2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

EDOPT: Event-camera 6-DoF Dynamic Object Pose Tracking

Arren Glover, *Member, IEEE*, Luna Gava, Zhichao Li, and Chiara Bartolozzi, *Member, IEEE*

***Abstract*— High-frequency, low-latency, 6-DoF object tracking is useful for grasping objects in motion, taking robots beyond pick-and-place tasks. We propose using an event-camera for tracking the objects to leverage the low-latency and continuous (i.e. not fixed-rate) data capture for high-frequency tracking. We propose the EDOPT algorithm, which maintains real-time operation with a variable event-rate (which occurs due to variation in camera velocity and scene texture) and avoids frame-jumps and motion-blur which are problematic in traditional computer vision solutions. EDOPT uses a strong object prior, leading to a novel solution possible only with the event-camera. To our knowledge, this is the first method for 6-DoF object pose estimation with only the event-camera. The proposed method achieves comparable results to a state-of-the-art DNN technique that fuses frames, depth, and events. We demonstrate smooth, online object pose tracking with a live camera feed at > 300 Hz.**

Fig. 1: Event-camera 6-DoF Dynamic Object Pose Tracking in a real-time live experiment moving both the camera and the object (toy car).

I. INTRODUCTION

A robot can use the precise pose of an object in order to grasp and manipulate it, or to plan a trajectory to avoid object collision. Tracking the 6-DoF pose of an object during fast motion can enable robotic object manipulation beyond pick-and-place tasks. For robots that interact with a known set of rigid objects, it is feasible to store models of the objects' 3D meshes *a-priori*, while the 3D mesh of new objects learnt during a robot's lifetime can also be added thanks to advances in scanning technology [1].

The state-of-the-art 6-DoF object pose estimation has been pushed by deep neural networks that exploit the availability of big-data and processing power, which is needed for training [2]–[5]. Data-driven approaches have helped to overcome difficult-to-model effects such as lighting, reflections and shadows and DNNs can also help with *motion-blur* in the input image, at the cost of increased training data, training time, and network size, thereby increasing operational costs, energy usage, and hardware requirements.

Event-cameras [6] are a more recent technology that can be exploited for sensing fast-moving objects at high frequency and potentially lower computational cost. The photon-integration time of a global shutter is replaced with individual pixel circuitry that detects *change* in light level and outputs a sparse asynchronous visual signal with low latency and at a possible MHz frequency. Event-cameras have the potential for lightweight, high-rate, 6-DoF object pose tracking and can avoid motion-blur in the sensor hardware.

Object tracking algorithms for event-cameras have mainly focused on tracking an object on the image plane with stationary cameras (e.g. [7]). In these cases, tracking an object with a stationary event camera is almost trivial as the background does not produce data and segmentation is almost free. Tracking with a moving, robot-mounted camera is also possible [8], but such algorithms do not provide the 6-DoF pose required for object grasping.

Estimation of object pose in 6-DoF has instead been performed with fusion techniques with RGB frames in a probabilistic framework [9], [10] and also with RGB and depth with neural networks [11]. In these cases, the events are typically only used to give an estimation of pose change between frames, while consistent pose alignment requires the image frame.

6-DoF tracking and pose estimation solutions have been proposed using event-based PnP solutions for key-points on flat fiducials [12], [13]. To extend such works to object pose estimation, key-point registration between the object and the projected 2D appearance would be required. The perspective transform also does not hold for fully rotating objects.

Object pose estimation with events-only could instead draw inspiration from 6-DoF *camera pose* estimation algorithms using optimisation [14], a Bayesian filter [15] or neural network [16] approaches, by inverting the relative motion between camera and environment. However, such algorithms would require enough modification to handle large rotation magnitudes in which self occlusion occurs, and to segment object and background events if both are changing simultaneously. Currently, there are no 6-DoF object pose estimation algorithms using only an event-camera.

All the authors are with Event-Driven Perception for Robotics, Istituto Italiano di Tecnologia, Italy. Corresponding author: arren.glover@iit.it

We propose **e**vent-camera 6-DoF **d**ynamic **o**bject **p**ose **t**racking (EDOPT) to leverage event-cameras for the following benefits:

- high frequency - event-cameras are frame-less and asynchronous and therefore not limited to the 60 Hz acquisition rate of most RGB cameras;
- motion blur robustness - event-cameras don't have fixed exposure periods and our algorithm exploits a blur-free representation that avoids temporal data accumulation;
- a lightweight algorithm - high-frequency tracking can exploit the asynchronous stream of event data to remove pose "jumps" between frames. Small pose updates can be assumed between algorithm updates.

