FEASIBILITY OF EVENT-BASED SENSORS TO DETECT AND TRACK UNRESOLVED, FAST-MOVING, AND SHORT-LIVED OBJECTS

Thesis

Submitted to

The School of Engineering of the

UNIVERSITY OF DAYTON

In Partial Fulfillment of the Requirements for

The Degree of

Master of Science in Electrical Engineering

By

Jonathan Luc Tinch

Dayton, Ohio

May, 2022

FEASIBILITY OF EVENT-BASED SENSORS TO DETECT AND TRACK UNRESOLVED, FAST-MOVING, AND SHORT-LIVED OBJECTS

Name: Tinch, Jonathan Luc

APPROVED BY:

ABSTRACT

FEASIBILITY OF EVENT-BASED SENSORS TO DETECT AND TRACK UNRESOLVED, FAST-MOVING, AND SHORT-LIVED OBJECTS

Name: Tinch, Jonathan Luc
University of Dayton

Advisor: Dr. Keigo Hirakawa

Event-based sensors are an emerging technology that hold the potential to change the computer vision field. Specifically, event-based sensors have increased dynamic range, increased temporal resolution, and lower power consumption. These qualities give a distinct advantage over traditional framing sensors in a variety of computer vision applications including high speed imaging, feature extraction and detection, etc. Event-based sensors have just recently begun to be tested in the context of space situational awareness, namely with the detection and tracking of slow moving objects (i.e., stars and satellites). The feasibility of event-based sensors to detect and track fast-moving, short-lived, and unresolved objects in the context of space situational awareness is a fairly novel approach with limited supporting literature. This question is investigated through the context of meteor detection and tracking with the purpose of pushing the event-based sensor capabilities to their limit. Two data collections are performed and a data set containing both meteors and distractors (i.e., stars, airplanes, etc.) is captured. A processing pipeline is developed to allow for denoising of raw event data and subsequent detection and tracking of meteors. A simple tracker with several thresholding methods is chosen to detect and track the meteors and is found to be the most computationally inexpensive method while providing reasonable results. A calibration process is also proposed to explain the logarithmic relationship between intensity and the generation of events. Lastly, the detection and tracking method along with calibration is compared to the hand-annotated ground truth obtained in the data collection to determine accuracy. Overall the simple tracker results in a majority of meteors being detected and accurately tracked over their entire duration. The calibration method also provides an accurate transformation from event magnitude to traditional pixel intensity.

I would like to dedicate this thesis to my fiancée, Hannah Ruth Gatlin, for always believing in and supporting me.

ACKNOWLEDGMENTS

I would like to first thank my fellow Intelligent Signal Systems Laboratory members at the Unveristy of Dayton for always being friendly and making the laboratory feel like home for the duration of my research. I'd also like to thank my wonderful support system, namely both my family and my fiancée's family, who have always encouraged me to continue my education. Last but certainly not least, I'd like to thank Dr. Keigo Hirakawa, my thesis advisor, for being an absolutely wonderful teacher, mentor, and friend.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

Event-based sensors are an emerging technology in the computer vision field that aim to mimic the retina of the human eye. The sensors, also called neuromorphic cameras or event cameras, operate on a high-level by detecting contrast changes within the camera field of view (FOV). These contrast changes are detected on an individual pixel level and the log-intensity changes are recorded as events. Also note that each pixel records a positive event when the pixel intensity increases and records a negative event when the pixel intensity decreases. The event-based sensors have several distinct advantages over the traditional framing sensors – namely increased dynamic range, lower latency, and lower power consumption. These advantages are achieved by recording each event asychronously and time-stamping each event with micro-second resolution. This provides a sparse three dimensional field containing all the important contrast changes within the sensor's FOV. Overall the event-based sensor is a novel technology in the image processing field and has been used in various applications ranging from aiding in autonomous vehicles to improving security [3] [4].

However, one area that has seen less investigation is the feasibility of event-based sensors to detect and track unresolved, fast-moving, and short-lived objects. In this thesis, meteor detection and tracking is used as a context in which we test the limits of the sensor. Meteors serve as a perfect example of an unresolved, fast-moving, and short-lived object – they can last anywhere from several milliseconds to several seconds and the faintest meteors only populate several pixels within the FOV of the event-based sensor. Meteors also cover a very large dynamic range that are challenging to image with conventional intensity cameras.

Since the sensors rely on pixel contrast changes, this provides an opportunity to explore the limits of this emerging technology.

1.1 Thesis Objective

Event-based sensors are a novel technology in the computer vision field and hold the potential to change many aspects of the field. The sensors operate on an individual pixel level and detect log-intensity changes of each pixel. The sensors themselves contain a relatively straight forward circuit that allows for low-latency and high temporal rate. Due to the novelty of this pixel design, a chapter is dedicated to explaining the intricacies of the event-based sensor including the advantages over conventional framing cameras. Research and publications focusing on the use of event-based sensors are also reviewed to allow for additional understanding of the technology and applications within the field of computer vision.

An additional field of event-based sensors focuses on their relation to space situational awareness. In short, space situational awareness relates to the detection of both man-made and naturally occurring objects in orbit. The primary objective is to prevent any collisions between these objects, whether it be space debris, satellites, etc. The primary motivation for this thesis is to assess the feasibility to detect unresolved fast-moving objects (i.e., meteors). The primary challenge with imaging such objects is namely the fast-moving and short-lived qualities. Specifically, meteors move at an extremely high velocity and are only visible for a second at most before burning out. Capturing meteors on traditional framing sensors is extremely difficult because of the low temporal rate relative to the event-based sensors. To prove feasibility we wish to at minimum provide evidence that event-based sensors can detect meteors. Additionally, we attempt to show that event-based sensors can be used to provide a track of the meteor's trajectory in the sky.

Limited data sets containing event data are available due to the novelty of event-based sensors. This is especially true when considering the unique data required for this problem. To provide a data set for both this problem and the scientific community at large, several data collections are performed to gather meteor-specific data sets. Two meteor showers are observed using event-based sensors, namely the Prophesee VGA and Megapixel sensors. The data collections occurred early August and mid December with the Perseids and Geminid meteor showers, respectively. The data collections also provide distractors such as sattelites, stars, and airplanes. Truth data is determined for both data sets and contain the spatio-temporal event information for both meteors and distractors. Overall approximately 51 meteors are observed across both data collections. The feasibility of simulated data is also considered later in this thesis.

While the event-based sensors produce sparse data, the initial data does include both hot pixels and noise. Hot pixels are defined as the pixels that generate events in the absence of irradiance changes and are comparable to pixels that consistently fire in traditional framing cameras. A method is developed to eliminate hot pixels without removing objects of interest such as meteors or any other distractors. Following the removal of hot pixels the data is then filtered using a technique called Inceptive Event Filtering [5]. The purpose of the inceptive event filtering is to separate all events within the event stream into three categories - inceptive events, trailing events, and noisy events. This filtering technique also provides substantial event reduction, allowing for greater computational efficiency moving forward. An event representation called Time-Ordered Recent Event (TORE) volumes is also considered to provide sparse time images to a convolutional neural network (CNN) [**?**].

The TORE volume images are generated for the Perseid data collection and due to time constraints have not yet been tested with a fully functioning CNN.

The pixels within the sensor will generate events according to the intensity of the contrast change within the respective pixel, subsequently generating events linked to the intensity of contrast change. The sensor hardware contains a low pass filter to ensure each pixel will generate a number of events based on the contrast change intensity. Due to the low-pass filter, it is hypothesized that a point source will cause an imbalance between the positive and negative events generated, in which negative events will "override" the positive events. This is due to the size of the object being the size of only one or several pixels. Since the object will enter and leave the pixel in a fraction of a second, the negative events are triggered before all of the positive events have fully generated (based on the intensity contrast change). This potential problem is analyzed in a secluded setting to verify its existence. A function generator and LED is used with a short pulsing square wave to mimic a short-lived point source object and the frequency of the function generator is varied to inspect the results on the positive and negative event generation. It is concluded through several tests that the negative events do override the positive events.

CHAPTER II

EVENT-BASED SENSOR BACKGROUND

2.1 General Overview

Event-based sensors provide high-speed sensing, low energy consumption, high dynamic range, and low computational requirements [1] [6]. Modeled after the photoreceptors in the human eye, event-based sensors record when detecting a change in intensity or contrast within a pixel. Specifically, the sensors detect changes of the log-intensity at a pixel. Each pixel within the sensor frame individually records events $e = (e_x, e_y, e_t, e_p)$, where the event e is described by its spatial location within the frame at pixel (e_x, e_y), time of occurrence at e_t, and polarity e_p indicating the event is either positive or negative. The positive and negative events describe an increase and decrease in pixel intensity respectively. The resulting event stream is able to clearly display the movement and size of an object through the positive and negative events. Further, the sensors do not rely on a fixed frame-rate, rather each pixel asynchronously records when a contrast change is detected. This reduces the overall amount of redundant information captured by the event-based sensor and also allows for lower computational cost and energy consumption. The ability of the sensor to operate on a pixel-by-pixel basis also allows for higher dynamic contrast because each pixel automatically adjusts to the initial sensor lighting.

Refer to Figure 2.1 for a high level example of the difference between the event-based sensor and a traditional frame-based sensor. Frame-based sensors generate a two dimensional image consisting of pixels defined by their (x, y) coordinate location at each time sequence. The frame-based sensor is limited by by their frames-per-second (FPS) rate, creating a discrepancy between each frame for objects in motion where the user may need to interpolate between frames to recover the missing spatial information. Event-based sensors, on the other hand, include a microsecond-level temporal resolution along with the standard (x, y) coordinate position. This quality of the sensor paired with the log-intensity change detection at each pixel allows for the event-based sensor to seamlessly and clearly track a fast moving object.

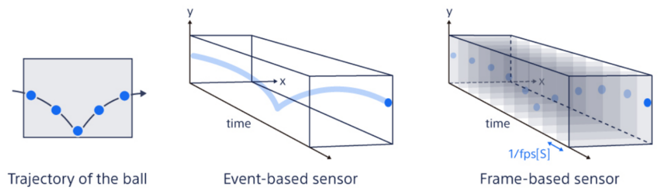


Figure 2.1: Comparison of event-based sensor and frame-based sensor [1]

The event-based sensors adapt to the contrast change at each pixel resulting in a logarithmic relationship between intensity and the number of events generated within a pixel. This is visualized in Figure 2.2, where a plot of *log intensity* versus *time* is shown. When a small increase and subsequent decrease is detected in intensity, a single positive and/or negative event may be generated. However when a greater change in intensity, whether positive or negative, multiple events of either polarity will be generated. Also note that the rate at which the events occur are reflective of the rate of change of the intensity. As in this case, the positive events are generated more rapidly to display a rapid increase in intensity while the negative events are generated slower to indicate a slower rate of decreased intensity.

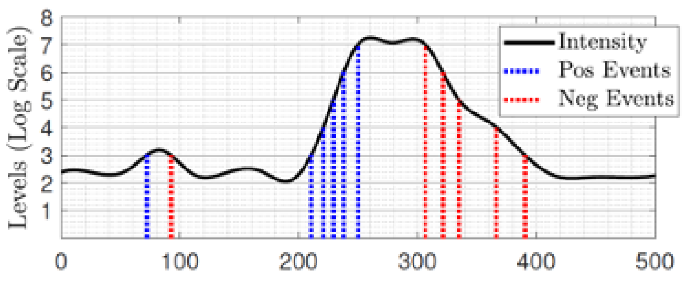
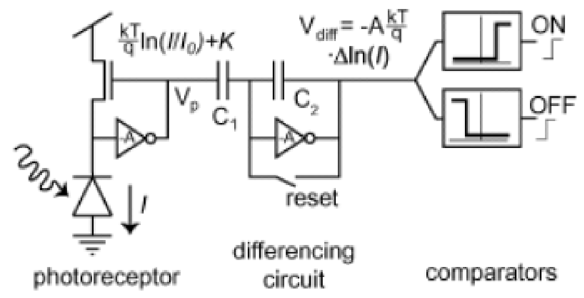


Figure 2.2: Visualization of the logarithmic relationship between events and intensity [1].

