SSC24-XI-04

Neuromorphic Star Tracking using Monte Carlo Localization

Connor Gisburne, Alex Warren, Steven Allen, Michael Bantel, Justin James
ExoAnalytic Solutions, Inc.
27042 Towne Centre Dr, Suite 250, Foothill Ranch, CA 92610; +1 (636) 667-0112
cgisburne@exoanalytic.com

ABSTRACT

Accurate attitude determination ensures precise payload sensor alignment and optimal solar panel positioning, enhancing data collection and power efficiency while ensuring stable orbit maintenance. The most accurate method of attitude determination is star tracking. Star trackers collect an image of a star field and utilize a catalog of known star positions to calculate the orientation of the system relative to the stars. The errors present in this method of attitude determination tend to be on the order of ten arcseconds or less. Using traditional collection techniques, attitude estimates can be provided at update rates of 1 to 10 Hz and angular velocities are typically limited to a few degrees per second.

Neuromorphic cameras, also known as event cameras, detect changes in the input visible signal on a per pixel basis, enabling microsecond temporal resolution and significantly lower data volume. This collection method is also capable of operating with a significantly reduced power usage (<~0.1W) and mass (<50g) compared to traditional high-speed cameras. These characteristics make neuromorphic cameras ideal candidates for autonomous navigation of small satellites that commonly have stricter size, weight, and power requirements. The application of neuromorphic cameras to autonomous navigation and localization is an active area of research. Many techniques leverage this unique dataset, such as Monte Carlo Localization (MCL).

MCL is a probabilistic technique used in robotics and autonomous navigation. It employs random sampling to estimate a system's position and orientation based on a known map and sensor data. Particle filters implement MCL by creating a dynamic model of potential states through a series of particles, which are iteratively updated and reweighted to align with sensor observations and refine the system's state estimate.

ExoAnalytic Solutions utilizes a particle filter to process event camera measurements generated by the stars, leveraging MCL to determine the attitude of the satellite. A star catalog is used as the known map that the particle filter compares measurements to. This process streamlines star identification and makes it continuous which saves time compared to traditional approaches like pattern matching algorithms. The high update rate of the neuromorphic star tracker can aid autonomous navigation of small satellites by providing improved stabilization, maneuver/station-keeping efficiency, and maneuverability. In the increasingly crowded space environment, highly accurate and responsive attitude determination technology is critical to ensuring a safe space environment for all operators. This paper presents the results of a MCL particle filter applied to simulated measurements from a neuromorphic star tracker.

BACKGROUND

Event Cameras

Event cameras, also called neuromorphic cameras, are a recent innovation in bio-inspired technology. Put simply, these cameras detect only the changes in intensity on the sensor, similar to how biological retinas respond to visual stimuli. Specialized sensor circuitry implemented on a per-pixel level gives event cameras their uniquely high dynamic range, low temporal resolution, and low power consumption. Work completed by the University of Zurich, Sony Electronics, and other contributors has transformed a technical novelty into a powerful imaging technology with multiple applications across computer vision. For a full description of the internal workings of event camera circuitry, see [1], [2], or [3].

In addition to computer vision technology, event cameras have been applied to space technologies, most notably for space situational awareness (SSA). The work done by Astrosite in Australia has pioneered this application [4]. In addition, the affiliated Western Sydney University International Centre for Neuromorphic Systems (ICNS) has sent an event camera to the International Space Station (ISS) with interesting results [5]. These efforts are promising, but they have merely scratched the surface of what is possible with

event cameras for space applications. Any dynamic imaging problem can be effectively solved by an event camera. In particular, star tracking shows promise as a space where event cameras can contribute effectively to space technologies.

Star Tracking

Star trackers are attitude determination devices that typically use a camera or other imaging device to locate the stars and then compare these detected stars to a known star catalog to find the space vehicle's orientation. Typical star trackers do this by collecting an image, processing it to find the star positions, identifying these stars with an identification algorithm, such as TETRA [6], then solving Wahba's Problem to find the optimal rotation between the two coordinate frames. This is the attitude used for any other attitude control tasks. Typically, star trackers are paired with other relative attitude measurement systems, such as gyroscopes. Together these sensors provide a complete picture of the attitude dynamics of the space vehicle. An example of a star tracker for a small satellite is shown in Figure 1Figure 1.

