

# Optical Flow-Guided 6DoF Object Pose Tracking with an Event Camera

Zibin Liu

National University of Defense  
Technology  
Changsha, China  
liuzibin@nudt.edu.cn

Shunkun Liang

National University of Defense  
Technology  
Changsha, China  
liangshunkun@nudt.edu.cn

Banglei Guan\*

National University of Defense  
Technology  
Changsha, China  
guanbanglei12@nudt.edu.cn

Zhenbao Yu

Wuhan University  
Wuhan, China  
zhenbaoyu@whu.edu.cn

Yang Shang\*

National University of Defense  
Technology  
Changsha, China  
shangyang1977@nudt.edu.cn

Qifeng Yu

National University of Defense  
Technology  
Changsha, China  
yuqifeng@nudt.edu.cn

## ABSTRACT

Object pose tracking is one of the pivotal technologies in multimedia, attracting ever-growing attention in recent years. Existing methods employing traditional cameras encounter numerous challenges such as motion blur, sensor noise, partial occlusion, and changing lighting conditions. The emerging bio-inspired sensors, particularly event cameras, possess advantages such as high dynamic range and low latency, which hold the potential to address the aforementioned challenges. In this work, we present an optical flow-guided 6DoF object pose tracking method with an event camera. A 2D-3D hybrid feature extraction strategy is firstly utilized to detect corners and edges from events and object models, which characterizes object motion precisely. Then, we search for the optical flow of corners by maximizing the event-associated probability within a spatio-temporal window, and establish the correlation between corners and edges guided by optical flow. Furthermore, by minimizing the distances between corners and edges, the 6DoF object pose is iteratively optimized to achieve continuous pose tracking. Experimental results of both simulated and real events demonstrate that our methods outperform event-based state-of-the-art methods in terms of both accuracy and robustness.

## KEYWORDS

Event Camera; Pose Tracking; Optical Flow; Hybrid Feature

## 1 INTRODUCTION

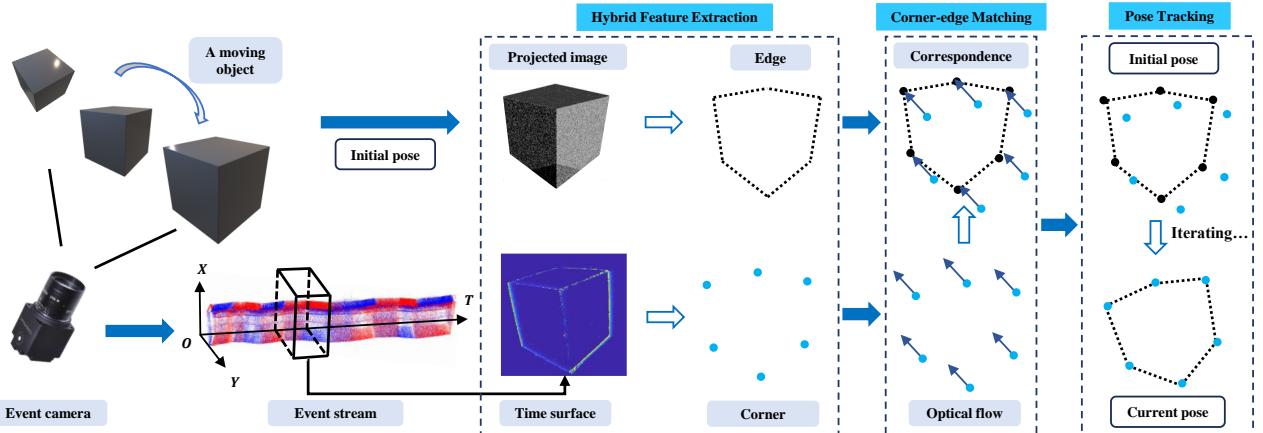
Object pose tracking is a hot research topic in the multimedia and computer vision communities, with significant real-world applications, such as augmented reality [19], robotic grasping [30], and autonomous navigation [11, 12]. The goal of object tracking is to continuously estimate six degrees of freedom (6DoF) that define the rotation and translation of an object relative to the sensor. Currently, there are numerous visual-based solutions that have been developed. Nonetheless, traditional cameras are constrained by various factors like low frame rates and limited dynamic range. Hence, the challenges in object tracking persist, such as drastic lighting change, motion blur due to rapid object movement, partial occlusion among objects, and interference from cluttered backgrounds.

A novel type of neuromorphic visual sensor, referred to as an "event camera", shows potential in addressing the aforementioned challenges. In contrast to traditional cameras that capture images at a fixed frame rate, event cameras have smart pixels that perceive variations of the logarithmic brightness, operating asynchronously and independently from one another [17]. Whenever a pixel detects a logarithmic intensity change in the scene that surpasses a certain threshold, it triggers an event [10]. Event cameras possess several advantages, including high temporal resolution, high dynamic range, low latency and minimal power consumption. These advantages make event cameras potentially superior to traditional cameras in challenging tasks, such as tracking fast-moving objects [9, 32]. Nevertheless, the unique asynchronous and discretized nature of events makes it difficult to directly apply traditional vision algorithms to address event-based tracking problems. Therefore, it is crucial to develop alternative algorithms to unlock the potential of event cameras for object tracking. The focus should be on addressing two issues: feature extraction and data association.

Feature extraction involves identifying typical features of the moving object from events, such as corners [34], lines [6], and circles [26]. These event-based features can precisely describe the object's appearance, providing critical geometric information for pose determination. Differing from traditional images, events exclusively report brightness changes asynchronously. Therefore, the feature extraction algorithms should fully leverage the distinctive spatio-temporal and photometric characteristics of events. Most events are predominantly triggered by edges of the moving object [10], thereby accurately capturing its contour. The vertices of edges are notably conspicuous, manifesting in events as corners. Furthermore, for 3D objects, the projection of their edges onto the event plane intuitively reflects their poses. This inspires us to comprehensively utilize multiple features, such as corners and edges, for the continuous 6DoF pose tracking of objects.

Data association refers to the establishment of correspondences between event features of the object and its model, namely the 2D-3D feature matching. For most tracking works, it is typically assumed that the initial pose of objects is known in advance [20], thus transforming data association into a matter of matching and updating 2D-2D features (event or features built from events) at

\*Corresponding authors.