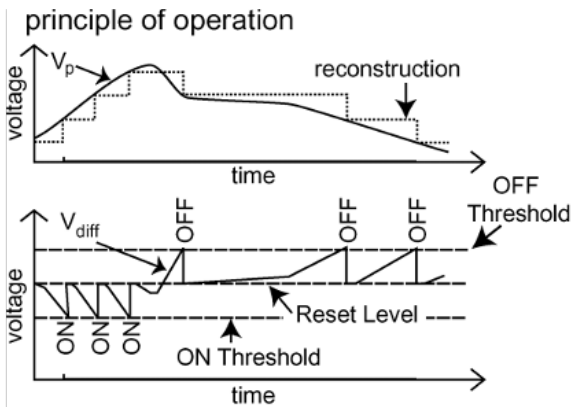
2.2 Circuit Overview

The complexity of the problem explored in this thesis also requires a basic knowledge of the circuitry within the event-based sensor and a simple explanation for the method in which the sensor determines an event has occurred (Figure 2.3). The sensor hardware consists of, at each pixel, a photoreceptor to obtain intensity information in the form of a log-intensity voltage, V_p. The voltage passes to a differencing circuit with reset which acts as a low-pass filter. The low-pass filter essentially allows for a threshold to be set in which the voltage is reset each time the threshold is exceeded (Figure 2.3b). The differencing circuit allows the charge accumulated by the $C1$ capacitor to trigger the reset multiple times, allowing V_{diff} to reset and retrigger the threshold. The difference voltage, V_{diff} is then intepretted by the comparators as a positive or negative change in intensity (i.e., contrast/luminance), and triggers one or several positive or negative events respectively.

The top plot of Figure 2.3b displays the rise and fall of the voltage as a relationship of time where the dotted line represents “reconstruction” of each threshold when triggered (mirrored in the bottom plot). The reconstruction also is reflective of when an event is generated (i.e., each positive unit step represents the generation of a positive event while each negative unit step represents the generation of a negative event). Note that both



(a) Basic circuit diagram of the event-based sensor.



(b) Plot displaying the relationship between voltage and ON/OFF thresholds.

Figure 2.3: Sensor hardware [1]

plots in Figure 2.3b are displayed to scale with one another in the x-axis. The bottom plot displays the ON threshold being triggered three times consecutively, shown as negative voltage due to the inverter of the circuitry in Figure 2.3a. This triggers three positive events and is followed by a fall in voltage (corresponding to a decrease in intensity) and triggers the OFF threshold, generating a negative event. Now a slow fall in intensity is observed and correspondingly, the last two OFF thresholds are slower to be triggered.

There are several parameters that can be tuned within the event-based sensor. The photoreceptor bias controls the rate at which the voltage increases in Figure 2.3a, effectively limiting the current from the photoreceptor. If the photoreceptor bias is high, it will respond to higher frequency noise and produce more noise in dimly illuminated scenes. On the other hand, if the photoreceptor bias is too low, the event-based sensor will be more insensitive to fast oscillations in intensity. Differential bias refers to the increase in voltage in Figure 2.3b, essentially controlling the sensitivity of the ON and OFF thresholds to intensity changes within the sensor and by extension the speed at which the sensor adjusts to the contrast changes. Too fast of a differential bias creates too many events per edge while too slow of a differential bias could contribute to one or no events corresponding to an edge. The threshold for a positive or negative event to be generated can also be tuned by the thresholds corresponding to ON and OFF comparators respectively. The refractory period corresponds to the time frame after an event is generated that a pixel does not respond to contrast changes, and is representative in Figure 2.3b by the short time between the voltage reset and the subsequent rise or fall in voltage.

CHAPTER III

LITERATURE REVIEW

3.1 Event-Based Time-Surfaces

Reducing dimensionality for event data streams is essential in allowing current state-of-the-art object classification algorithms to be used with novel event-based sensors. One such method of dimensional reduction is time-surfaces, in which an event is encoded with an intensity value relative to the contrast change at a pixel location [7]. Time-surfaces face the disadvantage of being sensitive to noise, recording multiple events for the same edge, etc. Conventional frame-based sensors often experience blurring and other undesirable artifacts within the image frame due to the motion of an object too quick for the sensor's integration period. Event-based sensors offer a solution for this problem in the form of high temporal resolution, low latency, etc. However current time-surface algorithms experience a similar problem of blurring. One solution proposed is Filtered Surface of Active Events (FSAE) [8]. FSAE focuses on corner detection and tracking, and succeeds in removing multiple events for a single edge. FSAE proves to be robust in representing simple features in the event space, such as strong edges. Another approach proposed by R. Wes Baldwin, At Al. to further improve the time-surface algorithm is called Inceptive Event Time-Surfaces (IETS) [5]. IETS improves upon FSAE to provide features robust to noise while removing approximately 70% of events, creating a significant increase in computational efficiency. The algorithm reaches nearly 100% accuracy for the N-CARS data set. The primary interest in the IETS for this thesis is the inceptive event filtering used to reduce the overall data in the event stream to nearly a quarter of the original data.

3.1.1 Inceptive Event Filtering

Filtering is essential in reducing the redundant and unusable information present in raw event stream data obtained from an event-based sensor. Inceptive event filtering allows for the events to separated into three event categories - inceptive events, trailing events, and noisy events. Inceptive events define the movement of objects within a scene and are defined as events without a temporally preceding event and with a temporally subsequent event, within a specified time threshold. Trailing events are defined as events that follow immediately after an inceptive event and are representative of the magnitude of the contrast change within a pixel. A larger contrast change within a pixel generates additional events, which are recorded in the form of trailing events. Last, noisy events are defined as events that do not have a temporally neighboring event within the time threshold and are therefore isolated and can be disregarded.

Table 3.1: Inceptive event filtering designations

	$t_i - t_{i+1} > \tau$	$t_i - t_{i+1} \leq \tau$
$t_i - t_{i-1} > \tau$	noisy event	inceptive event
$t_i - t_{i-1} \leq \tau$	trailing event	trailing event

Specifically, the three event designations can be described as found in Table 3.1 where t_i describes the time stamp of the ith event in the event stream and τ is a threshold representing the minimum time in which events are considered related. Additionally, Figure 3.1 displays a cross section of the event stream in which time is displayed on the y-axis and the x pixels are displayed on the x-axis. Note that the y pixel is kept constant in this demonstration. The inceptive events are denoted in green and correspond to an edge, while trailing events are displayed in blue and are seen temporally following the inceptive events. Noisy events are shown as red and do not have a temporally subsequent or preceding event.

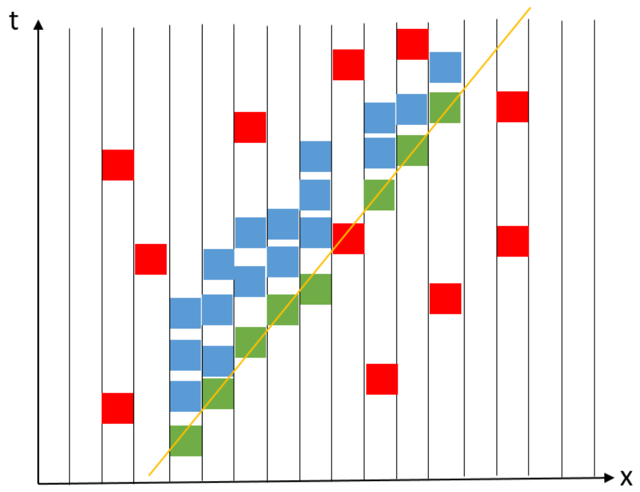


Figure 3.1: Visual representation of the inceptive event filtering technique.

Further, since multiple events are generated from a single edge encounter, we consider the number of trailing events to correspond to the edge/contrast magnitude. The event magnitude is assigned to the corresponding inceptive event to further reduce data size, and from here on out is referred to as the inceptive event magnitude.

3.2 TORE Volumes

Time-Ordered Recent Event (TORE) volumes is a two dimensional event representation designed to compress the sparse event data with minimal information loss [9]. TORE volumes uses local memory from past events to represent a “weighting” for the events shown in the two dimensional TORE representation (Figure 6.1). TORE volumes uses a “first in, first out” (FIFO) buffer of the n most recent events in a pixel (shown as $n = 3$)

to create the representation. Overall TORE volumes allows for dense reconstructions to be created so that conventional convolution neural networks (CNN) can be used with sparse data produced by event-based sensors.

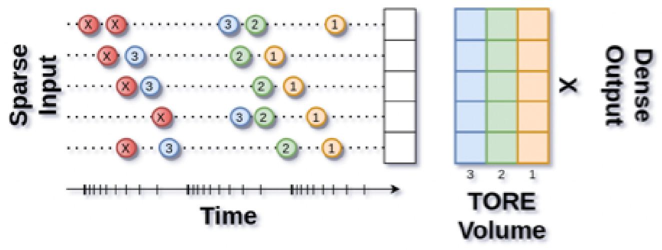


Figure 3.2: Visual representation of TORE volumes' useage of local memory

3.3 Event-Based Sensors and Space Situational Awareness

Several additional studies have been conducted to assess the feasibility of using event-based sensors for space situational awareness. Space situational awareness is the capability to detect and predict the physical location of both natural objects (i.e., meteors, stars, etc.) and man made objects (sattelites, etc.) in orbit of the planet with the main goal being to avoid collisions [10]. Simulated observation experiments have been performed to assess initial feasibility [2] [11]. In particular, Xiao, et. al. tests the feasibility for use of an event-based sensor attached to spacecraft in the atmosphere. Although the experiment was conducted in a simulated environment, the test setup included a sun simulator, slide rail, etc. with a 1 meter scale sized observation target (Figure 3.3). A DAVIS346 event-based sensor, produced by iniVation, is used for the experiment due to dual use of both the event-based and frame-based sensors. In other words, the DAVIS346 outputs both traditional and event data. Due to the high temporal rate and low power consumption, the event-based sensor is concluded to be feasible for use in space situational awareness.

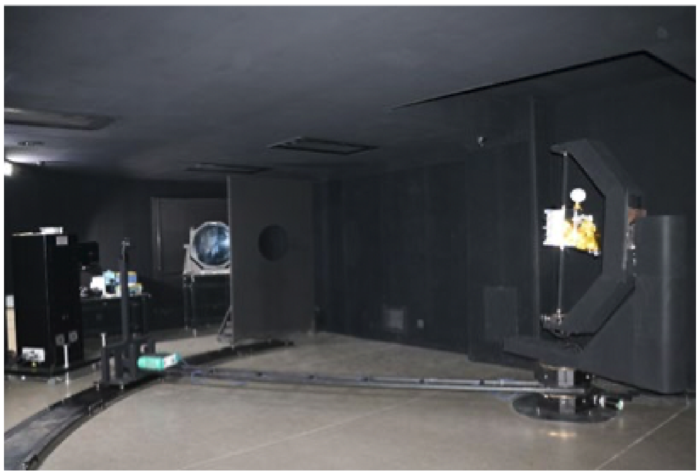


Figure 3.3: Simulation environment used for initial feasibility testing [2].