Figure 1: arcsec's Sagitta Star Tracker [7]

A simplified understanding of star tracking is that a star tracker is comparing the measured star positions to a set of known possible star positions. This broader view opens other potential methods for solving the celestial navigation problem, such as Monte Carlo Localization.

Monte Carlo Localization

Robotics and autonomous systems require accurate positioning and pose information to successfully complete many tasks. That can be as simple as basic obstacle avoidance or complex simultaneous localization and mapping (SLAM) of unknown scenes. MCL is an approach to estimating a robot or other autonomous system's position and pose in a known environment. The system takes measurements and compares them to a known map of the environment. The simplest example is a robot navigating a series of rooms while having some distance measuring sensor to determine the distance to walls. By providing these measurements over time while moving through the room, the robot can refine its estimate of its position and pose relative to the known map. This example is discussed and shown in Figures 2-5. MCL uses a particle filter, a Markov Chain Monte Carlo (MCMC) estimation method, to estimate the position and pose of the system. Particle filters are well known state estimation techniques that expand upon classical filtering methods, such as Kalman filters, by approximating the state's probability density function with a discrete set of particles, shown in Equation 1.

$$p(x) \approx \sum_{i=1}^{N_s} w^i (x - x^i) \quad (1)$$

The steps in an MCL algorithm are as follows:

Initialization:

Figure 2 shows a notional robotic system in a room with an initial particle distribution for an MCL algorithm. The particles are initially drawn from a uniform distribution across all possible positions in the room. This is because, with no a priori information, a uniform distribution is the most reasonable assumption. In more complicated problems or scenarios where additional information is available, the initial particle distribution can be modified to reflect the most representative prior.

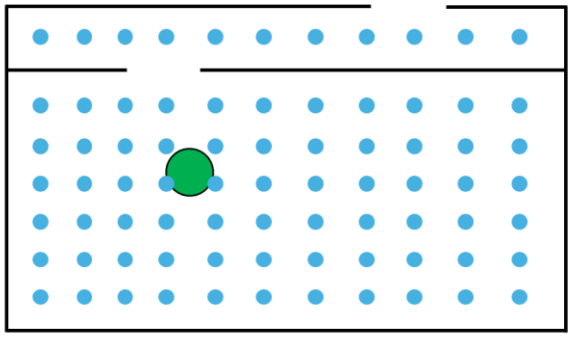

Figure 2: Initial Uniform Particle Distribution, Large Dot Represents Truth Position

Predict:

Next the model predicts forward in time based on the assumed system dynamics. Each particle has a state that is propagated forward to the next update time. This is shown in Equation 2.

$$\boldsymbol{x}_k = f_k(\boldsymbol{x}_{k-1}) + \boldsymbol{v}_{k-1} \quad (2)$$

f_k represents whatever system dynamics best models the evolution of the system over time. v_{k-1} is the assumed process noise to account for the inaccuracies present in the system dynamics model. The method of propagation depends on the system dynamics modeled and the associated assumptions. The toy problem illustrated in Figure 2 and Figure 3 assumes a constant position for ease of visualization.

Measure and Update:

Once the expected state of the particles is determined by prediction, a measurement is obtained from the system's sensors.

$$\boldsymbol{z}_k = \boldsymbol{h}_k(\boldsymbol{x}_k) + \boldsymbol{n}_k \quad (3)$$

z_k represents the measurement received, h_k is the measurement function, and n_k is the measurement noise. The selection and design of the measurement function h_k is critical for the creation of high-quality filtering algorithms. Further discussion of measurement functions is found in the Approach section of this paper. Figure 3 illustrates a measurement from the toy robot localization problem. The robotic system measures the distance to the walls. Since MCL assumes that the map of the environment is known, in this case that the layout of the walls is known, this set of distance measurements can be simply compared to the distance measurements that would be expected to be measured by the system if it was located at the particle's position.

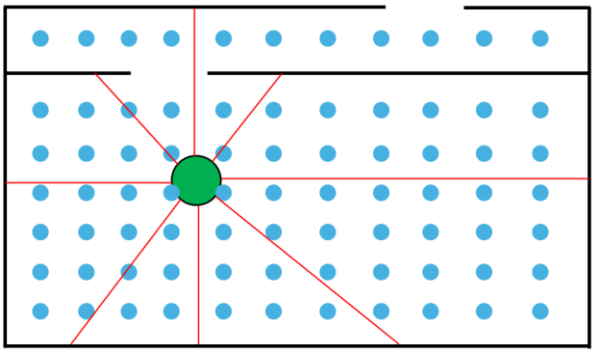

Figure 3: System Takes a Distance Measurement

This comparison is mapped to a probability distribution that gives the likelihood a particle state is the truth state. This allows the MCL algorithm to update the weights. This new weight is proportional to the probability calculated in the measurement function.

$$w_k^i \propto w_{k-1}^i p\left(\boldsymbol{z}_k \middle| \boldsymbol{x}_{k-1}^i\right) \quad (4)$$

The exact function which defines the proportionality relationship in Equation 4 depends on the measurement, sensors, and desired filter behavior. An illustration of this reweighting process is shown in Figure 4.

