Learning 6D Object Pose Estimation With Event Cameras Using Synthetic Data and Domain Randomization

Oussama Abdul Hay, Xiaoqian Huang, Muhammad Ahmed Humais, Abdulla Ayyad, Randa Almadhoun, and Yahya Zweiri

***Abstract*—Estimating the 6D pose of rigid objects is a critical upstream task in many robotics applications. Most existing methods rely on RGB or RGB-D sensing modalities, which suffer from limitations under challenging lighting conditions and high-speed motion. In contrast, event-based cameras offer unique advantages such as high temporal resolution and high dynamic range, making them well-suited for such scenarios. However, current event-based pose estimation methods are typically optimization-based, designed for relatively simple objects, and require hand-crafted parameters. In this work, we introduce the first learning-based approach for 6D object pose estimation using event cameras, employing an Augmented Event Encoder (AEE) trained entirely only on synthetic data and validated on the E-POSE dataset. Our model leverages an augmented autoencoder with domain randomization to map synthetic templates into a latent space, enabling accurate matching with real event query images. The method demonstrates robust performance across various scenarios, including changes in illumination and camera speeds, and achieves strong results on the ADD-S (Rotation) metric.**

***Index Terms*—Deep learning for visual perception, perception for grasping and manipulation.**

I. Introduction

RECOGNIZING the 6D pose of objects from sensors is an important task for many real-world applications. This entails estimating translation and orientation of the object in 3D space relative to the camera. Estimating the 6D pose of objects is fundamental to enabling autonomous systems in tasks such as grasping, bin-picking, and collaborative robotics [1]. In recent years, numerous approaches have been proposed for 6D object pose estimation using stereo and monocular cameras based on RGB and depth data. While these sensors provide rich visual features, they struggle in environments with rapidly changing lighting or fast motion due to limitations in dynamic range and shutter speed, as illustrated in Fig. 1. Event cameras are bio-inspired sensors that perceive the environment asynchronously with high temporal resolution [2]. They operate reliably under varying lighting, making them well-suited for pose estimation of texture-less objects and industrial parts with lighting-sensitive features [3]. Their asynchronous output enables diverse representations such as event histograms, frames, time surfaces, and voxel grids [4], which has been applied in ego-motion estimation [5], optical flow [6] and SLAM [7]. However, object pose estimation with event data remains underexplored, a gap this work aims to address.

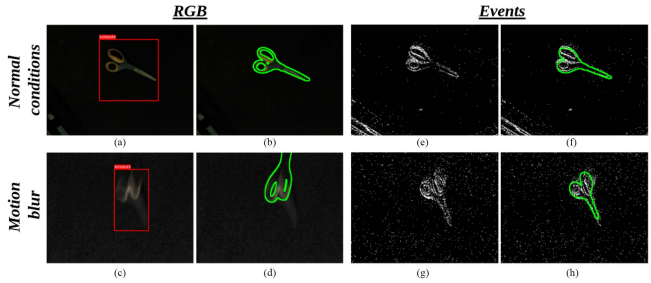


Fig. 1. Qualitative comparison of object pose estimation under normal and motion-blurred conditions using RGB and event cameras. In RGB, MegaPose [8] fails under motion blur, resulting in inaccurate overlays, while AEE on event data maintains accurate alignment due to the high temporal resolution of events. This highlights the robustness of event-based pose estimation in dynamic scenes.

Object pose estimation has been approached through various methods. While regression and correspondence-based techniques have shown strong performance, template-based methods have gained popularity for their ability to leverage CAD models to adapt pre-trained networks to new objects [9]. This enables pose estimation for unseen samples, improving scalability and reducing annotation and training time [10]. Template-based matching extracts a feature vector from the query image and compares it to pre-generated template embeddings, often organized in a codebook [11]. A key advantage is the ability to cover the full $SO(3)$ space through rendered templates, mapping all object rotations into the latent space [12]. This reduces reliance on extensive training data by exposing the model to all possible orientations in advance.

Received 13 August 2025; accepted 2 December 2025. Date of publication 15 December 2025; date of current version 22 December 2025. This article was recommended for publication by Associate Editor M. Vincze and Editor Y. Xiang upon evaluation of the reviewers' comments. This work was supported in part by STRATA Manufacturing PJSC and in part by Advanced Research and Innovation Center (ARIC), which is jointly funded by Aerospace Holding Company LLC, a holly-owned subsidiary of Mubadala Investment Company PJSC and Khalifa University of Science and Technology, United Arab Emirates. *(Corresponding author: Oussama Abdul Hay.)*

Oussama Abdul Hay, Xiaoqian Huang, Muhammad Ahmed Humais, and Abdulla Ayyad are with Advanced Research and Innovation Center (ARIC), Abu Dhabi, UAE (e-mail: oussama.hay@ku.ac.ae).

Randa Almadhoun is with the School of Computer Science & Engineering, Faculty of Business & Technology, University of Sunderland, SR6 0DD Sunderland, U.K..

Yahya Zweiri is with the Advanced Research and Innovation Center (ARIC), Abu Dhabi, UAE, and also with the Department of Aerospace Engineering, Khalifa University, Abu Dhabi, UAE.

Digital Object Identifier 10.1109/LRA.2025.3644151

© 2025 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

Several studies have explored pose tracking of simple objects with well-defined edges using event cameras, leveraging the sensor's inherent sensitivity to intensity changes along edges. The work in [13] estimated the pose of an icosahedron by associating events with 3D edge points on a known geometric model. While accurate, the method is tailored to shape-based objects with clear lines, simplifying event-to-edge matching. It minimizes the distance between detected events and projected 3D edge points along the camera's line of sight using experimentally tuned scaling parameters. However, its reliance on basic geometric shapes limits applicability to real-world objects with arbitrary shapes or textures.

