
Fully Spiking Neural Networks for Unified Frame-Event Object Tracking

Jingjun Yang* Liangwei Fan* Jinpu Zhang† Xiangkai Lian Hui Shen† Dewen Hu

College of Intelligence Science and Technology

National University of Defense Technology

{yjingjun, fanliangwei, zhangjinpu, lianxiangkai, shenhui, dwhu}@nudt.edu.cn

Codes and models: <https://github.com/Noctis-A/SpikeFET>

Abstract

The integration of image and event streams offers a promising approach for achieving robust visual object tracking in complex environments. However, current fusion methods achieve high performance at the cost of significant computational overhead and struggle to efficiently extract the sparse, asynchronous information from event streams, failing to leverage the energy-efficient advantages of event-driven spiking paradigms. To address this challenge, we propose the first fully Spiking Frame-Event Tracking framework called SpikeFET. This network achieves synergistic integration of convolutional local feature extraction and Transformer-based global modeling within the spiking paradigm, effectively fusing frame and event data. To overcome the degradation of translation invariance caused by convolutional padding, we introduce a Random Patchwork Module (RPM) that eliminates positional bias through randomized spatial reorganization and learnable type encoding while preserving residual structures. Furthermore, we propose a Spatial-Temporal Regularization (STR) strategy that overcomes similarity metric degradation from asymmetric features by enforcing spatio-temporal consistency among temporal template features in latent space. Extensive experiments across multiple benchmarks demonstrate that the proposed framework achieves superior tracking accuracy over existing methods while significantly reducing power consumption, attaining an optimal balance between performance and efficiency.

1 Introduction

Visual target tracking has important application value in areas such as automatic driving and intelligent surveillance. Although traditional frame-based tracking methods [1, 2, 3, 4] demonstrate satisfactory performance in conventional scenarios, robust tracking under complex environments involving low-light conditions, over-exposure, and high-speed motion remains a challenge. Event cameras offer new possibilities for tracking tasks due to their high dynamic range, microsecond response, and low power consumption [5], but their sparse and asynchronous characteristics makes it difficult to capture target appearance features. Some researchers [6, 7, 8] have attempted to improve the performance and robustness of tracking by integrating motion information from frames with appearance information from events. However, most Artificial Neural Networks (ANNs)-based methods rely on dense texture information for tracking and struggle to effectively model sparse event data. In contrast, Spiking Neural Networks (SNNs) demonstrate exceptional compatibility and ultra-low energy consumption advantages in event stream processing through their bio-inspired spatiotemporal dynamics and event-driven sparse computational paradigm [9, 10]. Recently, an

*These authors contributed equally to this work.

†Correspondence

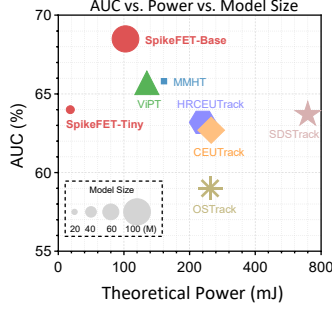


Figure 1: SpikeFET versus other tracking methods on COESOT.

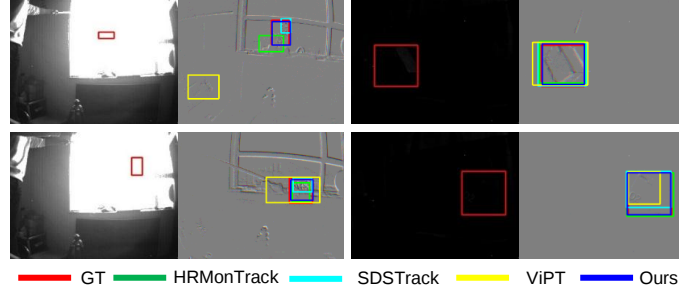


Figure 2: Visualization results of SpikeFET-Base in low light and overexposure scenarios.

E-SpikeFormer [11] based on Metaformer [12] demonstrated that SNNs can achieve comparable performance to classical ANN-based vision transformer models in classification tasks. However, the application of SNNs in object tracking remains underdeveloped, facing critical bottlenecks in architecture-modality co-optimization. Current research predominantly focuses on ANN-SNN hybrid architectures for unimodal tracking [13, 14]. While [15] extends this framework to multimodal tracking, it suffers from compromised energy efficiency. Notably, SDTrack [16] implements a fully spike-driven network but remains constrained to unimodal event-stream inputs, with interpolated noise in feature alignment causing error accumulation and performance degradation.

