Event-Based Feature Tracking Using the Iterative Closest Point (ICP) Algorithm

Hands-on Perception Course Project

Mohamed Khaled Othman, Goitom Abrha Leaku, Eliyas Kidanemariam Abraha,

***Abstract*—The emergence of event-based vision sensors, exemplified by the Dynamic and Active-pixel Vision sensor (DAVIS), signifies a transformative advancement in visual perception. These sensors excel in capturing asynchronous brightness changes and synchronous grayscale frames, granting unmatched advantages in temporal resolution, latency, and high dynamic range. This breakthrough opens avenues for achieving efficient and accurate feature tracking, pivotal for high-level applications like simultaneous localization and mapping (SLAM), structure from motion (SfM), and visual odometry (VO). This Paper introduces an innovative methodology leveraging the Iterative Closest Point (ICP) algorithm for event-based feature tracking. Through intelligent event grouping and transformation into image frames, the computational complexity is reduced, which facilitates integration into many robotics applications. The proposed approach integrates efficient event binning, Shi-Tomasi corner detector, and Canny edge detectors for feature detection; and the robustness of the ICP algorithm for accurate tracking. Rigorous evaluation underscores exceptional performance, signaling the profound impact of event-based vision sensors on real-time analysis in dynamic environments and laying the foundation for future advancements in robotics perception.**

***Index Terms*—Event-Based Camera, ICP, Feature Tracking.**

I. Introduction

THE development of event-based vision sensors, like the Dynamic and Active-pixel Vision Sensor (DAVIS), represents a paradigm shift in how humans perceive their environment It combines a traditional frame-based camera with an asynchronous event sensor, offering unparalleled advantages in temporal resolution, latency, and dynamic range. This integrated sensor presents a unique opportunity to leverage the complementary strengths of event and frame sensors, calling for innovative methodologies to harness their combined advantages and enhance solutions for robotics applications[1]. But in order to properly utilize these, specific algorithms that are suited to their unusual output characteristics need to be developed.

This paper proposes an innovative methodology for event-based feature tracking centered around the Iterative Closest Point (ICP) algorithm, aiming to address the computational challenges posed by the high frequency and large volume of events captured by these sensors. The methodology, characterized by grouped event processing, intelligently converts events into image frames, thereby reducing computational complexity and enhancing the performance of the feature tracker. Key components of the methodology include event binning for data condensation, corner detection using the Shi-Tomasi feature detector, and edge detection via the Canny edge detector. By aligning model points with data points from event frames, the ICP algorithm ensures accurate feature tracking. Through rigorous evaluation and analysis, this research aims to demonstrate the feasibility and effectiveness of event-based feature tracking, contributing valuable insights to the understanding and advancement of event-based vision sensors. Moreover, the outcomes of this study are expected to have significant implications for robotics applications, particularly in dynamic environments where real-time perception is essential for tasks such as simultaneous localization and mapping (SLAM) and visual odometry.

II. Related Works

Research efforts have focused on developing specialized algorithms and methodologies tailored to the unique characteristics of event-based cameras. Pioneering work by [1] introduced the concept of event-based sensors, highlighting their potential for low-latency and high-speed visual processing tasks. This foundational work set the stage for further exploration into the capabilities of event-based vision systems.

Studies by [2] have delved into various techniques for event processing, feature extraction, and object tracking using event-based cameras. Their research demonstrated the efficacy of these techniques in real-time robotics applications, underscoring the practical advantages of event-based vision in dynamic environments.

Significant contributions by [3] and [4] have advanced the field by developing event-based algorithms for complex tasks such as simultaneous localization and mapping (SLAM) and visual odometry (VO). These studies have showcased the robust and efficient navigation capabilities of event-based sensors, which are crucial for autonomous systems operating in unpredictable settings. The integration of event-based vision with SLAM and VO algorithms has opened new avenues for enhancing the precision and reliability of robotic perception and navigation.

Collectively, these research efforts lay the groundwork for ongoing innovation, driving the development of more advanced and efficient event-based vision applications in various fields. Hence, in this paper, the above line of research has been continued by proposing a basic methodology for event-based feature tracking utilizing the Iterative Closest Point (ICP) algorithm; addressing the computational challenges associated with the high frequency and large volume of events.