**Figure 1:** Flowchart of the proposed method. The core algorithm includes event-based feature extraction, corner-edge matching, and pose tracking. The method takes event streams, initial pose and the object model as input, continuously producing the 6DoF poses of the object as output.

consecutive times. One of the primary challenges for data association of event cameras is the lack of any intensity neighborhoods among asynchronous events, potentially reducing the effectiveness of the feature-descriptor approach. The second challenge lies in significant sensor noise and potential event clutter, which may lead to erroneous feature matching. Optical flow estimation is one of the classical algorithms in computer vision, already utilized in object tracking [5], Visual Odometry (VO) [24], and Simultaneous Localization and Mapping (SLAM) [31]. This motivates us to utilize optical flow to establish connections between events and features. Upon establishing the feature correspondences, it becomes feasible to formulate an optimization function under the constraint of the perspective projection model, thereby enabling the 6DoF object pose tracking.

In this paper, we propose an optical flow-guided 6DoF object pose tracking method using an event camera, which is summarized in the block diagram of Figure 1. Our method initially employs a hybrid feature extraction strategy, specifically detecting the object corners from the Time Surfaces (TSs) of events, and extracting the object edges from the projected point cloud. Then, we compute the optical flow of corners and search for correspondences with edges along the direction of the optical flow. Finally, we conceptualize object pose tracking as an optical flow-guided iterative optimization problem. The experimental results of simulated and real events indicate that our methods surpass the current state-of-the-art event-based approaches. The main contributions of this work can be summarized as three-fold:

- We introduce a 2D-3D hybrid feature extraction strategy for precisely characterizing object motion, which detects corners and edges from events and point clouds, respectively.
- We model event-based optical flow estimation as a maximization problem of event-associated probability, fully utilizing the spatio-temporal distribution of events.
- We propose an optical flow-guided object pose optimization method, minimizing the distances between corners and edges along the direction of optical flow.

## 2 RELATED WORK

Over the past several decades, there has been extensive research utilizing vision sensors for object pose estimation and tracking. We provide a brief overview of the two primary categories below: monocular-based and event-based object pose tracking.

### 2.1 Monocular-based Object Pose Tracking

Existing monocular pose tracking methods differ in their utilization of edges, features, image regions, direct optimization and deep learning. Feature-based pose tracking methods utilize characteristic features of objects, such as keypoints [28] and lines [23]. Therefore, it is essential to ensure that object surfaces possess ample textures to enable the extraction of a substantial number of features. Region-based methods track objects by maximizing image statistical characteristics between foreground and background regions. Direct methods optimize the photometric error for pose tracking, relying on the fundamental assumption of photometric constancy [7]. Deep learning methods have demonstrated remarkable efficacy in object pose estimation [4]. However, ensuring real-time capability in such data-driven methods remains challenging.

The research on edge-based methods exhibits a high relevance to our current works. The first edge-based object tracking method, RAPID, is proposed by Harris et al. [13], which projects the model edges to images and aligns with image edges. Following the pioneering RAPID, numerous advancements have been put forth to integrate color information or color statistics for precise edge correspondences. Bugaev et al. [1] develop an edge energy function that utilizes both the intensity and orientation of the raw image gradient to determine 2D-3D edge correspondences. While edge-based methods exhibit superior performance for low-textured objects, they frequently encounter challenges when confronted with significant occlusion and densely cluttered backgrounds.

The inherent challenges of monocular-based object estimation methods include susceptibility to motion blur and sensitivity to variations in lighting conditions.

## 2.2 Event-based Object Pose Tracking

Early event-based works are very simple, which track moving objects using a known simple shape, such as a blob [8], circle [16] or line [6]. These methods match newly triggered events with the closest existing blob or feature, and subsequently update the parameters of objects, including their location, size, etc. Nonetheless, the limitation of these methods lies in their effectiveness, which is confined to a narrow range of object shapes. For more complex objects, an event-based iterative closest point algorithm [25] is proposed to update object poses using event-by-event adaptations. Valeiras et al. [27] estimate and track moving objects by establishing connections between events and the 2D projection of objects, given prior knowledge of the wireframe model and initial pose. The recent advancements in machine learning have sparked interest in utilizing data-driven methods for event-based object pose tracking. Jawaid et al. [18] leverage established learning-based approaches to efficiently detect and predict the 2D positions of the satellite. Following this, the Perspective-n-Point (PnP) solver is employed for satellite pose estimation and tracking. Chen et al. [3] propose a novel deep neural network, RM-RNet, which enables the end-to-end estimation of 5-DoF object-level motion, using Time-Surface with Linear Time Decay (TSLTD) frames.

Most existing methods process events by converting them into a frame-like representation to leverage traditional image processing algorithms. However, this process also leads to the loss of temporal and spatial characteristics inherent in events. In this study, we estimate the optical flow by leveraging the spatio-temporal distributed probability of events, and further propose an optical flow-guided pose optimization method.

## 3 METHOD

In this section, the fundamental mathematical concepts and the symbolic representation are elucidated in Section 3.1. After a clear problem definition, an optical flow-guided object pose tracking method is proposed. A hybrid feature extraction method for extracting corners and edges from events and point clouds is presented in Section 3.2. Subsequently, corner-edge matching is employed to establish the correlation between corners and edges guided by optical flow in Section 3.3. Finally, by minimizing the distance between corners and edges, we achieve pose optimization and continuous tracking of objects, as discussed in Section 3.4.

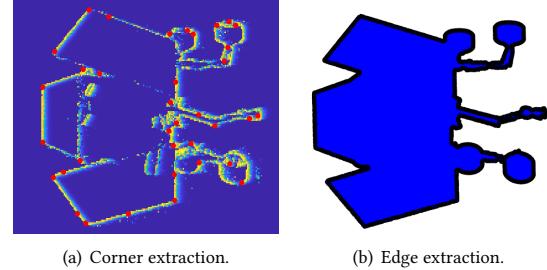
### 3.1 Problem Formulation

Event cameras possess intelligent pixels that independently react to variations in their logarithmic photocurrent  $L \doteq \log(I)$ . Mathematically, an event can be represented as  $e_k = (\mathbf{x}_k, t_k, a_k)$ , where  $\mathbf{x}_k = (x_k, y_k)^\top$  is pixel coordinates,  $t_k$  denotes the trigger time, and  $a_k \in \{+1, -1\}$  represents a binary polarity indicating whether the brightness is increasing ("ON") or decreasing ("OFF") [10]. An event is triggered when the brightness alteration of a single pixel surpasses a predetermined threshold, i.e.,

$$\Delta L(\mathbf{x}_k, t_k) \doteq L(\mathbf{x}_k, t_k) - L(\mathbf{x}_k, t_k - \Delta t_k), \quad (1)$$

reaches a temporal contrast threshold  $D > 0$ , i.e.,

$$\Delta L(\mathbf{x}_k, t_k) = a_k D. \quad (2)$$



**Figure 2: Feature extraction results. (a) Corners are extracted from TS, indicated in red. (b) Edges are extracted from the projected point cloud (blue), represented in black.**

Like traditional cameras, event cameras can be characterized by several camera models, with the pinhole camera model being the most widely used among them [16]. To approach this topic mathematically, let us begin with the ideal pinhole camera model. Suppose that the event  $e_k$  is triggered by a 3D point  $\mathbf{X}_k$  at time  $t_k$ . This projection process can be described as

$$\alpha \tilde{\mathbf{x}}_k = \mathbf{K} \mathbf{P}_{t_k} \tilde{\mathbf{X}}_k, \quad (3)$$

where  $\alpha$  is a scale factor,  $\tilde{\mathbf{x}}_k$  and  $\tilde{\mathbf{X}}_k$  are the homogeneous coordinates of  $\mathbf{x}_k$  and  $\mathbf{X}_k$ .  $\mathbf{K}$  represents the intrinsic parameters of the event camera, which is typically calibrated in advance.  $\mathbf{P}$  represents the extrinsic parameters, which describe the relative pose between the object and the event camera.