To address these challenges, we develop a novel fully **Spiking Frame-Event Tracking** framework, named **SpikeFET**. We develop a hierarchical tracking architecture with dual-branch feature extraction, single-branch fusion and dual-branch prediction modules. This design effectively integrates local feature representation of convolutional networks with global modeling capabilities of Transformers, enhancing cross-modal fusion performance. In order to address the adverse impact of positional bias introduced by padding [17, 18] in the convolution module on tracking accuracy, we propose a simple yet effective Randomized Patchwork Module (RPM). This module innovatively reorganizes target spatial distributions by randomly composing initial templates, online updated templates, and search frames into fused rectangular patches, while introducing learnable type encoding to explicitly identify the logical positions of each image patch. Without removing padding in residual units, RPM effectively mitigates padding-induced degradation of translation invariance, significantly enhancing model robustness against target positional variations. In addition, to further address the asymmetric boundary features of adjacent temporal template frames caused by RPM, we develop a Spatial-Temporal Regularization (STR) strategy that enhances both the temporal consistency between dual template target features in the latent space and the spatial consistency of boundary features, thereby improving the robustness of similarity measurement. In particular, SpikeFET-Tiny achieves a 2.7% higher AUC score on the FE108 [8] dataset compared to the state-of-the-art SDSTrack [19], while demonstrating 39× lower power consumption.

In summary, the main contributions of this paper can be summarized as follows:

- We propose a unified frame-event spiking tracker termed SpikeFET. To the best of our knowledge, it is the first work that employs a fully spiking neural network for unified frame-event object tracking.
- We propose a simple yet effective RPM that mitigates positional bias introduced by convolutional padding through randomized spatial reorganization, thereby significantly enhancing the model’s robustness to target position variations.
- We develop a STR strategy that overcomes similarity degradation from asymmetric features by enforcing spatio-temporal consistency among temporal template features in latent space, enhancing the tracking accuracy and stability of the model.
- Extensive experimental validation on multiple frame-event tracking benchmarks fully validates the effectiveness of our proposed SpikeFET, and achieves an optimal balance between computational power consumption and performance and parameters, as shown in Fig. 1.

2 Related Work

Visual Object Tracking. In recent years, deep learning has driven remarkable advancements in visual target tracking. Current mainstream approaches fall into three categories: First, CNN-based trackers like [1, 17] utilize convolutional backbones for feature extraction and employ correlation mechanisms for target localization. However, their limited receptive fields restrict long-range dependency modeling, leading to reduced robustness in scenarios with rapid motion and occlusions. Second, hybrid CNN-Transformer frameworks exemplified by TransT [2] and TMT [20] address this limitation by incorporating self-attention mechanisms into convolutional feature encoding, significantly improving spatio-temporal context modeling through complementary architecture design. Third, pure Transformer-based trackers [3, 21] demonstrate superior global relationship capture capabilities by unifying feature extraction and interaction within an attention-driven framework, showing promising potential for comprehensive context understanding.

Frame-Event based Object Tracking With the notable strengths of event data in low-light, fast-motion, and other complex scenes, frame-event fusion methods for single-object tracking have been gradually gaining attention in recent years. Some approaches [6, 7, 13, 22, 23] are based on cross-modal fusion frameworks, like CEUTrack [7], simplify traditional two-branch architectures with a unified Transformer for synchronized multimodal feature extraction. TENet [23] improves event feature representation with lightweight multi-scale pooling and cross-modal mutual guidance for enhanced complementarity. Meanwhile, other methods [24, 25] introduce prompt learning strategies, such as ViPT [25], which refines event features using pre-trained image models and attention mechanisms. In this paper, we propose a dual-single-dual framework for frame-event fusion, featuring independent feature extraction branches for each modality in the early stage and decoupled tracking heads after fusion to achieve efficient inter-modal complementarity and precise localization.

