

Event-based Motion & Appearance Fusion for 6D Object Pose Tracking

Zhichao Li^{1,2,3}, Chiara Bartolozzi¹, Lorenzo Natale², Arren Glover¹

Abstract—Object pose tracking is a fundamental and essential task for robotics to perform tasks in the home and industrial settings. The most commonly used sensors to do so are RGB-D cameras, which can hit limitations in highly dynamic environments due to motion blur and frame-rate constraints. Event cameras have remarkable features such as high temporal resolution and low latency, which make them a potentially ideal vision sensors for object pose tracking at high speed. Even so, there are still only few works on 6D pose tracking with event cameras. In this work, we take advantage of the high temporal resolution and propose a method that uses both a propagation step fused with a pose correction strategy. Specifically, we use 6D object velocity obtained from event-based optical flow for pose propagation, after which, a template-based local pose correction module is utilized for pose correction. Our learning-free method has comparable performance to the state-of-the-art algorithms, and in some cases outperforms them for fast-moving objects. The results indicate the potential for using event cameras in highly-dynamic scenarios where the use of deep network approaches are limited by low update rates.

I. INTRODUCTION

The majority of visual 6D object pose estimation literature leverages RGB and RGB-D sensors. Representative approaches include model-based techniques and template-matching methods [1], [2], [3], [4], [5], [6], with deep learning methods having pushed the state-of-the-art performance in recent years [7], [8], [9], [10], [11], [12], [13]. Deep learning-based methods have typically higher accuracy, but they employ sophisticated network architectures and require the availability of large-scaled annotated datasets and significant computational resources for network training. Because they employ large architecture, they are demanding in terms of computation and therefore have low inference frequency, especially on resource constrained hardware. Moreover, the intrinsic limitations of RGB or RGB-D sensors, which output frames at fixed rates of 30-60 FPS, give rise to motion blur when object moves at high speeds. This effect can substantially degrade the performance of object pose trackers that depend on conventional frame-based sensors.

Event cameras are relatively novel vision sensors that have a different operating principles from the global or rolling shutters used in frame-based sensors [14]. Event cameras measure brightness change of each pixel independently, once the pixel brightness has changed significantly an event is asynchronously output with sub-millisecon latency and precision. Due to such an operating principle, event cameras

can capture object motion information with high frequency and overcome motion blur that occurs when integrating light for fixed time periods. However, correctly processing the asynchronous and discrete data encoded in events comes with its own challenges. The success of 6D object pose tracker depends critically on the effective utilization of events information and the accurate decoding motion information from events.

Only a handful of approaches have been proposed for 6D object pose tracking with event cameras. Optical flow information is almost inherently encoded in the spatio-temporal patterns formed from events and leveraging it to infer object motion presents a feasible solution [15], [16]. Otherwise, events have been decoded into line information, for objects primarily formed by line structures [17], [18], or template matching techniques have been adopted for less constrained objects [19]. Using a propagation and correction method has been proposed requiring a fusion with an RGB-D camera in both model-based [16] and data-driven neural networks [20].

We believe effectively leveraging event information is critical for a robust event-based 6D object pose tracker. In this work, we combine event-based optical flow together with event-based template matching to implement a high frequency 6D object pose tracker. Specifically, first we decode raw events into optical flow information which is used to predict object 6D velocity using a Kalman filter formulation. To counteract the inevitable error coming from velocity integration, a local object pose corrector is introduced. Specifically, after the pose propagation, multiple hypothesis pose templates are generated by pose perturbation using the object mesh. These hypothesis templates are compared to current sensor measurements accumulated into a spatial representation to refine the estimate directly in the object pose configuration. Our proposed method therefore reduces tracking errors accumulating over time. We decode events separately in two distinct pipelines and then organically combine them through propagation and correction. The effectiveness of this approach is validated on synthetic and real camera data in experiments.

Our contributions are the following:

- We propose the Event-camera only propagation and correction method for 6D object pose tracking, that fuses an optical flow and a template matching approach.
- We remove the need for depth measurements in the event-driven pipeline required in 6D velocity estimation [16].
- We perform a comparison of event-camera pose tracking against recent state-of-the-art RGB-D deep learn-

¹Event-driven Perception for Robotics, Istituto Italiano di Tecnologia, Italy {first.last}@iit.it

²Humanoid Sensing and Perception, Istituto Italiano di Tecnologia, Italy {first.last}@iit.it

³University of Genoa, Genoa, Italy

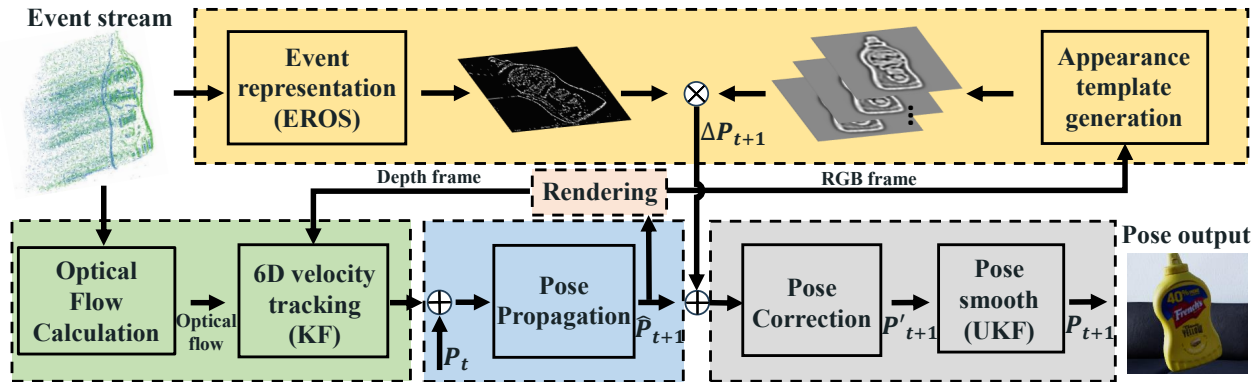


Fig. 1: Overview of the proposed pipeline for 6D object pose tracking with an event camera. Events captured by an event camera are utilized to update a velocity independent representation (EROS) [19], simultaneously with event-based optical flow extraction. The 6D object pose is propagated based on the 6D object velocity estimated from the optical flow measurement. Several appearance templates are generated based on the propagated pose but with additional small pose perturbations, from which edges are extracted. The propagated 6D pose is corrected based on the best matched template compared to the EROS representation. Finally, an Unscented Kalman filter is adopted to further smooth the corrected pose over time.