An alternative approach, LOPET [14], estimates object pose using a line-based initialization via a globally optimal Branch and Bound (BnB) algorithm, avoiding the need for 2D–3D correspondences. It then performs event-to-line matching and pose refinement, assuming a constant velocity model for motion prediction. Like [13], LOPET depends on clear object edges, making it less suitable for complex or textured objects, and it assumes constant angular and linear velocities. EDOPT [15] introduces a 6D pose tracking method based on the exponentially-reduced ordinal surface (EROS) [16], which captures light gradients along object edges. It evaluates mesh hypotheses around the current pose after initializing the object at the frame center. While fast, the method is less robust under challenging conditions like lighting changes, reflections, and occlusions. The authors also highlight the need for detection-based rather than tracking-based pipelines to eliminate reliance on initialization.

To address these limitations, we present the first learning-based method for event-based object pose estimation. First, a domain randomization approach is proposed to transform synthetic RGB data into event-like representations, enabling effective training of neural networks despite the limited availability of real event data. Second, the letter presents the *Augmented Event Encoder (AEE)*, the first learning-based method capable of estimating the full 6D pose of an object from a single event image without requiring prior initialization. The proposed AEE network is lightweight and achieves inference at approximately 5 ms, allowing real-time performance at approximately 200 Hz. Finally, the network is thoroughly evaluated under diverse conditions, including varying object speeds and illumination settings, demonstrating its robustness and generalization capability.

II. Related Work

In this work, we propose an object pose estimation method for event images inspired by template-based approaches. While various pose estimation paradigms exist, we follow the classic taxonomy: direct regression, correspondence-based, and template-based methods.

In *Direct Regression*, the network learns to predict the 3D translation and rotation of an object from image features. PoseCNN [17] directly regresses quaternions and object depth after estimating the object's center pixel. SSD-6D [18] infers in-plane rotation and viewpoint scores from 2D bounding boxes using multi-scale feature maps. DeepIM [19] refines an initial pose prediction through iterative updates of rotation and translation. Transpose [20] leverages graph convolution networks to regress pose directly from depth-image graphs. Although direct regression has shown remarkable results, the method can struggle under varying lighting, reflective surfaces (e.g., industrial parts), and often limits generalization to unseen cases.

Correspondence-based methods estimate pose by establishing 2D–3D point correspondences and solving a Perspective-n-Point (PnP) problem, requiring at least three matches. These methods explore both sparse and dense matching strategies. YOLO-6D [21] extends the YOLO architecture to regress keypoints for alignment with the 3D bounding box. Contour-Pose [22] and ER-Pose [23] leverage object edges for more reliable correspondences. EANet [24] introduces an auxiliary edge reconstruction loss to aid pose regression in textureless scenarios. Despite their strengths, correspondence-based methods often struggle with textureless objects and may degrade under varying viewing conditions due to reliance on reliable feature extraction.

Template-based methods render object templates and extract their latent representations. A query image is encoded and matched against these embeddings to estimate pose, bypassing explicit labeling of symmetric ambiguities. Sundermeyer et al. [12] discretized the *SO(3)* space and trained an augmented autoencoder on rendered templates with varied backgrounds to recover object orientation via dot-product similarity. However, their training did not enforce separation between embeddings. To address this, [11] introduced edge-based features and contrastive loss to improve robustness and increase intra-class separation. Despite these advances, template-based methods often require extensive synthetic data and can be sensitive to viewpoint coverage and template quality.

Template-based matching is adopted for its robustness to texture-less objects, which event cameras represent primarily through edges and corners. The absence of fine-grained features limits the effectiveness of direct regression and 2D–3D correspondence methods.

III. Methodology

A. General Overview

In this section, we present our proposed method, the Augmented Event Encoder (AEE), for 6D object pose estimation. The approach begins by generating dense viewpoints to construct a comprehensive set of templates for each object using RGB images. A domain randomization strategy is employed to convert these RGB images into event-like representations, enabling the network to learn a robust latent representation of each template. During inference, the network trained exclusively on synthetic data is presented with real event data, previously unseen during training, to estimate the object pose within the image. The complete method is outlined in Fig. 2.

B. Augmented Event Encoder

Template Generation: Template generation is the first step in training, enabling sim-to-real transfer by capturing each object from diverse viewpoints. Using Fibonacci sampling, 40,000 templates were generated—covering 4,000 viewpoints with 10 in-plane rotations each—spanning a discretized *SO(3)* space. To reduce computation, templates were limited to a hemisphere shown in Fig. 3, representing tabletop views, which lowers both template count and codebook size for faster inference. Following [10], Blender [25] and BlenderProc [26] were used to render objects under predefined material, lighting, and pose conditions.

Synthetic Data and Domain Randomization: The use of physics-based simulators such as Blender [25] offers powerful