Spiking Neural Networks for Object Tracking SNNs show superior performance and efficiency in vision tasks [11, 26, 27, 28] through event-driven, low-power operation, excelling in time-sensitive domains like dynamic object tracking with bio-inspired asynchronous computation. Early frame-based SNN trackers [29, 30, 31] integrated Siamese networks with temporal encoding using multi-step methods, and then compressed SNNs into single step models using knowledge distillation. For event-driven tracking [13, 14, 32, 33], STNet [13], and SNNTrack [14] enhanced spatiotemporal feature extraction but faced challenges in scenarios with insufficient texture information. SDTrack [16] is the first transformer-based spike-driven tracker, but also lacks robustness to spatial offsets. Frame-event fusion models MMHT [15] utilized ANN-SNN hybrid architectures for spatiotemporal integration but did not fully exploit the energy-efficient advantages of SNNs. To overcome these limitations, we present SpikeFET, the first fully spiking frame-event tracker. By adopting a fully SNN-based multimodal fusion framework, SpikeFET fully unleashes the spatiotemporal modeling capabilities of spiking networks while preserving their energy-efficient computational properties.

3 Methodology

As shown in Fig. 3, the overall framework of SpikeFET adopts a dual-single-dual structure based on SNN transformer. First, the image frames and event temporal frames are randomly combined using the RPM to mitigate the padding effects during feature extraction by convolutional layers. Subsequently, the combined multi-modal images are fed into a feature extraction network, where ConvFormer Spike Blocks are employed to comprehensively extract features. Positional encoding, modality encoding, and type encoding are then incorporated into the feature embeddings, followed by direction-consistent concatenation of the two modal images. The TransFormer Spike Block jointly processes and correlates these embeddings. Meanwhile, STR is introduced for dual template frames to maintain spatiotemporal consistency. Finally, the resulting feature embeddings are delivered to an SNN-based center-tracking head for prediction, with a constraint to ensure consistency between the response maps generated by both modalities.

3.1 Spiking Frame-Event Tracking Network

In order to enhance the cross-modal feature fusion capability, we design a framework that integrates a dual branch feature extraction network and a dual modal response graph fusion mechanism. This framework consists of three core modules: a dual branch modal specific feature extraction network, a

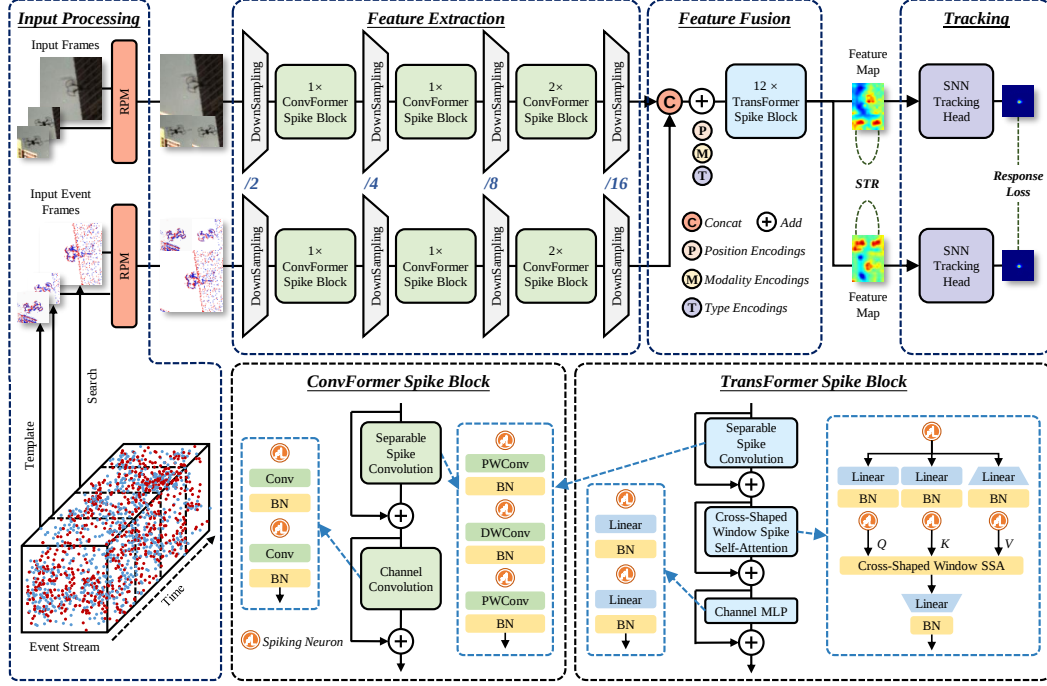


Figure 3: The overview of SpikeFET. Our SpikeFET consists of Input Processing, Feature Extraction, Feature Fusion and Tracking. Where the Event Stream is simply transformed into event temporal frames using [34]. Feature Extraction consists of DownSampling and ConvFormer Spike Block cascades. Feature Fusion consists of TransFormer Spike Block cascades. Tracking uses SNN Tracking Head, and Spiking Neuron use the Spike Firing Approximation (SFA) [11]

cross-modal fusion network, and a dual head decoupling tracking prediction with similarity fusion, as detailed in Fig. 3.