ing methods (including FoundationPose [13]) obtaining comparable or better results for fast-moving object scenarios.

II. RELATED WORK

Visual 6D object pose estimation involves estimating both the position and orientation of the object relative to the camera; 6D object pose tracking estimates and updates the object 6D pose over time. Previously, probabilistic modeling methods were used for solving this task [1], [2], [3]. Object pose can also be tracked, for example, by matching various templates, such as color histograms, and contours [4], [5], [6].

An increasing number of approaches have been proposed to address 6D object pose estimation using deep neural networks. The data-driven approaches were more effective at extracting features, enabling the generation of more efficient templates, as well as extraction of more informative key features or points for object pose tracking [7], [8], [9], [10]. Relying on a large amount of training data, deep neural networks can also learn and predict a pose variation between two continuous frames [11], [12], the object pose can be tracked by accumulating such pose variations over the time. With the combination of large-scale datasets and the high performance of transformers, end-to-end modeling for object pose tracking has become viable [13].

Approaches using RGB or RGB-D cameras can have performance degradation due to image blur for high-speed objects. In contrast, event cameras are promising sensors for 6D object pose tracking as they are largely immune to image blur, but instead lack information about color and spatial textures. One potential approach for 6D object pose tracking was to combine both RGB and event cameras using optimization and alignment techniques [21] or with cascading Kalman filters [22]. Using only event cameras [17], [18]

propose to extract straight line features which are tracked over time. Another work, called EDOPT [19], proposes a velocity-independent event representation method named EROS, which ensures consistent edge appearance for fast and slow motions. Template matching is used with EROS and object renders for pose tracking. In [23] a brightness increment image strategy based on motion interpolation is introduced. It can reconstruct brightness increments with events according to the object’s motion, then align the 3D object model with the incremented image directly for 6D object pose tracking. Among the works that are close to our work, ROFT [24] is an optical flow aided object pose tracker. It utilizes frame-based optical flow to implement an object 6D pose velocity tracker. The object pose is propagated with 6D velocity and corrected using a deep learning based object pose estimator DOPE [25]. Another work adopts a similar structure, but it utilizes event-based optical flow instead of frame-based optical flow for object 6D velocity tracking [16]. In this work, we combine EDOPT [19] strategy to make local pose correction using rendered object templates, and object 6D velocity tracking from event-based optical flow [16], which can provide motion information to the tracker. By using the rendered depth, the depth information is not required from cameras for 6D object velocity tracking. In this way, we build the structure of motion prediction and pose correction for 6D object pose tracking with an event camera.

III. METHOD

A. Overview

Given the 3D model, the initial 6D object pose of a target object O_{bj} and events stream $E : \{x_i, y_i, t_i\}$ from an event camera. Our goal is to track the 6D object pose of O_{bj} with respect to the event camera’s reference frame. Specifically, we want to continuously estimate 6D transformation $\mathbf{P} = \{t, \mathbf{R}\}$ that relates O_{bj} to the event camera, where $t \in \mathbb{R}^3$

and $\mathbf{R} \in \mathbb{R}^{3 \times 3}$ are the translation and rotation components. To solve this problem, we leverage the idea of propagation and correction. Specifically, an event-based optical flow 6D object velocity tracker is utilized to propagate the object 6D motion. Then in order to eliminate pose error from pose propagation, a template-based pose correction step using events stream is executed. In the end, to smooth the tracked pose, an Unscented Kalman filter is applied before output the final pose \mathbf{P}_{t+1} . The pipeline is shown in Fig. 1.

B. Event optical flow based velocity tracking

1) *Event-based optical flow*: In this work, the event-based optical flow is calculated by examining spatio-temporal relationship for each event within a region of interest (*RoI*). Then apply a registration technique to obtain the event-based optical flow [16]. Specifically, when an object moves in 3D space, one single point on the object crosses the 2D image plane and triggers events. Assume three events $e_i = \{\mathbf{x}_i, t_i\}$, e_j and e_k are triggered by the motion of the same point and in a very short time period. They follow spatio-temporal constraint:

$$\frac{\mathbf{x}_j - \mathbf{x}_i}{t_j - t_i} = \frac{\mathbf{x}_k - \mathbf{x}_j}{t_k - t_j + \tau} \quad (1)$$

\mathbf{x}_i , \mathbf{x}_j and \mathbf{x}_k are 2D pixel coordinates of triggered events e_i , e_j and e_k . τ is the parameter that represents the tolerance in time. These three events are spatio-temporal triplet matched events. The optical flow at pixel coordinate \mathbf{x}_i can be calculated as:

$$\mathbf{F}(\mathbf{x}_i) = (\mathbf{x}_k - \mathbf{x}_i) / (t_k - t_i) \quad (2)$$