The event-based pose tracking problem entails continuously solving for the pose  $\mathbf{P}$  from event streams, specifically determining the position and orientation of the object relative to the event camera. Inspired by the concept of *recursive pose estimation*, given the object pose  $\mathbf{P}_{t_{k-1}}$  at the previous time  $t_{k-1}$ , the current pose of the object at time  $t_k$  can be solved by

$$\mathbf{P}_{t_k} = \mathbf{P}_{t_{k-1}} \Delta \mathbf{P}^{-1}. \quad (4)$$

The pose change  $\Delta \mathbf{P}$  between adjacent times is relatively small. Therefore, we employ the pose  $\mathbf{P}_{t_{k-1}}$  as the initial value and iterate to optimize the current pose  $\mathbf{P}_{t_k}$ .

### 3.2 Hybrid Feature Extraction

To establish a correlation between 2D images and 3D models, the most common approach involves using 2D-3D feature points [20] or lines [22]. Different from the above methods, we propose a strategy for hybrid feature extraction that fully leverages the distinctive characteristics of events and point clouds. Specifically, it involves extracting corners from events and contours from point clouds, utilizing the most stable feature representations from both modalities.

We transform events into a 2D false-color map, known as the Time Surface (TS), where each pixel records the timestamp of the last event occurring at that location. Utilizing an exponential kernel, TSs prioritize recent events over past occurrences, displaying sensitivity to object edges and the direction of motion [10].

Let  $t_{end}$  denote the timestamp of the last triggered event, then the TS at any time  $t_k > t_{end}$  is defined as:

$$TS(\mathbf{x}_k, t_k) \doteq e^{-(t_k - t_{end}(\mathbf{x}_k)) / \tau}, \quad (5)$$

where  $\tau$  is the decay rate parameter, which is set to 30 milliseconds, consistent with [33]. Due to factors such as sensor noise, guided filtering is initially applied to TS for noise reduction while preserving edge features. Then, we detect Harris corners of the object on the filtered TS, as shown in Figure 2(a). We preserve uniformly distributed corners to reflect the current motion state of the object accurately.

Due to occlusion and significant rotations of objects, event-based corner tracking may be unreliable. Hence, we have abandoned the conventional method of establishing correspondences between 2D and 3D feature points, opting instead to directly associate corners with the projected edges of objects. Similar to state-of-the-art monocular pose tracking algorithms, possessing prior knowledge of the object's 3D model and initial pose is an essential prerequisite [20]. Initially, we employ uniform sampling on the 3D model of objects in order to generate a more lightweight point cloud. Following this, utilizing the initial pose, the 3D point cloud of objects can be projected onto the image plane of event cameras. Subsequently, determine the boundaries of the 2D projected point cloud to ascertain the object's edges. Then, retrieve the corresponding 3D edges associated with the detected 2D edges. During each iteration, it is sufficient to project these 3D edges and associate them with corresponding corners. The results of edge extraction are depicted in Figure 2(b), validating the simplicity yet efficacy of our approach.

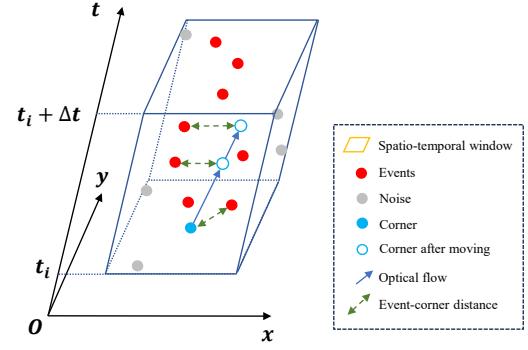
### 3.3 Corner-edge Matching

Before performing pose tracking, it is essential to address the problem of data association, specifically establishing the correspondences between object corners and edges. The most straightforward approach involves employing brute force matching or nearest neighbors matching to locate the edge points closest to the corners, thereby establishing correspondences. However, this approach may lead to a certain amount of mismatches. We aim to establish effective correspondences with edges through the motion direction of corners, and optical flow provides a promising approach to address this problem. Optical flow estimation is a classical method for describing the motion of pixels over time. As time passes and the object moves, corners shift within the event stream. Optical flow can trace the movement of these corners. Along the direction of optical flow, we seek edge points corresponding to these corners.

Let  $s$  denote the event corner, and its movement within the event plane can be described using optical flow,

$$s(t + dt) = s(t) + u \cdot dt. \quad (6)$$

When the time interval is sufficiently small, the optical flow  $u$  can be regarded as constant [35]. To seek the optical flow of corners, it is imperative to consider events that are located in the vicinity of corners. The geometric illustration of optical flow estimation is shown in Figure 3. Firstly, we establish an event spatio-temporal window  $\{e_k\}_{k=1}^m$  during the time interval  $[t, t + \Delta t]$ , centered around  $s_i$ . Ideally, events within a small window are mostly triggered by the same corner. However, due to factors such as noise, this assumption does not strictly hold true. Inspired by [34], we model the event-based optical flow estimation as a probability distribution problem. Under the guidance of optical flow, the distribution of events triggered by a corner should be more concentrated. We transform the maximization of event probability density into the minimization problem of