III. Methodology

In this section, a detailed methodology for event-based feature tracking using the Iterative Closest Point (ICP) algorithm had been presented. The computational challenges, posed by event-based cameras, had been tackled by adopting a grouped approach to process events collectively within specified intervals. Our feature detection employs the Shi-Tomasi feature detector for corner detection and the Canny edge detector for edge generation. Subsequently, the ICP algorithm facilitates feature tracking by aligning the model and event data points extracted from event frames. To manage feature tracking effectively, a feature re-initialization process is applied when features fall below a minimum threshold. The following subsections provide a thorough breakdown of each step in our methodology, encompassing event binning, feature detection, the feature tracker, and feature re-initialization, illustrating the techniques and algorithms employed for event-based feature tracking with the ICP algorithm.

A. Event Binning

Processing data from the DAVIS camera on an events-by-events basis can be computationally expensive due to the high frequency and large volume of events generated. Each event corresponds to a specific pixel location, timestamp and its polarity, resulting in a substantial amount of data that needs to be processed and analyzed. Performing feature tracking on an events-by-events basis would require handling and analyzing each event individually, imposing a significant computational cost. To optimize computational resources, a grouped approach had been adopted for event processing. Events within a certain interval were collectively converted into image frames, condensing the event data and reducing the computational load associated with feature tracking. This grouping approach allowed for more efficient handling and analysis of data, optimizing the overall performance of the feature tracker. Within each interval, pixel values were assigned to represent event occurrences, resulting in binary image frames where the presence of an event was represented by a white pixel. This method effectively balanced the thickness and sparsity of resulting edges in the image frames, enhancing the effectiveness of feature tracking, taking more events to create the image frame would result more cluttered events.

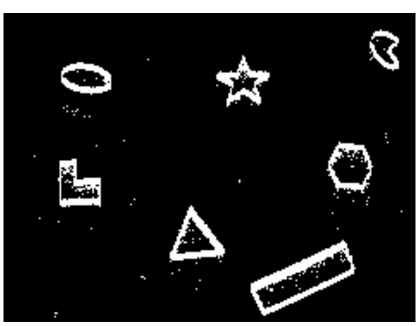


Fig. 1: Event Binning Binary Image

B. Feature Detection

The feature detection process in our algorithm encompasses several steps to identify salient points for feature tracking. Initially, the Shi-Tomasi feature detector is employed to locate corner points on the intensity image frame. This widely-used method operates by computing the minimum eigenvalue of the second-moment matrix of image gradients at each pixel, identifying corners as points with high minimum eigenvalues. Subsequently, the intensity frame is converted to grayscale, and a Canny edge detector is applied to generate a binary image where 1 represents edge pixels and 0 represents non-edge pixels. Following this, local edge-map patches are drawn around corner points, representing small regions of the image containing edges or corners. Specifically, square patches of size 21 x 21 had been used; centered on each corner point, chosen to balance accuracy and efficiency. The choice of patch size is adjustable and impacts feature tracking performance, with larger patches capturing more information but being computationally expensive, while smaller patches are faster but may lack sufficient information. Within each patch, edge pixels are used to define model points, extracted for all corner points detected by the Shi-Tomasi corner detector. To enhance accuracy, outlier corners with insufficient edge pixels are removed, considering corners with less than 10 model points as outliers and excluding them from the list of features tracked. This step improves the robustness of the feature tracking algorithm by eliminating corners detected by the Shi-Tomasi detector that may lack surrounding edge pixels.

C. Feature Tracker

To track features in the event-based frames, a feature tracking algorithm is employed following a flowchart depicted in Figure 2. This algorithm involves main steps including data point, model data point extraction and ICP Registration.

1) Data Point Extraction: Data points are extracted, for each feature, from the patch associated with that feature. These data points correspond to pixel locations where events were triggered, specifically pixels within the patch with a value of 255. This extraction process is performed for all patches in a given event frame. Outliers are eliminated to enhance the robustness of the subsequent ICP registration algorithm against noise. Data points located more than 3 pixels away from any model point within the patch are discarded. This filtering step ensures the retention of events triggered near edge pixel locations. The resulting model points and data set the basis for subsequent registration.