However, single triplet matched events is unable to guarantee events e_i , e_j and e_k are triggered by the motion of the same object point. Incorrect triplet matched events from wrong matching, background noise and sensor noise will introduce errors in the optical flow computation. To eliminate this error, we follow the work [16] using spatio-temporal registration strategy to improve the event optical flow from triplet matches. By splitting the image space into small grids. Search triplet matched events within each grid if events are triggered. Each grid is defined as a *RoI*. In each *RoI*, all searched triplet events are used for estimating a single optical flow vector \mathbf{F} using the following cost function:

$$\mathbf{F} = \arg \min_{\mathbf{F} \in \mathcal{M}} \left(\sum_{i=1}^n \min_{\mathbf{F}_{ic} \in M_i} |\mathbf{F} - \mathbf{F}_{ic}| \right) \quad (3)$$

\mathbf{F} denotes the target optical flow, which minimizes the cost function. \mathcal{M} represents the set of candidate flow vectors computed from the identified triplets using Eq. 2, within a region of interest (*RoI*) of n events. For a single event e_i , the subset $M_i \subset \mathcal{M}$ contains all candidate flow vectors associated with e_i . With the *RoI* and perform spatio-temporal registration more candidate triplet matches are incorporated, thereby enabling effective suppression of background and the sensor-induced noise.

2) *6D Object velocity estimation with optical flow*: Given event stream $E : \{x_i, y_i, t_i\}$ after decoded them as event optical flow \mathbf{F}_i . We estimate the 6D object velocity \mathbf{V}_t from \mathbf{F}_i using a Kalman filter as [16].

$$\mathbf{V}_t = [\mathbf{v}_{ot}, \boldsymbol{\omega}_{ot}] \quad (4)$$

Where, $\mathbf{v}_{ot} \in \mathbb{R}^3$ denotes the linear velocity of a point that momentarily coincides with the camera's origin and moves as though it were rigidly attached to the object. $\boldsymbol{\omega}_{ot} \in \mathbb{R}^3$ represents the object's angular velocity relative to that same point. Similarly to [16] a motion model for the Kalman filter is defined:

$$\mathbf{V}_{t+1} = \rho \mathbf{V}_t + \mathbf{w}, \quad (5)$$

where the propagation noise, $\mathbf{w} \sim N(0, \text{diag}(\mathbf{Q}_v, \mathbf{Q}_w))$, follows a zero-mean Gaussian distribution with covariance $\mathbf{Q}_v \in \mathbb{R}^{3 \times 3}$ and $\mathbf{Q}_w \in \mathbb{R}^{3 \times 3}$ associated to the linear and angular velocity, respectively. ρ is a decay parameter, following [16] we set it to 0.5.

The measurement model of Kalman filter is built as:

$$\mathbf{y} = (\mathbf{F}(u, v) - \hat{\mathbf{F}}(u, v)) \Delta_T \quad (6)$$

where $\mathbf{F}(u, v)$ is event optical flow at pixel coordinate (u, v) and $\hat{\mathbf{F}}(u, v) = \mathcal{J}(u, v) \mathbf{V}_t$, represents the mapped 2D pixel velocity from estimated 6D object velocity using the interaction matrix $\mathcal{J}(u, v)$ [16]. Δ_T represents the time interval between two consecutive updates of the object velocity tracker. Different from work [16], which uses depth captured from the camera, in this work, depth is obtained by rendering using the tracked 6D object pose.

3) *6D Object pose propagation*: Given the tracked 6D object velocity $\mathbf{V}_t(\mathbf{v}_{ot}, \boldsymbol{\omega}_{ot})$ from Kalman filter, the 6D object pose $\mathbf{P}_t = \{\mathbf{t}_t, \mathbf{R}_t\}$ can be propagated as $\hat{\mathbf{P}}_{t+1}$. $\mathbf{V}_t(\mathbf{v}_{ot}, \boldsymbol{\omega}_{ot})$ is splitted into translation and rotation components and used in the Eq. 7 separately.

$$\begin{aligned} \hat{\mathbf{t}}_{t+1} &= \mathbf{t}_t + \mathbf{v}_{ot} \cdot \Delta t \\ \hat{\mathbf{q}}_{t+1} &= \cos\left(\|\boldsymbol{\omega}_{ot} \frac{\Delta t}{2}\|\right) \mathbf{I}_4 + \frac{\sin\left(\|\boldsymbol{\omega}_{ot} \frac{\Delta t}{2}\|\right)}{\|\boldsymbol{\omega}_{ot}\|} \Omega(\mathbf{q}_t) \end{aligned} \quad (7)$$

For propagating the rotation part, we process it in quaternion format. \mathbf{q}_t is quaternion format of the rotation component \mathbf{R}_t . And $\Omega(\mathbf{q}_t)$ follows [26]:

$$\Omega(\mathbf{q}_t) = \begin{bmatrix} -\omega_x & -\omega_y & -\omega_z & 0 \\ \omega_x & 0 & -\omega_z & \omega_y \\ \omega_y & \omega_z & 0 & -\omega_x \\ \omega_z & -\omega_y & \omega_x & 0 \end{bmatrix} \quad (8)$$

C. Local pose correction

Following Eq. 7, inaccuracies in the estimated 6D object velocity \mathbf{V}_t lead to errors in the tracked pose. Over time, these errors accumulate, eventually leading to tracker failures. To solve this problem we introduce a local correction strategy. After propagating the 6D object pose using the estimated velocity, the 6D object pose is corrected by a template-based local pose corrector, inspired from [19].