The EDOPT algorithm uses the exponentially-reduced ordinal surface (EROS) [17] to represent the expectation of intensity gradients on the image plane. The representation maintains asynchronicity in the events (i.e. it is updated at sub-millisecond timing) while eliminating the need to tune parameters for speed or environmental clutter. The EROS does not use a fixed temporal period or fixed event-count accumulation method and is therefore robust to variable and simultaneous object and camera motion.

The EROS is compared to a small number of hypothesis pose change candidates in 6-DoF created by rendering the object mesh onto the image plane [18] and computing image gradients. EDOPT requires only a minimal set of hypothesis locations as previously shown to be an effective solution [19]; it can be assumed the target only changes pose slightly (i.e. a small ΔX) between updates due to the event-camera operating principle. Differently to [19], EDOPT removes the Gaussian-decayed fixed event-count parameter in favour of EROS, transitions from a 3-DoF to a 6-DoF solution, and introduces a method for robust templates from projected models. The solution also introduces real-time robustness by decoupling tracking from the processing of events.

The contributions of this work are the presentation of the first event-only algorithm for 6-DoF object pose estimation. The proposed EDOPT algorithm aims to exploit the advantages of event-cameras while being functional and useful for robotics applications. The algorithm provides a novel method to compare the asynchronous event-stream to mesh projections. We demonstrate the algorithm functions with a simple method for generating future pose hypotheses. We compare EDOPT to the RGB-D-E object tracking algorithm [11], demonstrate it generalises to any provided object mesh, and demonstrate the operation with an on-line experiment.

II. The EDOPT Algorithm

Event-based 6-DoF dynamic object pose tracking (EDOPT) requires the components shown in Fig. 2. The object state, X, is defined in the camera reference frame:

$$X = \begin{bmatrix} x & y & z & \alpha & \beta & \gamma \end{bmatrix}^T \tag{1}$$

where $[x\ y\ z]$ is the position and $[\alpha\ \beta\ \gamma]$ are the pitch, yaw and roll respectively.

The algorithm has three components:

- Producing the estimate of contrast gradients O_t at any point in time t from event-camera data.
- Projecting an object mesh onto the image plane at a set of hypothesis states, $\{X_t^+\}$, and subsequent processing to estimate image gradients to form the image set, $\{E_t\}$.
- Updating the estimated state X_t given the similarity between the sensor observation O_t and the set $\{E_t\}$.

A. Sensor Processing

The event-camera produces an asynchronous stream of pixel spikes dubbed *events* that occur when the light falling on the pixel changes beyond a threshold. The full history of event-camera data, $V_{0\to t}$, is processed iteratively to form the sensor observation, O_t, at the current time, t, using the exponentially reduced ordinal surface (EROS). Each element, $v_i \in V$, indicates a single pixel location $< v_x, v_y >$. An example of the EROS is shown in Fig. 1. The EROS facilitates a visual expectation comparison as no image is inherently produced by an event-camera.

$$O : (x, y) \mapsto [0.0, 1.0] \tag{2}$$

The EROS [17] forms an image-like representation in which each pixel indicates the presence of the *gradient* of light, for example, along the edges of objects and structures. The representation is updated asynchronously, with each event contributing a small, local change to the representation.

Algorithm 1 Event-by-event EROS update

```
for x = v_x − k_o : v_x + k_o
  for y = v_y − k_o : v_y + k_o
    O_xy ← O_xy × 0.3^(1/k_o)
O_{v_x v_y} ← 1.0
```

Algorithm 1 can process up to approximately $10 \cdot 10^6$ events per second, depending on the kernel size, k_o, and is therefore real-time viable for a wide range of conditions, even under camera motion. As no temporal forgetting is performed, the EROS offers persistence of edges that have not been updated recently. As no temporal parameter is needed, the algorithm is robust to variations in speed and amount of texture in the scene. The algorithm instead assumes there is a minimum distance (i.e. defined by k_o) between neighbouring edges.

