Event-based Detection, Tracking, and Recognition of Unresolved Moving Objects

Luc Tinch[1], Nitesh Menon[2], Keigo Hirakawa[1], Scott McCloskey[2]
[1] *University of Dayton*
[2] *Kitware, Inc.*

ABSTRACT

This paper explores the application of event-based sensing to the problem of wide field of view sky monitoring for unresolved, fast-moving optical targets. Event-based sensing captures, with high temporal precision, the changes in the target's brightness as it moves through the sky, and produces a much lower data volume than high-speed framing sensors. For our feasibility study, we consider the detection of meteors from a fixed ground-based camera. This paper describes the capture and curation of a dataset of event streams containing annotated meteors and distractor objects. We further describe a baseline software system to de-noise the event streams, track moving objects, and perform recognition to separate meteors from other moving objects. The performance of that system is evaluated against our real meteor data, on which our de-noising approach reduces more than 95% of the raw events (without removing events corresponding to real moving objects) and our lightweight recognition approach produces an accuracy of more than 85% on real data.

1. INTRODUCTION

Event-based sensors, which capture dynamic visual environments as asynchronous event streams, have recently attracted attention in the domain of space situational awareness (SSA). Relative to traditional framing sensors, event-based sensors have SSA-relevant advantages including increased dynamic range, lower latency, and lower power consumption due to the sparse nature of the orbital environment. Increased dynamic range has been shown [3] to enable monitoring of resident space objects (RSOs) during daytime and capture discriminating brightness variations against a dark background. In this paper, we emphasize the lower latency and higher temporal sampling of event streams, with the intention of detecting, tracking, and differentiating classes of unresolved, fast-moving, and short-lived objects. Specifically, we explore both the opportunities and challenges of event-based sensing in the context of meteor detection in a "wide field of view" setup, and present a dataset of real event streams captured during multiple recent meteor showers. We also introduce a baseline software pipeline to de-noise the resulting event streams, track moving objects, and perform recognition to discriminate meteors from other moving objects.

A key element of our software pipeline is a technique called inceptive event filtering partitions the raw data into inceptive events, trailing events and noise events[2]. When an event-based pixel sensor encounters an edge, it generates multiple events. Because multiple events occur in series and not simultaneously, the timestamp of the first event of this series—which we refer to as inceptive event—coincides with the actual edge arrival timing. The number of the subsequent events—referred to as the trailing events—is also useful because it is proportional to the log-intensity change stemming from the edge. However, the timestamps of the trailing events are ambiguous due to the latency. For this reason, the inceptive event filtering assigns the trailing event *count* value as the *magnitude* of the associated edge (i.e. contrast change). In this project, the inceptive event magnitude is interpreted as the log-intensity contrast of the meteor against the background sky. Finally the inceptive event filter labels events as noisy if it is an inceptive event with no corresponding trailing events. The rational is that low contrast edge whose magnitude is small is indistinguishable from a random occurrence of a positive or negative event. By requiring at least one trailing event, we are essentially thresholding by the edge contrast magnitude.

The remainder of this paper is organized as follows. We first review the basics of event-based sensing and its advantages compared to traditional framing sensors. Next, we review current approaches to distributed metor detection, to motivate the use of event cameras for this application. We describe in Section 3 attempts to simulate realistic event streams of meteors using a combination of night sky and event simulators. In section 4, we describe the real event data captured during the Perseid and Geminid meteor showers in 2021, and annotations of moving objects in that data. Section 5 describes a software pipeline to perform meteor detection on event streams, which is evaluated in section 6. The paper concludes with a summary and a discussion of future work.