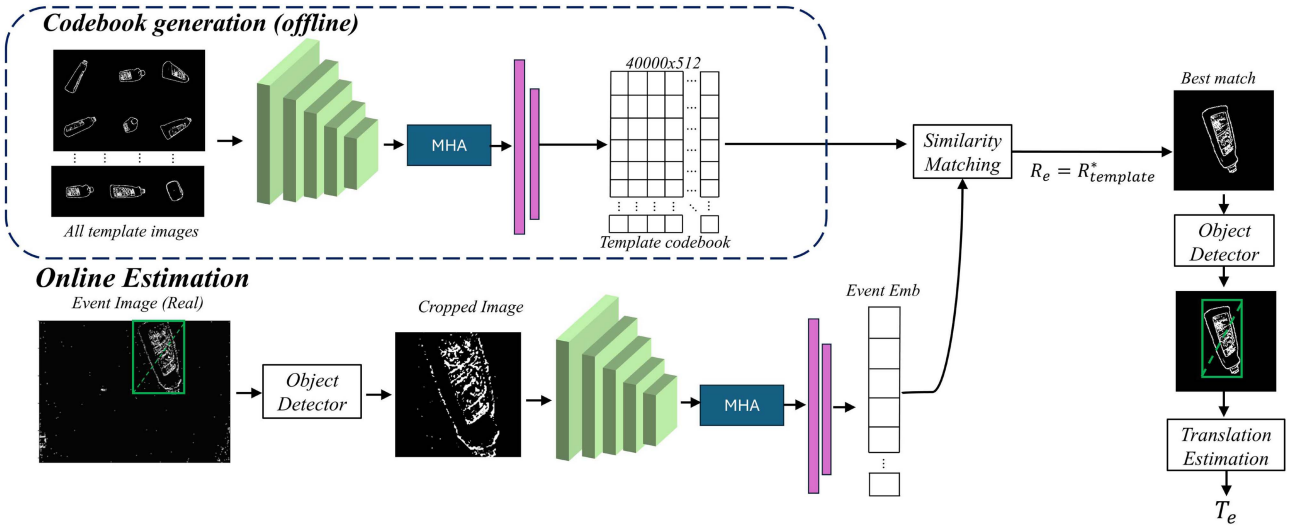


Fig. 2. Synthetic images are encoded by the AEE network and stored in a template codebook, with each column representing a specific object rotation. During inference, an object detector crops the region of interest from the real event image, which is then embedded by the AEE network. Cosine similarity retrieves the closest match, from which the rotation R_e and translation T_e are estimated.

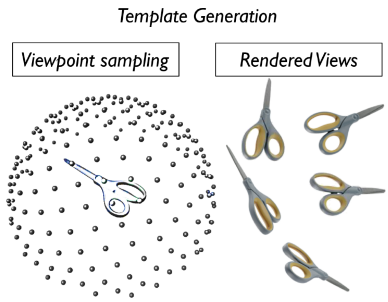


Fig. 3. Template generation for object. quasi-equidistant points on a hemisphere are generated as camera view points to capture the object at different rotations.

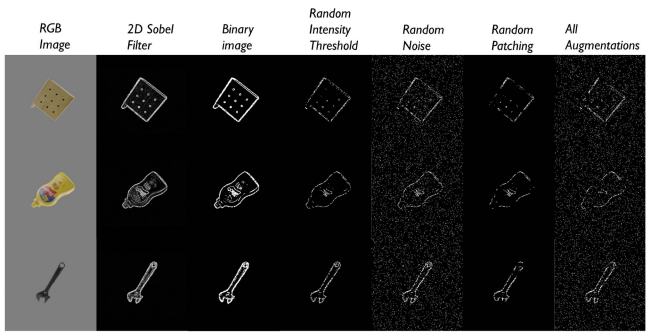


Fig. 4. Different augmentations applied to the RGB image to bring it closer to the event modality.

means to generate synthetic data samples along with their corresponding labels. This capability can significantly accelerate the development of robotic tasks and applications by enabling faster, more scalable, and more controlled data collection. However, the domain gap that exists between synthetic and real data prohibits the direct transfer of models trained on synthetic data to the real world. Domain randomization has been adopted in many cases [27], [28], [29] to bridge the gap between simulation and reality.

Most of the domain randomization approaches have usually been applied to RGB images by changing the illumination conditions and adding random background images to introduce variability into the model. However, another approach was adopted here to implement domain randomization, as events are to a certain extent invariant to light conditions and, unlike conventional vision sensors, exhibit a low signal-to-noise ratio. Therefore, after generating the templates in RGB, a 2D Sobel filter was applied on the RGB to convert them into edge images. Since the event camera only captures edges and corners, this step was important to get the templates closer to the real data. However, the edges captured in the event camera are not always complete, and thus, patches were added that would create discontinuities in the edges. Masking has proven exceptionally effective for learning image embeddings that transfer well to downstream tasks [30]. Moreover, a binary thresholded Sobel edge image generated with a random threshold removes different object features on each pass, injecting further variability and encouraging the network to learn core geometric cues. To emulate sensor noise, random background noise are superimposed so the model learns to filter clutter and concentrate on salient shapes. Conventional scaling and translation augmentations are also included, ensuring invariance to size and position changes at inference time. Furthermore, rather than just superimposing random background images as in most domain randomization approaches, we aim to mimic the characteristics of event cameras in the synthetic domain. By converting RGB templates into edge-based representations and introducing discontinuities and noise, the model learns to focus on geometric structure rather than appearance. This makes the learned embeddings more robust to real-world conditions where events are sparse and incomplete. This enables adapting template matching approaches to the event camera. All of these augmentations are illustrated in Fig. 4.

Reconstruction Network: The reconstruction network is designed to receive a binary-augmented image as input and generate its reconstructed output, with the objective of learning embeddings that are unique to the object's displayed rotation in the image. U-net style architectures have proven to be the best

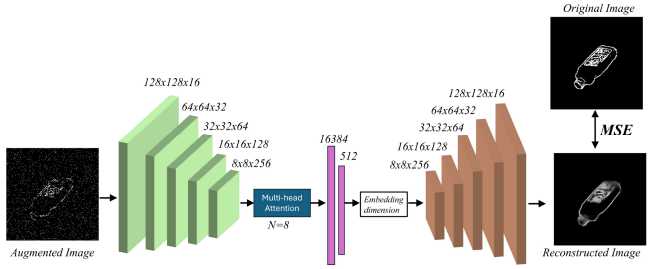


Fig. 5. Reconstruction network used to learn the features of the augmented image.

architectures for reconstruction and thus are employed here for this task. A 5-layer encoder is designed to reduce the dimension of the image. A decoder then reconstructs the image from the reduced dimension while employing a mean squared error loss between the reconstructed image and the ground-truth image that is without any augmentation. The detailed model used for reconstruction is depicted in Fig. 5. The ground-truth image was transformed to a binary image with a preset threshold that was manually tuned to obtain the best reconstruction results.

The reconstruction network was further improved by utilizing an attention block that allows for cross-attention for the embeddings produced by the encoder. Specifically, this mechanism focuses on the most important features of the embeddings, allowing the model to effectively capture and combine key information, while also learning inter-dependencies and focusing on the most salient features. As a result, the attention-driven refinement of embeddings leads to more robust feature representations and, consequently, improved performance. A main advantage of using a binary image is to avoid any pre-processing step for events when passing through the network. Ideally, neuromorphic processing is best suited for event cameras. Having the data in binary allows for the exploration of such data to estimate the pose. In order to confirm the distribution of the embeddings based on their rotation, PCA was used on the templates to reduce their dimension to a vector of 3 elements for visualization. Fig. 6 shows the intra-class and inter-class variations among the embeddings based on the shape of the template object. Templates with similar shapes are grouped together, while a different shape occupies a different region in the plot.