2) ICP Registration: The Iterative Closest Point (ICP) algorithm is a fundamental tool for aligning point clouds and is widely used in point cloud registration. In this study, we employed the point-to-point variant of the ICP algorithm from the open-source Python library, open3d. The main objective was to align model points (Q) with data points (P) to determine the optimal transformation (T) minimizing distances between corresponding points. The process begins with an

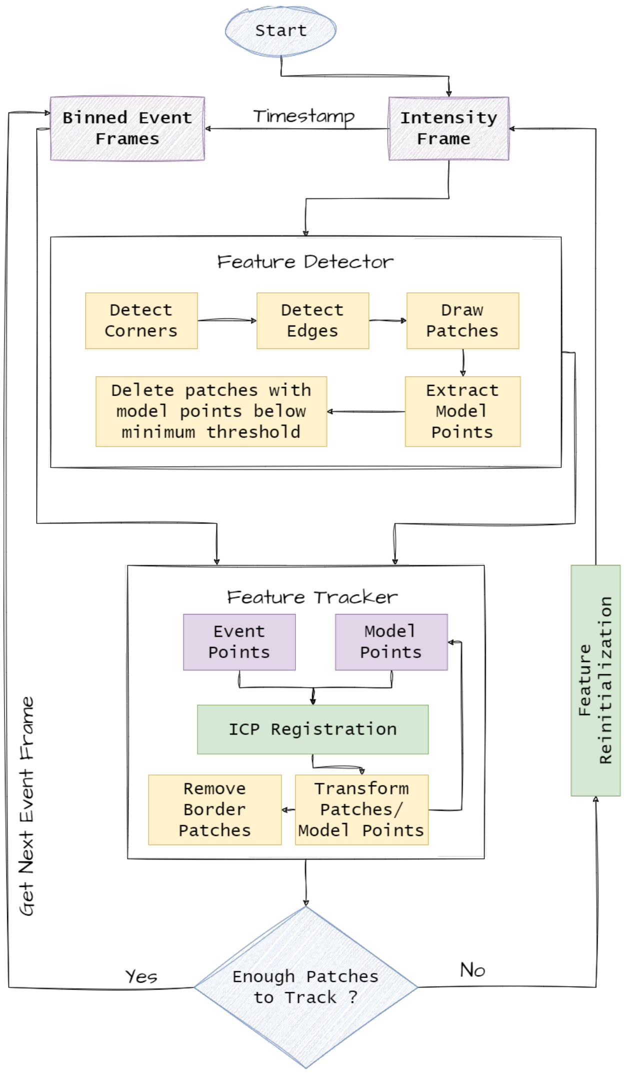


Fig. 2: Feature Tracker Flowchart Design

initial transformation estimate assumed to be identity. Point-to-point correspondences are established using Euclidean distance metrics, followed by an iterative refinement process minimizing point distances via the objective function:

$$E(T) = \sum_{(p,q)\in K} \|p - Tq\|^2 \tag{1}$$

Iterations continue until predefined convergence criteria are met. Upon convergence, the final estimated transformation (T) represents the alignment between the model and data points. This transformation is applied to features and model point sets, forming the basis for feature tracking in subsequent frames as in Figure 3. The ICP algorithm's iterative nature ensures robust and accurate point cloud registration.

D. Feature Reinitialization

In our tracking algorithm, certain conditions had been selected to apply the patch removal, such as patch proximity to the image border or an insufficient number of model or data points, which affect the ICP algorithm's performance. Hence,

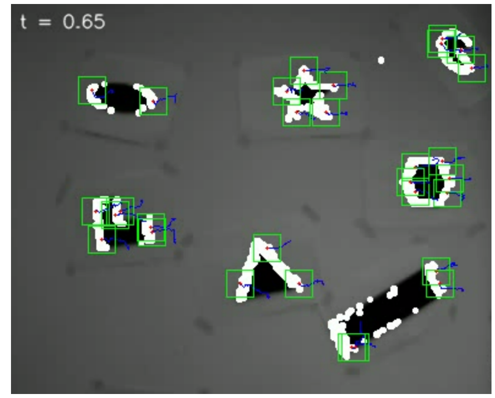


Fig. 3: Trajectory of patch centers between ten Event Frames.

