per pixel per second, illustrating how our dataset compares to others in terms of event generation under various conditions.

3.1.2 Flow Distribution

An ideal optical flow distribution for training should encompass the full range of motion patterns and magnitudes encountered in real-world scenarios. In Figure 3.3, we observe that TartanAir-v2 contains optical flow values that are greater than those found in other datasets like MVSEC and DSEC. This indicates that TartanAir-v2 provides a broader and more diverse set of motion patterns, which is crucial for developing models that generalize well to various environments.

In contrast to the flow data in MVSEC and DSEC which is collected from vehicles that primarily travel in straight lines with minimal rotation, mostly limited to the vertical axis, the motion patterns in TartanAir-v2 are random and include a wide variety of translations and rotations across all axes. This diversity in motion patterns in the tartanair-v2 ensures that the trained models are better equipped to handle the diverse motions encountered in real-world applications.

3.2 Flow Network

The performance of the flow network trained on TartanAir-v2 was evaluated by testing it on the DSEC and MVSEC datasets, which are real-world event camera datasets with optical flow ground truth.

3.2.1 Inference on DSEC and MVSEC

Table 3.2 presents a comparison of the models' performance on the DSEC dataset. Notably, the model trained on TartanAir-v2 demonstrates significantly better performance compared to networks trained on other datasets.

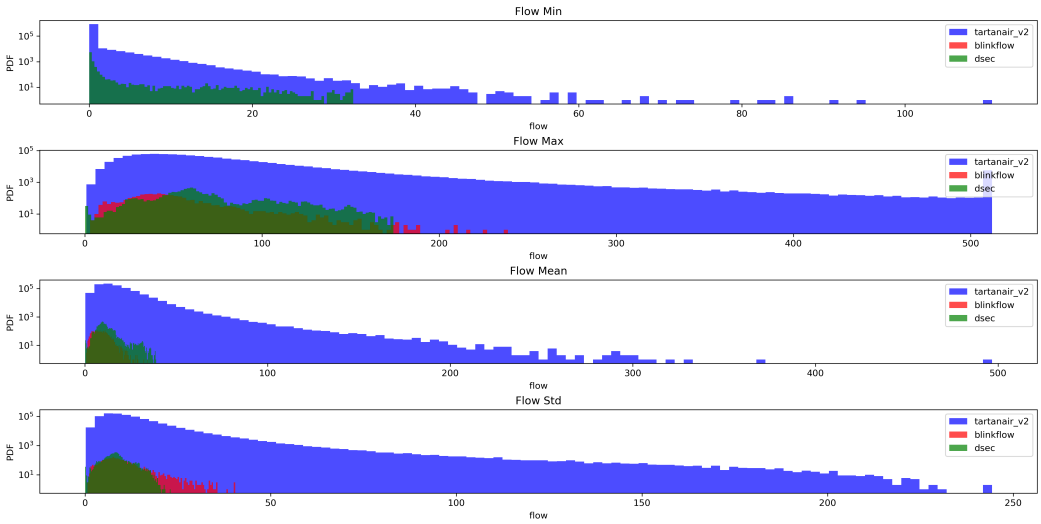


Figure 3.3: Flow analysis of TartanAir-v2 with different datasets

Since none of the networks were trained on DSEC data, this evaluation allows us to assess the generalization capabilities of the models to unseen real-world environments. A similar trend is observed with the MVSEC dataset, as shown in Table 3.3. The model trained on TartanAir-v2 outperforms others when tested on MVSEC, reinforcing the notion that training on TartanAir-v2 enhances generalization.

This superior performance can be attributed to the diverse motion patterns and lighting conditions present in TartanAir-v2, which enable the model to learn more robust and versatile features. These results suggest that the extensive and varied data provided by TartanAir-v2 contribute to a flow network that generalizes well across different datasets, making it more effective for practical applications involving event-based cameras.

3.3 TartanEVO

The performance of the TartanEVO network was evaluated on the following realworld dataset and also on a couple of unseen TartanAir-v2.