C. Event Image Generation

The event camera perceives the scene in the form of asynchronous events. Along a specific spatial resolution of $H \times W$ which correspond to the vertical and horizontal dimensions of the event sensor, each pixel location x, y is activated if the change in the log intensity of that pixel is higher than a certain threshold C as shown in (1).

$$\Delta L(x, y, t) = \log(L(x, y, t)) - \log(L(x, y, t - \Delta t)) \geq pC \tag{1}$$

The p parameter indicates the polarity of the event which indicates whether the change in the log intensity value was a positive change or a negative change. In total, an event e_i has 4 different variables that describe it, its spatial location x, y, its polarity p, and the time t at which it was received.

$$\mathcal{E} = \{e_i\}_{i=1}^{N} = \{(x_i, y_i, t_i, p_i)\}_{i=1}^{N}. \tag{2}$$

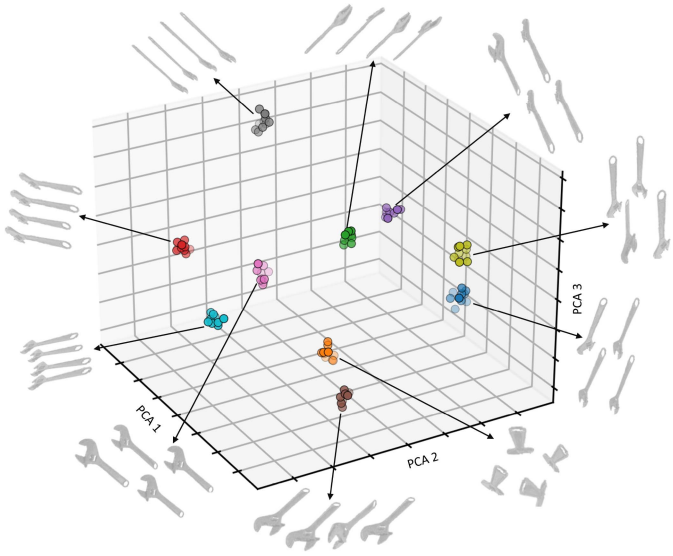


Fig. 6. Distribution of the embeddings for 100 different templates from the wrench codebook, visualizing 40 indices from 10 different clusters.

In order to visualize the event stream, we adapt event frames to convert an asynchronous stream of events into a frame-like representation for the input to the network. We choose an accumulation value N_{events} and a passing window size of w to create frames throughout the sequence. Although this representation disregards the temporal dimension, it was adopted as it closely resembles the domain-randomized templates, thereby enabling direct transfer of learned features from the templates to the event images.

D. Codebook Generation & Rotation Inference

After training the reconstruction network on the generated templates, each template is processed through the encoder and attention block to extract its corresponding embedding, which is then stored separately for each object. During inference, the event image is first passed through a YOLOv11 [31] detector network pre-trained to identify the objects. The bounding box obtained from YOLO is resized to a fixed size that matches the input dimensions of the reconstruction network. The real event image, referred to as the event query image, is also pre-processed similar to the template images, following the same procedure applied to the ground-truth template image during reconstruction. The resized image is passed to the encoder to obtain the embeddings which are used to find the closest match to it in the template codebook. The closest match is obtained using the cosine similarity (3), which measures the normalized dot product between the query image's embeddings A and the template images embeddings previously computed B_i where i corresponds to the number of templates available..

$$\text{Cosine Distance}(A, B_i) = \frac{A \cdot B_i}{\|A\|\|B_i\|}, \quad i = 1, \ldots, k \tag{3}$$

E. Translation Estimation

To estimate the translation of the object relative to the camera, the process begins by obtaining bounding boxes for both the real event image and the matching synthetic template images using a

YOLO-based detector. Once the correct template corresponding to the estimated rotation of the query event image is identified, the depth can be inferred by comparing the scale of the bounding boxes. Given the known camera intrinsic matrix for both the template and the event image, the relative scale provides an estimate of the object's depth. This allows for the computation of the full 3D translation vector, (4) shows how the depth is computed.

$$depth = t_{\text{template},z} \times \frac{\| bb_{\text{template}} \|}{\| bb_{\text{event}} \|} \times \frac{f_{\text{event}}}{f_{\text{template}}}. \tag{4}$$

where $\|bb\|$ denotes the diagonal length of the bounding box in both the synthetic template image and the real event query image, after these images are processed by an object detector, f is the focal length of those images.

After obtaining the depth of the real event query image, the other translation components *X,Y* are estimated using the following equation:

$$\Delta\hat{t} = \hat{t}_{\text{event},z}\, K^{-1}_{\text{event}}\, bb_{\text{event},c} \;-\; \hat{t}_{\text{template},z}\, K^{-1}_{\text{template}}\, bb_{\text{template},c}. \tag{5}$$

Where K is the intrinsic matrix and $bb_{\text{template},c}$, $bb_{\text{event},c}$ are the center pixel locations of the bounding box in homogeneous coordinates. The final translation vector is obtained as such:

$$T = depth + \Delta\hat{t} \tag{6}$$

This approach was similarly applied in other template matching approaches such as [12].

IV. Results

A. Network Details and Parameters Used

The network architecture is shown in Fig. 5. The AEE network comprises of five convolutional layers with 256×256 input, followed by an attention layer to capture long-range dependencies. The embedding size is then reduced to $E = 512$ through two successive linear layers. The reconstruction network comprises five deconvolution layers to restore the image dimensions for comparison with the ground truth using the mean squared error (MSE). Training was performed with a batch size of 32 for 200 epochs using an NVIDIA RTX 3070. The learning rate was 2×10^{-4} with weight decay 5×10^{-5}. Data was split 70% for training and 30% for validation per object, and each object was trained separately to learn its distinctive rotational features. YOLOv11 [31] was used for object detection, with a confidence threshold ensuring the object had sufficient features before passing to AEE. The AEE network achieves prediction in 5 ms for inference and retrieval due to its lightweight design, making it suitable for high-frame-rate event cameras. The event frames were constructed by integrating 5,000 events per frame with a 500-event sliding window, a configuration selected based on empirical evaluations conducted across diverse scenarios.