**Figure 3: The geometric interpretation of optical flow estimation. The optical flow of corners is computed by establishing correspondences between corners and events within a given spatio-temporal event window.**

statistical event-corner distance, modeled as

$$\min_{u_i} \sum_k^m \sum_i^n \omega_{ki} \left\| \mathbf{x}_k - \left( \mathbf{s}_i + \mathbf{u}_i t'_k \right) \right\|^2 = 0, \quad (7)$$

$$t'_k = t_k - t, \quad (7)$$

where  $\omega_{ki}$  represents the correlation probability between the event  $e_k$  and the corresponding corner  $s_i$  displaced along the optical flow  $u_i$ , which should inversely correlate with the distance between them. Taking bisquare weights as an example, it can be expressed as

$$\omega_{ki}(r_{ki}) = \begin{cases} \left[ 1 - \frac{r_{ki}^2}{b^2} \right]^2, & |r_{ki}| < b \\ 0, & |r_{ki}| \geq b \end{cases}, \quad (8)$$

where  $b = 4.685$  is a tuning constant.  $r_{ki}$  is the distance residual between  $e_k$  and  $s_i$  after moving along the optical flow  $u_i$ . After initializing the weights  $\omega_{ki}$ , we proceed to solve the linear least squares equation by addressing

$$\mathbf{u}_i \mathbf{A}^\top = \mathbf{B}, \quad (9)$$

where

$$\mathbf{A} = \left[ \sqrt{\omega_{11}} t'_1, \dots, \sqrt{\omega_{ki}} t'_k, \dots \right]^\top, \quad (10)$$

$$\mathbf{B} = \left[ \sqrt{\omega_{11}} (\mathbf{x}_1 - \mathbf{s}_2), \dots, \sqrt{\omega_{ki}} (\mathbf{x}_k - \mathbf{s}_i), \dots \right].$$

Then,  $\mathbf{u}_i$  can be determined by

$$\mathbf{u}_i = \frac{\mathbf{BA}}{\mathbf{A}^\top \mathbf{A}} = \frac{\sum_{k=1}^m \sum_{i=1}^n \omega_{ki} (\mathbf{x}_k - \mathbf{s}_i) t'_k}{\sum_{k=1}^m \sum_{i=1}^n \omega_{ki} t'^2_k}. \quad (11)$$

Through continuous iteration of Eqs. (8) and (9) until the threshold requirement is met or the maximum iteration limit is reached, the final value of  $\mathbf{u}_i$  is output. Subsequently, corners are propagated along the optical flow direction to locate the corresponding projected edge points  $\mathbf{c}_i$ .

### 3.4 Pose Tracking

Given the initial pose, the object model is projected onto the image plane and associated with detected corners by utilizing corner-edge matching. Subsequently, pose optimization is performed based on

---

**Algorithm 1** Event-Based Object Pose Tracking

---

**Input:** Events  $e$ , Intrinsic matrix  $\mathbf{K}$ , Initial pose  $\mathbf{P}_{t_0}$ , 3D point cloud of objects  $O$ ;

**Output:** Object poses  $\mathbf{P}_{t_1}, \mathbf{P}_{t_2}, \dots, \mathbf{P}_{t_k}, \dots$ :

- 1: **for** each timestamp  $t_1, t_2, \dots, t_k, \dots$  **do**
- 2:    $e_{t_k} \leftarrow \text{Generate\_Event\_Window}(e)$ ;
- 3:    $\text{TS} \leftarrow \text{Generate\_TS}(e_{t_k})$  using Eq. (5);
- 4:    $\text{Filtered\_TS} \leftarrow \text{Guided\_Filter}(\text{TS})$ ;
- 5:    $\mathbf{s} \leftarrow \text{Harris\_Corner\_Detection}(\text{Filtered\_TS})$ ;
- 6:   **for**  $i = 1$  to  $n$  **do**
- 7:      $\omega \leftarrow \text{Calculate\_Weight}(\mathbf{s}_i, e_{t_k})$  using Eq. (8);
- 8:      $\mathbf{u}_i \leftarrow \text{Calculate\_Optical\_Flow}(\omega, \mathbf{s}_i, e_{t_k})$  using Eq. (9);
- 9:     **if** threshold condition is met **then**
- 10:       **return**;
- 11:     **end if**
- 12:   **end for**
- 13:    $\text{2D\_Point\_Cloud} \leftarrow \text{Project\_3D\_Point\_Cloud}(O, \mathbf{P}_{t_0}, \mathbf{K})$ ;
- 14:    $\mathbf{c} \leftarrow \text{Extract\_Edges}(\text{2D\_Point\_Cloud})$ ;
- 15:    $\mathbf{c}_i, \mathbf{s}_i \leftarrow \text{Corner\_Edge\_Matching}(\mathbf{c}, \mathbf{s}_i, \mathbf{u}_i)$ ;
- 16:    $\mathbf{P}_{t_1} \leftarrow \text{Nonlinear\_Optimization}(\mathbf{c}_i, \mathbf{s}_i, \mathbf{P}_{t_0})$  using Eq. (12);
- 17: **end for**

---

the corresponding corners and edges, where corners record the object status at the current moment and edge points represent the object pose in the previous moment. By minimizing the distance  $d(\cdot)$  between corners and edges, we can optimize the object pose  $\mathbf{P}_{t_k}$  at the current time, i.e.,

$$\min_{\mathbf{P}_{t_k}} \sum_{i=1}^n d(\mathbf{c}_i, \mathbf{s}_i). \quad (12)$$

For event-based pose tracking, we achieve continuous object tracking by iteratively updating  $\mathbf{P}_{t_k}$ , which can be represented as

$$\xi^\wedge = \begin{bmatrix} \mathbf{\phi}^\wedge & \boldsymbol{\rho} \\ \mathbf{0}^\top & 0 \end{bmatrix} \in \mathbb{R}^{4 \times 4}, \xi = \begin{bmatrix} \mathbf{\phi} \\ \boldsymbol{\rho} \end{bmatrix} \in \mathbb{R}^6, \quad (13)$$

$$\mathbf{\phi} \in \mathfrak{so}(3), \boldsymbol{\rho} \in \mathbb{R}^3.$$

The pose variation vector  $\xi$  comprises both the rotation vector  $\mathbf{\phi}$  and translation vectors  $\boldsymbol{\rho}$ . We aim to search for the optimal set of parameters that minimizes the objective function (12). This is a challenging problem because the objective function may be highly nonlinear and non-convex, with multiple local minima. To address this issue, one can iteratively update the pose by moving in the direction of the steepest descent. The cost function can be approximated by its first-order Taylor expansion

$$d(\xi + \Delta\xi) \approx d(\xi) + J(\xi)^\top \Delta\xi. \quad (14)$$

The Jacobian matrix of the current pose  $\xi$  is

$$J(\xi) = \frac{\partial d(\xi)}{\partial \xi}. \quad (15)$$

We utilize the Levenberg-Marquardt (LM) algorithm, which restricts the step size of variable updates within a specified trust

region, ensuring that the Taylor expansion provides a reliable approximation. We have

$$(H(\xi) + \lambda I) \Delta\xi = g(\xi), \quad H(\xi) = J(\xi) J(\xi)^\top, g(\xi) = -J(\xi) d(\xi). \quad (16)$$

The increment  $\Delta\xi$  is calculated by

$$\Delta\xi = -(H(\xi) + \lambda I)^{-1} g(\xi), \quad (17)$$

where  $\lambda$  represents the regularization parameter and  $I$  is the identity matrix. We achieve pose tracking by composing the matrix exponential of the corresponding twist  $\Delta\xi^\wedge$  with the previous pose. When the threshold is satisfied, we can obtain the optimal object pose  $\mathbf{P}_{t_k}$  at the current moment. Upon arrival of new events, leveraging  $\mathbf{P}_{t_k}$  as the initial pose, we persistently refine it to attain the new pose  $\mathbf{P}_{t_{k+1}}$  at next moment for continuous pose tracking. The overall flow of the proposed method is illustrated in Algorithm 1.