Further, a processing pipeline is developed in which the events will be reconstructed through a CNN to recover images which will then undergo traditional image processing techniques such as feature extraction, matching, pose estimation, etc. The findings primarily display the strengths of event-based sensing in the context of space situational awareness, namely low-power consumption, high dynamic range, and high temporal resolution.

Space situational awareness using event-based sensing has also been assessed from the ground [12] [13] [14]. Unfortunately a major disadvantage of event-based sensors is the lack of a large dataset, and this is especially true in the context of space situational awareness. The study conducted by Afshar, et al. provides expansion to an emerging field of study, Event-Based Space Imaging (EBSI). A novel dataset is collected by connecting an event-based sensor to a telescope and consists of 236 independent recordings and 572 label resident space objects. The dataset is the first event-based space imaging dataset in literature and provides over 8 hours and 377 million events. An additional 152 unlabel data streams containing 5 hours of recording is also collected using several different DAVIS sensors. Ground truth data is hand-labeled and allows for a subsequent detection and tracking pipeline to be proposed using several methods. Preprocessing involves a time-surface approach in which a region of interest (ROI) is found around an event of interest for use in further processing. The ROI contains all events within the radius around the event of interest. Feature detection is next performed by transforming the ROI to an angular activation vector, and is compared to an angular activation threshold before being passed onto the next stage of processing. The purpose of the angular activation threshold is to remove any events not associated with a streak on the time-surface. At a high level, the feature extractor acts as a denoiser for the surface activation filter.

Event-based tracking is performed by modeling each object as a neuron and membrane potential which decays over time. The membrane potential is increased by detecting events, and if the amount of detected events passes a threshold the object is considered activated. Likewise, if the membrane potential decays to 0, the object is deleted. Once an object is activated, it stays activated regardless of membrane potential decay. The objects are then used in a event-based line fitting tracker and effectively removes almost all false detections and correctly clusters events for each object. Several other method of event-based feature detection methods were also analyzed - namely Global Maximum Detection (GMD), Hough transform, and a third method that combines GMD and Hough transform along with the ground truth data. The feature detection method proposed by Asfar, et al. outperforms in informedness and specificity while while performing third and fourth best in sensitivity and processing time ratio, respectively.

3.4 Event-Based Sensor Applications

The novelty of event-based sensors has allowed for breakthrough applications in various fields of computer vision. Event-based sensors have been used to solve the problem of motion blurring in monitoring systems to classify targets using Histogram of Oriented Gradient (HOG) feature extraction and Support Vector Machine (SVM) [4]. Event-based sensors have also been used in many sensor fusion based experiments. Notably, an Dynamic and Active-pixel Vision Sensor (DAVIS) event-based sensor has been used alongside a Micro-Controller Unit (MCU) and First Person View (FPV) camera to allow for first autonomous driving platform via CNN [3]. Using both a LiDAR and event-based sensor also allows for a solution to sparse LiDAR depth information and the combination of sensors can be used to create a dense three dimensional depth reconstruction with the goal of improving intelligent robots [15]. The feasibility of event-based sensors to aid in identifying human gestures to teleoperate Unmanned Aerial Vehicles (UAVs) has also been analyzed with success due to the fast response at gesture recognition [16].

CHAPTER IV

DATA COLLECTION

4.1 Perseid and Geminid Meteor Showers

The Perseid meteor shower is one of the most popular meteor showers of the year, visible annually sometime between July and August. The meteor shower is caused by Earth passing through the tail of Comet Swift-Tuttle, a comet that passes right beside Earth every 133 years, the last time being 1992. At peak times the meteor shower is said to produce 150-200 meteors an hour, assuming little moonlight and light pollution interference.

This meteor shower was selected for the high meteor counts per hour in order to maximize potential meteors caught within the data collection. Two Prophesee VGA cameras were used for this data collection with a Computar 8.5mm f/1.3 and Computar 8mm f/1.4 lens. The sensors both have a 640x480 pixel resolution and 123.68 degree field of view (0.19325 degrees per pixel).

Note that due to the temporal low-pass filter within the event-based sensor hardware, trailing events are still generated after an edge arrival. Since stars and meteors are point-sources, meaning each occupy only one or few pixels at a given moment, the object will pass a pixel (generating a positive event) and then immediately leave a pixel (generating a negative event). This causes the negative events to potentially override the trailing positive events, creating an "edge-competition" between the two event polarities. This is explored in detail later in this thesis, however for the purpose of the first data collection the camera biases are tuned to allow for sensitivity towards positive events, allowing for a majority collection of positive events over negative events in each event stream. In other words, both cameras are tuned to be more sensitive to positive events and relatively insensitive to negative events. However, in the Geminid data collection the biases are tuned to allow for roughly equal sensitivity towards both positive and negative events.

Several practice runs of the data are conducted before the main data collection on the peak nights of the Perseid meteor shower in 2021. The purpose of the practice runs was to tune the biases of the sensors and find optimal settings for the aforementioned positive event sensitivity. The information for the location and date of both the practice runs and main data collection can be found in Table 4.1. The John Bryan State Park practice session was conducted in a large open field with no obstruction and provided no light pollution to the east (primary direction of the meteor shower). The Caesar Creek State Park practice session and main data collection session was conducted in a wide open parking lot with no light pollution, but did however have tall trees to the east that limited the field of view. The two sensors were placed in separate directions with one facing directly east and the other facing north. Due to the density of data from noise, the event streams of both cameras are limited to 2-3 minutes in which the event stream would be saved locally and restarted. This allowed for manageable size data files for later analysis. Note that unfortunately the second day of data collection had to be cancelled due to inclement weather conditions. The weather conditions on the date of the main data collection (August 11, 2021) consisted of high, thin clouds and a hot, humid atmosphere. Several air-crafts approached throughout the data collection and appear as distractors in the data set.

Table 4.1: Perseid meteor shower data collection dates

Practice Run	
Date	Location
2021-08-08	Caesar Creek State Park (Waynesville, OH)
2021-08-10	John Bryan State Park (Yellow Springs, OH)
Main Data Collection	
2021-08-11	Caesar Creek State Park (Waynesville, OH)
2021-08-12	Caesar Creek State Park (Waynesville, OH)

The main data collection provided over 5 hours of recorded data streams and 300 million events captured by the two Prophesee VGA cameras. Table 4.2 summaries the findings of the Perseid meteor shower data collection.

Table 4.2: Perseid meteor shower data collection summary

Meteors Captured	31
Instances of multiple meteors in same event stream	4
Median duration of meteors	290 ms
Shortest duration	110 ms
Median distance of meteors	17.39 pixels (3.36 degrees)
Median velocity of meteors	105.71 $\frac{pixels}{second}$ (20.43 $\frac{degrees}{second}$)
Fastest velocity	14907 $\frac{pixels}{second}$ (2880.78 $\frac{degrees}{second}$)

Also note one particular meteor of interest with a distance of 193.74 pixels (37.44 degrees) and lasted 1.34 seconds (Figure 4.1). Note that a distractor (airplane) is present above the meteor.

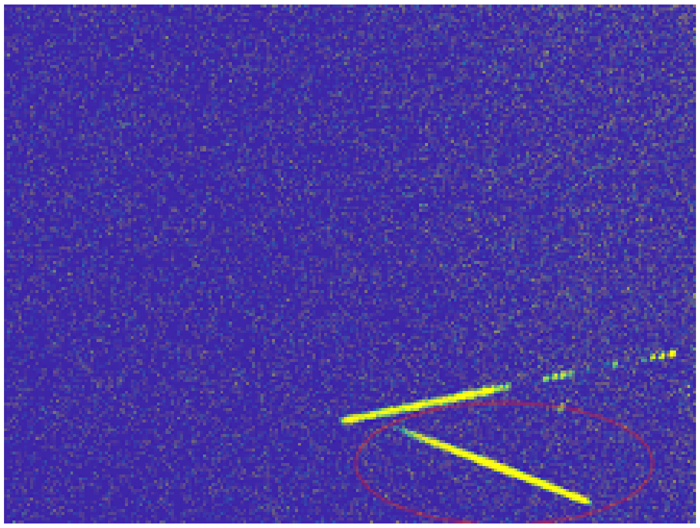


Figure 4.1: Largest meteor captured by the Prophesee VGA event-based sensor

The Geminid meteor shower occurs every year around mid-December and is caused by the 3200 Phaethon asteroid. With a peak hourly meteor rate of approximately 120 meteors, the Geminid meteor shower is selected for the second data collection. This data collection is performed over a single night (December 14, 2021). Two event-based sensors are again used for this collection - the Prophesee Megapixel sensor with a 5mm lens and the Prophesee VGA sensor with a fisheye lens. The purpose of this data collection is to expand the initial dataset with higher resolution images via the Megapixel sensor and also to find the feasibility of using a fisheye lens in future data collections. Additionally, the biases of both sensors are tuned to allow for equal sensitivity to both positive and negative events. The advantage of the fisheye lens is the wide field of view, being approximately 370 degrees. Approximately 29 meteors are caught within the FOV of either sensor over a 3 hour time span.

Overall a wide range of meteors are captured on Prophesee VGA and Megapixel sensors using a variety of lenses. This provides a reasonable dataset for initial analysis into the feasability of event-based sensors to detect and track unresolved, fast-moving, and short-lived objects.

4.2 Manual Annotation of Truth Data

The events corresponding to the meteor were annotated in a semi-supervised manner. We hereafter refer to this as ”ground truth data” as it can be used to verify the accuracy of automatic track extraction. For these initial cases the truth data is found through manual annotation. First, each time stream is condensed along the time axis to produce a two dimensional color map image displaying the intensity of the events. The images are visually analyzed to determine which contain a meteor, which in turn determines which event streams move onward for further analysis. Next, the event streams themselves are analyzed within the three dimensional space. Two events are chosen as a beginning and end events for the meteor, respectively. The properties of the events are recorded, specifically (x, y) position of the beginning and end events and their corresponding time difference. Note that for the rest of this chapter the ground truth values are referred to in capital letters as,

$$E^{st} = (E_x^{st}, E_y^{st}, E_t^{st}), \tag{4.1}$$

$$E^{end} = (E_x^{end}, E_y^{end}, E_t^{end}), \tag{4.2}$$

for the start and end events of the meteors, respectively. All truth data is recorded in a comma delimited file for use in tracking (Figure 4.2).

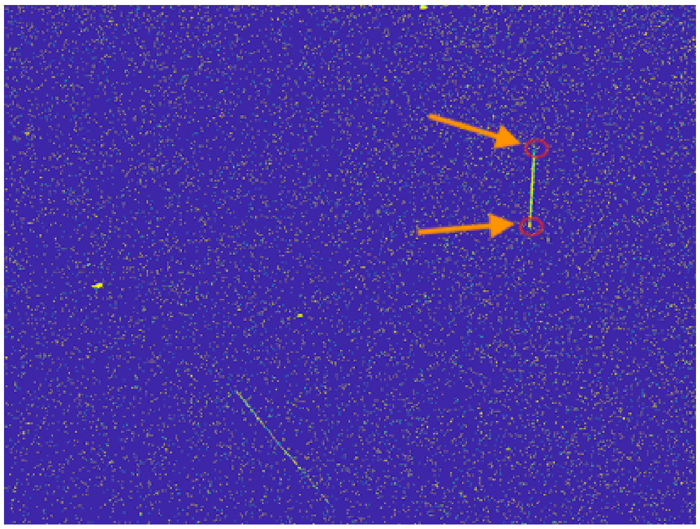


Figure 4.2: Start and end events of the meteor are recorded.