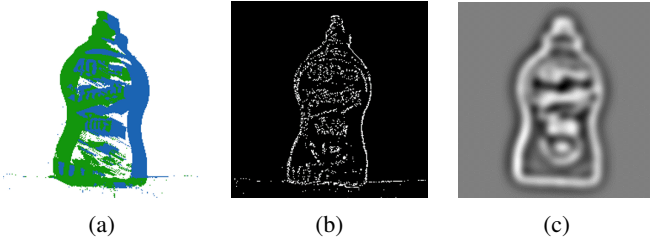


Fig. 2: Visualization of (a) raw events, (b) EROS representation, and (c) template generated using the model rendered at a perpetuated pose with edge gradient extraction.

1) *Event representation*: To allow for efficiently extracting object appearance, including outlines and edges, whenever the object is moving, the asynchronous stream is converted to EROS [19]. Different from raw events (Fig. 2a), EROS creates an image-like representation where each pixel encodes the presence of a light change (see Fig. 2b). This velocity-independent representation is updated asynchronously, with each event producing a small, localized adjustment. Specifically, we define an array \mathcal{E} with the same size as the image plane, where each element takes a value in the range $[0.0, 1.0]$. When a new event is triggered at (u, v) , we set $\mathcal{E}(u, v) = 1.0$, and decay the values of all neighboring pixels in the region $\mathcal{N}(u, v)$ by a factor λ .

$$\begin{aligned} \mathcal{E}(x, y) &= \lambda \times \mathcal{E}(x, y) \\ \forall (x, y) \in \mathcal{N}(u, v) \end{aligned} \quad (9)$$

(x, y) represents the pixel within the neighborhood of pixel (u, v) .

2) *Gradient visual expectation*: We generate the expected object image gradients for the propagated object pose \hat{P}_{t+1} . By projecting the known 3D mesh onto the 2D image plane using the camera's intrinsic parameters, we render a synthetic image that represents how the object should appear from that viewpoint. This projection simulates the object's outline and surface details as seen by the camera. Next, to extract meaningful feature from this synthetic image, following [19] we apply a Sobel filter, which highlights the edges and gradients-areas where the image intensity changes sharply and could trigger events (see Fig. 2c). The resulting expectation map serves as an appearance template against the observation in event representation of the object. A Difference of Gaussians filter is applied to broaden edges, allowing some matching tolerance, produce a gradient to differentiate strong and weak matches, and to produce a negative matching region to reject badly matching observations.

3) *Template-based pose correction*: We aim to predict how the current object pose could have evolved such that it would generate the events observed by the event camera. Since an event camera has no integration period we assume small pose changes between two consecutive estimations. Specifically, we assume that the templates resulting from the projection of two consecutive poses have no more than one pixel of displacement.

The relationship between changes in the object pose

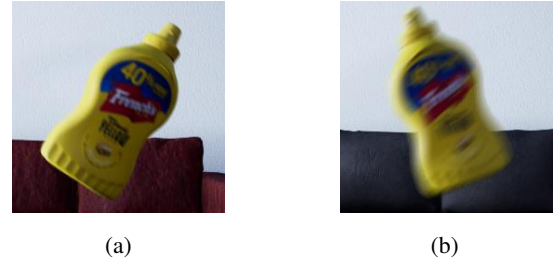


Fig. 3: Snapshots of objects in synthetic data sequences: (a) regular motion sample, and (b) fast motion sample.

$\Delta\{x, y, z, \alpha, \beta, \gamma\}$ (specific for translations) and changes in the image plane $\Delta\{u, v\}$ (pixel motion) can be captured by the image Jacobian, which describes how each degree of freedom (move forward/backward, left/right, up/down along the axes) translates into pixel shifts in the image plane.

$$\begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \begin{bmatrix} -\frac{f_x}{z} & 0 & \frac{u-c_x}{z} \\ 0 & -\frac{f_y}{z} & \frac{v-c_y}{z} \end{bmatrix} \begin{bmatrix} \Delta x & \Delta y & \Delta z \end{bmatrix}^T \quad (10)$$

We perturb ΔP_{t+1} for the propagated object pose \hat{P}_{t+1} such that the resulting motion on the image plane is at Δp pixels (in this work Δp set as one pixel). By inverting the Jacobian and decoupling each dimension, we derive the amount of translation along x or y axis for pose corresponds to Δp pixel image motion:

$$\left\{ \pm \Delta x = \frac{\mp z \Delta p}{f_x} \quad \pm \Delta y = \frac{\mp z \Delta p}{f_y} \quad \pm \Delta z = \frac{\pm z \Delta p}{\bar{p}} \right\} \quad (11)$$

$\bar{p} = \|(\Delta u, \Delta v)\|$. For rotations, assume the object rotates with a small rotation degree θ (in this work θ is set as 0.5°). Then

$$\{\pm \Delta \alpha = \pm \theta \quad \pm \Delta \beta = \pm \theta \quad \pm \Delta \gamma = \pm \theta\} \quad (12)$$

These equations generate 12 perturbed poses (positive and negative directions along each motion axis). If considering an additional null motion hypothesis, these give a total 13 hypothesis poses. The hypothesis pose is defined as $\{H_i | i \in [1, \dots, 13]\}$.

To reduce the pose error accumulated by pose propagation in eq.7, 13 hypothesis pose H_i is generated based on propagated object pose \hat{P}_{t+1} , and compare them with current object pose observation EROS O_{t+1} from the camera. The hypothesis pose H_i with the highest similarity with the observation O_{t+1} is selected and the pose \hat{P}_{t+1} is refined as:

$$P'_{t+1} = \hat{P}_{t+1} + \underset{\Delta P_{t+1}}{\operatorname{argmax}} \{H_i \cdot O_{t+1}\}, i \in [1, \dots, 13]. \quad (13)$$

D. Pose smooth with UKF

We employ an Unscented Kalman Filter (UKF) to smooth the refined object pose P'_{t+1} and output the final pose P_{t+1} . The UKF effectively handles the non-linearities inherent in the pose dynamics by propagating a set of sigma points through the system model. It improves the temporal



Fig. 4: Dual-camera setup. The left device is an event camera with resolution 1280×720 . The right RGB-D camera is a RealSense D415, only used for evaluating base-line algorithms and doesn't form an input to our algorithm.

consistency of the refined pose while reducing the effects of perturbations in the appearance template pose correction, as well as mitigating pose observation noise from the camera.