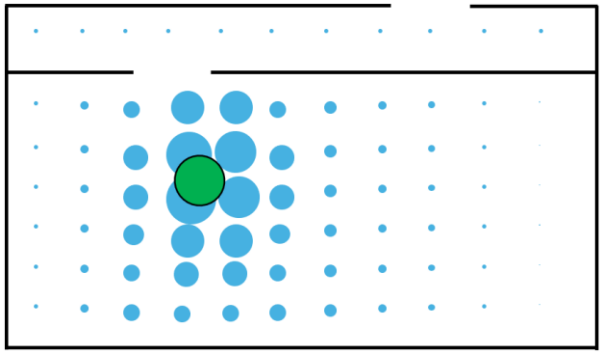

Figure 4: Measurement Function Updates Particle Weights based on Likelihood of Received Measurement

Resampling:

Particle filters often have issues where the weights can overly weigh a small set of particles, leading to a collapse of performance. This has led to resampling algorithms that periodically determine the current distribution of particles and resample throughout that distribution. An illustration of the product of this resampling process is shown in Figure 5.

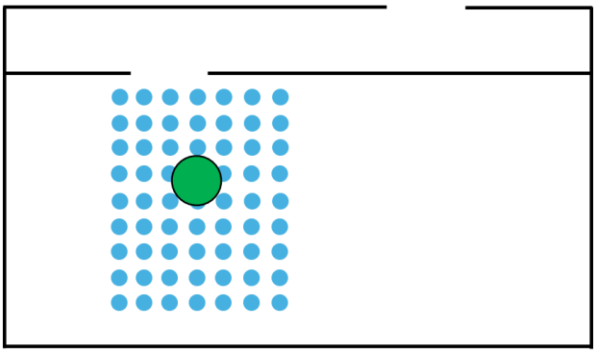

Figure 5: Particle Distribution After Resampling

Outputs:

One of the advantages of the particle filter is its ability to model nonlinear, non-Gaussian processes and distributions. This also makes representing the output state more difficult, as the mean of a bimodal or other complex distribution is not as meaningful as that of a Gaussian distribution. For the comparisons shown in the results, the weighted mean of the particle states and the top ten highest weighted particles will be used. Other

more complex methods are available, but this will generally be acceptable for converged particle filters.

For further in-depth information on particle filters and their implementation, see [8].

APPROACH

To develop and test the neuromorphic star tracking MCL algorithm, an in-house event camera simulation is used to generate representative event camera data of the stars. The section below describes the assumptions, simulation, and parameters used to determine the performance of the neuromorphic star tracker.

Simulation

An in-house event camera simulation is used to generate data for model testing. This event camera simulation draws significantly from [9] and authors' experience in simulating star signatures and intensities. Given input attitude dynamics, the model determines the star field that would be visible and calculates the per-pixel intensities of that field over the simulation time. This is done at a high temporal resolution to ensure that the space is well sampled. An example of one of these intensity frames is shown in Figure 6. During the intensity frame generation, the visible stars are limited to stars of visual magnitude 6 or brighter. This was determined by the authors' experiments and is also observed by other researchers [10].

Figure 6: Intensity Frame Generated for Event Data Simulation

For each pixel, a spline is calculated using the logarithm of the intensities viewed by that pixel over time. This produces a continuous per-pixel intensity simulation over the entire simulation time. Using this spline, the exact times, to microsecond resolution, of events can be determined. An event is determined to occur when the log intensity increases or decreases by a predefined contrast threshold. Often the positive threshold is greater than the negative; however, in the simulated frames used here, they are the same. An illustration of this event calculation process is shown in Figure 7.