B. Evaluation Metrics

In order to evaluate the performance of the developed method, the **Average distance** $ADD - S$ metric was used, which was proposed in [17]. The $ADD - S$ metric is derived from the ADD metric which uses the predicted rotation R and translation T to transform the 3D model points and compute the mean of the pair-wise distance between the predicted and ground-truth transformation. In $ADD - S$ the symmetries in the objects are taken into account and the closest point distance is computed for each predicted transformation and ground-truth transformation of the 3D model points.

$$\text{ADD-S} = \frac{1}{m} \sum_{\mathbf{p}_1 \in \mathcal{M}} \min_{\mathbf{p}_2 \in \mathcal{M}} \left\| (\mathbf{R}\mathbf{p}_1 + \mathbf{T}) - (\tilde{\mathbf{R}}\mathbf{p}_2 + \tilde{\mathbf{T}}) \right\| \tag{7}$$

m denotes the total number of points, and $\mathcal{M}$ denotes the set of points belonging to the CAD model of the object. $\mathbf{R}$ and $\mathbf{T}$ represent the ground-truth rotation and translation, respectively, while $\tilde{\mathbf{R}}$ and $\tilde{\mathbf{T}}$ are the predicted estimates. In this work, the AEE framework allows us to directly predict the object's rotation and subsequently estimate its translation. To fairly evaluate the network's rotational inference capability, We report the $ADD\text{-}S$ 10% threshold success rate using rotation only and full transformation to evaluate each module independently.

When computing $ADD\text{-}S(\text{Rotation})$, using the predicted rotation $\tilde{\mathbf{R}}$ together with the ground-truth translation $\mathbf{T}$ in most cases achieves a high success rate, demonstrating that the network is highly accurate in predicting the object's orientation. However, when evaluating the full $ADD\text{-}S$ metric that incorporates both the predicted rotation and translation $(\tilde{\mathbf{R}}, \tilde{\mathbf{T}})$, performance drops primarily due to translation errors. These errors often originate from factors unrelated to the rotation inference itself, such as inaccuracies in the detector's estimated bounding box, uncertainties in the camera intrinsic parameters obtained during calibration, missed alignment of the bounding box center and the object center in unsymmetrical objects and depth ambiguities in the generated template.

C. Performance

1) Qualitative Evaluation: The qualitative evaluation done in this work is to visually evaluate the retrieved template index that best matches the event image embeddings passed to the network. As seen in Fig. 7, all 10 objects were tested with random samples from the E-POSE dataset [32]. The network retrieved samples that resemble the orientation of the object in the events domain. Although the event images are in some cases very sparse and noisy, the network was able to surpass the noise in the event image and not be affected by the discontinuities available in the object shape.

2) Quantitative Evaluation: Table I presents the $ADD\text{-}S$ 10% threshold success rate using rotation only and full transformation across all scenarios outlined in Table II for 10 different objects. The best-performing objects were the cracker box, bleach, sugar box, and nine-hole peg. Their strong performance is attributed to their distinctive features, such as irregular distinctive shapes in different orientations and non-uniform color patterns, which enabled the event camera to capture a rich set of events for each object. This facilitated accurate matching by AEE, resulting in ADD-S success values across all scenarios of 86.0%, 83.2%, 82.2% and 61.8% respectively.

The scissors and wrench also performed reasonably well achieving an accuracy of 53.0% and 60.4% across all scenarios respectively; however, due to their smaller size and the limited number of distinguishable features captured by the event camera, accurate template inference was more challenging in many samples. In contrast, the wood block, drill, and mustard bottle exhibited weaker performance with a 2.5%, 35.5% and 25.7% overall accuracy, primarily due to their texture-less surfaces and, in the case of the drill and mustard bottle, color uniformity, which limited the camera's ability to capture fine-grained event details necessary for accurate template matching.

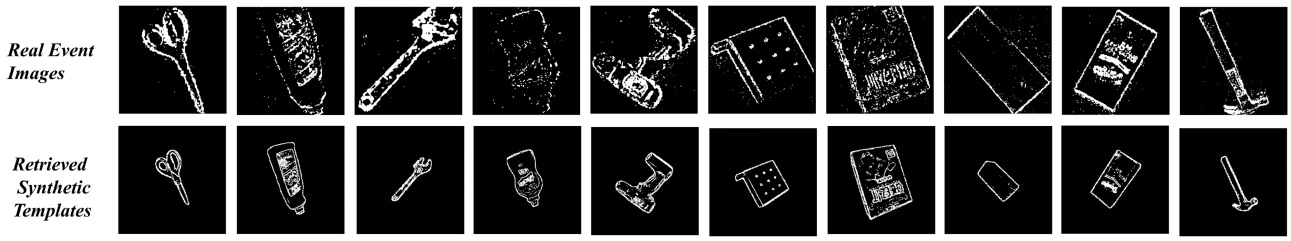


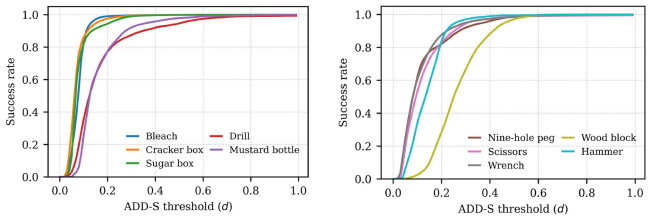
Fig. 7. First row shows the real event images passed to the template matching network, and the second row shows the templates retrieved from AEE matching the rotation of the object.