a feature reinitialization process was employed to manage situations where the number of remaining patches falls below the minimum threshold. This involves selecting the next closest timestamp intensity frame in the sequence with the closest timestamp to the last event frame where tracking was interrupted. Feature detection is then performed on the selected frame, and tracking is resumed. If the number of features is still insufficient for tracking, the frame is skipped, and the process moves to the subsequent frame. This iterative approach continues until an intensity frame with an adequate number of features is found.

The minimum number of features required for tracking is dynamically set at 50% of the features detected in the intensity image. This threshold can be adjusted based on specific application needs or preferences. This reinitialization strategy ensures the tracking algorithm maintains a sufficient number of features for effective and reliable tracking, thereby enhancing the overall accuracy and robustness of the process, even when features are lost or become inadequate for successful tracking.

IV. Results and Discussion

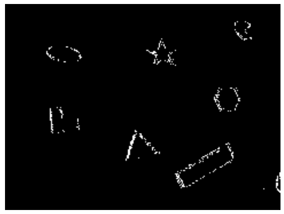
The performance of the event-based feature tracker was evaluated across different datasets, primarily involving objects with distinct and well-defined edges. To incrementally showcase the progress and effectiveness of our methodology, the subsequent subsections present intermediate results from different components of the integration pipeline as illustrated in Figure 2.

A. Event Binning

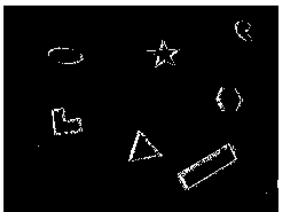
The size of each event bin is a critical parameter that needs to be adjusted according to the specific dataset and the object of interest that needs to be tracked. If the bin size is too small, it may not capture the full transformation of the features of interest, leading to insufficient data points. Conversely, a large bin size can introduce motion blur in the event data, particularly during periods of rapid camera movement. This results in the inability to capture subtle but significant intermediate motions, causing noticeable 'jumps' in the tracking output.

For our implementation, with some trials of implementation a bin size of 1000 events was selected. This choice balances the need to capture detailed transformations without introducing excessive motion blur, ensuring smooth and accurate tracking results as shown in Figure 4c.

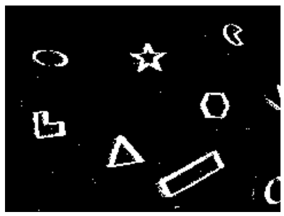
(a) 100 Events per frame



(b) 500 Events per frame



(c) 1000 Events per frame

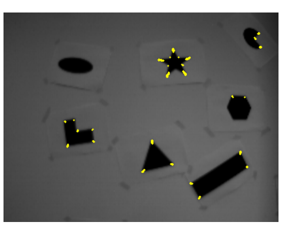


(d) 5000 Events per frame

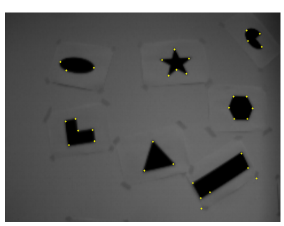
Fig. 4: Tracking results with different binning sizes.

B. Model Points

1) Corner Detector: In evaluating corner detection algorithms, we compared the Harris and Shi-Tomasi methods, with their results illustrated in figure 5. Detected corners are marked with yellow crosses. Despite expectations from literature, where Shi-Tomasi is touted to identify fewer spurious corners, our findings showed it detected significantly more corners than the Harris algorithm. Specifically, the Harris detector, shown in Figure 5a, missed several real corners that were detected by the Shi-Tomasi detector, as seen in Figure 5b. This discrepancy highlights the robustness of the Shi-Tomasi detector in our application. The ability of Shi-Tomasi to detect more relevant corners was instrumental in our choice to use it, as it minimizes the need for feature re-initialization, thereby maintaining the tracker's performance over prolonged periods.