IV. EXPERIMENT

To evaluate the performance of our proposed method, we execute experiments on synthetic and real world data from event cameras and compare with multiple algorithms including RGB/RGBD frame-based, events-based and hybrid methods. We assume a static camera setting, and the object in the scene is moving. Under this assumption, cluttered environments with background motion are uncommon, as static background structures do not generate events and do not interfere with the proposed method. In the experiments, we set both event-based optical flow update frequency and event representation EROS update rate at 500Hz.

A. Testing data

Two datasets sequences are evaluated in the experiments. The first is a synthetic dataset from [16]. It is generated using a simulation engine, while the second dataset is acquired with a dual-camera setup, consisting of an event camera (1280×720 resolution) and an Intel RealSense D415, shown in Fig. 4. Several objects from YCB object sets, such as cracker_box, mustard_bottle, potted_meat_can, etc, are used for data collection.

For synthetic data sequences, the data includes triggered events, RGB-D frames recorded at 500 FPS, and corresponding ground-truth object poses also provided at 500 Hz. To simulate the frames with motion blurs as collected from a real camera, frames are averaged and achieves 60 FPS output, Fig. 3. Moreover, in order to simulate the real depth captured by the RealSense, a Gaussian noise with 25 mm standard deviation is added. For real-world data sequences, objects are manipulated by a human operator. The data contains events induced by object motions, as well as RGB and depth frames captured at 60 FPS by the RealSense. In these data sequences, ground-truth object poses are unavailable.

B. Metrics

In the experiment, the tracking performance for synthetic data sequences is evaluated quantitatively using root mean square error (RMSE) with respect to the ground-truth pose.

Positional errors are quantified with the Euclidean distance, whereas rotational errors are expressed as angular vectors. The standard deviation (σ) is also calculated for evaluating the smoothness of the tracking trajectories. For the real-world sequences, qualitative comparison results are reported. The estimated poses are examined by rendering the object according to the tracked pose and overlaying it onto the captured events stream or RGB frames.

C. Comparison against RGB/RGBD frame-based model

We evaluate the performance of our algorithm and compared it with several frame-based algorithms including ROFT [24], se(3)-TrackNet [12] and FoundationPose [13] on synthetic data sequences. ROFT is frame-based optical flow aided method with dual filter structure, it combines the 6D object velocity calculated from optical flow with the pose estimated by the learning-based pose estimator DOPE [25]. While se(3)-TrackNet and FoundationPose are both learning based methods. The object motion sequences can be split into *regular* and *fast* groups by pixel motion speeds. From Table I, algorithms that use RGBD modality outperform our method in *regular* motion speed group. When the motion speed is not very high, the captured frames show little or almost no noticeable motion blur, higher tracking accuracies demonstrate the effectiveness of learning-based methods. However, in the *fast* motion group, high speed motions cause significant frame blurs, which reduces the accuracy of frame-based optical flow and also affects the effectiveness of neural network-based feature extractors. In the end, it degrades the performance of ROFT and se(3)-TrackNet. FoundationPose adopts a transformer architecture, which has stronger and stable feature extraction capability. Therefore, the impact of the high speed motion on the performance less significant in this experiment. The advantage of our method becomes clear in the high speed motion group and achieves comparable results to FoundationPose. This is because high speed helps trigger a sufficient number of events for optical flow extraction, which further improves the accuracy of 6D object velocity tracker.

D. Comparison against hybrid method

We also compared against a hybrid method [16], which employs both events and frames: it tracks 6D object velocity from event optical flow and fuses it with a learning-based object pose estimator DOPE [25], the same algorithm used in ROFT [24]. Our method outperforms this hybrid method in tracking accuracy, except for the tomato sequence with normal motion speed.

This performance difference can be attributed to the local pose correction in our method and the global object pose estimator DOPE. Our local pose correction relies on the velocity-independent representation, it will not be affected by motion blur. In contrast, DOPE relies on RGB frames, which are susceptible to motion blur. This comparison demonstrates the effectiveness of our local pose correction strategy.