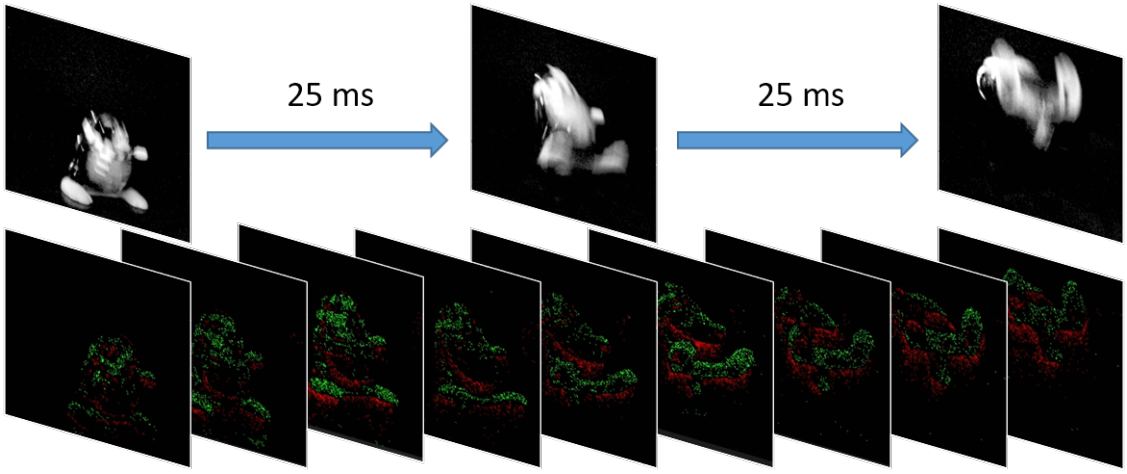


Fig. 1: A wind-up flipping toy captured by (Top Row) conventional active pixel sensor or APS and (Bottom row) dynamic vision sensor or DVS. Unlike APS that records a frame of pixel intensities (including blurred regions on moving targets and background areas with no motion), DVS asynchronously reports changes in log pixel intensities (green dots indicating increasing intensity, red dots decreasing) at temporal precision around 1μs. As a result, the spatial-temporal resolution of event-based cameras are far superior to APS.

2. EVENT-BASED SENSING AND RELATED WORK

The dynamic vision sensor (DVS) in event cameras is a type of so-called *silicon retina* loosely emulating the changes that take place in membrane potential of biological photoreceptors. The encoding of the log pixel intensity changes as "events" mimic the spiking neurotransmission within biological visual pathways. Instead of performing wasteful, repeated sampling of all points in its field of view at a fixed temporal resolution (typically 30Hz), event cameras asynchronously report spatio-temporal events when the intensity at a given pixel changes significantly. Each event is a tuple of (x, y, t, p) indicating the (x, y) location of the event, the time t at which it experienced a change, and a polarity sign p indicating whether the brightness increased or decreased. In that event cameras combine optical sensing with change detection, they are a type of neuromorphic sensor, and provide significant benefits with respect to power consumption, speed, and dynamic range. Event cameras have shown significant performance improvements across a range of applications, including highly-relevant work tracking distant and dim targets such as stars and satellites in low earth orbit [3].

Figure 1 illustrates the difference between standard intensity cameras and event cameras. Because a standard intensity camera repeatedly samples each pixel at a fixed frame rate, power is repeatedly expended by exposing, reading out, and digitizing parts of the field of view where nothing changes, and significant processing is required in software to identify changes which may indicate targets of interest. Event cameras, on the other hand, consume much less power in detecting where things are changing, i.e. a meteor. Like intensity cameras, event cameras provide spatial coordinates via pixel locations. Unlike intensity cameras, though, event cameras only provide information at that location when the irradiance on that part of the sensor has changed significantly. Besides the timing of when the change happened, the only other information provided by an event is its polarity, indicating whether the irradiance increased or decreased. Relative to traditional sensing approaches, event cameras offer three key benefits:

1. Relative to standard cameras, which fail to provide useful readings when small targets saturate, event cameras have **dramatically higher dynamic range** (140 dB vs. 60 dB of standard cameras). This will allow an event camera-based system to detect and track dim meteors with lower contrast against a wider range of backgrounds [4] than would be possible with a standard camera.

2. Inspired by biological vision, event cameras detect changes with **high temporal resolution, and have much lower latency** than standard cameras. Pixels report change events asynchronously and are timestamped to microsecond resolution [6], enabling accurate tracking of high-speed targets and providing more timely detection and more detailed phenomenology.

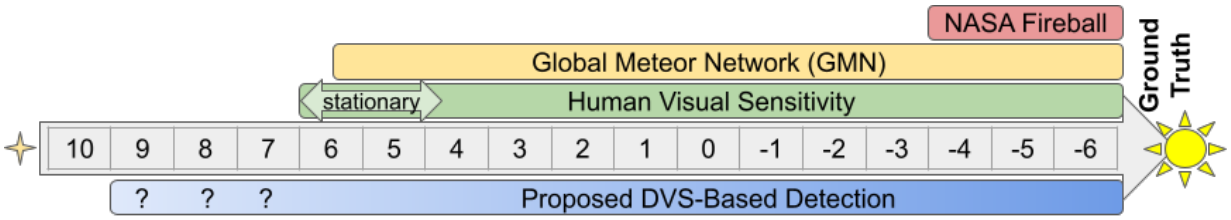


Fig. 2: Detection ranges of meteor detection approaches. The central axis shows stellar magnitude, which is a reverse logarithmic scale where smaller numbers indicate brighter targets. NASA's Fireball project only detects meteors brighter than a stellar magnitude of about -4, whereas GMN detects meteors with magnitude < 6, and human vision has a cutoff near 7 for stationary objects; human sensitivity to moving meteors would be less than the GMN range. We have not yet established the limiting magnitude of our DVS-based approach, but have anecdotally detected at least one meteor that wasn't seen by multiple human observers.