B. Visual Expectation

The expectation, E of image gradients for any given state, X, is calculated by projecting the target object mesh onto the image plane and applying a Sobel filter. The set of hypothesis states $\{X^+\}$ is calculated as small offsets from X_t - this reliance on a strong prior is made possible only when using the event-camera.

a) Model Projection: Given any object pose, and the camera parameters, $\boldsymbol{K}$, an object mesh can be projected (as in Fig. 3a) onto the image plane [18] forming an image, I. Obtaining the object mesh can be easily achieved with computer-aided drawing software or object scanning

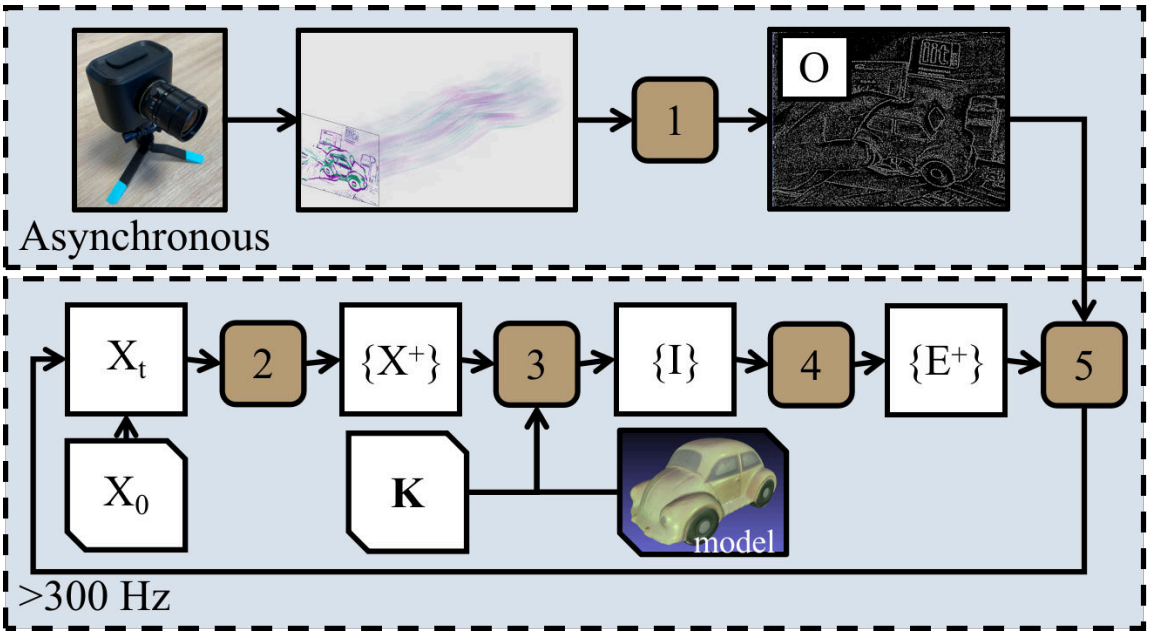

Fig. 2: The EDOPT pipeline has two decoupled pipelines. The first pipeline is the processing of the event-driven data to form the EROS representation, O, that indicates likelihoods of image gradients at the temporal resolution of the event-camera. The second pipeline is the production of image-gradient representations, $\{E^+\}$, associated with hypothesis state estimates, $\{X^+\}$, and the comparison of the observation with hypotheses to update the state, X_t. The indicated processes are described in Sections: (1) II-A, (2) II-B.0.c, (3) II-B.0.a, (4) II-B.0.b and (5) II-C.

technology [1], and many object interaction datasets also provide mesh files as a convenience (e.g. [20]).

b) Gradient Extraction: To facilitate a comparison between 0 (image gradients from the sensor), the image gradients are extracted from the visual expectation, I:

1) Identify gradients in I using a 3×3 Sobel operation (`cv::Sobel()` [21]).
2) Smooth the result with a Gaussian Blur of width B (`cv::GaussianBlur()`).
3) Subtract from the result a secondary smoothed result with a Gaussian Blur of width $2B$.