Table 3.2: Flow network inference on the DSEC dataset

Methods	**Training Datasets**	**AEE**	**1px**	**3px**	**5px**
EV-FlowNet	MVSEC	2.32	0.448	0.702	0.814
S. Shiba et al.	—	2.12	—	—	—
E-RAFT	BlinkFlow	1.34	—	—	—
E-RAFT (Ours)	BlinkFlow + TartanAir_v2	1.07	0.689	0.950	0.980

Table 3.3: Flow network inference on the MVSEC dataset

Methods	**Training Datasets**	**AEE**	**1px**	**3px**	**5px**
E-RAFT	DSEC	> 30.0	-	-	-
E-RAFT	BlinkFlow	3.48	0.19	0.65	0.82
E-RAFT (Ours)	BlinkFlow + TartanAir_v2	2.46	0.20	0.74	0.91

3.3.1 Inference on TartanAir-v2

The performance of TartanEVO is compared to a bunch of other algorithms like the TartanVO, ORB-slam (Mono). The data for this is provided in table 3.4, 3.6, and 3.5. We see that the geometry-based methods like ORB and EVO fail on all the trajectories in the environment. This is because the camera motion in these datasets are large and classical algorithms tend to fail. In most of the trajectories the TartanVO performs better than TartanEVO but in few trajectories of MiddleEast and ShoreCaves the tartanEVO performs better, this is because for environments like shorecaves have lower lighting and TartanEVO tend to perform better in these conditions.

3.3.2 Inference on VECTor

The VECTor is a dataset collected by Prophasee Gen3 which provide good sensitivity and resolution, when compared to the popular DAVIS346. Because of the VECTor dataset should have had dense number of events but that is not the case, as the environment is pretty sparse and featureless. Despite this lack of data in certain frames the TartanEVO perform pretty well.

Dataset	ORB-SLAM		EVO		TartanVO		TartanEVO (Ours)	
	MPE	ATE	MPE	ATE	MPE	ATE	MPE	ATE
CountryHouse DE_P000	-	-	-	-	**7.56**	**0.51**	24.06	0.534
CountryHouse DE_P001	-	-	-	-	**9.61**	**0.37**	16.42	0.464
CountryHouse DE_P002	-	-	-	-	**12.27**	**0.88**	17.52	0.573
CountryHouse DE_P003	-	-	-	-	**8.77**	0.515	19.55	**0.497**
CountryHouse DE_P004	-	-	-	-	**7.83**	**0.446**	24.21	0.632
CountryHouse DE_P005	-	-	-	-	**4.14**	**0.260**	14.48	0.495

Table 3.4: Comparison of methods on the TartanAir-v2:CountryHouse environment.

Dataset	ORB-SLAM		EVO		TartanVO		TartanEVO (Ours)	
	MPE	ATE	MPE	ATE	MPE	ATE	MPE	ATE
ShoreCaves_DE_P001	-	-	-	-	11.94	10.907	**5.75**	**9.42**
ShoreCaves_DE_P002	-	-	-	-	34.61	28.11	**10.80**	**22.17**
ShoreCaves_DE_P003	-	-	-	-	31.03	**21.40**	**12.11**	30.26
ShoreCaves_DE_P004	-	-	-	-	15.19	12.103	**4.32**	**8.86**
ShoreCaves_DE_P005	-	-	-	-	27.49	5.692	**7.08**	**2.34**
ShoreCaves_DE_P006	-	-	-	-	24.02	11.423	**8.11**	**5.56**
ShoreCaves_DE_P007	-	-	-	-	13.15	8.3402	**3.90**	**5.85**
ShoreCaves_DE_P008	-	-	-	-	5.37	2.194	**4.35**	**1.85**

Table 3.5: Comparison of methods on the TartanAir-v2:ShoreCaves environment.