TABLE I
ADD-S 10% SUCCESS RATE FOR 10 OBJECTS. THE ROTATION SUCCESS RATE IS REPORTED SEPARATELY WITH PARENTHESES SHOWING FULL POSE SUCCESS RATE INCLUDING ROTATION AND TRANSLATION

Scenarios	1	2	3	4	5	6	7	8	9	10	11	12	Average
Bleach	99.7% (95.5%)	98.9% (90.2%)	99.7% (96.2%)	98.6% (94.3%)	99.6% (95.7%)	99.0% (93.5%)	99.9% (74.2%)	100% (80.9%)	100% (85.0%)	100% (67.9%)	100% (80.5%)	99.8% (49.7%)	99.6% (83.6%)
Wood block	72.1% (2.6%)	67.4% (3.1%)	71.1% (4.5%)	68.8% (2.7%)	70.6% (5.1%)	68.9% (4.8%)	74.3% (0%)	83.6% (1.1%)	75.5% (3.3%)	80.3% (0.2%)	78.7% (0%)	83.9% (2.5%)	74.6% (2.5%)
Cracker box	97.7% (91.1%)	89.7% (75.2%)	98.4% (95.9%)	96.7% (87.9%)	98.6% (96.3%)	97.9% (89.0%)	99.8% (93.0%)	98.3% (79%)	99.8% (95.1%)	99.5% (72.3%)	99.9% (94.9%)	94.3% (62.3%)	97.6% (86.0%)
Sugar box	99.7% (92.1%)	95.7% (77.7%)	99.7% (94.3%)	98.3% (91.5%)	99.6% (91.4%)	98.1% (86.8%)	96.9% (81.2%)	92.7% (80.2%)	97.7% (87.6%)	75.8% (61.8%)	97.7% (82.3%)	92.8% (59.4%)	93.4% (82.2%)
Drill	75.3% (45.4%)	34.3% (15.5%)	80.2% (35.9%)	52.5% (33.3%)	84.6% (32.8%)	59.9% (30.2%)	89.7% (54.3%)	27.2% (8.4%)	84.7% (62.3%)	49% (26%)	83.3% (57.5%)	51% (24.3%)	64.3% (35.5%)
Hammer	57.1% (46.0%)	28.6% (17.0%)	58.4% (44.2%)	44.5% (21.7%)	59% (42.1%)	42% (25.1%)	49% (30.6%)	62.7% (37.7%)	52.3% (39%)	66.8% (43.4%)	56.5% (38.3%)	60.7% (21.7%)	53.1% (34.0%)
Mustard bottle	82.9% (36%)	5.2% (0.6%)	79.3% (43.1%)	75.7% (29.5%)	79% (44.1%)	70.8% (23.6%)	98.7% (20%)	89.9% (22.2%)	94.4% (27.7%)	85.5% (20.3%)	90.4% (27.7%)	78.8% (13.3%)	77.6% (25.7%)
Nine-hole peg	99.6% (93.9%)	96.4% (46.6%)	99.8% (96.1%)	97.8% (62.3%)	99.7% (96.6%)	95.1% (65.4%)	99.2% (79.2%)	98.5% (50.5%)	99.8% (56.1%)	92.3% (20.1%)	99.9% (55.6%)	91.4% (18.8%)	91.4% (61.8%)
Scissors	94.5% (68.4%)	87.4% (43.2%)	94.5% (58.4%)	94.1% (61.1%)	93.2% (57.0%)	91.9% (53.6%)	97.6% (45.7%)	100% (62.8%)	100% (43.7%)	100% (42.6%)	99.9% (50.6%)	99.7% (48.4%)	96.1% (53.0%)
Wrench	93.2% (72.3%)	93.9% (49.1%)	89.5% (69.7%)	90.3% (57.1%)	88.9% (65.6%)	91.6% (55.9%)	99.9% (74%)	99.9% (60.9%)	100% (76.8%)	99.9% (37%)	100% (70.3%)	99.8% (36.6%)	95.6% (60.4%)

TABLE II
DESCRIPTIONS OF ALL SCENARIOS BASED ON MOTION TYPE, LIGHTING CONDITION, AND SPEED

Scenario #	Motion	Lighting	Speed
1	*R+T*	830 lux	0.1 m/s
2	*R+T*	170 lux	0.1 m/s
3	*R+T*	830 lux	0.5 m/s
4	*R+T*	170 lux	0.5 m/s
5	*R+T*	830 lux	1 m/s
6	*R+T*	170 lux	1 m/s
7	*T*	830 lux	0.1 m/s
8	*T*	170 lux	0.1 m/s
9	*T*	830 lux	0.5 m/s
10	*T*	170 lux	0.5 m/s
11	*T*	830 lux	1 m/s
12	*T*	170 lux	1 m/s



(a) Success rate for Bleach, Cracker Box, Sugar Box, Drill and Mustard Bottle

(b) Success rate for Nine-hole peg, Scissors, Wrench, Wood block and Hammer

Fig. 8. ADD-S success rate under different d thresholds

A slight decrease in performance was observed under diminished-light scenarios (2, 4, 6, 8, 10, 12), attributed to the significantly lower signal-to-noise ratio in these conditions. This degradation can be mitigated by employing event denoising algorithms that better preserve fine-grained object features. Notably, increasing the dynamics of the scene did not negatively impact performance, underscoring the advantage of event cameras in handling high-speed motion scenarios.