## 4 EXPERIMENTAL EVALUATION

In this section, we assess the effectiveness of our methods using simulated and real events. The experimental details of event collection are provided in Section 4.1. Subsequently, we proceed to execute simulated and real event experiments for method validation as outlined in Sections 4.2 and 4.3, respectively.

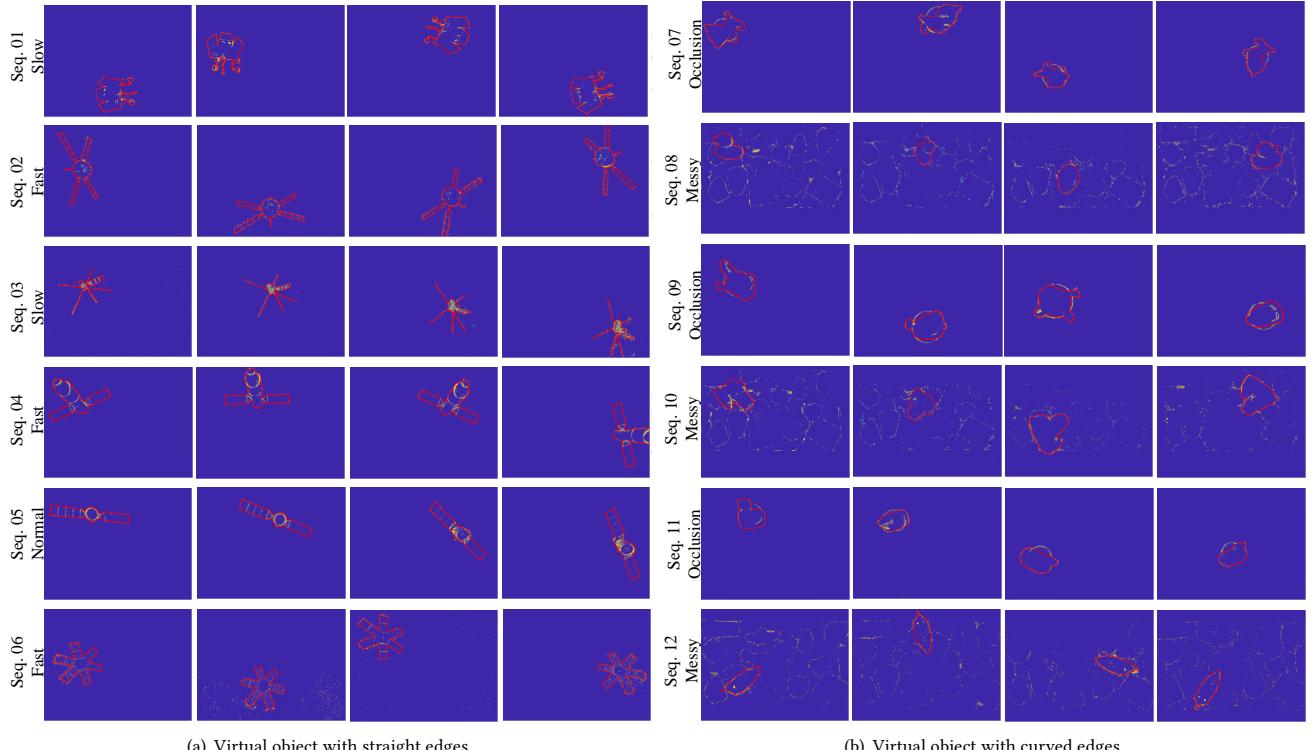
### 4.1 Experimental Preparation

Initially, experiments with simulated events are prepared. We select several representative objects with straight edges, as well as those with curved edges from [14]. Blender is utilized to render RGB videos of object motion, subsequently transforming them into event streams using V2E [15]. The events of objects with straight edges are designated as sequence 01-06, while those of objects with curved edges are designated as sequence 07-12. To simulate various complex scenarios that objects may encounter during motion, numerous challenging conditions are introduced, such as messy backgrounds, extensive occlusions, and fast movements.

To validate the feasibility of the proposed method, experiments with real events are subsequently conducted. Firstly, we manually manipulate the motion of objects and capture their movement using a fixed event camera (Prophesee EVK4), which is pre-calibrated [21]. A variety of objects, such as planar patterns, cubes, books, satellite models, and astronaut models, are used for testing, each corresponding to the sequence 13-17. Subsequently, the tracking methods are validated using event streams and compared against other advanced algorithms. The object models are known a priori, and the ground truth of the pose is acquired using the OptiTrack system. The initial pose is obtained in a similar manner as described in [2].

In order to perform a quantitative evaluation of the errors associated with each method, we present the object pose tracking results using two well-established metrics [29]:  $\delta_T = \|T - T_{gt}\|_2$ ,  $\delta_R = \cos^{-1}((\text{trace}(R^\top R_{gt}) - 1)/2)$ . The evaluation of these two errors involves comparing the estimated translation vector  $T$  and rotation matrix  $R$  with their corresponding ground truth values  $T_{gt}$  and  $R_{gt}$ .

To evaluate the performance of the proposed method, we compare against state-of-the-art event-based methods. For objects with straight edges, we employ the following three baseline methods: (i) Line-Based. Inspired by the work of Chamorro et al. [2] on event



**Figure 4: Pose tracking results of virtual objects in simulated event experiments.** The red edges represent the edges of objects, which are reprojected onto TSs using the estimated pose solved by our methods. (a) Tracking results of objects with straight edges, from top to bottom, corresponding to Sequence 01-06. (b) Tracking results of objects with curved edges, from top to bottom, corresponding to Sequence 07-12.