TartanEVO is the best performing monoluar event camera odometry algorithm. In couple of trajectories like the ‘units-dolly‘ our algorithms performs better than ORB and in trajectories like ‘school-dolly‘ and ‘school-scooter‘ it performs better that the TartanVO.

3.3.3 Inference on MVSEC

The MVSEC dataset collected the popular DAVIS346 which is low resolution and low sensitivity when compared to the Propahsee Gen3. Despite this the TartanEVO perform pretty decently well. The TartanEVO. The result for this can be seen in the table 3.8.

Dataset	ORB-SLAM		EVO		TartanVO		TartanEVO (Ours)	
	MPE	ATE	MPE	ATE	MPE	ATE	MPE	ATE
MiddleEast_DE_P000	-	-	-	-	**5.62**	**2.23**	6.53	3.626
MiddleEast_DE_P001	-	-	-	-	**3.56**	2.110	4.60	**1.689**
MiddleEast_DE_P002	-	-	-	-	**5.31**	1.707	10.07	**1.358**
MiddleEast_DE_P003	-	-	-	-	6.68	4.661	**6.07**	**2.433**

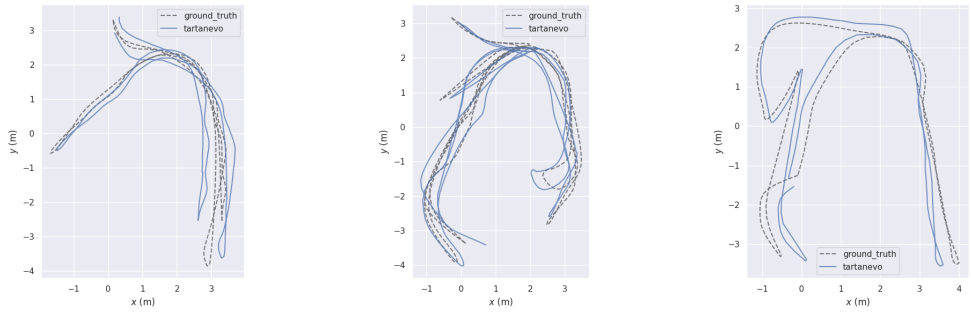
Table 3.6: Comparison of methods on the TartanAir-v2:MiddleEast environment.

Trajectory	ORB3 (StereoVO)		ESVO (StereoEO)		EVO (MonoEO)		TartanVO (MonoVO)		TartanEVO (MonoEO)	
	MPE	ATE	MPE	ATE	MPE	ATE	MPE	ATE	MPE	ATE
corridors-dolly	**1.03**	**0.80**	-	-	-	-	1.76	1.00	4.95	3.75
corridors-walk	**1.32**	**1.03**	-	-	-	-	7.23	1.15	10.38	2.27
school-dolly	**0.73**	**0.92**	10.90	13.71	-	-	20.41	1.38	8.37	3.94
school-scooter	**0.70**	**0.75**	9.21	9.83	-	-	7.43	1.86	4.30	1.61
units-dolly	7.64	18.06	-	-	-	-	**5.38**	**8.24**	6.71	8.89
units-scooter	**6.22**	14.50	-	-	-	-	7.96	**8.13**	9.13	10.79

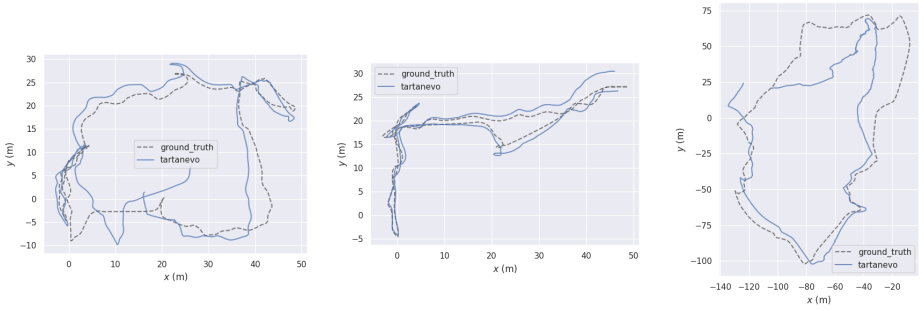
Table 3.7: Comparison of methods on VECTor trajectories. Data other than TartanVO and TartanEVO are adapted from [2][13].