Sequence		mustard_regular		potted_regular		cracker_regular		tomato_regular	
Methods	Modality	$e_p(\sigma)$ (cm)	$e_r(\sigma)$ (deg)	$e_p(\sigma)$ (cm)	$e_r(\sigma)$ (deg)	$e_p(\sigma)$ (cm)	$e_r(\sigma)$ (deg)	$e_p(\sigma)$ (cm)	$e_r(\sigma)$ (deg)
ROFT [24]	RGBD	1.16(0.59)	4.88(1.62)	3.41(1.44)	19.81(8.78)	1.54(0.40)	2.61(1.04)	1.77(0.63)	9.29(3.87)
se(3)-TrackNet [12]	RGBD	0.77(0.09)	2.78(1.11)	0.84(0.34)	11.37(5.81)	0.86(0.52)	5.20(3.32)	28.27(8.81)	34.90(9.41)
FoundationPose [13]	RGBD	0.86(0.40)	6.29(3.36)	0.56(0.18)	1.72(0.75)	0.40(0.15)	0.94(0.35)	0.48(0.21)	1.53(0.70)
Hybrid method [16]	RGBDE	1.61(0.68)	5.78(1.75)	4.42(3.31)	12.96(7.99)	1.95(1.09)	7.30(3.66)	2.13(0.84)	9.25(4.08)
EDOPT [19]	Events	1.93(1.04)	14.17(8.40)	15.55(8.90)	21.89(10.57)	1.10(0.59)	4.11(2.31)	13.15(7.33)	71.96(36.60)
EDOPT_Vel	Events	10.04(7.08)	29.92(15.60)	9.69(6.51)	17.56(9.44)	1.26(0.66)	3.49(1.44)	25.65(8.80)	68.74(19.63)
Ours (only Vel)	Events	8.34(2.80)	25.87(12.48)	7.77(3.13)	14.01(4.38)	5.73(2.02)	20.82(6.40)	4.01(1.12)	21.62(4.76)
Ours (w/o ukf)	Events	1.79(1.03)	9.53(5.90)	1.68(0.97)	11.92(6.68)	0.98(0.49)	3.01(1.29)	8.26(3.87)	61.15(29.76)
Ours (ukf)	Events	1.66(0.94)	7.84(4.40)	1.90(0.80)	14.14(6.65)	1.13(0.61)	3.32(1.50)	7.66(3.10)	58.36(24.38)
Sequence		mustard_fast		potted_fast		cracker_fast		tomato_fast	
ROFT [24]	RGBD	4.95(2.64)	21.49(10.95)	74.12(35.59)	80.28(38.73)	6.76(3.21)	27.73(17.51)	3.25(1.41)	17.40(7.58)
se(3)-TrackNet [12]	RGBD	89.82(47.29)	87.64(49.20)	15.54(6.37)	16.24(6.35)	24.13(9.62)	29.90(11.86)	26.04(9.67)	22.48(8.40)
FoundationPose [13]	RGBD	1.49(0.67)	12.69(7.36)	1.04(0.44)	6.33(3.99)	1.42(0.57)	3.70(2.61)	0.89(0.50)	4.51(3.11)
Hybrid method [16]	RGBDE	4.66(2.29)	11.90(4.63)	4.90(2.13)	14.82(6.09)	3.30(1.93)	9.13(4.96)	2.89(1.27)	10.38(4.35)
EDOPT [19]	Events	26.31(9.45)	46.77(22.41)	12.03(5.79)	20.03(8.58)	24.78(8.99)	33.95(13.30)	19.25(11.74)	31.79(18.11)
EDOPT_Vel	Events	38.24(13.18)	82.40(30.43)	2.22(1.01)	12.12(5.34)	33.76(14.72)	27.88(11.97)	75.64(53.78)	42.25(21.18)
Ours (only Vel)	Events	11.18(5.34)	13.23(5.56)	12.77(5.76)	15.67(6.88)	16.82(7.58)	24.43(13.28)	7.60(2.57)	24.67(13.69)
Ours(w/o ukf)	Events	2.54(1.71)	15.50(9.27)	1.59(0.70)	14.12(6.53)	1.12(0.54)	3.06(1.29)	1.51(0.74)	5.68(2.20)
Ours(ukf)	Events	1.14(0.57)	11.53(5.68)	1.30(0.64)	13.61(6.44)	0.90(0.47)	3.05(1.23)	1.74(0.84)	4.68(1.89)

TABLE I: 6D object pose tracking performance of multiple algorithms using different information modalities on synthetic object motion sequences. The results of translations and rotations are measured by root mean square error (RMSE) and standard deviation (σ). EDOPT_Vel means that the EDOPT algorithm integrates the velocity calculated from two recent continuously tracked object poses, and Ours (only Vel) refers to propagating the pose using only 6D velocity.

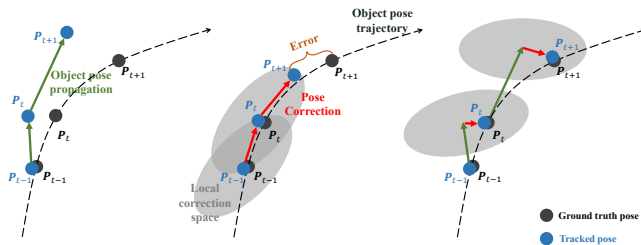


Fig. 5: Propagation and local correction strategy. The (left) figure shows error accumulation if velocity integration is used alone. The (middle) figure demonstrates a failure case of perturbation correction, when the noise or discretisation error allow the true pose to leave the perturbation region without correction. The (right) figure shows the proposed method combining both approaches, enabling the pose variation between the propagated pose and the ground truth pose to more likely remain within the bounds of the local correction space, and conversely correct integration error.

E. Comparison against Event-based model

Finally, we compared our proposed method with the event-based method EDOPT [19]. EDOPT relies on the velocity-independent representation EROS and produces templates by applying perturbations. In comparison with our method, EDOPT relies solely on a local correction strategy, without incorporating a motion model from motion cues. When the object moves at higher speed, the local pose correction space may fail to cover the ground truth pose, leading to tracking failure. As shown in Table I, EDOPT tracks the *mustard_regular* and *cracker_regular* sequences, but fails the remaining sequences. In contrast, our method achieves superior tracking performance in terms of tracking accuracy across all evaluated sequences.

F. Real camera data testing

In the real camera data testing, we compare our method with EDOPT [19] and FoundationPose [13]. As the testing sequences do not provide ground truth object poses, we report only qualitative results for the evaluated algorithms. In particular, the tracked object pose is visualized by rendering the object using its mesh onto the collected events streams for event-based algorithms and onto the RGB frames. The performance of the algorithms is evaluated by assessing how well the rendered object aligns with the events streams over a short temporal window or with individual RGB frames.

As shown in Fig. 6, the rendered objects align well with the event streams for our algorithm and with the RGB frames for FoundationPose over time. The performance of the two algorithms is comparable on the testing sequences. In contrast, the rendered object aligns well with events stream at the beginning, but misalignments become evident over time for EDOPT. Those results are consistent with the findings from the experiments on synthetic data sequences.