It should be noted that the separation between the results of ADD-S (Rotation) and ADD-S full pose in the Table. I is mainly to isolate the errors induced by the detector. Due to the low resolution of the event camera images and template images used a 346×260 camera resolution with a $1m$ depth would cause a 3 mm error per pixel. Therefore, a very small error in the detector could lead to an error in centimeters. This was particularly evident in the drill case, where a consistent offset between the drill's center and its bounding box introduced a bias that was accounted for to obtain an accurate full transformation estimate. Additionally, Fig. 8 shows the change in success rate for all objects averaged across all scenarios as the threshold is changed. Other than the wood block case, even a small increase in the threshold (e.g., to 0.2 d) leads to a considerable rise in the

TABLE III
EFFECT OF MHA AND DOMAIN RANDOMIZATION ON ADD-S 10% FULL-POSE SUCCESS RATE ACROSS ALL OBJECTS AND SCENARIOS

Objects	AEE	w/o MHA	w/o DR
Bleach	83.6%	81.6 %	19.2%
Wood block	2.5%	0%	0%
Cracker box	86%	85.9%	2.6%
Sugar box	82.2%	80.9%	1.6%
Drill	35.5	34.6 %	5.85%
Hammer	34%	34%	4%
Mustard bottle	25.6%	12.6%	0.3%
Nine-hole peg	61.8%	58%	24.5%
Scissors	53%	45.6%	9%
Wrench	60.4%	59.8%	9.4%

success rate, with almost all objects reaching around 80%. The ADD 10% overall success rate across all scenarios was also computed for rotation for all objects except the wood block since its symmetric. The metric was applied on the objects bleach, cracker box, sugar box, drill, hammer, mustard bottle, nine-hole peg, scissors and wrench. The results were 95.2 %, 81.7%, 26.8%, 21.8%, 52.4%, 11.04%, 60.2%, 93.2% and 84.4% respectively.

D. Ablation Study

An ablation study was additionally conducted to study the importance of different blocks in the template matching network developed. Mainly, the effect of the attention (MHA) and domain randomization in Table III. All objects were included in the study to evaluate the effect of both the MHA block and domain randomization. The ablation was performed across all objects, and the $ADD\text{-}S$ 10% threshold success rate was reported. To assess the impact of domain randomization on the Augmented Event Encoder (AEE), we retrained the network with edge images without the domain randomization techniques proposed. The impact of domain randomization is clearly evident, as the results obtained without it were significantly lower than those achieved with AEE. Although the effect of the MHA block is less evident, however, it still resulted in marginal improvements for certain objects such as the scissors and mustard bottle.

E. Benchmark

Benchmarking the obtained results against other methods is essential to assess the performance of AEE. To this end, the network's performance was compared on a subset of the data with other RGB-based approaches, specifically YOLO-6D [33], PoseCNN [17], and EfficientPose [34]. These networks were adapted to the event domain by fine-tuning them on event-based data. Table IV shows the results of the comparison performed. It can be seen the AEE outperforms other RGB based methods on all object except the wood block and is able to estimate the full pose. The inference times are also reported for each network, showing the lightweight AEE achieving lower inference time compared to other networks as per their reported inference times.

In addition to comparing AEE with RGB-based methods, we also evaluated it against LOPET [14], a line-based pose estimation approach operating on the same event data modality. The method, however, struggled to estimate the pose for several objects due to the lack of clear lines to track. Furthermore, the reduction in image resolution from the 1280×720 resolution on which LOPET [14] was originally optimized to 346×260 in our dataset further increased the difficulty of the task.

TABLE IV
COMPARISON OF ADD-S 10% FULL-POSE SUCCESS RATE AND INFERENCE TIME ACROSS ALL OBJECTS WITH YOLO-6D [33], POSECNN [17] AND EFFICIENTPOSE [34]

Object	AEE (Ours)	YOLO-6D[34]	PoseCNN[17]	EfficientPose[35]
Bleach	**95.7%**	0.4%	85.4%	3.2%
Wood block	5.1%	0%	**14.51%**	3.74%
Cracker box	**96.3%**	0.2%	0%	15.4%
Sugar box	**91.4%**	0.35%	60.27%	16.4%
Drill	**32.8%**	0.2%	0%	0%
Hammer	**42.1%**	0%	6.65%	11.8%
Mustard bottle	**44.1%**	0.8%	10.02%	2.1%
Nine-hole peg	**96.6%**	0.14%	1.13%	36.79%
Scissors	**57.0%**	0%	4.61%	0.46%
Wrench	**65.6%**	0.4%	0%	11.37%
Inference Time (ms, approx.)	5	20	200	37

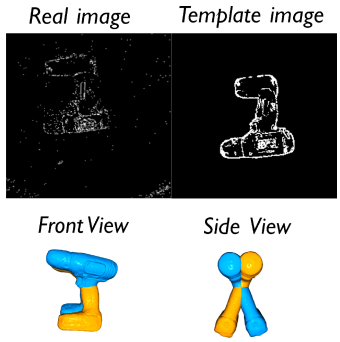


Fig. 9. Front view and side view of the drill object between the real event image pose and the template image.

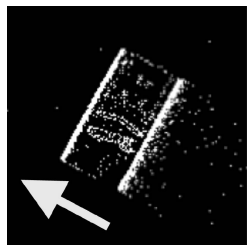


Fig. 10. Real event image with camera motion (arrow). Motion-parallel edges are weak, leading to incorrect template retrieval.

F. Limitation and Future Work

The method developed, although it exhibits accurate matching in many cases as seen in Fig. 7, still has some failure cases that were noticed during the evaluation. Out of plane rotation: Fig. 9 illustrates an example of a failure case, showing the real event image alongside the corresponding template inferred by the template matching network. While the inferred template aligns well with the real event image in terms of yaw and pitch angles, accurately estimating the roll angle proved challenging. This difficulty is primarily due to the sparsity of events captured by the event camera, which limits the availability of fine-grained spatial features needed to resolve rotational ambiguities, particularly around the roll axis. As a result, although the network could correctly infer major orientation components, subtle misalignment in roll led to an increase in the overall pose error.

Missing Edges: The other failure case that was noticed was when the camera motion is directly parallel to the edges of the object. This scenario which can be seen in Fig. 10 shows how the

edges perpendicular to the motion are captured vividly, However the parallel edges are barely captured by the event camera. The features extracted from such an image do not represent the rotation of the object, which leads to errors in the template matching procedure. In the future, we aim to look into the event representation that best suits such an approach to circumvent the lack of features in event images, such as motion-compensated images. In addition, additional augmentation techniques will be adopted to enhance the domain randomization approach and bridge the gap between synthetic and real images.