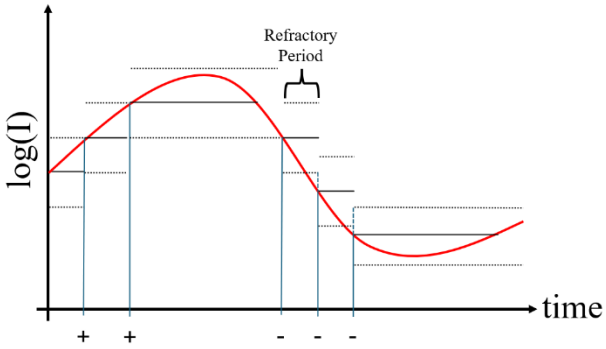

Figure 7: Event Calculation From log(Intensity) Spline

Each event calculated can only occur if it is outside of the refractory period of the previous event. This is shown in Figure 77. This refractory period reflects actual event camera performance and can be modified within the simulation and via biases on commercial event cameras.

Table 1 summarizes the inputs to the simulation.

Table 1: Simulation Inputs

Name	Description	Units
Initial Attitude	The Right Ascension and Declination (RA/DEC) in the sky to start the simulation pointed toward.	RA/DEC (radians)
Angular Velocity	The rotational velocity of the space vehicle.	Radians/second
Field of View (FOV)	The angle describing the cone viewable by the event camera.	Degrees
Resolution	The number of pixels in each axis.	Pixels
Positive Contrast Threshold	The relative change needed to produce a positive event.	Percent
Negative Contrast Threshold	The relative change needed to produce a negative event.	Percent
Refractory Period	The minimum time between events for a single pixel.	Microseconds

Several different datasets have been generated to support this development effort. They include slewing and boresight rolling cases. Their inputs are described before the results are shown in the Results section.

Particle Filter Implementation

The particle filter used for this experiment assumes a constant angular velocity for the system dynamics. The particles are resampled with a constant 0.01 radian covariance and are resampled with every new event packet. During resampling, the particles are propagated and scored again. The highest weighted particles are then resampled to replace the lower scoring particles. Noise is added to the attitude and velocities of high weight particles before replacing the low weight particles to add additional variation in the particle distribution.

Both test cases described in the Results section utilize event packets that are 0.025 seconds in length. This corresponds to an update rate of 40 Hz. The event packet size chosen is selected to balance accuracy with computation time. The measurement function used for the update step is described in detail below.

Measurement Function

Particle filters are particularly sensitive to measurement function selection. The measurement function calculates the probability of the measurement observed given the predicted state distribution.

The measurement function compares the expected focal plane of each of the particles with the focal plane generated by the event packet it has been fed. The events contained in the first 0.01 seconds of the event packet are accumulated to generate a synthetic focal plane. This accumulation frame is then used to find the centroids of the stars on the focal plane at the beginning of the event packet. A star catalog is then used to calculate the expected centroids for the simulated focal planes of each of the particles.

A modified triangle algorithm is used to compare features from the event data to the predicted star image. First, the stars centroids are sorted by their distance from the center of the focal plane, and then are used to generate every possible triangle that can be made with those points. The two outer most points are excluded from this calculation to avoid issues with one or two stars not being visible.

Next, the angles defining each triangle are calculated, and stored as a matrix. In order to map these angles to a metric, a scoring function measures the sum of differences between the angles of the measured and predicted triangles to score the particles. This score is then inverted, meaning the lowest distances will have the highest scores, and used to update the particle weights.

Particle Distribution

If starting with no a-priori attitude or velocity knowledge a uniform sphere of 20,000 particles is generated to find an initial weighting using the measurement function. The 0.1% of highest weighted attitudes are then determined and used to generate the particles that will be utilized in the filter. Each of these attitudes is then matched with a uniform distribution of velocity vectors to generate a set of 5,000 total particles that each have their own attitude and velocity vectors.

If any a-priori attitude knowledge is known, the particles can be initially distributed in a significantly smaller region. This allows for a smaller number of particles to be used to save computational resources, or for the velocity distributions to be larger for each initial attitude to save time to convergence.

RESULTS

The data passed to the particle filter for attitude tracking performance assessment has been generated with the parameters in Table 2.