4.2.1 Meteor Ground Truth Tracking

Ground truth data for meteor tracks are determined using the manually annotated spatial and temporal ground truth data for the start and end events of each meteor. A linear line fitting is performed using this data and all events are first temporally aligned and shifted relative to the meteor end event found in Chapter 4.2 following,

$$e^i_{t\Delta} = E^{end}_t - e^i_t, \tag{4.3}$$

where $e^i_{t\Delta}$ denotes the shifted time stamp of the ith event in the data stream, E^{end}_t represents the time stamp of meteor end event, and e^i_t represents the original time stamp of ith event in the event stream. Note that each event in the event stream now is denoted by,

$$e^i_\Delta = (e^i_x, e^i_y, e^i_{t\Delta}) \tag{4.4}$$

where e^i_x and e^i_y are the ith event's original coordinate position. This is repeated for each event in the original event stream, including the meteor start event. The cross product is then found between each time-shifted event and the time-shifted meteor start event using,

$$\zeta^i = cross(e^i_\Delta, E^{st}_\Delta) = (\zeta^i_x, \zeta^i_y, \zeta^i_z) \tag{4.5}$$

where ζ^i is the cross product of the ith time-shifted event and the time-shifted meteor start event. The projection error of each event can then be found by using,

$$error = \frac{N(\zeta^i)}{N(E^{st}_\Delta)}, \tag{4.6}$$

where $N(\zeta^i)$ is the euclidean norm of the ith cross product in the event stream and $N(\zeta^i)$ is the euclidean norm of the time-shifted meteor start event. The Euclidean norm is further described as,

$$N(e^i) = \sqrt{\sum_{k=1}^{3} |e^i_k|^2}, \tag{4.7}$$

where $N(e^i)$ is the Euclidean norm of the ith event with 3 total elements - (e^i_x, e^i_y, e^i_t). The projection error for each event then undergoes an error threshold to determine if the event is included in the meteor tracks. The thresholding is described by,

$$\begin{cases} e^i \in \Gamma_j, & \text{if } error < \tau_{error} \\ e^i \notin \Gamma_j, & \text{if } otherwise \end{cases}, \tag{4.8}$$

where the Γ_j is the jth track and τ is a user defined threshold measured in pixels. The projection error threshold is empirically determined to be $\tau_{error} = 4$ pixels. The simple thresholding is found to work well in most cases, however due to the density of the event stream the thresholding isn't perfect and consequently the meteor tracks may contain trace amounts of noisy events. The ground truth meteors tracks are found for all meteors in the data collection detailed in Chapter III. An example of the ground truth meteor tracks can be seen in Figure 4.3.

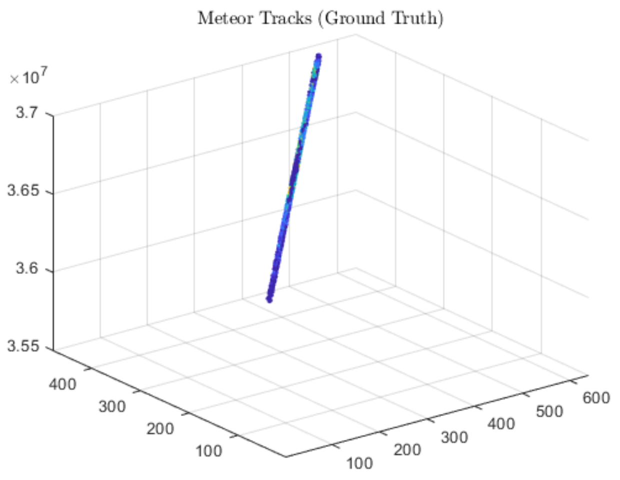


Figure 4.3: Example ground truth for meteor tracks.

4.3 Simulated Data

Several methods of simulating event-based meteor data are possible. One particular method utilizes Stellarium, an open source planetarium that runs on all major operating systems. The program strives to provide a realistic sky containing stars and meteors in a 3D dimensional space, attempting to mimic a telescope or binoculars. Within Stellarium, 3 dimension object models for objects such as stars and meteors can be used along with system settings to replicate meteor showers. However, the simulated meteors speed and resolution are limited by the hardware running the simulation. Specifically, this means that the FPS is limited by what the graphics card can render. Testing has shown a maximum of around 120 FPS is the obtained, which lacks in comparison to real-world data in which the event-based sensors can capture meteors with a much higher speed and shorter duration. The simulations also display many computational artifacts, a result of the interpolation algorithms that are not designed for sparse features like meteors. This combination of computational limits provide a simulated dataset that appears to be disconnected and choppy, compared to the clean continuity of lines found in the real world data.

Another reason for real-world data over simulated data is to preserve the hardware qualities of the event-based sensors. As mentioned before, an edge competition between positive and negative events is present in the real-world data due to the lowpass filter within the hardware and the point particle nature of the meteors and stars. Unfortunately the simulated data fails to reproduce this quality of the real-world data. Overall, the simulated meteors do not accurately represent the meteors collected through real-world data collections.

CHAPTER V

DATA PREPROCESSING

The data obtained from the meteor collections provide an abundance of data, however due to the characteristics of the event-based sensor hardware several methods of filtering are required. The processing pipeline consists of several stages to yield data feasible for object detection and tracking. First, two methods are developed and tested for hot pixel and star suppression, respectively. Hot pixels are defined as the pixels that generate events in the absence of irradiance changes. Stars are extremely slow moving objects within the event-based sensor FOV. Stars are considered for filtering in the same step as hot pixels due to their slow spatial qualities. Both the hot pixels and stars provide unusable events that cause an excess amount of data and consequently through hot pixel and star suppression the data is reduced. A method of hot pixel and star suppression is proposed and the resulting data is then filtered using a novel denoising technique described in Chapter III, inceptive event filtering.

The initial data rate for the output of the event-based sensors is roughly 16,000 events per second. The purpose of filtering the event streams is to both reduce the number of total events before subsequent processing and to provide more accurate tracks of the meteors and distractors in the three dimensional space. Distractors consist of stars, satellites, and airplanes.

5.1 Hot Pixel and Star Suppression

Two methods of hot pixel and star suppression are developed and tested using the meteor data collection. Both hot pixel and star suppression methods rely on thresholding by events per second rate. In other words, each pixel within the frame of the event stream have their event count summed according to,

$$C_k(x, y) = \sum_{i=1} \delta(e_x - x)\delta(e_y - y), \tag{5.1}$$

where C_k denotes the total counts at each (x, y) pixel location in the kth event stream. The resulting hot pixel suppression mask contains a value of 1 if the number of events in a pixel location exceeds a user defined threshold. The threshold is determined empirically and is in the form of events per second.

The subsequent thresholding follows,

$$M^k(x, y) = \delta(C_k(x, y) > \tau_C), \tag{5.2}$$

where (x, y) indicates the coordinate position and M^k denotes the mask for the kth event stream. The thresholding is described by,

$$\tau_C = \frac{\lambda max(e_t^i)}{1e6} \tag{5.3}$$

where λ is in events per second and $max(e_t^i)$ is the event stream length of time in microseconds. Note that stars are typically included in the mask detailed in Equation 5.2 because of the simple temporal thresholding. Typically two event stream masks are found before moving forward (i.e., M_1 and M_2). The first event stream corresponds to M_1 and contains the meteor to be analyzed. The second event stream corresponds to M_2 and is chosen to be an event stream of arbitrary length in which the bias settings are the same as the first event stream, but contains a scene with no contrast changes (i.e., sensor with lens cap). This is to provide the additional event sequence with identical hot pixels, as they are sensor and bias specific, but not contain any stars.

The first method uses two separate sequences as described above to determine if a pixel should be designated as a hot pixel or star. The function of the first method can be summarized by the following equation,

$$M_{hot}(x,y) = \delta(M_1(x,y))\delta(M_2(x,y)), \tag{5.4}$$

where $M_{hot}(x,y)$ is the resulting two dimensional hot pixel mask. The hot pixel mask, M_{hot}, is then compared to M_1 to find the subsequent star mask, M_{star} of the first event stream according to

$$M_{star}(x,y) = \delta(M_1(x,y))\delta(1 - M_{hot}(x,y)), \tag{5.5}$$

The star mask is essentially assessing the (x,y) pixel locations of both masks where M_1 is designated a hot pixel or star and M_{hot} is not designated a hot pixel. This combination of logic filters the hot pixels out of the hot pixel and star mask M_1 and results in exclusively the star locations.

Both the hot pixel mask and star mask are then applied according to,

$$\{e^i \notin e^j \mid \delta(M_{hot}(e^i_x, e^i_y))\delta(M_{star}(e^i_x, e^i_y)) = 1\}, \tag{5.6}$$

where e^i is the ith event in the event stream and e^j is the jth event stream. Also note that this expression is invariant of time and only depends on the pixel position in the sensor frame. The advantage of using two individual hot pixel masks is the increased confidence in the final hot pixel mask while a disadvantage is the two event streams must be used, increasing the computational complexity of the algorithm and subsequent time needed for execution.

5.2 Inceptive Event Filtering

Although covered in Chapter III, inceptive event filtering is briefly covered here as well. Inceptive event filtering was developed by Wes Baldwin, et. al. and provides a simple and intuitive method for creating a sparse data set from an event-based sensor data stream while retaining the advantageous information provided by the raw data set, namely event magnitude. Inceptive event filtering divides the events that comprise the data stream into three designations - inceptive events, trailing events, and noisy events. The inceptive event filtering is used on the event stream following hot pixel and star filtering techniques.

5.3 Results

The method of hot pixel and star filtering followed by the inceptive event filtering is applied to multiple raw event streams gathered from the Perseids meteor shower. Note the raw sensor data displayed in Figure 5.1. The event stream is viewed with the time axis expanded and the x and y axes along the horizontal plane. As expected, the raw event stream is extremely condensed and is the motivation for data minimization in the preprocessing stage.

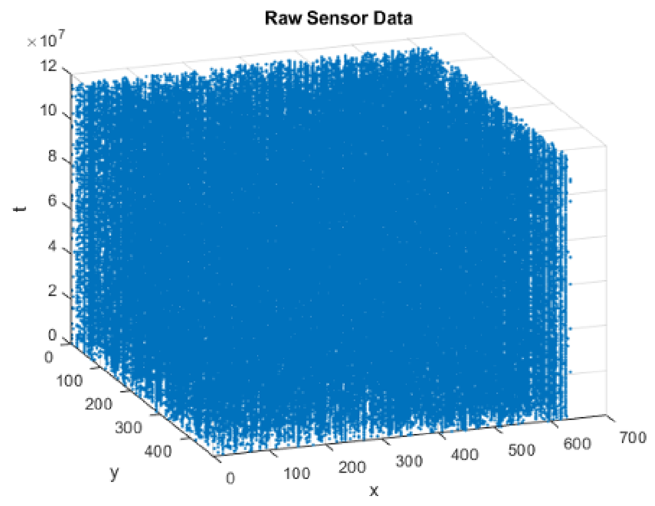


Figure 5.1: Raw data from the Prophesee VGA event-based sensor.

Figure 5.2 display the method of hot pixel and star suppression. Upon visual analysis, the results appear to be more sparse than the original raw data, as expected. Additionally, a distractor can be observed in the results, shown as a curved line (more prominently seen towards the beginning of the time axis at approximately $(x, y) = (200, 400)$).