**Table 1: Pose tracking error of objects with straight edges in simulated event experiments ( $\delta_R : {}^\circ$ ,  $\delta_T : cm$ ).**

Sequence	01				02				03			
	Slow		Fast		Normal		Fast		Slow		Fast	
Method	$\delta_R$	$\delta_T$										
Line-Based	3.01	7.29	3.10	8.35	2.67	4.24	6.47	7.48	4.46	18.23	5.25	22.22
LS-Based	<b>2.19</b>	7.10	2.86	8.10	2.41	<b>3.67</b>	5.69	6.24	3.96	13.25	4.54	20.44
NNS-Based	3.11	7.82	6.95	8.54	2.95	5.18	6.94	7.38	5.95	17.15	6.12	23.52
Ours	2.24	<b>3.42</b>	<b>1.94</b>	<b>5.56</b>	<b>1.53</b>	4.47	<b>1.75</b>	<b>4.43</b>	<b>2.12</b>	<b>5.44</b>	<b>2.65</b>	<b>8.56</b>
Sequence	04				05				06			
	Normal		Fast		Normal		Fast		Normal		Fast	
Method	$\delta_R$	$\delta_T$										
Line-Based	1.39	2.51	2.95	7.19	0.68	7.44	<b>1.48</b>	8.33	<b>1.29</b>	4.26	<b>1.17</b>	5.74
LS-Based	1.36	2.94	2.49	6.78	<b>0.52</b>	5.54	1.68	6.05	1.58	3.99	2.00	4.58
NNS-Based	2.45	3.06	2.75	7.12	1.76	5.94	2.87	7.65	2.22	4.82	2.45	5.65
Ours	<b>1.35</b>	<b>2.56</b>	<b>1.67</b>	<b>4.86</b>	0.88	<b>3.62</b>	1.80	<b>4.30</b>	2.21	<b>3.76</b>	2.24	<b>4.14</b>

**Table 2: Pose tracking error of objects with curved edges in simulated event experiments ( $\delta_R : ^\circ$ ,  $\delta_T : cm$ ).**

Sequence	07						08						09					
	Original		Messy		Occlusion		Original		Messy		Occlusion		Original		Messy		Occlusion	
Method	$\delta_R$	$\delta_T$																
NNS-Based	4.12	6.44	2.89	8.92	3.54	8.24	3.51	7.47	3.87	10.11	4.87	<b>8.25</b>	2.71	7.24	4.21	9.68	3.91	<b>8.26</b>
Ours	<b>2.08</b>	<b>5.21</b>	<b>2.62</b>	<b>7.51</b>	<b>2.56</b>	<b>7.62</b>	<b>2.12</b>	<b>7.21</b>	<b>2.65</b>	<b>8.68</b>	<b>2.22</b>	8.87	<b>2.14</b>	<b>6.87</b>	<b>3.57</b>	<b>7.98</b>	<b>2.86</b>	8.47
10						11						12						
Sequence	Original		Messy		Occlusion		Original		Messy		Occlusion		Original		Messy		Occlusion	
Method	$\delta_R$	$\delta_T$																
NNS-Based	2.64	6.96	3.48	8.14	4.15	9.34	2.70	7.75	3.69	9.86	4.02	10.88	2.59	6.02	3.05	9.72	<b>3.62</b>	10.45
Ours	<b>2.01</b>	<b>6.35</b>	<b>3.15</b>	<b>7.54</b>	<b>3.04</b>	<b>8.15</b>	<b>2.04</b>	<b>6.51</b>	<b>2.98</b>	<b>7.20</b>	<b>3.05</b>	<b>7.05</b>	<b>2.35</b>	<b>5.72</b>	<b>2.96</b>	<b>8.93</b>	3.88	<b>8.52</b>

camera line-SLAM, we employ the fundamental technologies presented in their method for object pose tracking. (ii) LS-based. The least squares-based approach optimizes the pose and tracks the object by minimizing the distance between events and lines. (iii) NNS-based. Initially, employ our method to extract corners and edges, then establish the correlation between corners and edges using the nearest neighbor search approach, for the purpose of comparison with our optical flow-guided method. For objects with curved edges, we compare our method with NNS-based methods.

## 4.2 Simulated Event Experiments

Firstly, we conduct extensive experimental tests focusing on objects characterized by a notable abundance of lines. We utilize varying levels of object motion velocities, including slow, normal, and fast, to conduct a comparative analysis of methods. The pose tracking results are visually depicted in Figure 4(a), providing an intuitive display of the performance of the proposed methods. Based on the solved pose, the 3D point cloud of the object is projected onto the image plane while preserving its 2D edges. Moreover, we quantitatively evaluate the pose error of each method, as shown in Table 1. Overall, the proposed methods surpass the baselines in terms of accuracy. In sequence 03, there is a notable increase in errors across all methods. This can be attributed to the considerable distance between the object and the camera, the limited occurrence of events poses a challenge for pose tracking.

Next, we conduct tests on objects with curved edges. Since the edges of these objects are curved, the methods that rely on lines for pose tracking cannot be directly applied. However, our methods remain capable of consistently pose tracking of these objects, as depicted in Figure 4(b). To thoroughly validate the robustness of the methods, we also introduce challenging conditions such as messy backgrounds and object occlusions. Our methods both effectively address these challenges by establishing associations between corners and edges of objects. In Figure 4(b), the object edges and events exhibit a tight alignment utilizing the solved poses. We also quantitatively assess the pose tracking error of our methods, as demonstrated in Table 2. Our method yields smaller pose errors, as optical flow effectively guides the establishment of correlations between corners and curved edges. On the other hand, NNS-based methods solely seek out the nearest edge points,

which can potentially lead to inaccuracies in matching corners with curved edges.