3. Standard intensity cameras are inefficient, wasting power by recording intensities of unchanging parts of their field of view. Event cameras enable **low-SWaP detectors** by consuming less power in sensing and dramatically sparsifying the data analyzed downstream. Even incorporating complex algorithms, embedded event-based gesture recognition systems consume only 200mW [1].

2.1 Distributed Meteor Detection

Because of the scientific questions they help answer about the composition and trajectory of the comets that leave them behind, the detection of meteors has motivated the development of multiple networks of sensors. The NASA Fireball Network (NFN) [1] has 17 specialized cameras distributed across the southern and eastern United States, each having a hemispheric field of view. The Global Meteor Network (GMN) [8] is a citizen science project enabling enthusiasts to purchase a hardware kit including a low-light Sony framing sensor, lens, camera enclosure, and Raspberry Pi processing board. Each node runs an open source detection algorithm and communicates with a centralized server to localize meteors across multiple observation sites. Relative to this prior work, our objectives are to assess the benefits of event-based sensing for meteor detection, which may have three facets. First, the event sensor's increased dynamic range may improve the detectability of dim meteors, as compared to the sensors used in NFN and GFN (see Fig. 2). Second, the increased temporal resolution may capture more detailed meteor phenomenology, providing insight into the composition, size, or other attributes of meteors. Finally, the reduced power consumption may allow for battery-operated detection nodes, allowing a proliferation of sensors.

3. SIMULATING EVENT RECORDINGS OF METEORS

Given that our unique combination of application (meteor detection) and sensor (event camera) hasn't been researched before, we explored the use of simulation to generate development, training, and testing data. Separate methods exist that generate framing video of simulated meteors and simulate event streams from framing video, the composition of which can produce simulated events of meteors.

To generate meteor videos, we use Stellarium [9], 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, the simulated meteors' temporal 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. 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.

From the Stellarium video, we simulated events using Video-2-Events [5]. The available implementation produces an event stream that reflects changes captured in the input video. Note that, because the input video consists of synchronous frames at 120 FPS, the temporal resolution information in the resulting event stream is temporally coarse.

Figure 3 shows example frames and events from this approach. Unfortunately the simulated data fails to reproduce this quality of the real-world data, which is shown in the next section. Specifically, the simulated event streams are

[1] https://fireballs.ndc.nasa.gov/

Fig. 3: (Left) Example output frame from Stellarium, showing a simulated fisheye view of the night sky with meteors. (Right) Visualization of events simulated from the Stellarium video using [5]. Gray areas show no change, white pixels reflect the presence of positive events, and black pixels reflect the presence of negative events.

much cleaner than real event streams, lacking events from hot pixels and generally having many fewer noise events. The simulated events corresponding to the meteors are much cleaner and smoother than actual meteor data, and seem not to reflect the temporal structure of real meteors. Overall, the simulated meteors do not accurately represent the meteors collected through real-world data collections, which motivated us to capture real meteor data.

4. DATA COLLECTION

4.1 Perseid and Geminid Meteor Showers

The Perseid meteor shower was selected for the high meteor counts per hour in order to maximize potential meteors caught within the data collection. 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.

The data collection for the Perseid meteor shower took place on 2021-08-11 at Caesar Creek State Park in Waynesville, OH. The data collection 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 the radiant of the meteor shower (roughly 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. 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 airplanes approached throughout the data collection and appear as distractors in the data set. Note that a second planned day of data collection was cancelled due to inclement weather conditions.

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). The main data collection provided over 5 hours of recorded data streams and 300 million events captured by the two Prophesee VGA cameras. Table 1 summaries the findings of the Perseid meteor shower data collection. Also note one particular meteor of interest with a distance of 193.74 pixels (37.44 degrees) and lasted 1.34 seconds (Figure 4). Note that a distractor (airplane) is present above the meteor.

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 Megapixel 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.

Table 1: Meteor shower data collection summary

	Perseid	**Geminid**
Meteors Captured	31	29
Instances of multiple meteors in same (2-3 minute) 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 $\frac{degrees}{second}$)	

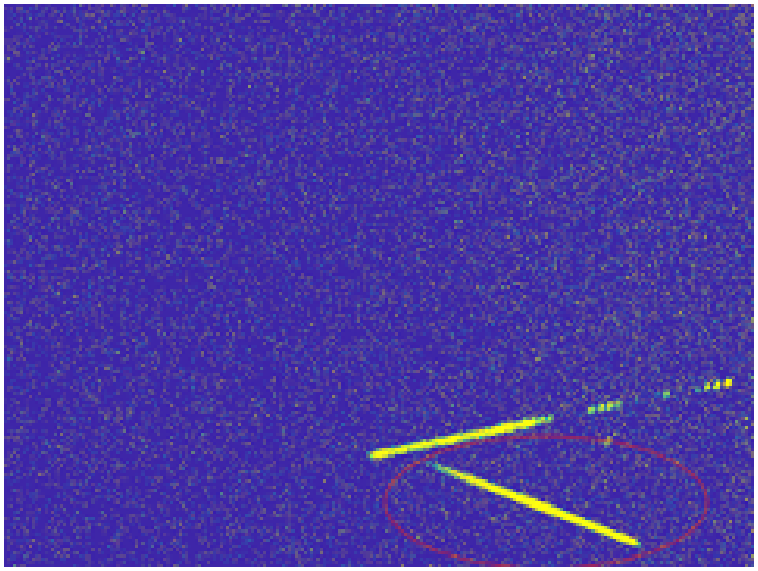


Fig. 4: Largest meteor captured by the Prophesee VGA event-based sensor

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 required 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 timestamps. Ground truth values are referred to in capital letters as,