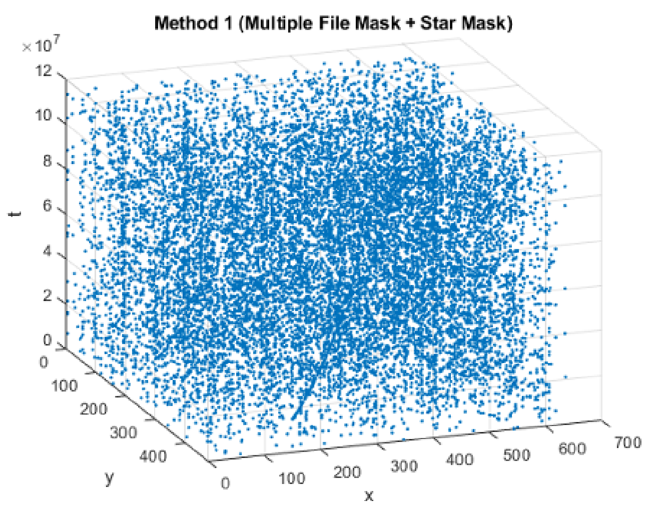


Figure 5.2: Raw data after hot pixel and star filtering.

Last, the inceptive event filtering is applied to the images. In Figure 5.3 the meteor and distractor are clearly visible, indicated by the straight horizontal line and curved downward line respectively. The stars, indicated by the vertical light blue events, are successfully filtered out but added here to display their position in the event stream.

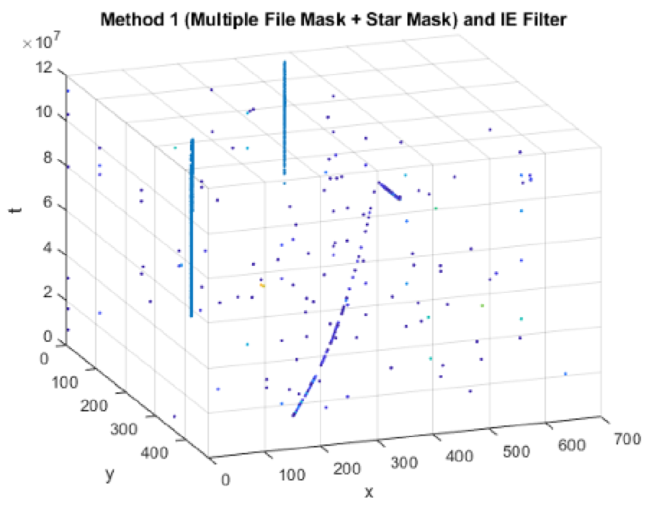


Figure 5.3: Raw data and subsequent hot pixel/star filtering and inceptive event filtering.

CHAPTER VI

DETECTION AND TRACKING METHODS

6.1 Simple Tracker

A simple tracker in the context of event-based data provides a computationally inexpensive method of detecting and tracking objects in the three dimensional space. To begin, a grid with specified density (i.e., space between grid intersections) is created from the event-based sensor's frame dimensions. The grid is necessary because no first event exists to begin a meteor track when the tracker is initially activated. The grid essentially acts as a starting point for the tracker and begins at the first time step of the event stream. The intersections of the grid are considered "placeholder" events and are removed from the result of each object track within the event stream.

Next, each event in the original event stream is individually assessed. To improve computational efficiency and take advantage of MATLAB's linear indexing, the event is expanded into a matrix with each row consisting of the spatial and temporal information of the particular event and each column a copy of the spatial and temporal information. The length of rows of the point matrix is determined by the total number of the events in the event stream.

A second point matrix is also created in which each row corresponds to a placeholder event within the grid and the columns similarly consist of the the spatial and temporal information of each placeholder event. The element-wise three dimensional euclidean distance is found between the two point matrices and effectively produces a distance calculation between the particular event and each of placeholder events. Next, the event of minimum distance is compared to a minimum distance threshold to determine if the event of minimum distance is the next event in the object tracks. If the threshold condition is passed, the event of minimum distance is recorded to be the second event of the object track (following the initial event being assessed). The second event of the object track then takes the place of the placeholder event in the second point matrix and the process repeats for each event. Events that are within the distance threshold of one another will be recorded in the same track.

If the particular event is equidistant from two other events, the spatial distance of the two events will be compared to the particular event, with the event of minimum distance being the next event in the object track. If the two events are spatially equidistant from the particular event, then the next event in the track is chosen at random. This feature unfortunately adds an element of randomness to the simple tracker, however is unavoidable due to the nature of the simple tracker.

Due to the conditional statements of the simple tracker, namely for when the tracker is considering two events as the next event in the object tracks, the simple tracker may select an event as part of the object tracks that is temporally unlikely. An additional filter is added to the simple tracker that allows for temporal outlier events to be removed from their respective object tracks. Outliers are defined as events with temporal elements more than 3 Median Absolute Deviations (MAD) away from the median and the outlier filter is described by

$$\{e^i \notin \Gamma^j \mid e_t^i > 3MAD\}, \tag{6.1}$$

where Γ^j is the meteor track corresponding to the jth event stream. The object tracks next undergo velocity filtering. The time difference between the beginning and end of the object track is recorded as well as the three dimensional euclidean distance of the object tracks in pixels. The velocity is then then found according to

$$v^j = \frac{distance}{speed}, \tag{6.2}$$

where distance is in pixels and speed is in seconds, and v^j is the velocity of the jth event stream track in pixels per second. The velocity is then normalized by the focal length of the lens used for data collection and follows

$$v^j_{norm} = \frac{v^j}{f}, \tag{6.3}$$

where f is the focal length of the lens used to collect the event-based data. The normalization of the velocity allows for the same simple tracker algorithm to be used with the all data taken by an event-based sensor, regardless of the lens used during data collection. The Perseid and Geminid meteor data collection allows for a minimum and maximum normalized velocity threshold to be set. Consequently, the tracks that are deemed noise or distractors are filtered out using,

$$\{e^i \in \Gamma^j \mid v^j_{norm} < v_{max} \wedge v^j_{norm} > v_{min}\}, \tag{6.4}$$

where v_{norm}, v_{max}, v_{min} represent normalized velocity of the track, maximum normalized velocity threshold, and minimum normalized velocity threshold respectively.

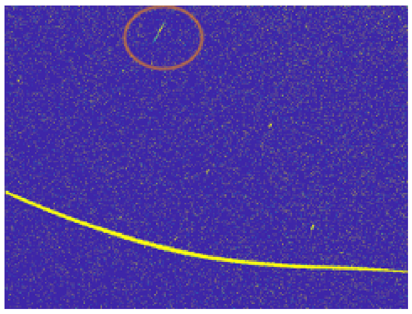
Last, the object tracks are passed through a simple filter and disregards object tracks that contain under a minimum number of events. This allows for additional robustness to noise and noisy tracks.

6.2 Additional Detection Methods

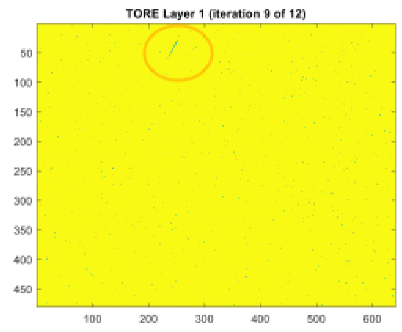
6.2.1 TORE Volumes and CNN

An additional option to detect and track meteors is through TORE volumes. The algorithm, originally developed by Baldwin, et al., allows for a traditional two dimensional representation of the event stream. The two dimensional images would then be used in a conventional CNN to allow for the detection and tracking of meteors. The algorithm is briefly considered and several TORE volume representations are generated for the meteors (Figure 6.1).

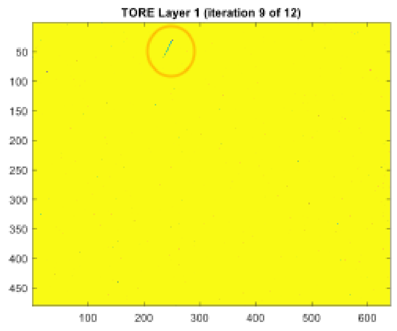
While the TORE volumes representations initially look promising, the simple tracker allows for an alternative that does not require a CNN which overall allows for much quicker computations.



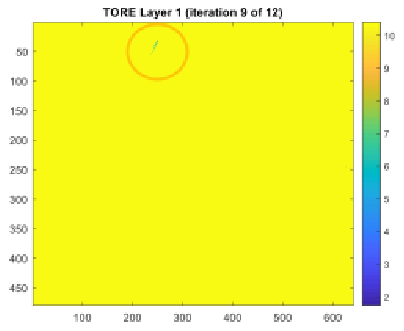
(a) Two dimensional representation of raw event data.



(b) TORE volumes representation of raw event data.



(c) TORE volumes representation of event data after hot pixel and star suppression.



(d) TORE volumes representation of event data after hot pixel and star suppression and inceptive event filtering.

Figure 6.1: TORE volumes representations after each step in the preprocessing pipeline.

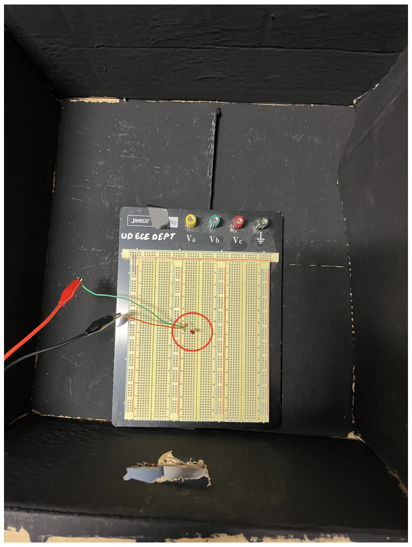
CHAPTER VII

EVENT-BASED SENSOR CALIBRATION

7.1 Point Source Analysis

Event-based sensors record positive events and negative events within a pixel when a increase and decrease in intensity is detected, respectively. A low-pass filter is present in the sensor hardware to allow for the number of events generated to reflect the intensity of the contrast change. It is hypothesized that due to the point particle nature of meteors, the subsequent negative events will overwrite the trailing positive events. This is accounted for in Chapter IV by tuning the event-sensor biases to allow more sensitivity towards positive events in the initial data collection.

An experiment is performed to confirm the validity of the edge competition hypothesis. The procedure involves observing a point source illumination in a controlled environment and measuring the subsequent positive and negative events. The point source is chosen to be a red LED connected to a breadboard and powered at a constant voltage by a square function generator (Figure 7.1a). The interior of a cardboard box is painted with BLK3.0 paint to ensure as little reflections as possible and a small hole is cut into the side of the box to allow the event-based sensor to observe the flashing LED (Figure 7.1b). The LED is driven at increasing frequencies and the resulting positive and negative events are recorded.



(a) Point source red LED and breadboard.

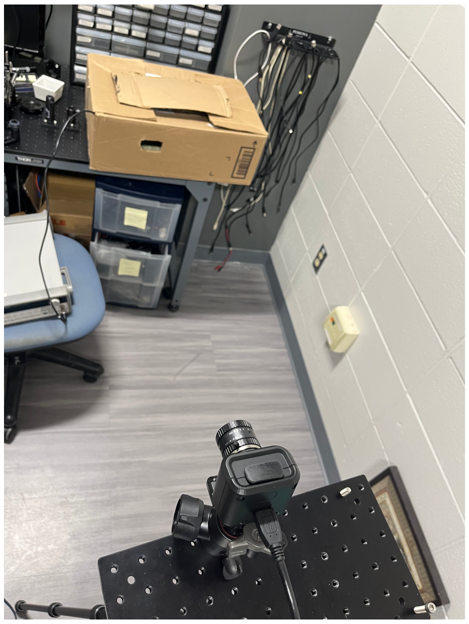

(b) Test setup with controlled lighting and event-based sensor.