The final result, E, is a type of difference-of-Gaussians in which positive regions should coincide with gradient locations in O to achieve a strong matching score.

c) State Expectation: A key process in the EDOPT algorithm is the selection of the set of states $\{X^+\}$ from which to create the visual expectations $\{E^+\}$. Due to the mechanisms of the event-camera we can select states within one pixel of visual change from X_t. To estimate a ΔX that corresponds to a small change in image space we use the image Jacobian:

$$\begin{bmatrix}\dot{u}\\ \dot{v}\end{bmatrix} = \begin{bmatrix} -\frac{f'}{z} & 0 & \frac{\bar{u}}{z} & \frac{\bar{u}\bar{v}}{f'} & -\frac{f'^2+\bar{u}^2}{f'} & \bar{v} \\ 0 & -\frac{f'}{z} & \frac{\bar{v}}{z} & \frac{f'^2+\bar{v}^2}{f'} & -\frac{\bar{u}\bar{v}}{f'} & -\bar{u} \end{bmatrix} \begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\\ \dot{\alpha}\\ \dot{\beta}\\ \dot{\gamma}\end{bmatrix} \tag{3}$$

where $\bar{u} = u - c_x$ and $\bar{v} = v - c_y$.

However, we want to find a ΔX that corresponds to a fixed $\Delta|u, v|$ such that the maximum visual change is Δ_p pixels (1 in our experiments). By taking each axis independently and using the inverse Jacobian, the 12 $\{X^+\}$ can be calculated from:

$$\begin{aligned} \pm\Delta x &= \frac{\mp z\Delta_p}{f'} & \pm\Delta y &= \frac{\mp z\Delta_p}{f'} & \pm\Delta z &= \frac{\pm z\Delta_p}{\bar{p}} \\ \pm\Delta\alpha &= \frac{\pm f'\Delta_p}{\bar{p}^2} & \pm\Delta\beta &= \frac{\mp f'\Delta_p}{\bar{p}^2} & \pm\Delta\gamma &= \frac{\pm\Delta p}{\bar{p}} \end{aligned} \tag{4}$$

where $\bar{p}$ is the $\max(\bar{u}, \bar{v})$ for a pixel on the outer edge of the object. The 13th hypothesis state is the null hypothesis in which $X_i^+ = X_t$.

C. State Update

The EROS, O, and a processed visual expectation, E, are not identical measures and an L1 norm cannot be applied. Instead, a simple approach to finding similarity between the representations is to take the dot product between the observation O_t and the visual expectations, $\{E^+\}$. To update the state the ΔX that obtains the maximum similarity can be used:

$$X_t = X_{t-1} + \underset{\Delta X}{\operatorname{argmax}}\{E_i^+ \cdot O_t\}. \tag{5}$$

As an object may move along multiple axes simultaneously, a faster convergence can be made by updating the state additively with *any* ΔX for which $E_i^+ \cdot O_t > E_X \cdot O_t$, where E_X is the expectation corresponding to no motion (the expectation of X_t).

D. Implementation

Two implementation considerations are worthwhile mentioning regarding algorithm complexity and computational savings:

a) Algorithm Complexity: Real-time operation requires an algorithm complexity that allows for a constant, and predictable update. However, event-cameras produce a variable data-rate making a constant, predictable output-rate non-trivial. In EDOPT we separate algorithm processing into two components that have no direct computational dependency, both run asynchronously, as fast-as-possible:

1) An event-by-event update of the EROS with a complexity dependent on the event-rate, $\dot{V}$, and the EROS kernel size: $\mathcal{O}(\dot{V}.k_o)$. Event-throughput is maximised by minimising processing performed per event.
2) The object pose update with a complexity dependent on the number of hypothesis states and the image resolution, h, w: $\mathcal{O}(y.h.w)$, where y is the size of $\{X^+\}$. Tracking is independent of event-rate as the EROS is sampled in 2D only, at the most up-to-date time, t, for each update.

In practice, we can run the two algorithm components in parallel using a multi-threaded approach to boost the overall update rate of the algorithm. The output of the algorithm, X_t, occurs at a rate independent of the (variable) input event-rate.

b) Projection Warping: Computation can be saved by using affine warps to produce $\{E^+\}$ instead of the full procedure of projection and image processing. The visual expectation of no-motion, E_X, is firstly created, and then other visual expectations are produced using Affine warps along x, y, z, and γ. The related ΔX is computed as in Section II-B.0.c. However, for axes α and β (pitch and yaw), the image Jacobian is highly non-linear due to self-occlusion and Affine warping is not sufficient. Therefore the full image projection and processing is performed for these two axes for each step.