Table 2: Test Case 1 Inputs

Parameter	Value
Initial Attitude	30 degrees RA, 30 degrees DEC
Angular Velocity	[5, 0, 0] degrees per second
Field of View (FOV)	13 degrees
Resolution	321 x 321 pixels
Positive Contrast Threshold	0.1
Negative Contrast Threshold	0.1
Refractory Period	100 µs

The angular displacement between the particle with the highest weight and the truth was determined using Equation 5:

$$\theta = 2\cos^{-1}\left(q_{particle}q^{*}_{truth}\right) \quad (5)$$

Where the * operator represents the conjugate of the quaternion. Angular velocity error was calculated using the root mean square error (RMSE) between the truth velocity and the highest weight particle velocity.

$$RMSE = \sqrt{\sum_{i=1}^{n}\left(v^{i}_{truth} - v^{i}_{particle}\right)^{2}} \quad (6)$$

Figure 8 shows these error values over time.

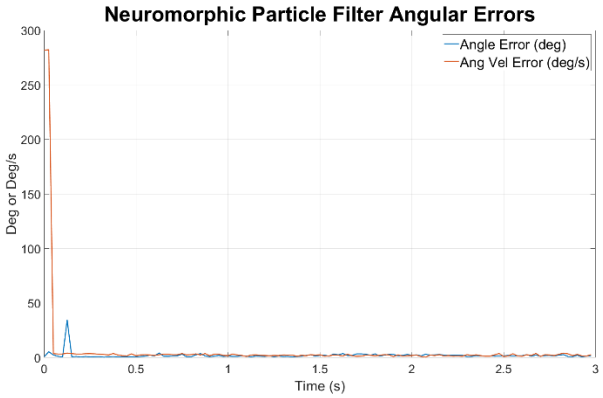

Figure 8: Errors of Particle Filter over Time for Test Case 1

The high initial error settles quickly. Figure 9 shows a zoomed in plot showing the performance over time.

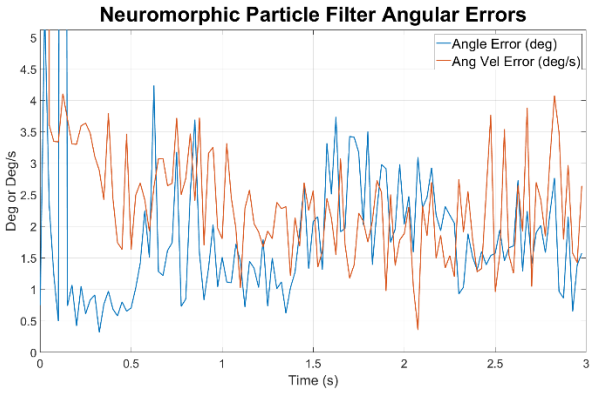

Figure 9: Errors of Particle Filter over Time for Test Case 1 – Zoomed

The angular error for test case 1 settles between 1 degree and 2 degrees. While this value would not be considered competitive for modern star trackers, this value has been obtained with a relatively low-resolution camera.

Test case 2 used the parameters in Table 3.

Table 3: Test Case 2 Inputs

Parameter	Value
Initial Attitude	30 degrees RA, 30 degrees DEC
Angular Velocity	[0, 0, 5] degrees per second
Field of View (FOV)	13 degrees
Resolution	321 x 321 pixels
Positive Contrast Threshold	0.1
Negative Contrast Threshold	0.1
Refractory Period	100 µs

The difference between test case 1 and test case 2 is that the rotation of the vehicle is about the boresight instead of slewing across the night sky. The errors are calculated according to Equation 5 and Equation 6. The error over time is shown in Figure 10.

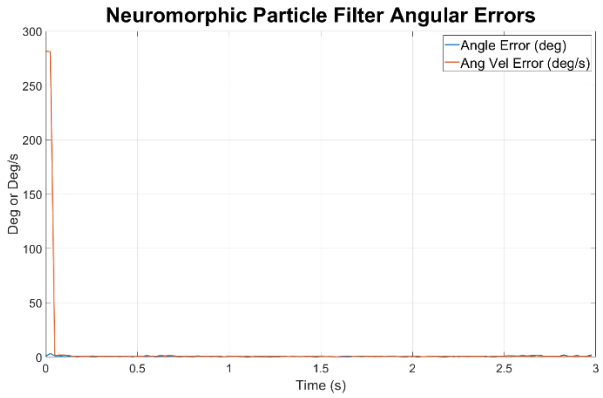

Figure 10: Errors of Particle Filter over Time for Test Case 2

Again, this plot is skewed by the high initial error which decays quickly. For ease of visualization, Figure 11 shows a zoomed in version of the errors.