$$E^{st} = (E_x^{st}, E_y^{st}, E_t^{st}), \tag{1}$$

$$E^{end} = (E_x^{end}, E_y^{end}, E_t^{end}), \tag{2}$$

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 5).

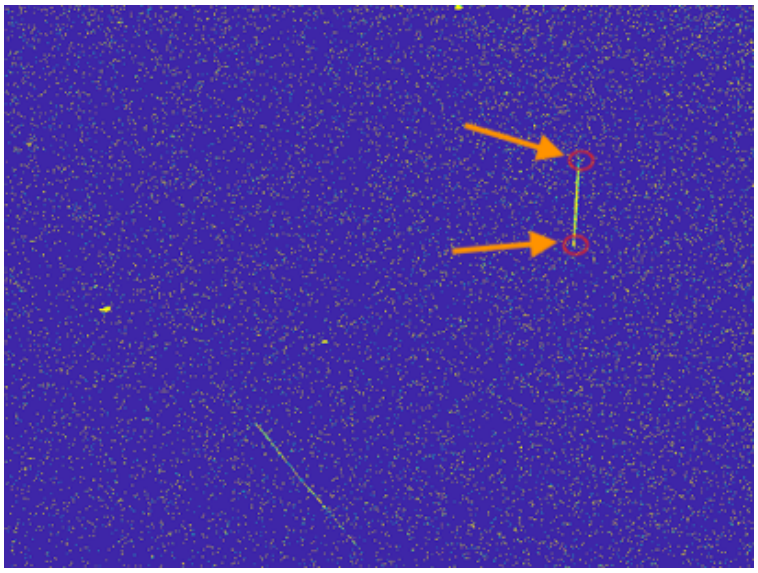


Fig. 5: Start and end events of the meteor are recorded.

4.3 Meteor Ground Truth Tracks

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. We fit a line to this data and all events are first temporally aligned and shifted relative to the meteor end event found in Section 4.2 following,

$$e^i_{t\Delta} = E^{end}_t - e^i_t, \tag{3}$$

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,

$$e^i_\Delta = (e^i_x, e^i_y, e^i_{t\Delta}) \tag{4}$$

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,

$$\zeta^i = cross(e^i_\Delta, E^{st}_\Delta) = (\zeta^i_x, \zeta^i_y, \zeta^i_z) \tag{5}$$

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,

$$error = \frac{N(\zeta^i)}{N(E^{st}_\Delta)}, \tag{6}$$

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,

$$N(e^i) = \sqrt{\sum_{k=1}^{3} |e^i_k|^2}, \tag{7}$$

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,

$$\begin{cases} e^i \in \Gamma_j, & \text{if } error < \tau_{error} \\ e^i \notin \Gamma_j, & \text{if } otherwise \end{cases}, \tag{8}$$

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 dataset. An example of the ground truth meteor tracks can be seen in Figure 6.

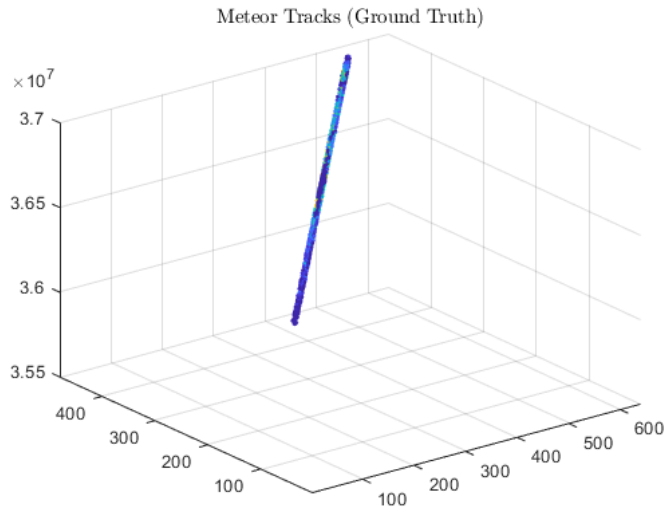


Fig. 6: Example ground truth for meteor tracks.