III. EXPERIMENTS AND RESULTS

EDOPT was evaluated in comparison to the current state-of-the-art [11], and an evaluation of pose tracking consistency over 5 objects (O_{1-5}). Both experiments use simulated data to have the precise ground-truth generated by placing the object mesh in an unreal engine environment and the simulated event-stream produced by a standard logarithmic-image-difference technique at 500 Hz. The RGB, depth and ground-truth information is available at 60 Hz. The object meshes are moved along the 6 major axes in separate motions approximately 20 cm/s or 28 degrees/s, and simultaneously along all axes at a maximum of 1 m/s and 57 degree/s. The typical motion results in an image motion of $>$ 200 pixels/s, which takes 2 to 3 seconds to traverse the full field-of-view. Approximate total displacements for the dataset are 0.25 m translation and 67 degrees rotation. A final experiment shows the qualitative result of 6-DoF Tracking using a live camera, running real-time.

Experiments were performed with an Intel i7 @ 2.60GHz and an NVIDIA GeForce GTX 1650 Ti (for [11]).

(a) Dragon (O_1) (b) Jell-o (O_2) (c) Mustard (O_3)

(d) Soup Can (O_4) (e) SPAM (O_5) (f) Toy Car (O_6)

Fig. 3: Rendered object meshes used in (a) comparison to state-of-the-art (b-e) object variation, and (f) live on-line experiments.

	RGB-D-E	**EDOPT**
Sensors	frames, depth, events	events only
Mean position error (offline)	0.73 cm	0.70 cm
Mean rotation error (offline)	2.44 deg	2.30 deg
Update rate (online)	5 Hz	300 Hz
Update type	RGB-D coupled	asynchronous
Auto Initialisation/failure recovery	✗	✗
Requires training	✓	✗
Requires object mesh	✓	✓

TABLE I: Comparison of EDOPT and RGB-D-E requirements and capabilities.

A. Comparison to the state-of-the-art

The RGB-D-E [11] algorithm is the only event-based 6-DoF tracking algorithm that is open-source and possible to compare against. The algorithm has a two-step process in which the events are first used to predict the object motion between consecutive RGB frames. A refinement of the pose using the RGB frame is then performed in a second step. RBG-D-E uses events, RGB frames, and depth information while EDOPT functions only using the event data. The original datasets used in [11] were not available open-source, nor was the code to re-train the network with more objects. Therefore new synthetic datasets were created using the original object mesh (dragon) for performance comparison.

Both algorithms could successfully track the 6-DoF pose of the dragon object, as shown in Fig. 4, with typical errors of < 1 cm and < 2.5 degrees, as shown in Fig. 5. Specifically, EDOPT was more consistent for position tracking on the image plane (x, y axes) but had a higher variation of error when moving towards/away from the camera (z axis), or under yaw motion, β. When considering simultaneous motion along all axes, both algorithms had a similar median error, but EDOPT had a higher error variance.

The inference time of RGB-D-E is reported as 33 ms [11], however, in our experiments RGB-D-E only operated at approximately 200 ms per frame (5 Hz). The RGB-D-E implementation was taken from the author's source and run with identical parameters, but required some modifications