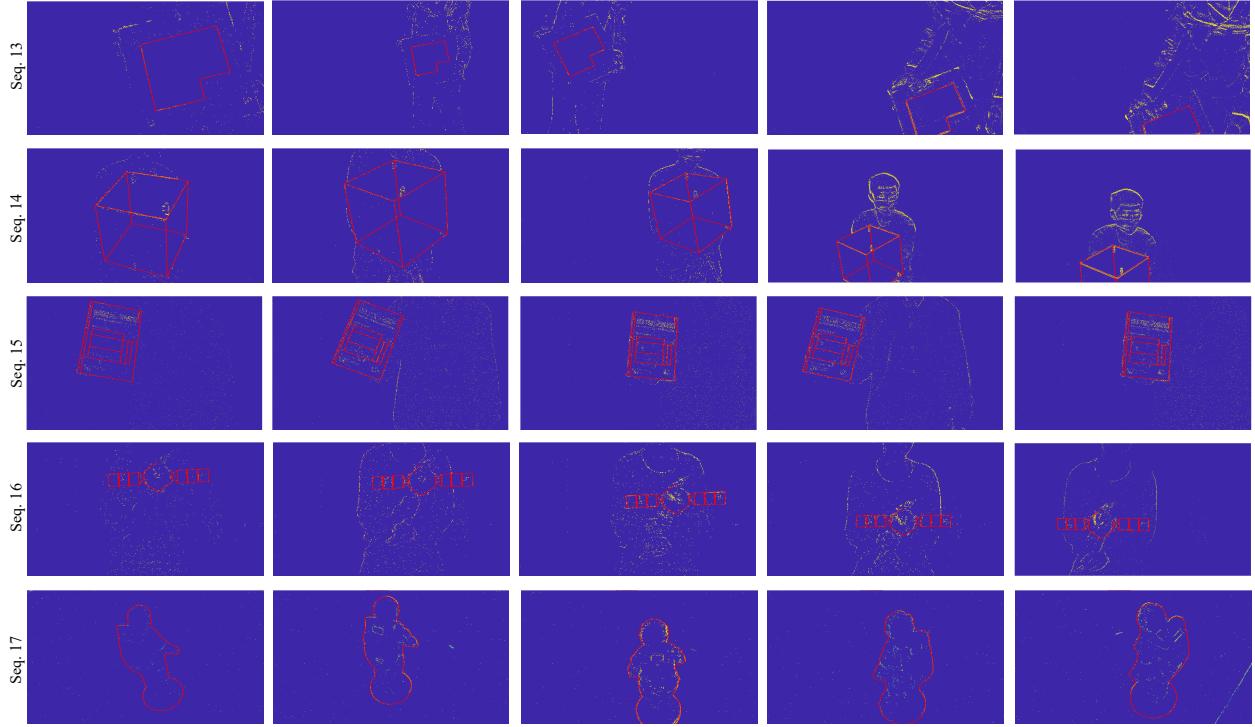
## 4.3 Real Event Experiments

Initially, we conduct real event experiments on objects with straight edges, including the planar pattern, cube, book, and satellite model. In comparative methods, partial lines of objects are employed for pose tracking. To ensure a fair comparison, our methods utilize corresponding partial lines as object edges for tracking. The tracking results of our method are illustrated in Figure 5. The color red denotes the lines of objects, which are projected onto TSs utilizing the solved poses obtained through our methods. The error statistics of pose tracking are shown in Table 3. Experimental results demonstrate that our methods outperform the compared methods in terms of accuracy.

We select the astronaut model as a representative subject for conducting object pose tracking experiments. In contrast to the previous experiment, this time the astronaut model remains stationary while the event camera is moved. This experimental setup enables us to test both the scenario of a stationary model with a moving camera and the scenario of a stationary camera with a moving model. We conduct statistical analysis on the pose tracking errors, as presented in Table 3. It is observed that our methods achieve higher accuracy. The pose tracking results are visually displayed in Figure 5. The projected edges of the astronaut model remain tightly aligned with events throughout the entire tracking process, providing an intuitive demonstration of the accuracy of our tracking methods.

## 5 CONCLUSION

This paper proposes an optical flow-guided 6DoF object pose tracking method using an event camera. Firstly, a hybrid feature extraction strategy is employed to detect corners from events and extract edges from the projected point cloud. Then, the optical flow of corners is calculated based on the spatio-temporal probability distribution of events. Subsequently, we establish associations between corners and edges along the direction of the optical flow. Furthermore, by minimizing the distance between edges and corners, an iterative pose refinement procedure is employed for continuous



**Figure 5: Pose tracking results of test objects in real-world experiments. The red represents the projected edges of their 3D models, which are reprojected onto TSs using the estimated poses obtained from our methods. The TSs are shown from top to bottom, corresponding to Sequence 13–17.**

**Table 3: Pose tracking error of objects in real event experiments ( $\delta_R : {}^\circ$ ,  $\delta_T : cm$ ).**

Sequence	13		14		15		16		17	
Type	Planar line		Nonplanar line		Planar line		Nonplanar line		Nonplanar curve	
Method	$\delta_R$	$\delta_T$	$\delta_R$	$\delta_T$	$\delta_R$	$\delta_T$	$\delta_R$	$\delta_T$	$\delta_R$	$\delta_T$
Line-Based	<b>1.16</b>	2.99	1.44	3.81	<b>1.35</b>	3.94	3.02	6.58	-	-
LS-Based	2.46	3.49	1.46	4.14	1.39	3.91	2.98	5.94	-	-
NNS-Based	3.46	5.52	2.35	5.90	2.21	6.45	4.05	7.12	7.78	11.25
Ours	1.78	<b>2.76</b>	<b>0.89</b>	<b>2.75</b>	1.52	<b>2.64</b>	<b>2.02</b>	<b>4.67</b>	<b>3.42</b>	<b>7.66</b>

tracking of objects. The experimental results of both simulated and real events indicate that our methods outperform the state-of-the-art event-based methods, particularly in challenging scenarios involving severe occlusion and messy backgrounds.

## ACKNOWLEDGMENTS

The authors would like to acknowledge the funding support provided by project 12372189 by the National Natural Science Foundation of China and project 2023JJ20045 by the Hunan Provincial Natural Science Foundation for Excellent Young Scholars.

## REFERENCES

- [1] Bogdan Bugaev, Anton Kryshchenko, and Roman Belov. 2018. Combining 3D Model Contour Energy and Keypoints for Object Tracking. In *European Conference on Computer Vision*. 53–69.
- [2] William Chamorro, Joan Solà, and Juan Andrade-Cetto. 2022. Event-based line SLAM in real-time. *IEEE Robotics and Automation Letters*. 7, 3 (2022), 8146–8153.
- [3] Haosheng Chen, David Suter, Qiangqiang Wu, and Hanzi Wang. 2020. End-to-end Learning of Object Motion Estimation from Retinal Events for Event-based Object Tracking. In *AAAI Conference on Artificial Intelligence*. 10534–10541.
- [4] Haosheng Chen, Pichao Wang, Fan Wang, Wei Tian, Lu Xiong, and Hao Li. 2022. EPro-PnP: Generalized End-to-End Probabilistic Perspective-N-Points for Monocular Object Pose Estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*. 2781–2790.
- [5] Haosheng Chen, Qiangqiang Wu, Yanjie Liang, Xinbo Gao, and Hanzi Wang. 2019. Asynchronous Tracking-by-Detection on Adaptive Time Surfaces for Event-based Object Tracking. In *ACM International Conference on Multimedia*. 473–481.
- [6] Jörg Conradt, Matthew Cook, Raphael Berner, Patrick Lichtsteiner, Rodney J Douglas, and Tobi Delbrück. 2009. A pencil balancing robot using a pair of AER dynamic vision sensors. In *IEEE International Symposium on Circuits and Systems*. 781–784.
- [7] Alberto Crivellaro and Vincent Lepetit. 2014. Robust 3D Tracking with Descriptor Fields. In *IEEE Conference on Computer Vision and Pattern Recognition*. 3414–3421.