5. EVENT-BASED METEOR DETECTION APPROACH

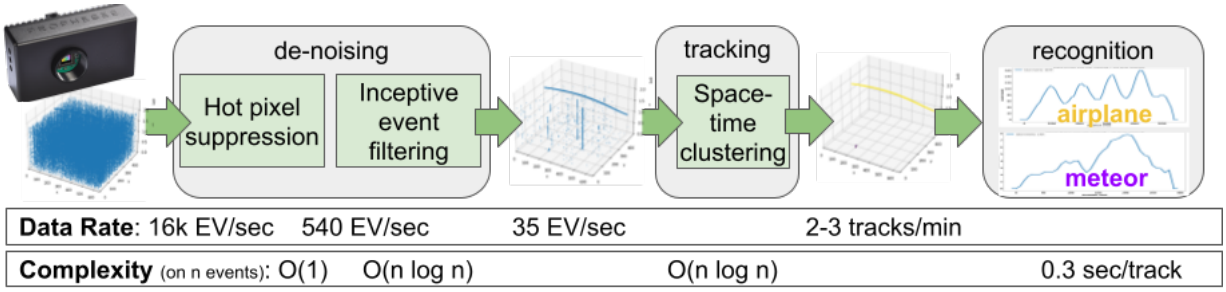


Fig. 7: Event-based detection approach, illustrated on a 2 minute event sequence including both an airplane (yellow) and a meteor (purple). Example data rates after each step are presented, along with the computational complexity of each processing step.

As illustrated in Fig. 7, our event-based meteor detection approach starts from raw events of the night sky, captured with high sensitivity. This results in a large density of noise events that don't relate to meteors, stars, airplanes, or any other real-world object. We employ multiple de-noising approaches to reduce the volume of noise events without

discarding events from real targets. Within the thinned spatio-temporal volume, we combine individual events into tracks corresponding to moving objects, and also filter out stationary stars[2]. The remaining tracks, corresponding to real moving objects, are then analyzed with respect to their temporal appearance, allowing us to separate meteors from non-meteors (primarily airplanes in our dataset). Each of these steps are detailed in the sections that follow.

5.1 Hot Pixel and Star Suppression

A method of hot pixel and star suppression are developed and tested using the meteor data collection. Additionally, the hot pixel and star suppression method relies 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,

$$C_k(x,y) = \sum_{i=1} \delta(e_x^i - x)\delta(e_y^i - y), \tag{9}$$

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,

$$M^k(x,y) = \delta(C_k(x,y) > \tau_C), \tag{10}$$

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

$$\tau_C = \frac{\lambda max(e_t^i)}{1e6} \tag{11}$$

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 10 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 suppression 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 can be summarized by the following equation,

$$M_{hot}(x,y) = \delta(M_1(x,y))\delta(M_2(x,y)), \tag{12}$$

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

$$M_{star}(x,y) = \delta(M_1(x,y))\delta(1 - M_{hot}(x,y)), \tag{13}$$

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 M_1 (hot pixel and star mask) and results in exclusively the star locations.

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

$$\{e^i \notin \varepsilon^j \mid \delta(M_{hot}(e_x^i, e_y^i))\delta(M_{star}(e_x^i, e_y^i)) = 1\}, \tag{14}$$

[2]Stars generate events in a stationary camera primarily due to their twinkle in the mid-summer humidity.

where e^i is the ith event in the event stream and ε^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, slightly increasing the computational complexity of the algorithm and subsequent time needed for execution.

5.2 Inceptive Event Filtering

An edge encounter by a pixel (such as meteor intersecting a pixel) generates several events. Let $e^i = (e^i_x, e^i_y, e^i_p, e^i_t)$ denote an ith event with x-y coordinate (e^i_x, e^i_y), polarity e^i_p, and the timestamp e^i_t. Inceptive event filtering organizes the events in a manner that reflects the physical phenomenon using heuristics described below[2]. The first of the series of events generated by an edge encounter is referred to as an inceptive event, and its timestamp corresponds to the exact timing of the edge arrival. An event e^i is classified as an inceptive event if following conditions are met:

$$IE = \left\{ e^i \middle| \nexists j < i \ni e^i_x = e^j_x, e^i_y = e^j_y, e^i_p = e^j_p, e^i_t - e^j_t < \kappa \right\} \wedge \left\{ e^i \middle| \exists k > i \ni e^i_x = e^j_x, e^i_y = e^j_y, e^i_p = e^j_p, e^k_t - e^i_t < \kappa \right\}, \tag{15}$$

where $\kappa > 0$ is some predefined threshold. Intuitively, an inceptive event e^i is the first event in at least κ seconds that has at least one event e^k occurring within κ seconds (trailing event, defined below). Similarly, trailing events are those satisfying the following conditions:

$$TE = \left\{ e^k \middle| \exists i < k \ni e^i_x = e^j_x, e^i_y = e^j_y, e^i_p = e^j_p, e^k_t - e^i_t < \kappa \right\}, \tag{16}$$

that is, there exists an event e^i preceding e^k within κ seconds. Because inceptive events and trailing events occur in succession, there is an ambiguity to the timing of the trailing event relative to the edge arrival.