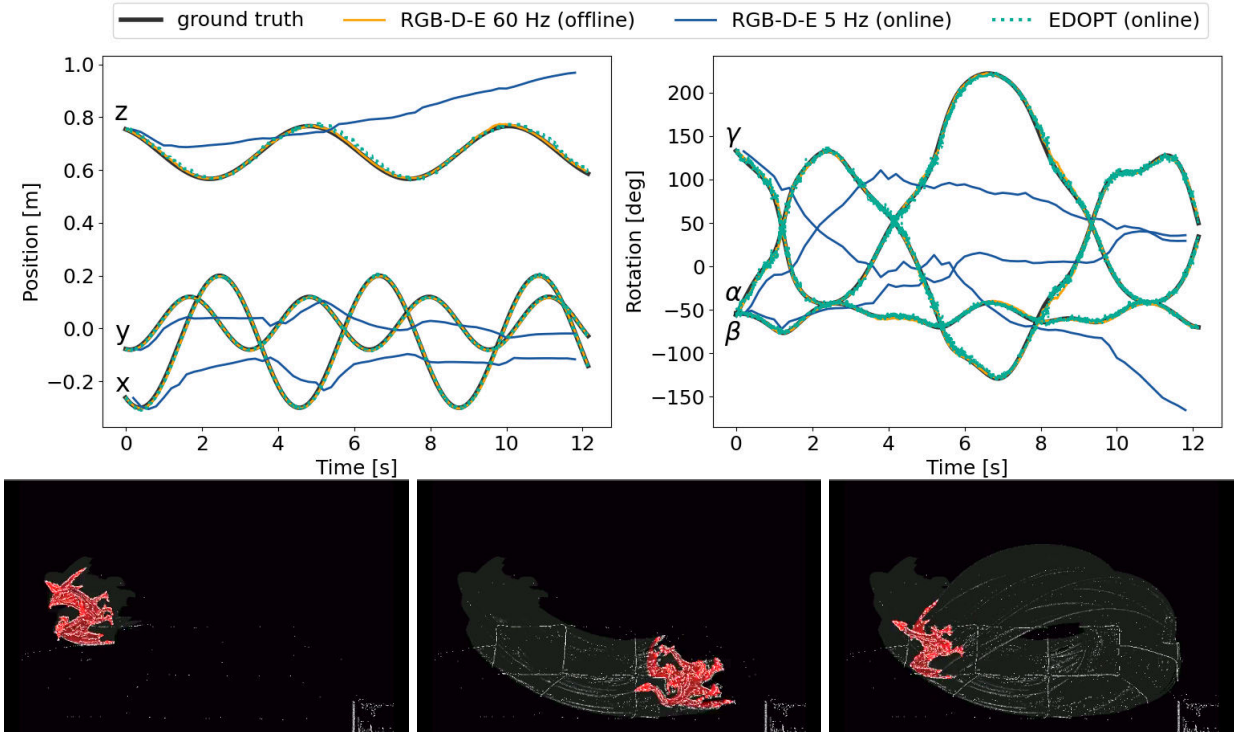

Fig. 4: Tracking trajectories compared to RGB-D-E [11] at 60 Hz (non real-time) and 5 Hz (real-time). The dragon (O1) is moved in 6-DoF simultaneously with some example frames shown below. RGB-D-E 5 Hz fails as the dragon moves too far between updates; 5 Hz is chosen as the fastest we could run RGB-D-E in real-time at 200 ms per update.

that could have changed run-times. We could not achieve the reported 30 Hz as in [11].

If both algorithms are run in real-time (EDOPT v.s. RGB-D-E 5 Hz in Fig. 4) it can be seen that the RGB-D-E algorithm fails at tracking as the jump in motion over the 200 ms period is too large for the algorithm to handle. EDOPT tracks in real-time at 300 Hz and manages to maintain tracking throughout the dataset. Additionally, EDOPT has fewer sensors, and does not require training; see Table. I.

B. Robustness to Object-shape

We further evaluated EDOPT on 5 different objects, 4 of which are from the YCB [20] dataset to evaluate the algorithm robustness to object type. Simulated datasets for these objects are produced from the available YCB meshes and 6 motion sequences are produced for each object. Errors for each object are averaged for translation and rotation separately and shown in Fig. 6.

RGB-D-E cannot solve these datasets as the network is not trained for these objects, instead, EDOPT only requires the available object mesh and does not require further training.

The median positional error is again < 1 cm for all objects, indicating a consistent performance independent of the object. The mean rotational error is < 6 degrees for all objects, which is higher than for the dragon (O1) at 2.3 degrees. EDOPT may perform better when tracking an object with irregular edges, as unique positions of high contrast (object edges) are extracted from the model.

C. Live, On-line Experiment

The previous experiments are performed with simulated objects and simulated datasets but still processed in real-time using EDOPT (but non-real-time with RGB-D-E). However, for robotics use-cases on-line experiments with real objects are required to demonstrate the algorithm operates beyond perfect conditions.

For the live experiments, we used the Metavision ATIS HVGA Gen3 (640x480 pixels) [22]. A toy car mesh was generated using a 3D scanner and initialised at the centre of the field of view. A person is holding the toy car and moving it in front of the camera in unconstrained motion. Fig. 1 shows a series of frames from a recorded video in which the events and projected car model overlap. The car mesh visually matches the event-stream, however it is not perfect, for example, as shown in 3rd frame. The live experiments demonstrate that the algorithm successfully tracks motion along multiple axes simultaneously.