Trajectory	ORB3 (StereoVO)		ESVO (StereoEO)		EVO (MonoEO)		TartanVO (MonoVO)		TartanEVO* (MonoEO)	
	MPE	ATE	MPE	ATE	MPE	ATE	MPE	ATE	MPE	ATE
indoor flying 1	5.31	142.0	4.00	107.0	5.09	136.0	**2.19**	**14.48**	10.86	90.30
indoor flying 2	5.65	170.0	3.66	110.0	-	-	**3.90**	**54.85**	9.15	82.46
indoor flying 3	2.90	154.0	**1.71**	91.0	2.58	137.0	2.20	**32.08**	10.71	41.82
indoor flying 4	6.99	58.0	-	-	-	-	7.43	**33.61**	**6.62**	39.80

Table 3.8: Comparison of methods on MVSEC indoor flying trajectories. Data other than TartanVO and TartanEVO are adapted from [2] [13].



(a) CountryHouse-P000 (b) CountryHouse-P002 (c) CountryHouse-P005



(d) MiddleEast-P000 (e) MiddleEast-P003 (f) ShoreCaves-P002

Figure 3.4: Example trajectories generated by TartanEVO on TartanAir-v2 Environments

Chapter 4

Conclusion

4.1 Summary of Contributions

In conclusion, this thesis introduces the world's largest event camera dataset, providing orders of magnitude more data than previously available. It has improved the performance of the flow network E-RAFT, enhancing its ability to generalize across different environments. Additionally, TartanEVO proves that event camera odometry can also provide robust odometry, offering evidence for the generalization capabilities of adapting frame-based architecture to event camera data. This work demonstrates significant advancements in event-based odometry by introducing new datasets and improving deep learning models to handle the unique characteristics of event camera data.

4.2 Future Work

Although TartanEVO demonstrates robustness and generalization in event camera odometry, its processing time and accuracy currently lag behind those of other camera-based methods. This gap highlights significant room for improvement. Therefore, exploring the following research directions holds considerable potential for improving the odometry performance.

4.2.1 Transformer based Networks

Recently, transformer-based flow networks like FlowFormer[11] and GMFlow[32] have gained popularity due to their effectiveness in capturing the global con-

text in optical flow estimation. Implementing such transformer-based algorithms for event cameras and should improve the performance. Adapting these models to the tartanEVO could potentially lead to significant improvements in accuracy for odometry tasks.

4.2.2 Optimizer with FlowNet

The flow network serves as a dense feature-matching front-end, providing detailed correspondence between events across frames. Instead of relying solely on a pose network for pose estimation, integrating with a backend optimization framework can provide more accurate motion estimates as this approach allows for optimization over multiple frames, not just pairwise comparisons.

Bibliography

[1] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016.

[2] Peiyu Chen, Weipeng Guan, and Peng Lu. Esvio: Event-based stereo visual inertial odometry. *IEEE Robotics and Automation Letters*, 8(6):3661–3668, 2023.

[3] F. Fraundorfer D. Scaramuzza. Visual odometry: Part i - the first 30 years and fundamentals. In *IEEE Robotics and Automation Magazine*, volume 18, 2011.

[4] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. In *arXiv:1607.02565*, July 2016.

[5] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *European Conference on Computer Vision (ECCV)*, September 2014.

[6] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[7] Daniel Gehrig, Mathias Gehrig, Javier Hidalgo-Carrió, and Davide Scaramuzza. Video to events: Recycling video datasets for event cameras. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, June 2020.

[8] Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza. DSEC: A stereo event camera dataset for driving scenarios. *IEEE Robotics and Automation Letters*, 2021.

[9] 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.