G. Ablation study

Our method integrates object velocity from event-based optical flow with a Kalman filter and a local pose correction module based on templates. We conduct ablation studies on synthetic data sequences to assess components of our proposed method. All results are obtained by averaging over all sequences and reported in terms of RMSE and σ as mentioned in Sec. IV-B.

1) *Integration of 6D object velocity and local pose correction*: We evaluate the effectiveness of each module in our method. Fig. 5 illustrates the principle of each module. In Table. IIa, we report the results of using only local correction, using only 6D object velocity propagation, and our proposed method, which combines the two modules. The results indicate that, with our method, tracking errors

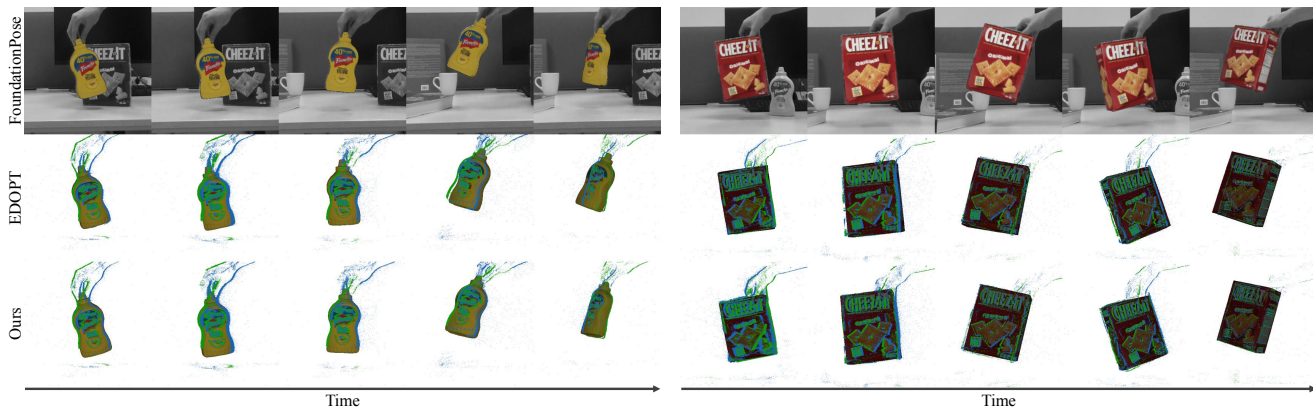


Fig. 6: Qualitative results of mustard_bottle and cracker_box from real camera data.

(a) Tracking performance with different modules.

Method	$e_p(\sigma)$ (cm)	$e_r(\sigma)$ (deg)
Local correction	14.26(6.73)	30.58(15.04)
Velocity propagation	9.28(3.79)	20.04(8.43)
Ours	2.18(1.00)	14.57(6.52)

(b) Tracking performance using different motion information.

Method	$e_p(\sigma)$ (cm)	$e_r(\sigma)$ (deg)
Local correction 6D object velocity	24.56(13.22)	35.55(14.38)
Optical flow based 6D object velocity	2.18(1.00)	14.57(6.52)

(c) Performance improvement achieved by introducing UKF.

Method	$e_p(\sigma)$ (cm)	$e_r(\sigma)$ (deg)
w/o UKF	2.43(1.26)	15.50(7.87)
w UKF	2.18(1.00)	14.57(6.52)

TABLE II: Ablation study.

(RMSE) for both translation and rotation decrease substantially. Additionally, the tracking trajectories are smoother, as the standard deviation (σ) is significantly reduced.

2) *Event optical flow based object velocity*: In the Table. IIb We evaluate the integration of 6D object velocity tracker using event-based optical flow. We compare it with the usage of 6D object velocity calculated from two continuous 6D object poses tracked by local correction. By integrating event-based optical flow based 6D object velocity tracker, the system achieves lower error and better performance compared to using 6D object velocity derived from two continuous poses. This demonstrates the effectiveness of integrating event-based optical flow 6D object velocity tracker.

3) *Integrate Unscented Kalman filter*: We also evaluate the effectiveness of integrating the Unscented Kalman filter. The filter, applied before the output of the tracked pose, aims to further smooth the pose trajectories. From Table. IIc, although the reduction of the standard deviation σ of the tracking error for both translation and rotation is modest, it nonetheless represents a positive improvement. Additionally, the tracking error (RMSE) is also slightly reduced.

V. DISCUSSION

FoundationPose demonstrates stable and accurate performance on the test sequences, particularly when there is almost no image blur; however, achieving high-frequency operation requires a high-performance GPU. Additionally, its performance is ultimately constrained to the sensor’s output rate (i.e. 30-60 Hz). In contrast, our method is not affected by image blur, benefiting from the high temporal resolution of event cameras.

We currently do not have an full on-line pipeline that would take advantage of the event-camera for real-time 6-DoF object pose tracking. However, the components of our pipeline were measured independently: velocity estimation (optical flow and KF) takes on average 2 ms, template pose correction takes on average 5 ms, the pose propagation and final UKF take less than 2 ms. Therefore we expect an on-line pipeline should be realized at approximately 110 Hz, while also taking advantage of event camera properties of removed motion blur and high dynamic range.

While our approach relies on an initial pose, in this work we first focus on the tracking algorithm that is most suited to be solved with event cameras due to their nature of measuring appearance change. For an eventual complete pipeline, a dedicated pose estimator could be introduced (e.g. as in [22]) to provide the initial pose and help with failure recovery, however no robust object pose detection networks have been proposed for event cameras as of the time of this publication.

A real-world 6D object pose dataset captured with event cameras, particularly including fast moving objects and annotated with ground truth poses, would be highly valuable for advancing research on 6D object pose tracking or estimation using event cameras.