Modality-Specific Feature Extraction Network The proposed dual-branch modality-specific feature extraction network consists of an image branch and an event branch operating in parallel, where each branch comprises multiple stacked ConvFormer Spike blocks. Specifically, the ConvFormer Spike Block can be formulated as:

$$\mathbf{U}' = \mathbf{U} + \text{SepSpikeConv}(\mathbf{U}) \quad (1)$$

$$\mathbf{U}'' = \mathbf{U}' + \text{ChannelConv}(\mathbf{U}') \quad (2)$$

$$\text{SepSpikeConv}(\mathbf{U}) = \text{Conv}_{\text{pw}2}(\text{SN}(\text{Conv}_{\text{dw}}(\text{SN}(\text{Conv}_{\text{pw}1}(\text{SN}(\mathbf{U})))))) \quad (3)$$

$$\text{ChannelConv}(\mathbf{U}') = \text{Conv}(\text{SN}(\text{Conv}(\text{SN}(\mathbf{U}')))) \quad (4)$$

where $\text{SepSpikeConv}(\cdot)$ denotes the token mixer, $\text{ChannelConv}(\cdot)$ represents the channel mixer, $\text{Conv}_{\text{pw}1}(\cdot)$ and $\text{Conv}_{\text{pw}2}(\cdot)$ are pointwise convolutions, $\text{Conv}_{\text{dw}}(\cdot)$ refers to depthwise convolution [35], and $\text{Conv}(\cdot)$ indicates standard convolution. $\text{SN}(\cdot)$ stands for the spiking neuron layer. Note that BN layers are omitted here for notation simplicity.

Cross-Modal Fusion Network Before feeding the dual-modality feature maps into the fusion network, we introduce three standard learnable encodings: positional encoding E_p , modality encoding E_m , and type encoding E_t (see Sec.3.2). The dual-modality feature maps are then summed with their corresponding three encodings:

$$\mathbf{U}' = \mathbf{U} + E_p + E_m + E_t \quad (5)$$

The image feature map U_i and the event feature map U_e are then concatenated as $U = [U_i; U_e]$. The generated feature map U serves as the input to the fusion network for cross-modal interactive learning.

The cross-modal fusion network consists of Transformer Spike Blocks, which can be formulated as:

$$\mathbf{U}' = \mathbf{U} + \text{SepSpikeConv}(\mathbf{U}) \quad (6)$$

$$\mathbf{U}'' = \mathbf{U}' + \text{CSWin-SSA}(\mathbf{U}') \quad (7)$$

$$\mathbf{U}''' = \mathbf{U}'' + \text{ChannelMLP}(\mathbf{U}'') \quad (8)$$

where $\text{CSWin-SSA}(\cdot)$ denotes the Cross-Shaped Windows [36] Spiking Self-Attention module. Specifically, the CSWin-SSA module can be expressed as:

$$\mathbf{Q}_S = \text{SN}(\text{Linear}(\mathbf{U})), \mathbf{K}_S = \text{SN}(\text{Linear}(\mathbf{U})), \mathbf{V}_S = \text{SN}(\text{Linear}_\gamma(\mathbf{U})) \quad (9)$$

$$\mathbf{U}' = \text{Linear}_{\frac{1}{\gamma}}(\text{CSWinSSA}(\mathbf{Q}_S, \mathbf{K}_S, \mathbf{V}_S)) \quad (10)$$

$$\text{SSA}(\mathbf{Q}_S, \mathbf{K}_S, \mathbf{V}_S) = \text{SN}(\mathbf{Q}_S \mathbf{K}_S^\top \mathbf{V}_S * \text{scale}) \quad (11)$$

where $\text{Linear}(\cdot)$ denotes a linear transformation, γ is the channel expansion factor [37], $\text{CSWinSSA}(\cdot)$ represents the CSWinSSA operator, and $\text{SSA}(\cdot)$ is the spiking self-attention operator within the CSWinSSA operator. To address the challenge of large values generated by matrix multiplication, a scaling factor (scale) is introduced to regulate the magnitude of the results. Further details of the CSWinSSA operator are provided in the Appendix E.