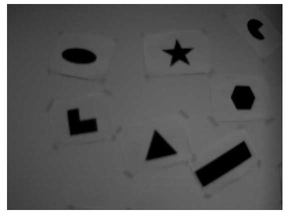
(a) Harris Corners



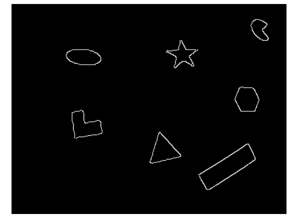
(b) Shi-Tomasi Corners

Fig. 5: Different Corner Detectors Results

2) Edges Detector: For the edge detection phase, the Canny edge detector had been employed to extract edges from the original intensity image as shown in fig 6a . Figure 6b depicts the resulting edge image, where strong edges are emphasized while weak edges are scored based on their connectivity to strong edges. This binary edge-map representation, shown in Figure 6b was generated through canny edge detector. The utilization of the Canny edge detector facilitated the extraction of relevant edge information, crucial for subsequent feature tracking processes.



(a) Reference Image

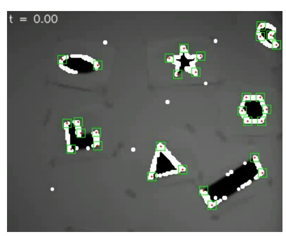


(b) Edges' Image

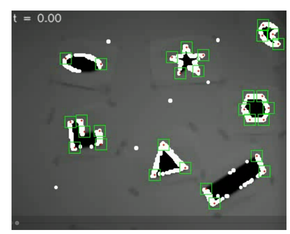
Fig. 6: Canny Edges Detector.

C. Patch Size

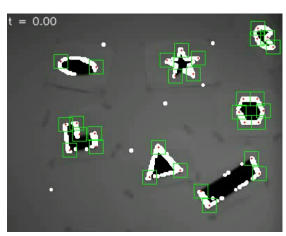
The patch size serves as a crucial parameter in our implementation, impacting the performance of feature tracking. Experimentation is essential to determine the optimal patch size, as showcased in Figure 7, where various patch sizes are visualized as green boxes overlaid on detected corners and edges. A larger patch size results in higher computational requirements and increased patch overlap, particularly in scenarios where candidate corners are closely situated. This overlap can lead to issues if the ICP algorithm returns incorrect transformations due to noisy data, such as spurious edges, affecting neighboring patches. Conversely, a smaller patch size may provide insufficient data for ICP registration and may exacerbate the aperture problem, misleading the ICP algorithm.



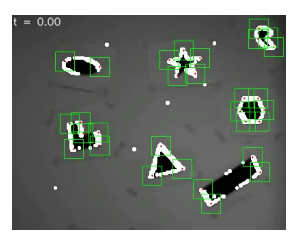
(a) 13x13 Pixel Patches



(b) 17x17 Pixel Patches



(c) 21x21 Pixel Patches



(d) 29x29 Pixel Patches

Fig. 7: Local Patch Sizes.

Based on the comparison presented in figure 7, a patch size of 21x21 was selected as it gave a better result in a balance between capturing sufficient information for accurate feature tracking and minimizing computational overhead. This size ensures that an adequate amount of data is provided for ICP registration while mitigating the risk of excessive patch overlap which all results in longer tracking time.