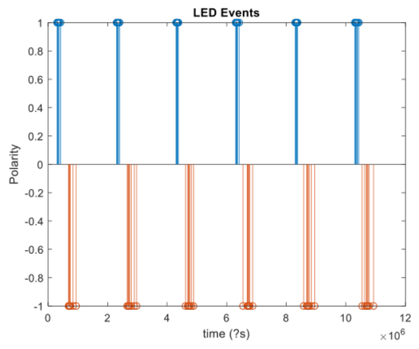
Figure 7.1: Test setup for point source analysis.

A single pixel is chosen to ensure consistent illumination for all frequencies analyzed. The LED is also analyzed with a gray-scale Forward Looking Infrared (FLIR) camera to ensure the intensity of the pixel remains consistent across all frequencies.

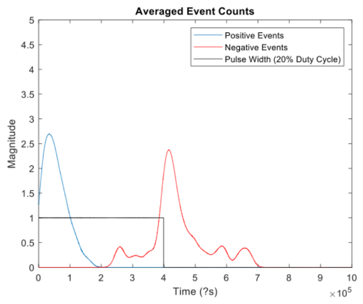
As mentioned in Chapter II, an increase in intensity within a pixel generates a positive event while a decrease in intensity generates a negative event. Therefore we can observe the behavior of the positive and negative events by gradually increasing the frequency. The LED is driven at a constant voltage for all frequencies to ensure constant illumination and the duty cycle is set to 20%. Note that the duty cycle is set to the minimum value allowed by the function generator and setting the duty cycle as low as possible allows for a short "on" period of LED illumination.

The plots on the left of Figure 7.2 display the number of events with each square wave pulse throughout the entire recording. The y-axis displays the polarity of events, with blue being positive and red being negative. The x-axis represents the time axis of the event stream and is measured in microseconds. As expected, when the LED is “on” positive events are generated and are followed by negative events when the LED turns “off”. The plots on the right of Figure 7.2 display the average positive and negative event count over a one second period, with blue being positive events and red being negative events. A low-pass smoothing filter has been applied to the averaged event counts to allow for easy analysis of the amount of events within the pulse. A unit step function has also been added to the plot to represent the pulse width of the square wave, and is shown in black. Note that the beginning of the plot is considered the start of the “on” period of the LED.

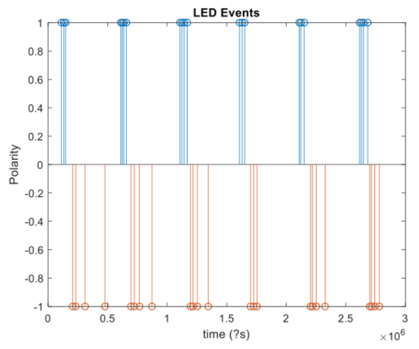
Ideally, the beginning of the positive and negative event generation will correspond exactly with the “on” and “off” periods of the square wave in right plots of Figure 7.2. However due to real-world hardware constraints, this is not quite the case. Regardless, the result is sufficient for point particle analysis. As the frequency is increased, the period of the square wave decreases (as depicted by the black pulse width unit-step line). As stated above the LED is verified to maintain constant illumination across all frequencies, indicating an ideal scenario in which the number of events across all frequencies remain constant. However, as frequency increases the number of events actually decrease. As the period of the square wave decreases, the time between when positive and negative events are generated becomes smaller. The positive and negative events begin to “compete” within the hardware with a final result of negative events almost completely overriding the positive events in Figure 7.2e.



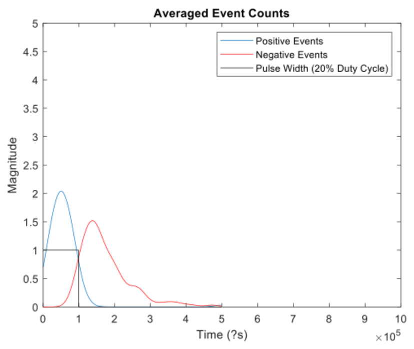
(a) Positive/negative event counts at 0.5Hz.



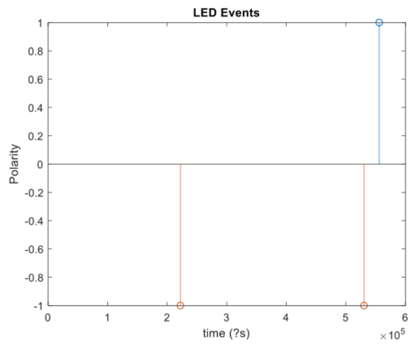
(b) Event Representation at 0.5Hz.



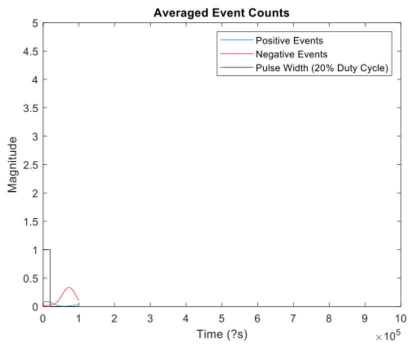
(c) Positive/negative event counts at 2Hz.



(d) Event Representation at 2Hz.



(e) Positive/negative event counts at 8Hz.



(f) Event Representation at 8Hz

Figure 7.2: Results of the point source analysis

7.2 Calibration process

The event-based sensors produce events based on a logarithmic intensity scale. This results in the minimum contrast change required to generate an initial event to be much lower than a traditional frame-based sensor. However, this also creates the need for a calibration process to accurately depict an intensity when compared to a traditional frame-based sensor. Since the event generation is dependent on bias and the sensor, this calibration process is initially performed with the same sensor and biases as the initial Perseid data collection, namely the Prophesee VGA event-based sensor.

The process involves translating the inceptive events and their corresponding magnitudes to intensity values by using a similar process as Chapter 7.1. The calibration process uses both a event-based sensor and a FLIR camera working in tandem to record the same illumination source (Figure 7.3).

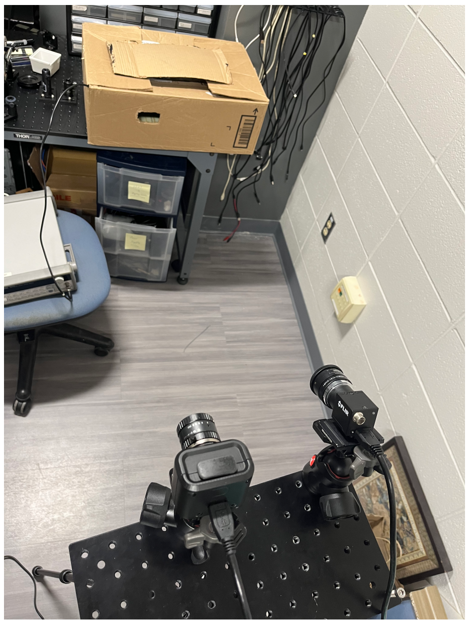


Figure 7.3: Event-based sensor and FLIR camera setup for the calibration experiment.

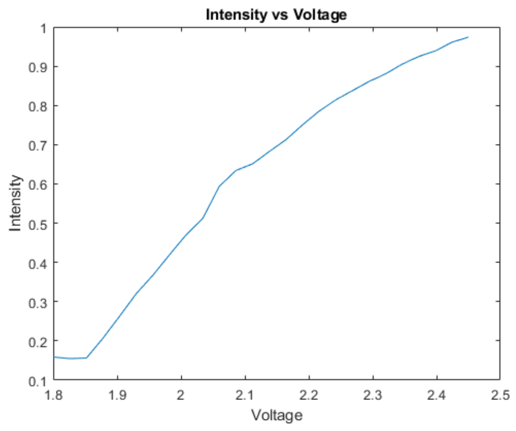
As with the point source analysis, a red LED is used as the illumination source. Note the LED is set up the same as shown in Figure 7.1a. The FLIR camera gamma is set to $\gamma = 1$ for the entire procedure as to enable us to disregard its non-linearity. The gain and exposure times are empirically tuned to allow for a peak total event count of roughly 45 events before the FLIR camera image saturates. Note that due to the biases being tuned to be more sensitive to positive events, a peak of roughly 26 positive events and 19 negative events are observed.

Three images are captured with the FLIR camera at each voltage and a single pixel representing maximum intensity of the corresponding image is chosen as the measure of intensity. The three intensity values for each voltage are averaged to provide a more accurate measure of luminance. A single event stream is also captured at each voltage for roughly 12 seconds and the function generator is kept at a constant 0.5Hz square wave with 50% duty cycle allowing for roughly 6 periods to pass. The events at each period are averaged to find an average event count for each voltage.

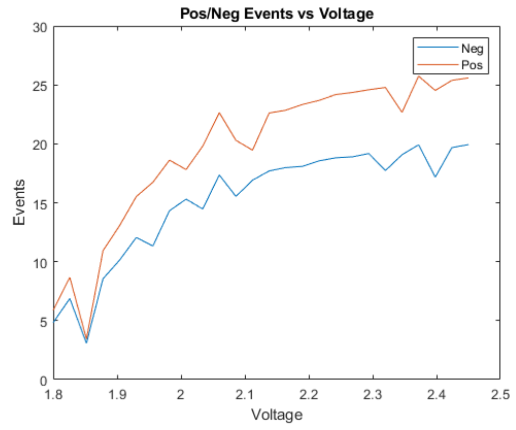
The range of voltages are chosen based off the intensity values to ensure a full range between the noise floor and saturation point of the FLIR camera. The values are relatively arbitrary and provide a full range of intensity values as desired (Figure 7.4a). Note the intensity values are normalized by the following,

$$intensity = \frac{intensity_{init}}{intensity_{max}}, \tag{7.1}$$

where $intensity_{init}$ are the initial intensity values generated from the FLIR camera and $intensity_{max}$ is the saturation point of the FLIR camera (255 for an 8-bit grayscale image) and allows for a peak value of 1. Ideally, we expect intensity to increase linearly as voltage



(a) Calibration test intensity versus voltage plot.

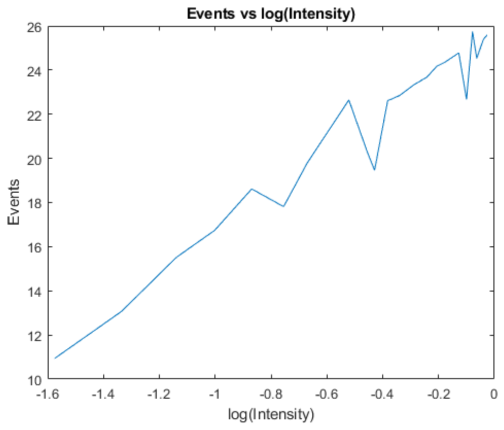


(b) Calibration test positive and negative events versus voltage plot.

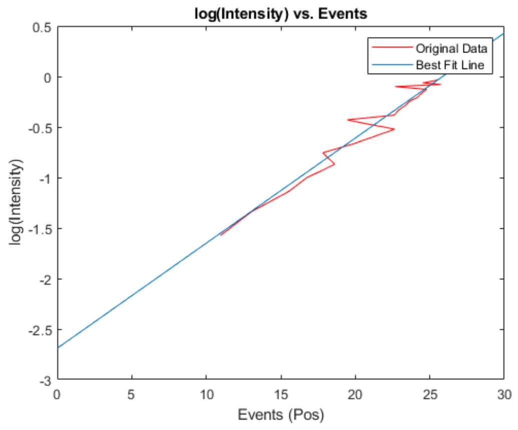
Figure 7.4: Initial analysis for event-based sensor calibration.

increases, which is relatively correct. The slight deviation can be attributed to the FLIR sensor hardware and small number of images captured at each brightness level. Note the first several intensity values are constant, indicating the noise floor has been hit. These values are disregarded moving forward.