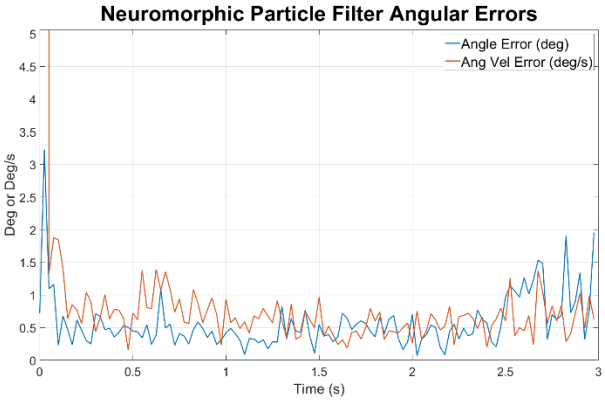

Figure 11: Errors of Particle Filter over Time for Test Case 2 – Zoomed

For test case 2, the error appears to settle between 0.25 and 0.75 degrees. This is closer to lower accuracy, commercially available star trackers, but it should be noted that these updates are provided at the high update rate of 40 Hz.

CONCLUSIONS

The simulation results show that event cameras can use novel algorithms to support the star tracking and attitude determination needs of satellites. It is believed that this application in particular shows promise for space applications, especially if the algorithm performance can be improved. The authors believe the following would improve the algorithm:

- Further iteration on the measurement function and its parameters would yield higher attitude estimation accuracy. Taking advantage of other features that can be calculated from the event data, such as optical flow, could dramatically improve performance.
- Improvements to the state transition function could improve filter performance by responding more effectively to the expected system dynamics.
- Exploring other resampling methods and proposal distributions for the particle filter.
- Optimization of the initial prior distribution could significantly reduce the needed particles. This would in turn significantly reduce runtime and needed computational power.

With these improvements, it is believed that this method could be viable for application with the current set of commercially available event cameras.

Using current commercially available event cameras and processors, this could be performed with a relatively low size, weight, and power (SWaP) system. Due to its

ability to operate at much higher rates, this type of system could provide a high-quality single attitude determination system for a small satellite, eliminating the need for gyroscopes or other attitude determination systems.

Neuromorphic computing represents a new class of potential space hardware that can dramatically improve space vehicle mission success and performance. Small satellites could gain from the rise of low power event-based computing architectures to allow for much more efficient operations. This approach only represents the beginning of possible applications of event cameras in the space domain.

Acknowledgments

The authors would like to thank ExoAnalytic Solutions, Inc. for supporting this work. In particular, the authors thank Doug Hendrix, Jim Myers, and Nancy Rhoades for championing the internal research efforts into event cameras at ExoAnalytic Solutions, Inc. The authors would also like to thank Jeffrey Ganley of the Air Force Research Laboratory's Space Vehicles Directorate for prior advice and guidance on event cameras and their applications to small satellites.

References

1) P. Lichtsteiner, C. Posch and T. Delbruck, "A 128 × 128 120 dB 15 μ s Latency Asynchronous Temporal Contrast Vision Sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566-576, 2008.

2) T. Delbruck and C. Mead, "Analog VLSI phototransduction," California Institute of Technology Computation and Neural Systems Program, Pasadena, CA, 1996.

3) T. Finateu and e. al., "A 1280x720 Back-Illuminated Stacked Temporal Contrast Event-Based Vision Sensor with 4.86 um Pixels, 1.066 GEPS Readout, Programmable Event-Rate Controller and Compressive Data-Formatting Pipeline," in *2020 IEEE International Solid-State Circuits Conference - (ISSCC)*, San Francisco, CA, USA, 2020.

4) N. O. Ralph, "Exploring space situational awareness using neuromorphic event-based cameras," Western Sydney University, Sydney, 2023.

5) G. Cohen and e. al., "Event-based vision sensor observations of lightning from the International Space Station: The Falcon Neuro mission," in *AGU23*, 2023.

6) J. Brown and K. Stubis, "TETRA: Star Identification with Hash Tables," in *Small Satellite Conference*, Logan, UT, 2017.

7) arcsec Space, "arcsec: Products," 2024. [Online]. Available: https://www.arcsec.space/web/products. [Accessed 5 June 2024].

8) M. S. Arulampalam, S. Maskell, N. Gordon and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174-188, 2002.

9) R. Graça, B. McReynolds and T. Delbruck, "Shining light on the DVS pixel: A tutorial and discussion about biasing and optimization," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Vancouver, 2023.

10) 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.