Decoupled Tracking Prediction With Similarity Fusion In the design of the tracking head, we maintain structural continuity with the dual-modal feature extractor. Specifically, we construct the Decoupled Spike Tracking Heads similar to SDTrack [16], utilizing parallel dual branches to process heterogeneous modal features.

In object tracking tasks, the response maps output by the network play a critical role in target localization. To achieve sufficient fusion of image and event modalities, we propose enhancing feature alignment by constraining the similarity between the response maps of the two modalities, implemented via a weighted focal loss function $\mathcal{L}_{\text{GWF}}$ [38] (see the Appendix C.1 for more details). The final loss function is expressed as: $\mathcal{L}_{\text{Res}} = \mathcal{L}_{\text{GWF}}(\mathbf{R}_F/\tau, \mathbf{R}_E/\tau)$, where $\mathbf{R}_F$ and $\mathbf{R}_E$ are the response maps of two modes, τ is the temperature coefficient, and the empirical setting is 2.

3.2 Randomized Patchwork Module

The widely used padding in residual networks [17, 18] significantly degrades single-object tracking performance by breaking translation invariance, while traditional methods of removing padding [17, 18] would disrupt the network structure. To address this critical issue, while preserving the advantages of the original network core architecture, we have innovatively proposed a simple yet effective Randomized Patchwork Module (RPM). Specifically, prior to modality-specific feature extraction, Two adjacent temporal template images $Z_1, Z_2 \in \mathbb{R}^{T \times C \times H_z \times W_z}$ and the search image $X \in \mathbb{R}^{T \times C \times H_x \times W_x}$ are fed into the RPM. As shown in Fig. 4, RPM randomly employs either horizontal concatenation $U_h \in \mathbb{R}^{T \times C \times H_z \times (W_z + W_x)}$ or vertical concatenation $U_v \in \mathbb{R}^{T \times C \times (H_z + H_x) \times W_z}$ to generate rectangular fused inputs. This design randomly distributes the search image across four orientations (top, bottom, left, right) of the concatenated region, creating diversified input patterns. Meanwhile, we enforce consistent concatenation direction (horizontal or vertical) between image frames and event streams to ensure spatial consistency across modalities.

Additionally, as shown in Fig. 3, the learnable type encoding E_t explicitly identifies the logical positions of image patches through additive integration with visual features, eliminating spatial ambiguities caused by random patch permutation while suppressing noise. Through visual analysis of the heatmap distribution in output images (Fig. 6), the effectiveness of the module is clearly demonstrated.

3.3 Spatial-Temporal Regularization

Although RPM dynamically reorganizes the spatial distribution of input data to mitigate target localization bias, the boundary regions of adjacent temporal template frames (e.g., Z_1 and Z_2) still manifest asymmetric feature sources, such as distinct features from other images or padding regions induced by convolutional networks. As illustrated in Fig. 5, the left boundary of Z_1 corresponds to padding regions, while the left boundary of Z_2 coincides with the right region of Z_1 , while their bottom boundaries respectively align with different regions of the search frame X . This spatial distribution

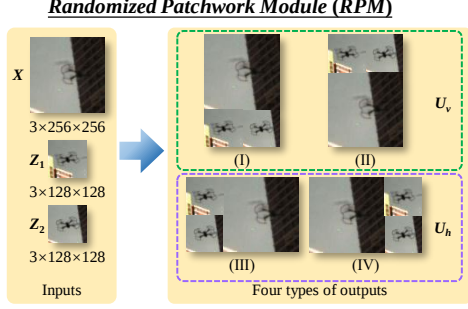


Figure 4: Detailed design of proposed RPM. Randomly combine template frames Z_1 and Z_2 with search frame X .

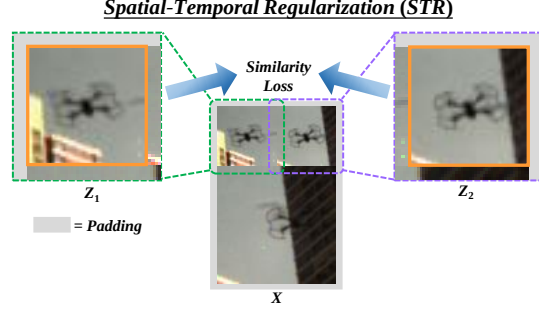


Figure 5: Illustration of the proposed STR. Applying similarity loss regularization to two temporally adjacent template frames.

inconsistency inherently disrupts the strict translation invariance of convolutional networks, leading to feature representation deviations at identical spatial positions across template frames Z_1 and Z_2 and consequently causing significant degradation in similarity measurement. Notably, we also found that constraining the similarity of latent feature space between adjacent template frames in the temporal domain can establish a more robust similarity metric space.