IV. DISCUSSION

Data-driven v.s. Modelled

While the modelled approach proposed in EDOPT doesn't require training, it can have disadvantages compared to a data-driven approach (i.e. neural network) in that it may be less robust to difficult-to-model conditions such as lighting, reflections, and occlusions. The event-camera achieves some robustness to illumination due to the dynamic range and sensitivity of the pixels but further experimentation is re-

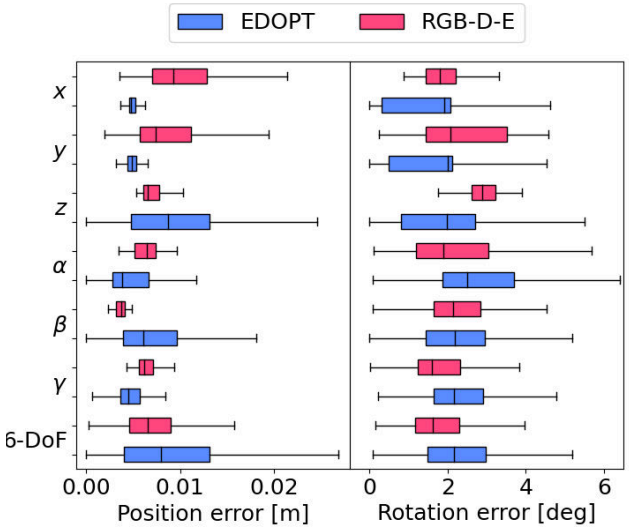

Fig. 5: EDOPT and RGB-D-E [11] comparison for median position and rotation error for motions along the six major axes. Whiskers indicate minimum and maximum errors.

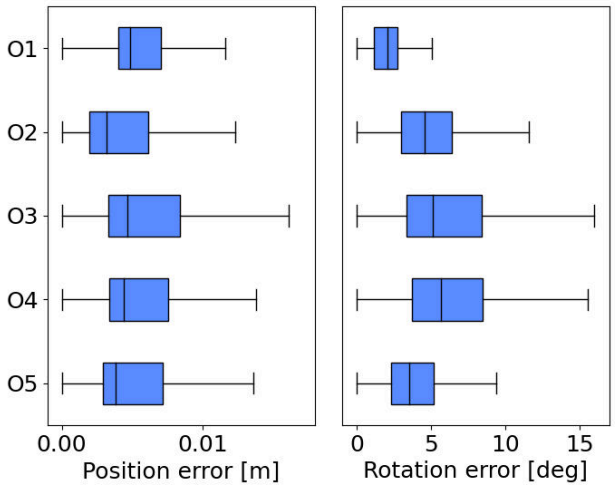

Fig. 6: EDOPT median position and rotation errors for different objects (O_{1-5}). Each object is tested on 6-DoF motions and results are concatenated for each object.

quired to fully characterise the algorithm amongst a variety of conditions.

EROS v.s. Image Gradients

The EDOPT algorithm assumes that the EROS, O, and the image gradients, ∇I, are comparable, in order to find the best matching state hypothesis. However, while both indicate the presence of a gradient, they are not directly identical measures. We therefore didn't use an error measure (i.e. L1 norm) to compute similarity. Instead, the proposed dot-product of E and O performs a similarity score without requiring identical measures. Negative values present in E reduce the matching score in specific pixels for dissimilar values.

Initialisation

The scope of the proposed method is limited to the tracking portion of a full visual pipeline, which typically also requires an algorithm for initialisation and failure recovery in the form of a *pose detector*. A global detector needs to be developed for event-camera data (i.e. similar to DOPE [23] for RGB cameras), however, no 6-DoF pose detection algorithms currently exist for event cameras and remains an open problem.

Failures and Limitations

Tracking failure can occur due to a mismatch between the observation and the projected object model. Imperfections in the visual signal, or incorrect assumptions about the visual signal as made by the EROS algorithm, manifest as noise and artefacts in the EROS representation. However, these artefacts are less likely around moving edges with strong contrast. Therefore the object moving typically has a clear and reliable visual signal from the EROS.

The algorithm has a hard maximum velocity cap due to a fixed ΔX per update. The design is purposeful as increasing the ΔX can exponentially increase the hypothesis states required, and exponentially increase the time per update. Instead, a small ΔX with fast tracker updates is chosen to enable a real-time solution; robustness may increase by finding a more optimal trade-off point.

In general, we have shown the algorithm correctly converges and can run in real-time. However, failure can be caused by any number of difficult conditions. Understanding how to make the algorithm more robust is dependent on the exact scenarios and applications for which the algorithm would be deployed.