[10] Y Hu, S C Liu, and T Delbruck. v2e: From video frames to realistic DVS events. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2021.

[11] Zhaoyang Huang, Xiaoyu Shi, Chao Zhang, Qiang Wang, Yijin Li, Hongwei Qin, Jifeng Dai, Xiaogang Wang, and Hongsheng Li. Flowformer: A transformer architecture and its masked cost volume autoencoding for optical flow, 2023.

[12] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 225–234, 2007.

[13] Simon Klenk, Marvin Motzet, Lukas Koestler, and Daniel Cremers. Deep event visual odometry, 2023.

[14] Beat Kueng, Elias Mueggler, Guillermo Gallego, and Davide Scaramuzza. Low-latency visual odometry using event-based feature tracks. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 16–23, 2016.

[15] Ruihao Li, Sen Wang, Zhiqiang Long, and Dongbing Gu. Undeepvo: Monocular visual odometry through unsupervised deep learning. *CoRR*, abs/1709.06841, 2017.

[16] 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. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3881–3888, 2023.

[17] Songnan Lin, Ye Ma, Zhenhua Guo, and Bihan Wen. Dvs-voltmeter: Stochastic process-based event simulator for dynamic vision sensors. In *ECCV*, 2022.

[18] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *CoRR*, abs/1502.00956, 2015.

[19] Henri Rebecq, Daniel Gehrig, and Davide Scaramuzza. ESIM: an open event camera simulator. *Conf. on Robotics Learning (CoRL)*, October 2018.

[20] Henri Rebecq, Timo Horstschaefer, Guillermo Gallego, and Davide Scaramuzza. Evo: A geometric approach to event-based 6-dof parallel tracking and mapping in real time. *IEEE Robotics and Automation Letters*, 2(2):593–600, 2017.

[21] Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza. Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization. In *British Machine Vision Conference (BMVC)*, 2017.

[22] Cedric Scheerlinck. *How to See with an Event Camera*. PhD thesis, College of Engineering and Computer Science, The Australian National University, 2021.

[23] T. Stoffregen, C. Scheerlinck, D. Scaramuzza, T. Drummond, N. Barnes, L. Kleeman, and R. Mahoney. Reducing the sim-to-real gap for event cameras. In *European Conference on Computer Vision (ECCV)*, 2020.

[24] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[25] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow, 2020.

[26] Zachary Teed and Jia Deng. DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras. *Advances in neural information processing systems*, 2021.

[27] Zachary Teed, Lahav Lipson, and Jia Deng. Deep patch visual odometry. *Advances in Neural Information Processing Systems*, 2023.

[28] Antoni Rosinol Vidal, Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza. Ultimate slam? combining events, images, and imu for robust visual slam in hdr and high speed scenarios. In *IEEE Robotics and Automation Letters (RA-L)*, 2018.

[29] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 2043–2050. IEEE, 2017.

[30] Wenshan Wang, Yaoyu Hu, and Sebastian Scherer. Tartanvo: A generalizable learning-based vo. In *Conference on Robot Learning (CoRL)*, 2020.

[31] 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. *IROS International Conference on Intelligent Robots and Systems*, 2020.

[32] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching, 2022.

[33] Nan Yang, Lukas von Stumberg, Rui Wang, and Daniel Cremers. D3VO: deep depth, deep pose and deep uncertainty for monocular visual odometry. *CoRR*, abs/2003.01060, 2020.

[34] Georges Younes, Daniel Asmar, Elie Shammas, and John Zelek. Keyframe-based monocular slam: design, survey, and future directions. *Robotics and Autonomous Systems*, 98:67–88, 2017.

[35] Alex Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Evflownet: Self-supervised optical flow estimation for event-based cameras. In *Robotics: Science and Systems XIV*, RSS2018. Robotics: Science and Systems Foundation, June 2018.

[36] Alex Zihao Zhu, Dinesh Thakur, Tolga Özaslan, 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(3):2032–2039, 2018.