The remaining events are classified as noisy events:

$$NE = \left\{ e^i \middle| \nexists j < i \ni e^i_x = e^j_x, e^i_y = e^j_y, e^i_p = e^j_p, e^i_t - e^j_t < \kappa \right\} \bigcap \left\{ e^i \middle| \nexists k > i \ni e^i_x = e^j_x, e^i_y = e^j_y, e^i_p = e^j_p, e^k_t - e^i_t < \kappa \right\}. \tag{17}$$

Intuitively, noisy events are events that occur with no other events within temporal window $(-\kappa, \kappa)$.

The *contrast* scales proportionally to the number of trailing events generated. We therefore assign the number of trailing events as *edge magnitude* e^i_m of an inceptive event e^i, as follows:

$$e^i_m = 1 + \#\left\{ \left\{ e^j \in TE \middle| e^i_x = e^j_x, e^i_y = e^j_y, e^i_p = e^j_p, e^i_t < e^j_t \right\} \bigcap \left\{ e^j \in TE \middle| \nexists e^k \in IE \ni e^i_x = e^k_x, e^i_y = e^k_y, e^i_p = e^k_p, e^i_t < e^k_t < e^j_t \right\} \right\}. \tag{18}$$

Intuitively, the trailing events (e^j) after the inceptive event (e^i) that proceed any other inceptive events (e^k) are counted. The *add one* in (18) is to count the inceptive event itself as part of the edge magnitude.

In this project, the positive inceptive event denotes the timing of the meteor arrival, followed by the negative inceptive event to mark the meteor departure. The edge magnitude of the positive inceptive event represents the contrast level of the meteor relative to the background/environmental scatter; and the timestamp differences between the successive positive and negative inceptive events represents the meteor's *dwell time* at the pixel. Owing to the fact that only inceptive events and their edge magnitudes are retained, inceptive event also serves as a data thinning technique that reduces the data volume by considerable amount while preserving salient characteristics of the meteor.

5.3 Tracking

The events form a track to reconstruct the meteor path. The asynchronous events in the event-based sensors continuously update the meteor location, unlike the conventional framing camera where the the movement of the meteor is lost between the frames. Tracking meteors with events is not without challenges, however. Owing to the uncertainty in the timestamps of the trailing event, plotting the raw events yield ambiguous tracks. Tracking positive inceptive events exclusively improves accuracy by resolving the timing.

Thanks to the asynchronous nature of the event stream, a very simple "spatial-temporal nearest neighbor" meteor tracking strategy proved effective. Let $\{T_1, \ldots, T_N\}$ denote existing tracks, with their last events

$$\tilde{e}^n := \arg\max_{e \in T_n} e_t. \tag{19}$$

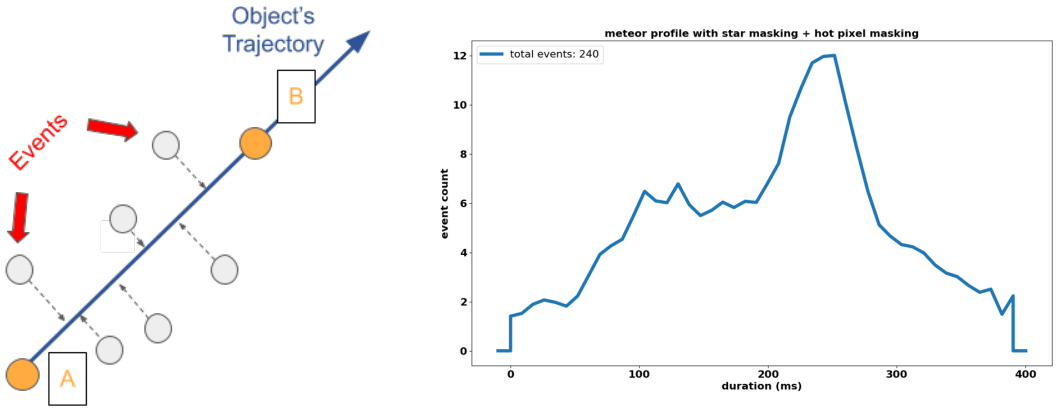


Fig. 8: (Left) Given a trajectory with associated events, we could the number of positive events (gray circles) between the start and endpoints. This results in an event-time profile (right), illustrated for one of the annotated meteors.