V. CONCLUSION

To the best of our knowledge, we have presented the first event-camera-only 6-DoF object pose tracking algorithm. The algorithm achieved comparable performance to the available state-of-the-art, RGB-D-E, which used both frames and depth information in addition to the events and required training of the network. EDOPT could run in real-time with $>$ 300 Hz output compared to RGB-D-E which ran at 5 Hz in our experiments. We demonstrated the algorithm was robust to different objects, tracks robustly in 6-DoF simultaneously, and operated with a live, on-line camera.

In the event-camera vision domain, we have solved problems that occur when using fixed event-batches (which can lead to motion-blur or under-exposed scenes even with an event-camera) through the use of EROS. The method is therefore also robust to a moving camera and variation in object/camera velocity. The method is novel in the way it proposes to compare the EROS with state expectations.

In order to use the algorithm in a robot 'in-the-wild', an additional object pose detection layer using data-driven approaches would provide an initialisation, and a failure recovery mode. In addition, fusion with a data-driven method can lead to improved robustness to lighting variation, occlusions, shadows, and reflections.

The code and datasets are made available open-source for the community to adopt EDOPT[1].

[1]https://github.com/event-driven-robotics/EDOPT

REFERENCES

[1] M. E. Gurses, A. Gungor, S. Hanalioglu, C. K. Yaltirik, H. C. Postuk, M. Berker, and U. Türe, "Qlone®: a simple method to create 360-degree photogrammetry-based 3-dimensional model of cadaveric specimens," *Operative Neurosurgery*, vol. 21, no. 6, pp. E488–E493, 2021.

[2] J. Issac, M. Wthrich, 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] W. Chen, X. Jia, H. J. Chang, J. Duan, L. Shen, and A. Leonardis, "Fs-net: Fast shape-based network for category-level 6d object pose estimation with decoupled rotation mechanism," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 1581–1590.

[4] 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, 2022.

[5] 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," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 10 367–10 373.

[6] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, 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.

[7] A. Z. Zhu, N. Atanasov, and K. Daniilidis, "Event-based feature tracking with probabilistic data association," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 4465–4470.

[8] A. Glover and C. Bartolozzi, "Robust visual tracking with a freely-moving event camera," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 3769–3776.

[9] H. Li and J. Stueckler, "Tracking 6-dof object motion from events and frames," in *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9561760

[10] 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)*. IEEE, 2023, pp. –.

[11] 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.

[12] D. Reverter Valeiras, S. Kime, S.-H. Ieng, and R. B. Benosman, "An event-based solution to the perspective-n-point problem," *Frontiers in neuroscience*, vol. 10, p. 208, 2016.

[13] A. Loch, G. Haessig, and M. Vincze, "Event-based high-speed low-latency fiducial marker tracking," in *Fifteenth International Conference on Machine Vision (ICMV 2022)*, vol. 12701. SPIE, 2023, pp. 323–330.

[14] H. Rebecq, T. Horstschaefer, G. Gallego, and D. Scaramuzza, "Evo: A geometric approach to event-based 6-dof parallel tracking and mapping in real time," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 593–600, 2017.

[15] G. Gallego, J. E. Lund, E. Mueggler, H. Rebecq, T. Delbruck, and D. Scaramuzza, "Event-based, 6-dof camera tracking from photometric depth maps," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 10, pp. 2402–2412, 2018.

[16] Y. Zhou, G. Gallego, and S. Shen, "Event-based stereo visual odometry," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1433–1450, oct 2021. [Online]. Available: https://doi.org/10.11092Ftro.2021.3062252

[17] G. Goyal, F. Di Pietro, N. Carissimi, A. Glover, and C. Bartolozzi, "MoveEnet: Online High-Frequency Human Pose Estimation With an Event Camera," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2023, pp. 4023–4032.

[18] C. Fantacci, "Superimposemesh library," https://github.com/robotology/superimpose-mesh-lib, 2019.

[19] I. Alzugaray and M. Chli, "Haste: multi-hypothesis asynchronous speeded-up tracking of events," in *31st British Machine Vision Virtual Conference (BMVC 2020)*. ETH Zurich, Institute of Robotics and Intelligent Systems, 2020, p. 744.

[20] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks for manipulation research," in *2015 international conference on advanced robotics (ICAR)*. IEEE, 2015, pp. 510–517.

[21] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[22] C. Posch, D. Matolin, and R. Wohlgenannt, "A qvga 143 db dynamic range frame-free pwm image sensor with lossless pixel-level video compression and time-domain cds," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 1, pp. 259–275, 2010.

[23] 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