- [8] David A. Drazen, Patrick Lichtsteiner, Philipp Dominik Häfliger, Tobi Delbrück, and Atle Jensen. 2011. Toward real-time particle tracking using an event-based dynamic vision sensor. *Experiments in Fluids* 51 (2011), 1465–1469.
- [9] Falanga Davide, Kleber Kevin, and Scaramuzza Davide. 2020. Dynamic obstacle avoidance for quadrotors with event cameras. *Science Robotics* 5, 40 (2020), eaaz9712.
- [10] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J. Davison, Jorg Conradt, Kostas Daniilidis, and Davide Scaramuzza. 2022. Event-Based Vision: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 1 (2022), 154–180.
- [11] Banglei Guan, Ji Zhao, Daniel Barath, and Friedrich Fraundorfer. 2023. Minimal Solvers for Relative Pose Estimation of Multi-Camera Systems using Affine Correspondences. *International Journal of Computer Vision* (2023), 1–22.
- [12] Banglei Guan, Ji Zhao, Zhang Li, Fang Sun, and Friedrich Fraundorfer. 2022. Relative Pose Estimation With a Single Affine Correspondence. *IEEE Transactions on Cybernetics* (2022), 1–12.
- [13] Chris Harris and Carl Stennett. 1990. RAPID - a video rate object tracker. In *British Machine Vision Conference*. 15.1–15.6.
- [14] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. 2012. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian Conference on Computer Vision*. 548–562.
- [15] Yuhang Hu, Shih-Chii Liu, and Tobi Delbrück. 2021. v2e: From Video Frames to Realistic DVS Events. In *IEEE Conference on Computer Vision and Pattern Recognition Workshop*. 1312–1321.
- [16] Kun Huang, Yifu Wang, and Laurent Kneip. 2021. Dynamic Event Camera Calibration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 7021–7028.
- [17] Tiejun Huang, Yajing Zheng, Zhaofei Yu, Rui Chen, Yuan Li, Ruiqin Xiong, Lei Ma, Junwei Zhao, Siwei Dong, Lin Zhu, et al. 2023. 1000× faster camera and machine vision with ordinary devices. *Engineering* 25 (2023), 110–119.
- [18] Abdul Mohsi Jawaid, Ethan Elms, Yasir Latif, and Tat-Jun Chin. 2022. Towards Bridging the Space Domain Gap for Satellite Pose Estimation using Event Sensing. In *IEEE International Conference on Robotics and Automation*. 11866–11873.
- [19] Max Krichenbauer, Goshiro Yamamoto, Takafumi Taketomi, Christian Sandor, and Hirokazu Kato. 2018. Augmented Reality versus Virtual Reality for 3D Object Manipulation. *IEEE Transactions on Visualization and Computer Graphics* 24, 2 (2018), 1038–1048.
- [20] Vincent Lepetit and Pascal V. Fua. 2005. Monocular Model-Based 3D Tracking of Rigid Objects: A Survey. *Foundations and Trends in Computer Graphics and Vision* 1 (2005).
- [21] Zibin Liu, Banglei Guan, Yang Shang, Zhenbao Yu, Yifei Bian, and Qifeng Yu. 2024. LECalib: Line-based event camera calibration. *Measurement* 235 (2024), 114900.
- [22] Zibin Liu, You Li, Chunhui Wang, Liang Liu, Banglei Guan, Yang Shang, and Qifeng Yu. 2024. AstroPose: Astronaut pose estimation using a monocular camera during extravehicular activities. *Science China Technological Sciences* (2024), 1–13.
- [23] Zibin Liu, Haibo Liu, Zhaokun Zhu, and Junyao Song. 2021. Relative pose estimation of uncooperative spacecraft using 2D-3D line correspondences. *Applied Optics* 60, 22 (2021), 6479–6486.
- [24] Zhixiang Min, Yiding Yang, and Enrique Dunn. 2020. VOLDOR: Visual Odometry From Log-Logistic Dense Optical Flow Residuals. In *IEEE Conference on Computer Vision and Pattern Recognition*. 4897–4908.
- [25] Zhenjiang Ni, Aude Bolopion, Joël Agnus, Ryad Benosman, and Stéphane Regnier. 2012. Asynchronous Event-Based Visual Shape Tracking for Stable Haptic Feedback in Microrobotics. *IEEE Transactions on Robotics* 28, 5 (2012), 1081–1089.
- [26] Zhenjiang Ni, Cécile Pacoret, Ryad Benosman, Siohoi Ieng, and Stéphane RÉGNIER\*. 2012. Asynchronous event-based high speed vision for microparticle tracking. *Journal of Microscopy* 245, 3 (2012), 236–244.
- [27] David Reverter Valeiras, Garrick Orchard, Sio-Hoi Ieng, and Ryad B Benosman. 2016. Neuromorphic event-based 3d pose estimation. *Frontiers in Neuroscience* 9 (2016), 522.
- [28] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. 2011. ORB: An efficient alternative to SIFT or SURF. In *IEEE International Conference on Computer Vision*. 2564–2571.
- [29] Henning Tjaden, Ulrich Schwanecke, Elmar Schömer, and Daniel Cremers. 2019. A Region-Based Gauss-Newton Approach to Real-Time Monocular Multiple Object Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41, 8 (2019), 1797–1812.
- [30] Tianren Zhang, Shangqi Guo, Tian Tan, Xiaolin Hu, and Feng Chen. 2023. Adjacency Constraint for Efficient Hierarchical Reinforcement Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 4 (2023), 4152–4166.
- [31] Tianwei Zhang, Huayan Zhang, Yang Li, Yoshihiko Nakamura, and Lei Zhang. 2020. FlowFusion: Dynamic Dense RGB-D SLAM Based on Optical Flow. In *IEEE International Conference on Robotics and Automation*. 7322–7328.
- [32] Junwei Zhao, Jianming Ye, Shiliang Shiliang, Zhaofei Yu, and Tiejun Huang. 2023. Recognizing High-Speed Moving Objects with Spike Camera. In *ACM International Conference on Multimedia*. 7657–7665.
- [33] Yi Zhou, Guillermo Gallego, and Shaojie Shen. 2021. Event-based stereo visual odometry. *IEEE Transactions on Robotics* 37, 5 (2021), 1433–1450.
- [34] Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis. 2017. Event-based feature tracking with probabilistic data association. In *IEEE International Conference on Robotics and Automation*. 4465–4470.
- [35] Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis. 2017. Event-Based Visual Inertial Odometry. In *IEEE Conference on Computer Vision and Pattern Recognition*. 5816–5824.