When a new inceptive event e^i is detected, it is compared to the last event of each existing tracks, as follows:

$$d^2 = \min_n (e^i_x - \tilde{e}^n_x)^2 + (e^i_y - \tilde{e}^n_y)^2 + \gamma(e^i_t - \tilde{e}^n_t)^2. \tag{20}$$

If $d^2 > \nu$ for some predefined Lagrangian γ and threshold ν, then this new event is far from $\{\tilde{e}^1_x, \ldots, \tilde{e}^N_x\}$, and so a new track ($T_{N+1}$) is formed. On the other hand, if $d^2 < \nu$, then e^i is assigned as the last event of the nearest track.

While such simplistic nearest-neighbor tracking approach would typically fail for unresolved fast targets in the frame-based cameras (because $d^2 > \nu$ even if the new event e^i belongs to an existing track), event-based tracking keeps up the pace with fast moving targets such as meteors.

5.4 Recognition

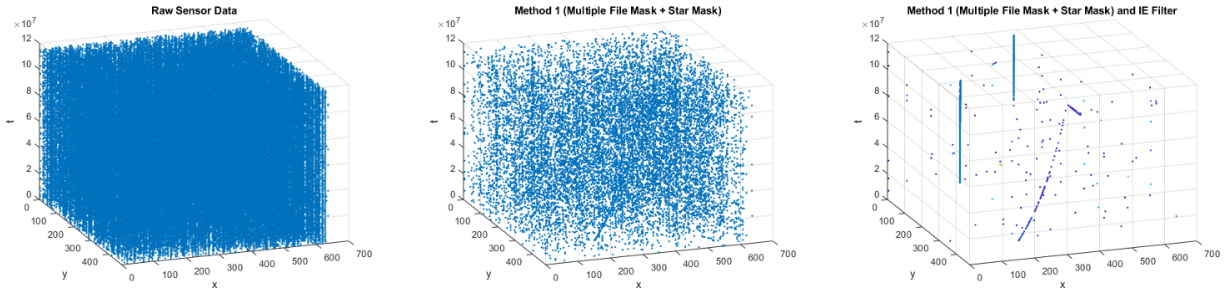
Because the moving objects in our dataset are primarily airplanes and meteors, recognition of meteors from the event trajectories is not as difficult as it would be in our envisioned application. Specifically, the meteors and aircraft tracks in our dataset can be perfectly separated into the two classes by simply applying a threshold to the duration of the trajectory (aircraft tracks were all longer than 10 seconds, and all meteor tracks were shorter than this). In order to be more robust to real-world conditions, e.g. a airplane track becomes occluded after a short time, avoiding false alarms requires us to develop a more robust meteor recognition approach. To that end, we normalized the airplane and meteor track durations to understand how discriminative the temporal brightness profiles are.

From a track of events, computed as described in the previous subsection, we generate a temporal profile of the moving object's brightness by counting the number of positive events that it generates as it moves across the sensor. This process is illustrated in Fig. 8. For representation and recognition, we used the TimeSeriesSVC model from the tslearn [7] package, along with a gak kernel. Each event time profile was represented as a time series (normalized to eliminate the duration), and a support vector machine (SVM) was fit in a two class (airplane vs. meteor) classifier. Because our dataset over-represents meteors (having been collected during the peak of the largest meteor shower of the year), we chose to use all 20 airplane tracks and 21 randomly-selected meteor tracks from the Perseid meteor shower.

6. EXPERIMENTS

6.1 Data Volume Reduction

A major motivation for the initial filtering steps outlined in Section 5.1 and Section 5.2 are to reduce the volume of data for the tracking and recognition algorithms. The raw data stream contains approximately 16,000 events per second (Figure 9a). Applying the hot pixel and star suppression masks reduces the data stream to roughly 540 events per second, a 96.8% reduction (Figure 9b). Applying the inceptive event filtering further reduces the event stream to 35 events per second, for an overall 99.7% reduction in events (Figure 9c). The application of the initial filtering techniques proves to be both crucial and effective in the reduction of data while retaining the inceptive events and corresponding magnitudes needed to track meteors and distractors.



(a) Raw data from the Prophesee VGA event-based sensor.

(b) Raw data after hot pixel and star filtering.

(c) Raw data and subsequent hot pixel/star filtering and inceptive event filtering.

Fig. 9: Illustration of data reduction from various filtering algorithms

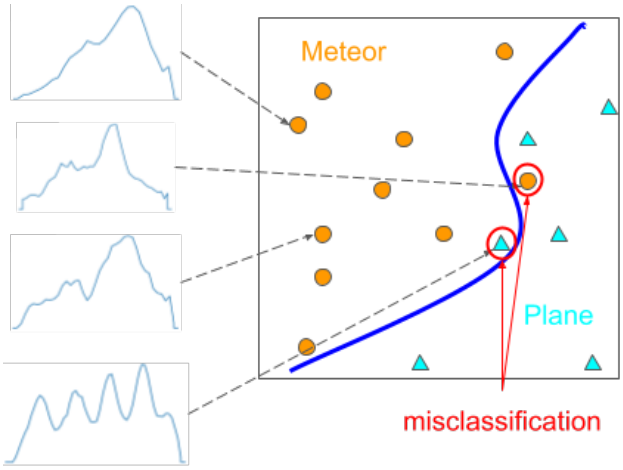


Fig. 10: Real event-time profiles (thin blue lines on left), showing two correct meteor detections and one of each type of error: a false positive meteor (bottom-most profile) and false negative meteor (second from the top). A cartoon illustration of the SVM decision boundary is shown as a bold blue curve.