V. Conclusion

This work looked into object pose estimation using event cameras through a template matching approach. The work utilized an image generated from RGB solely to train a template matching network that was induced with domain randomization to bridge the domain gap between RGB images and event images. The network was able to retrieve the best-matching template that represents a specific rotation and thus get an accurate estimate of the rotation of the object in the event image. The translation was also estimated and the ADD-S (Rotation) and ADD-S 10% full pose success rates were presented for each object under 12 different scenarios that exhibited light variation and motion variations.

References

[1] S. Thalhammer, D. Bauer, P. Hönig, J.-B. Weibel, J. García-Rodríguez, and M. Vincze, "Challenges for monocular 6-d object pose estimation in robotics," *IEEE Trans. Robot.*, vol. 40, pp. 4065–4084, 2024.

[2] G. Gallego et al., "Event-based vision: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 1, pp. 154–180, Jan. 2022.

[3] H. Liu, S. Peng, L. Zhu, Y. Chang, H. Zhou, and L. Yan, "Seeing motion at nighttime with an event camera," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2024, pp. 25648–25 658.

[4] N. Zubić, D. Gehrig, M. Gehrig, and D. Scaramuzza, "From chaos comes order: Ordering event representations for object recognition and detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 12846–12856.

[5] C. Zhao, Y. Li, and Y. Lyu, "Event-based real-time moving object detection based on imu ego-motion compensation," in *Proc. IEEE Int. Conf. Robot. Automat*, 2023, pp. 690–696.

[6] M. Gehrig, M. Millhäusler, D. Gehrig, and D. Scaramuzza, "E-RAFT: Dense optical flow from event cameras," in *Proc. Int. Conf. 3D Vis.*, 2021, pp. 197–206.

[7] J. Jiao, H. Huang, L. Li, Z. He, Y. Zhu, and M. Liu, "Comparing representations in tracking for event camera-based SLAM," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1369–1376.

[8] Y. Labbé et al., "MegaPose: 6D pose estimation of novel objects via render & compare," in *Proc. Conf. Robot Learn.*, 2023, pp. 715–725.

[9] V. N. Nguyen, T. Groueix, M. Salzmann, and V. Lepetit, "GigaPose: Fast and robust novel object pose estimation via one correspondence," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*, 2024, pp. 9903–9913.

[10] V. Nguyen 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6771–6780.

[11] Y. Wen, H. Pan, L. Yang, and W. Wang, "Edge enhanced implicit orientation learning with geometric prior for 6D pose estimation," *IEEE Robot. Automat. Lett.*, vol. 5, no. 3, pp. 4931–4938, Jul. 2020.

[12] M. Sundermeyer et al., "Implicit 3D orientation learning for 6D object detection from RGB images," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 699–715.

[13] D. Reverter Valeiras et al., "Neuromorphic event-based 3D pose estimation," *Front. Neurosci.*, vol. 9, 2016, Art. no. 522. [Online]. Available: https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2015.00522

[14] 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 Trans. Image Process.*, vol. 33, pp. 4765–4780, 2024.

[15] A. Glover, L. Gava, Z. Li, and C. Bartolozzi, "EDOPT: Event-camera 6-dof dynamic object pose tracking," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2024, pp. 18200–18206.

[16] G. Goyal, F. Di Pietro, N. Carissimi, A. Glover, and C. Bartolozzi, "MoveeNet: Online high-frequency human pose estimation with an event camera," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2023, pp. 4024–4033.

[17] Y. Xiang et al., "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," in *Proc. Robot., Sci. Syst. XIV*, 2018.

[18] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1521–1529.

[19] Y. Li et al., "DeepIM: Deep iterative matching for 6D pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 695–711.

[20] X. Lin et al., "TransPose: 6D object pose estimation with geometry-aware transformer," *Neurocomputing*, vol. 589, 2024, Art. no. 127652.

[21] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6D object pose prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 292–301.

[22] Z. He et al., "ContourPose: Monocular 6D pose estimation method for reflective textureless metal parts," *IEEE Trans. Robot.*, vol. 39, no. 5, pp. 4037–4050, Oct. 2023.

[23] X. Yang et al., "Er-Pose: Learning edge representation for 6D pose estimation of texture-less objects," *Neurocomputing*, vol. 515, pp. 13–25, 2023.

[24] Y. Zhang, Y. Liu, Q. Wu, J. Zhou, X. Gong, and J. Wang, "EANet: Edge-attention 6D pose estimation network for texture-less objects," *IEEE Trans. Instrum. Meas.*, vol. 71, 2022, Art. no. 2504413.

[25] Online Blender Community, "Blender - A 3D modelling and rendering package," Blender Foundation, Blender Institute, Amsterdam, (n.d.), 2018. [Online]. Available: http://www.blender.org

[26] M. Denninger et al., "BlenderProc2: A procedural pipeline for photorealistic rendering," *J. Open Source Softw.*, vol. 8, no. 82, 2023, Art. no. 4901, doi: 10.21105/joss.04901.

[27] S. Zakharov, W. Kehl, and S. Ilic, "DeceptionNet: Network-driven domain randomization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 532–541.

[28] A. Loquercio, E. Kaufmann, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Deep drone racing: From simulation to reality with domain randomization," *IEEE Trans. Robot.*, vol. 36, no. 1, pp. 1–14, Feb. 2020.

[29] D. Gehrig, M. Gehrig, J. Hidalgo-Carrió, and D. Scaramuzza, "Video to events: Recycling video datasets for event cameras," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 3586–3595.

[30] K. He et al., "Masked autoencoders are scalable vision learners," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*, 2022, pp. 16000–16009.

[31] G. Jocher et al., "Ultralytics YOLO," Jan. 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[32] O. A. Hay et al., "E-pose: A large scale event camera dataset for object pose estimation," *Sci. Data*, vol. 12, no. 1, 2025, Art. no. 245.

[33] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6D object pose prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 292–301.

[34] Y. Bukschat and M. Vetter, "EfficientPose: An efficient, accurate and scalable end-to-end 6D multi object pose estimation approach," 2020, *arXiv:2011.04307*.