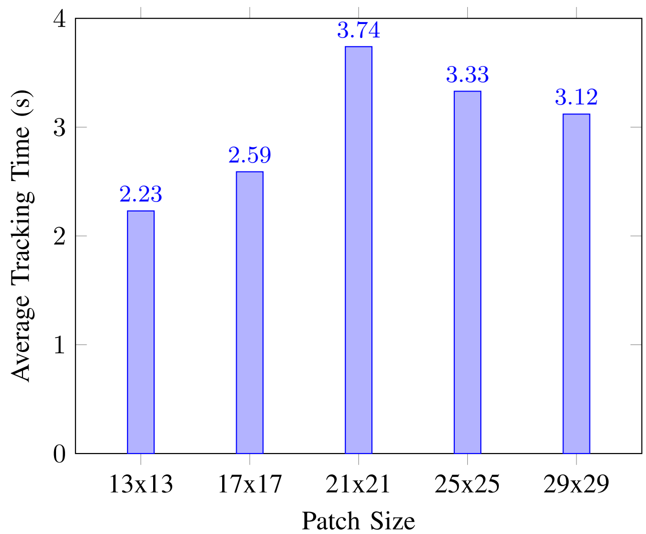
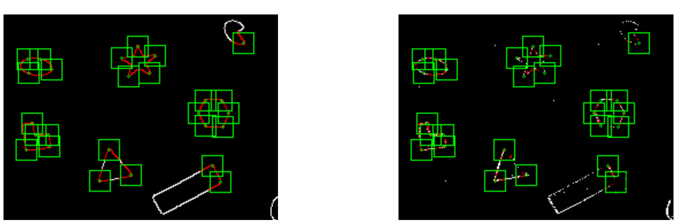


Fig. 8: Comparison of Patch Sizes and Average Tracking Time

D. Iterative Closest Point (ICP)

The Iterative Closest Point (ICP) algorithm was utilized to align model points and event points for precise feature tracking. In our implementation, both sets of points, as shown in Figures 9a and 9b, were fed into the ICP algorithm. The process began with the identity matrix as the initial transformation. Through iterative refinement, the ICP algorithm computed an optimal transformation that effectively aligned the model points with the event points. This alignment, achieved by minimizing the distances between corresponding points, is crucial for maintaining accurate and robust feature tracking throughout the sequence of event frames. The transformation results, showcasing the alignment of the two point sets, indicating the effectiveness of the ICP algorithm in our feature tracking pipeline.



(a) Model Points (b) Event Points

Fig. 9: ICP Source and Target Input Points

V. Conclusion and Future Work

In conclusion, this study introduces a novel methodology for event-based feature tracking using the Iterative Closest Point (ICP) algorithm. By leveraging the capabilities of the Dynamic and Active-pixel Vision Sensor (DAVIS), the approach effectively addresses the challenges posed by high-frequency, low latency, and high-volume event data. Key components of the methodology include event binning for data reduction, feature detection through the Shi-Tomasi corner detector and Canny edge detector, and the robust application of the ICP algorithm for accurate feature tracking. Evaluation results demonstrate the method's efficiency and accuracy in various dynamic environments, underscoring the potential of event-based vision sensors to significantly enhance real-time robotic perception. This work lays the foundation for future advancements in robotics applications, particularly in areas requiring precise and low-latency visual processing.

Future research could focus on enhancing the robustness and scalability of the proposed methodology. One promising direction is the integration of neural networks for feature extraction and event data processing. Neural networks, particularly convolutional neural networks (CNNs), have demonstrated exceptional performance in image processing tasks and can be trained to handle the unique characteristics of event-based data. Incorporating neural networks could improve feature detection accuracy and reduce the computational load, thereby enabling real-time processing for more complex and dynamic environments. Additionally, exploring unsupervised learning techniques may offer further advancements in adaptive feature tracking and recognition, enhancing the overall capability of event-based vision systems in robotics applications.

Acknowledgment

The authors would like to thank their supervisors Nuno Ricardo and Jad Mansour for their invaluable support and guidance throughout this research.

References

[1] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240x180 130dB 3us latency global shutter spatiotemporal vision sensor," IEEE J. of Solid-State Circuits, vol. 49, no. 10, 2014.

[2]A. Mitrokhin, C. Ferm"uller, C. Parameshwara, and Y. Aloimonos, "Event-based moving object detection and tracking," in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 1-9.

[3] D. Weikersdorfer and J. Conradt, "Event-based particle filtering for robot self-localization," in IEEE Int. Conf. on Robotics and Biomimetics (ROBIO), 2012.

[4] A. Censi and D. Scaramuzza, "Low-latency event-based visual odometry," in IEEE Int. Conf. on Robotics and Automation (ICRA), 2014.

[5]B. Kueng, E. Mueggler, G. Gallego and D. Scaramuzza, "Low-latency visual odometry using event-based feature tracks," 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea (South), 2016, pp. 16-23, doi: 10.1109/IROS.2016.7758089.