Positive and negative events are also analyzed as a function of voltage (Figure 7.4b). Since event-based sensors operate on a log-intensity scale, we expect the events to follow the logarithmic scale. This is also relatively true, while disregarding the first several data point as with the *intensity* versus *voltage* plot.



(a) Plotting events versus log(intensity).



(b) Plotting log(intensity) versus events and finding a linear line of best fit.

Figure 7.5: Further analysis for event-based sensor calibration.

The initial data collection uses only the positive events for the subsequent preprocessing and tracking, so the positive events are exclusively used for the initial calibration as well. The events are next plotted against as a function of log(intensity). We expect to see a linear relationship between the events and log(intensity) and this holds relatively true (Figure 7.5a). Finally, the axes are inverted to display log(intensity) as a function of positive event count and a linear line of best fit is found. The linear relationship between log(intensity) and positive event magnitude is found to be,

$$log(intensity) = 0.1041e_m^i - 2.6904, \tag{7.2}$$

where e_m^i describes the magnitude of the ith inceptive event in the event stream. The linear line of best fit will allow us to convert the event magnitudes to log(intensity) and a subsequent intensity value. The full relationship is described by,

$$intensity = e^{(0.1041e_m^i - 2.6904)}. \tag{7.3}$$

CHAPTER VIII

RESULTS

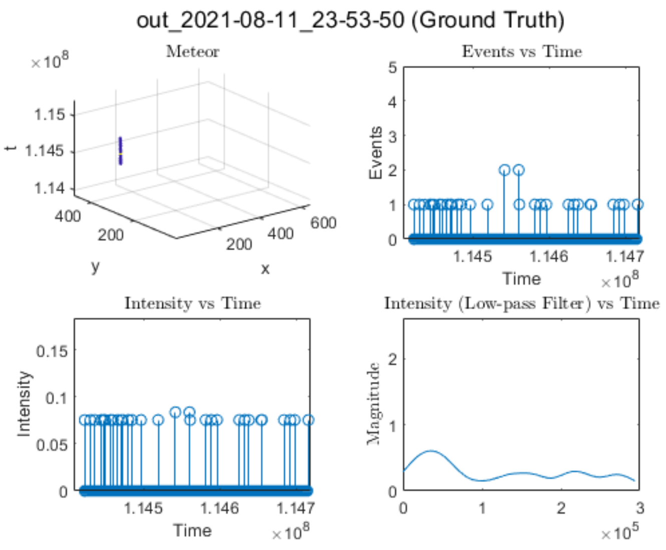
The raw event stream from the data collection is used as the input into the meteor detection and tracking algorithm. Several parameters are tuned to allow for optimized results, namely the grid density and minimum distance between pixels to allow an event to be added to a track. These parameters are described in Chapter VI and can be found for the VGA sensor in Table 8.1.

Table 8.1: Simple tracker parameters

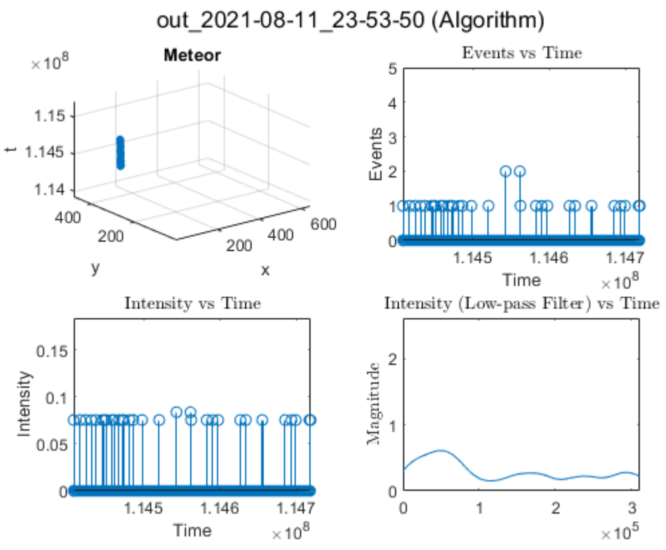
Grid Density	10 pixels
Pixel Threshold	4.2

The tracker parameters are empirically tuned to find the optimal detection and tracking accuracy for all VGA tracks found in the Perseids data collection.

Figure 8.1a displays an example of the ground truth tracks for a low intensity meteor. The top left plot displays the meteor tracks in the full three dimensional space. The sparsity can be observed in the top right plot, which displays the inceptive event magnitudes as a function of time. Note that for this particular meteor, the event magnitudes never exceed approximately 3. The bottom left plot displays the events transformed into intensity using Equation 7.3 in Chapter VII. Note the extremely low intensity – at a peak of approximately 0.1. Finally, the bottom right plot displays the intensity values with a low-pass filter to better reflect areas in which more events are active. As mentioned, the particular meteor is extremely sparse and remains at a relatively constant intensity throughout.



(a) Example of ground truth sparse meteor tracks.



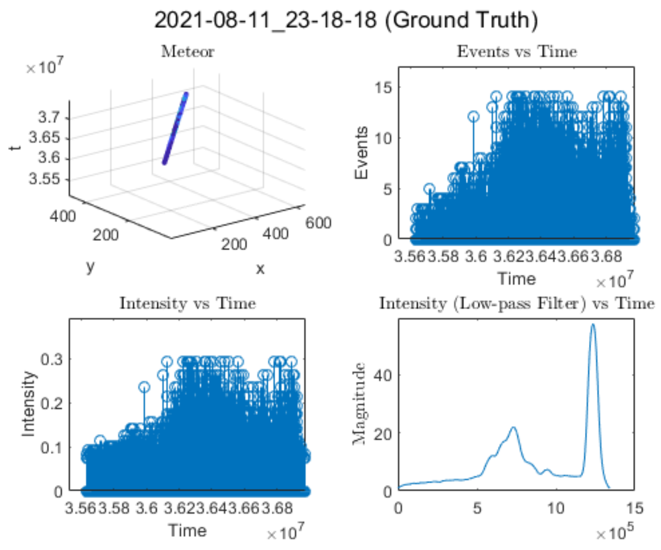
(b) Example of sparse meteor tracks as found by the simple tracker.

Figure 8.1: Example of sparse meteor tracks.

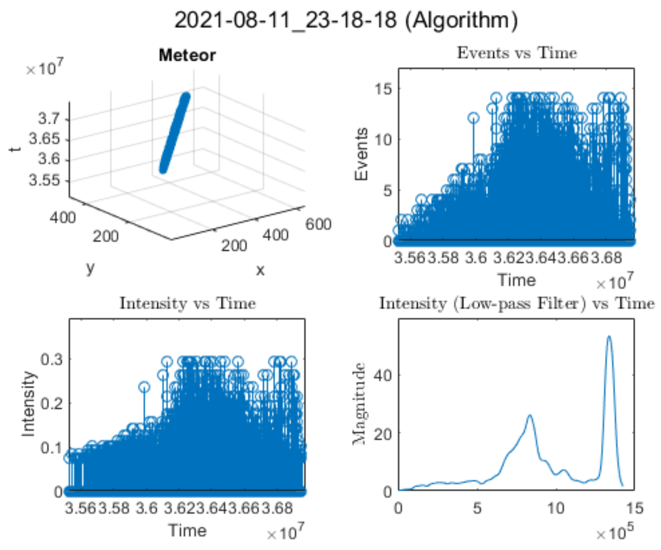
The sparse meteor is processed through the entire algorithm pipeline – starting with preprocessing (hot pixel and star suppression), then inceptive event filtering, followed by the meteor tracking and subsequent track callibration (Figure 8.1b). Upon inspection, the meteor appears to have been detected when compared to the ground truth data of Figure 8.1a. However, through the subsequent plots we observe that the tracking method may not have gathered all the events related to the ground truth meteor tracks. Upon analysis of the individual events, the simple tracker has detected 83.3% of the same events as the ground truth for this particular sparse meteor.

Due to the sparsity of this meteor, the intensity stays relatively consistent with the ground truth data in the subsequent callibration from the events to intensity. Note that due to the calibration setup for the VGA Prophesee sensor being focused on observing events in the 10 to 40 range, meteors with much higher levels of intensity will be fairly inaccurate in the intensity representation.

A set of dense meteor tracks are also shown (Figure 8.2). The simple tracker provided 90.57% of the same events depicted in the ground truth meteor tracks (Figure 8.2a). The densely populated meteor depicts peak event magnitudes of roughly 14 events, which corresponds to an intensity of approximately 0.3 when performing the calibration. Overall the simple tracker performed well on the densely populated meteor and depicts the same intensity trends in the test data as shown in the ground truth data.



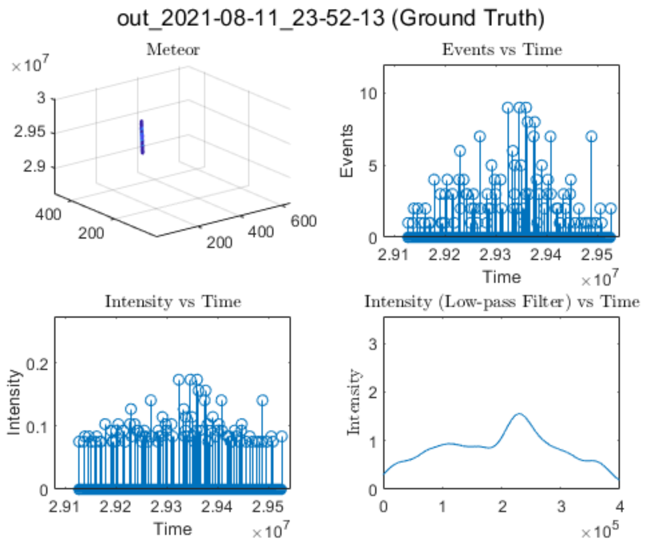
(a) Example of ground truth dense meteor tracks.



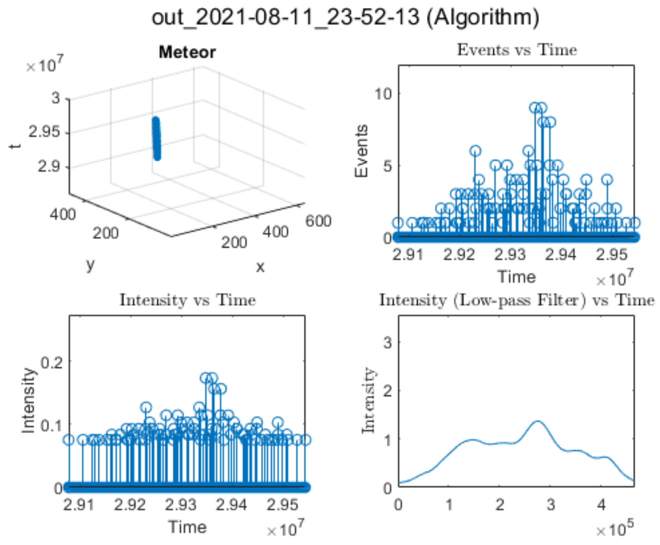
(b) Example of dense meteor tracks as found by the simple tracker.

Figure 8.2: Example of meteor tracks dense with events.

Only several meteor tracks are mentioned above for brevity, however additional examples of the meteors and their comparisons to the hand annotated ground truth data are available in Figures 8.3, 8.4, 8.5.