VI. CONCLUSIONS

In this work, we introduce the event-camera-only method using propagation and correction for 6D object pose tracking, which combines event-based optical flow and a template matching approach. By rendering depth using the tracked 6D object pose, our method removes the need for explicit

depth measurements required in 6D velocity estimation. We demonstrate the effectiveness of our approach through comparisons on both synthetic and real camera data sequences against existing event-camera pose trackers and state-of-the-art frame-based deep learning methods.

REFERENCES

- [1] M. Wüthrich, P. Pastor, M. Kalakrishnan, J. Bohg, and S. Schaal, "Probabilistic Object Tracking using a Range Camera," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 3195–3202.
- [2] J. Issac, M. Wüthrich, C. G. Cifuentes, J. Bohg, S. Trimpe, and S. Schaal, "Depth-based Object Tracking Using a Robust Gaussian Filter," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 608–615.
- [3] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox, "PoseRBPF: A Rao–Blackwellized Particle Filter for 6-D Object Pose Tracking," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1328–1342, 2021.
- [4] H. Tjaden, U. Schwanecke, and E. Schomer, "Real-Time Monocular Pose Estimation of 3D Objects Using Temporally Consistent Local Color Histograms," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 124–132.
- [5] N. Payet and S. Todorovic, "From Contours to 3D Object Detection and Pose Estimation," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 983–990.
- [6] S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua, and N. Navab, "Dominant Orientation Templates for Real-Time Detection of Texture-Less Objects," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 2257–2264.
- [7] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes," in *Asian conference on computer vision*. Springer, 2012, pp. 548–562.
- [8] V. N. Nguyen, Y. Hu, Y. Xiao, M. Salzmann, and V. Lepetit, "Templates for 3D Object Pose Estimation Revisited: Generalization to New Objects and Robustness to Occlusions," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 6771–6780.
- [9] Y. Lin, J. Tremblay, S. Tyree, P. A. Vela, and S. Birchfield, "Single-Stage Keypoint-Based Category-Level Object Pose Estimation from an RGB Image," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 1547–1553.
- [10] —, "Keypoint-Based Category-Level Object Pose Tracking from an RGB Sequence with Uncertainty Estimation," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 1258–1264.
- [11] R. Ge and G. Loianno, "VIpose: Real-time Visual-Inertial 6D Object Pose Tracking," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 4597–4603.
- [12] B. Wen, C. Mitash, B. Ren, and K. E. Bekris, "se(3)-TrackNet: Data-driven 6D Pose Tracking by Calibrating Image Residuals in Synthetic Domains," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2020. [Online]. Available: <http://dx.doi.org/10.1109/IROS45743.2020.9341314>
- [13] B. Wen, W. Yang, J. Kautz, and S. Birchfield, "Foundationpose: Unified 6d pose estimation and tracking of novel objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 17 868–17 879.
- [14] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conrath, K. Daniilidis *et al.*, "Event-Based Vision: A Survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 1, pp. 154–180, 2020.
- [15] Z. Liu, B. Guan, Y. Shang, S. Liang, Z. Yu, and Q. Yu, "Optical Flow-Guided 6DoF Object Pose Tracking with an Event Camera," in *Proceedings of the 32nd ACM International Conference on Multimedia*, 2024, pp. 6501–6509.
- [16] Z. Li, A. Glover, C. Bartolozzi, and L. Natale, "6-DoF Object Tracking with Event-based Optical Flow and Frames," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2025, pp. 18 880–18 887.
- [17] Z. Liu, B. Guan, Y. Shang, Q. Yu, and L. Kneip, "Line-Based 6-DoF Object Pose Estimation and Tracking With an Event Camera," *IEEE Transactions on Image Processing*, 2024.
- [18] Z. Liu, B. Guan, Y. Shang, Y. Bian, P. Sun, and Q. Yu, "Stereo Event-Based, 6-DOF Pose Tracking for Uncooperative Spacecraft," *IEEE Transactions on Geoscience and Remote Sensing*, 2025.
- [19] A. Glover, L. Gava, Z. Li, and C. Bartolozzi, "EDOPT: Event-camera 6-DoF Dynamic Object Pose Tracking," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 18 200–18 206.
- [20] E. Dubeau, M. Garon, B. Debaque, R. d. Charette, and J.-F. Lalonde, "RGB-D-E: Event Camera Calibration for Fast 6-DOF Object Tracking," in *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2020, pp. 127–135.
- [21] H. Li and J. Stueckler, "Tracking 6-DoF Object Motion from Events and Frames," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 14 171–14 177.
- [22] Z. Li, N. A. Piga, F. Di Pietro, M. Iacono, A. Glover, L. Natale, and C. Bartolozzi, "Hybrid Object Tracking with Events and Frames," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 9057–9064.
- [23] Y. Kang, G. Caron, R. Ishikawa, A. Escande, K. Chappellet, R. Sagawa, and T. Oishi, "Direct 3D model-based object tracking with event camera by motion interpolation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 2645–2651.
- [24] N. A. Piga, Y. Onyshchuk, G. Pasquale, U. Pattacini, and L. Natale, "ROFT: Real-Time Optical Flow-Aided 6D Object Pose and Velocity Tracking," *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 159–166, 2021.
- [25] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects," in *Conference on Robot Learning (CoRL)*, 2018. [Online]. Available: <https://arxiv.org/abs/1809.10790>
- [26] N. A. Piga, "Hybrid Architectures for Object Pose and Velocity Tracking at the Intersection of Kalman Filtering and Machine Learning," Ph.D. dissertation, Università degli studi di Genova, 2022.