6.2 Recognition Performance

Figure 10 shows an illustration of the SVM's feature space and decision boundary. We evaluated classification performance using leave-one-out cross-validation. Because of the slight class imbalance (21 meteors and 20 airplanes), a random classifier would produce a 51.2% accuracy. By contrast, our timeseries SVM, ignoring the track duration, achieved 85.4% accuracy. As can be seen from the illustration, the airplane track mis-classified as a meteor shows a similar event-time profile, generating more events over time as the airplane gets closer to the camera. Both meteors and airplanes have local minima and maxima in their event time profiles, but the airplane profiles are more periodic due to the flashing of marker lights.

7. CONCLUSIONS AND FUTURE WORK

In this work, we described an initial feasibility analysis of using event sensors to monitor the night sky for meteors. We investigated the use of synthetic data, and found that the domain gap between it and real event streams of the night sky was very large. We collected an initial dataset of real event streams, within which we annotated meteors and airplanes. Airplanes are used as a negative class for recognition because airplanes and meteors are confused in other meteor detection approaches [8]. We described a software pipeline to de-noise event streams taken with high sensitivity, an approach to group events into spatio-temporal trajectories, and experiments with a support vector machine to recognize which trajectories correspond to meteors. While the sample size is small, our experiments demonstrate that our de-noising approach reduces more than 95% of the raw events without impeding downstream recognition. Recognition

performance on fixed-length trajectory sub-sequences has greater than 85% accuracy with a lightweight classifier.

We believe this demonstrates the feasibility of using event-based sensing in the place of traditional framing sensors for night sky monitoring and meteor detection. In order to more fully quantify the potential benefits to meteor detection networks, subsequent work must:

1. Quantify the apparent magnitude range of event-based meteor detection. While we have anecdotal evidence that out event camera captured a meteor that was undetected visually, the exact detection range is unknown.

2. Quantify the utility of the additional temporal information captured by an event camera, as compared to a framing sensor, in terms of its ability to capture scientifically meaningful phenomenological information.

3. Quantify the power reduction achieved by replacing a traditional framing sensor for an event sensor.

REFERENCES

[1] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza, J. Kusnitz, M. Debole, S. Esser, T. Delbruck, M. Flickner, and D. Modha. A low power, fully event-based gesture recognition system. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[2] R Wes Baldwin, Mohammed Almatrafi, Jason R Kaufman, Vijayan Asari, and Keigo Hirakawa. Inceptive event time-surfaces for object classification using neuromorphic cameras. In *Int'l Conf. on Image Analysis & Recognition*, 2019.

[3] Gregory Cohen, Saeed Afshar, André van Schaik, Andrew Wabnitz, Travis Bessell, Mark Rutten, and Brittany Morreale. Event-based sensing for space situational awareness. In *Advanced Maui Optical and Space Surveillance Conf.*, 2017.

[4] A. Glover and C. Bartolozzi. Event-driven ball detection and gaze fixation in clutter. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2203–2208, 2016.

[5] Y Hu, S C Liu, and T Delbruck. v2e: From video frames to realistic DVS event camera streams. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021.

[6] P. Lichtsteiner, C. Posch, and T. Delbruck. A 128$\times$ 128 120 db 15 μs latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-State Circuits*, 43(2):566–576, 2008.

[7] Romain Tavenard, Johann Faouzi, Gilles Vandewiele, Felix Divo, Guillaume Androz, Chester Holtz, Marie Payne, Roman Yurchak, Marc Rußwurm, Kushal Kolar, and Eli Woods. Tslearn, a machine learning toolkit for time series data. *Journal of Machine Learning Research*, 21(118):1–6, 2020.

[8] Denis Vida, Damir Šegon, Peter S Gural, Peter G Brown, Mark J M McIntyre, Tammo Jan Dijkema, Lovro Pavletić, Patrik Kukić, Michael J Mazur, Peter Eschman, Paul Roggemans, Aleksandar Merlak, and Dario Zubović. The Global Meteor Network – Methodology and first results. *Monthly Notices of the Royal Astronomical Society*, 506(4):5046–5074, 08 2021.

[9] Georg Zotti, Susanne M. Hoffmann, Alexander Wolf, Fabien Chéreau, and Guillaume Chéreau. The simulated sky: Stellarium for cultural astronomy research. *Journal of Skyscape Archaeology*, 6(2):221–258, Mar. 2021.