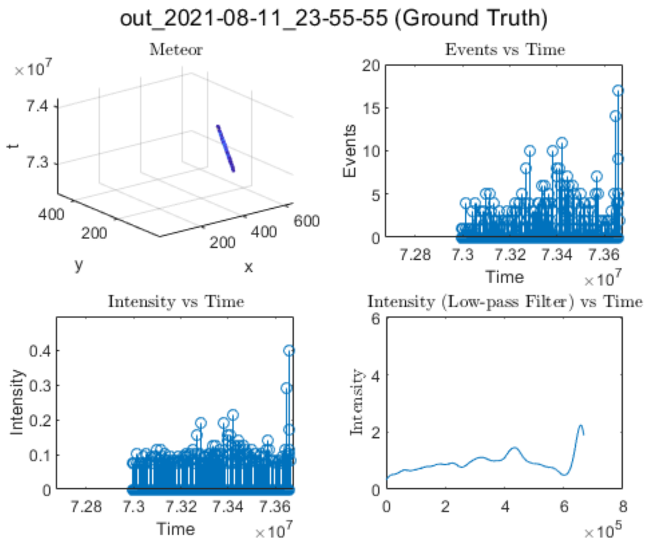


(a) Example of ground truth meteor tracks.

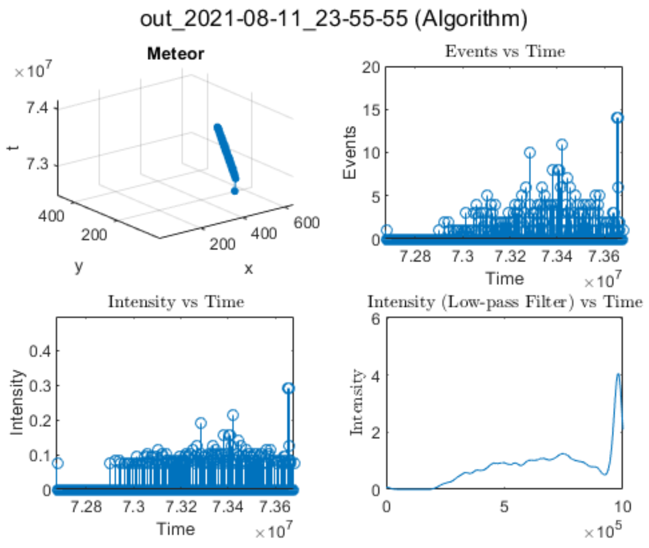


(b) Example of meteor tracks as found by the simple tracker.

Figure 8.3: Additional example of meteor tracks.

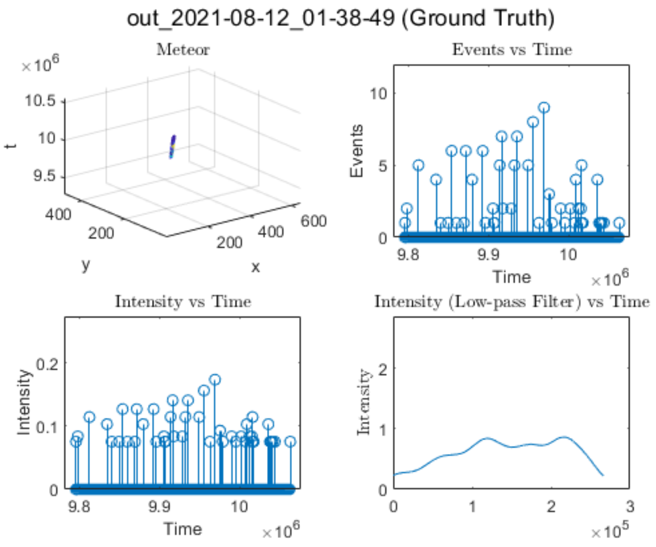


(a) Example of ground truth meteor tracks.

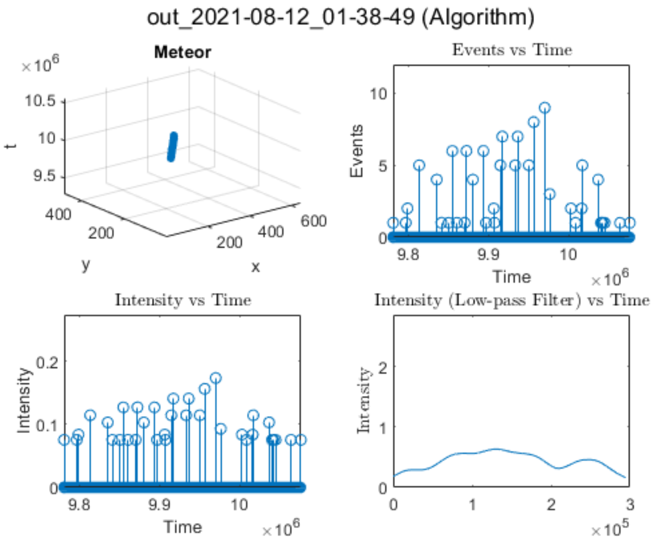


(b) Example of meteor tracks as found by the simple tracker.

Figure 8.4: Additional example of meteor tracks.



(a) Example of ground truth meteor tracks.



(b) Example of meteor tracks as found by the simple tracker.

Figure 8.5: Additional example of meteor tracks.

CHAPTER IX

CONCLUSION AND FUTURE WORK

The purpose of this thesis is to investigate the feasibility of using event-based sensors to detect and track short-lived, fast-moving, unresolved objects. Specifically, this problem is explored in the context of meteors. A novel data set is collected and manually annotated to provide real-world data for this experiment. The dataset consists of two separate meteor showers and utilizes the Prophesee VGA and Megapixel sensors with several different lenses to provide a variety of data. A data processing pipeline is proposed and developed that first denoises and filters the raw event data by means of hot pixel and star suppression, as well as inceptive event filtering to minimize data and increase computational efficiency. The resulting data is then processed by a simple tracking algorithm to provide meteor tracks for each event stream captured in the data collection. A subsequent calibration method is also developed to further explain the logrithmic relationship between event magnitudes and pixel intensity, and is applied to the event profile of the meteor tracks. This allows us to observe the meteor tracks in the context of intensity rather than events.

Overall the hot pixel and star suppression techniques remove nearly all hot pixels and stars from the event stream. The inceptive event filtering also provides an immense reduction in data by providing each inceptive event a corresponding magnitude to represent the number of trailing events and therefore a representation of intensity. The simple tracker provides a computationally efficient method of meteor tracking and with several additional thresholding techniques allows for both meteor detection and a single cohesive meteor track while disregarding any noise still present in the event stream. Overall the simple tracker reaches a peak accuracy of 90.57% of events captured in the meteor track compared to the ground truth meteor track. Additionally, the calibration technique to allow for conversion from the events to intensity provides an accurate transformation for positive event magnitudes ranging from approximately 11 to 25. Applying the calibration to the meteor tracks provides an intuitive understanding of the pixel intensity of the meteor and the corresponding brightness.

Unfortunately due to time constraints the point pixel analysis did not gain any additional attention. Moving forward, we plan to expand the calibration process to include and compensate for the edge competition observed within the hardware of the event-based sensor. This calibration process will be tested on the Geminid collection data, where the Prophesee Megapixel and VGA sensors were tuned to allow for relatively equal sensitivity between positive and negative events. This will compose of two separate calibration studies to obtain a transformation equation for both the Megapixel and VGA sensors. The relationship between positive and negative events will be further analyzed to gain an understanding of object movement – namely acceleration or deceleration of the meteors.

In conclusion, we have provided an algorithm pipeline to allow for the detection and tracking of fast-moving, short-lived, unresolved targets, specifically meteors, that could potentially be expanded to additional objects in the future.

BIBLIOGRAPHY

[1] K. Hirakawa and D. Migliore, "Introduction to event detection cameras." IEEE Computer Society, 2021.

[2] K. Xiao, P. Li, G. Wang, Z. Li, Y. Chen, Y. Xie, and Y. Fang, "A preliminary research on space situational awareness based on event cameras." 2022.

[3] Y. Hu, H. M. Chen, and T. Delbruck, "Slasher: Stadium racer car for event camera end-to-end learning autonomous driving experiments," in *2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 2019, pp. 29–33.

[4] P. Li, J. He, Y. Wu, and D. Zhou, "Design of monitoring system based on event camera." *2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA), Information Technology, Big Data and Artificial Intelligence (ICIBA), 2021 IEEE 2nd International Conference on*, vol. 2, pp. 519 – 522, 2021.

[5] R. W. Baldwin, M. Almatrafi, J. R. Kaufman, V. Asari, and K. Hirakawa, "Inceptive event time-surfaces for object classification using neuromorphic cameras." 2020.

[6] G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza, "Event-based vision: A survey." *IEEE Transactions on Pattern Analysis and Machine Intelligence, Pattern Analysis and Machine Intelligence, IEEE Transactions on, IEEE Trans. Pattern Anal. Mach. Intell*, vol. 44, no. 1, pp. 154 – 180, 2022.

[7] X. Lagorce, G. Orchard, F. Galluppi, B. Shi, and R. Benosman, "Hots: A hierarchy of event-based time-surfaces for pattern recognition." *IEEE Transactions on Pattern Analysis and Machine Intelligence, Pattern Analysis and Machine Intelligence, IEEE Transactions on, IEEE Trans. Pattern Anal. Mach. Intell*, vol. 39, no. 7, pp. 1346 – 1359, 2017.

[8] I. Alzugaray and M. Chli, "Asynchronous corner detection and tracking for event cameras in real time." *IEEE ROBOTICS AND AUTOMATION LETTERS*, vol. 3, no. 4, pp. 3177 – 3184, 2018.

[9] R. W. Baldwin, R. Liu, M. Almatrafi, V. Asari, and K. Hirakawa, "Time-ordered recent event (tore) volumes for event cameras." 2021.

[10] J. Kennewell and V. Ba-Ngu, "An overview of space situational awareness." *Proceedings of the 16th International Conference on Information Fusion, Information Fusion (FUSION), 2013 16th International Conference on*, pp. 1029 – 1036, 2013.

[11] P. N. McMahon-Crabtree and D. G. Monet, "Commercial-off-the-shelf event-based cameras for space surveillance applications." *APPLIED OPTICS*, vol. 60, no. 25, pp. G144 – G153, 2021.

[12] S. Afshar, A. P. Nicholson, A. van Schaik, and G. Cohen, “Event-based object detection and tracking for space situational awareness.” 2019.

[13] G. Cohen, S. Afshar, and A. van Schaik, “Approaches for Astrometry using Event-Based Sensors,” in *The Advanced Maui Optical and Space Surveillance Technologies Conference*, S. Ryan, Ed., Sep. 2018, p. 38.

[14] T.-J. Chin, S. Bagchi, A. Eriksson, and A. van Schaik, “Star tracking using an event camera.” *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Computer Vision and Pattern Recognition Workshops (CVPRW), 2019 IEEE/CVF Conference on, CVPRW*, pp. 1646 – 1655, 2019.

[15] M. Cui, Y. Zhu, Y. Liu, G. Chen, and K. Huang, “Dense depth-map estimation based on fusion of event camera and sparse lidar.” *IEEE Transactions on Instrumentation and Measurement, Instrumentation and Measurement, IEEE Transactions on, IEEE Trans. Instrum. Meas*, vol. 71, pp. 1 – 11, 2022.

[16] J. Rodriguez-Gomez, R. Tapia, A. G. Eguiluz, J. R. Martinez-de Dios, and A. Ollero, “Uav human teleoperation using event-based and frame-based cameras.” *2021 Aerial Robotic Systems Physically Interacting with the Environment (AIRPHARO), Aerial Robotic Systems Physically Interacting with the Environment (AIRPHARO), 2021*, pp. 1 – 5, 2021.