Building upon the aforementioned insights, we propose the Spatial-Temporal Regularization (STR) strategy to constrain the similarity measurement of dual-template frame features. The core idea behind this strategy is to leverage two temporally adjacent template frame features with spatial-temporal consistency for relation modeling with the search frame features, thereby enhancing model robustness. Specifically, after the cross-modal fusion network completes multi-modal information integration, we construct a similarity constraint on the fused features output by temporally adjacent template frames. By applying mean squared error (MSE) to regularize the feature discrepancies between dual templates, we establish a similarity supervision signal with spatial-temporal consistent characteristics. This constraint can be formulated as the following loss function:

$$\mathcal{L}_{\text{sim}} = \frac{1}{hw} \sum_{i=1}^h \sum_{j=1}^w \left(F_1^{(i,j)} - F_2^{(i,j)} \right)^2 \quad (12)$$

where $F_1 \in \mathbb{R}^{h \times w}$ and $F_2 \in \mathbb{R}^{h \times w}$ denote the output feature maps of two temporally adjacent template frames after passing through the feature fusion network, where h and w represent the spatial dimensions of the feature maps. The loss function guides the network to extract more generalizable feature representations from spatiotemporally continuous video data by minimizing the discrepancies between spatially corresponding points in the dual-template features.

3.4 Training and Inference

For training, we adopt the same training procedure as OTrack [3], employing weighted focal loss [38] for classification and a combination of L1 loss and generalized IoU loss [39] for regression. For cross-modal fusion and the STR strategy, we employ the similarity-based loss functions $\mathcal{L}_{\text{Res}}$ and $\mathcal{L}_{\text{Sim}}$ as previously described. Finally, the overall loss function is formulated as:

$$\mathcal{L} = \mathcal{L}_{\text{cls}}^F + \lambda_{\text{iou}} \mathcal{L}_{\text{iou}}^F + \lambda_{\text{L1}} \mathcal{L}_{\text{L1}}^F + \mathcal{L}_{\text{cls}}^E + \lambda_{\text{iou}} \mathcal{L}_{\text{iou}}^E + \lambda_{\text{L1}} \mathcal{L}_{\text{L1}}^E \quad (13)$$

$$\mathcal{L}_{\text{track}} = \mathcal{L} + \alpha \mathcal{L}_{\text{res}} + \beta \mathcal{L}_{\text{sim}} \quad (14)$$

where $\lambda_{\text{iou}} = 2$ and $\lambda_{\text{L1}} = 5$ are regularization parameters as in [4]. $\alpha = 1$ and $\beta = 0.5$ are hyperparameters used to balance the contributions of different loss functions. Ablation studies on hyperparameters are shown in Appendix F.

For inference, we perform a weighted summation and fusion of the image frame and event temporal frame response maps output by the dual-head decoupled tracking head to produce robust target localization results. This is expressed as:

$$\mathcal{R} = \lambda \mathcal{R}_F + (1 - \lambda) \mathcal{R}_E \quad (15)$$

where $\lambda = 0.5$ is a hyperparameter to balance the responses of different modalities. It should be noted that we do not employ a dynamic template update strategy. Instead, we consistently replicated

the first frame to use two template frames (both derived from the first frame) and one search frame for inference. At the same time, we follow common practice by applying a Hanning window penalty to leverage positional priors in tracking [2, 3].

4 Experiments

4.1 Implementation details

Our proposed SpikeFET was implemented with PyTorch 1.12 in Python 3.8 and trained on two NVIDIA RTX 4090 GPUs. The network input consists of a triplet image group comprising one search image and two distinct template images. When constructing training samples from randomly sampled video sequences, we expanded the search regions and template regions to $4\times$ and $2\times$ their original bounding box sizes, respectively, followed by resizing them to 256×256 and 128×128 . Common data augmentation techniques such as horizontal flipping and brightness jittering were applied to the image sets. The model was trained using the AdamW optimizer. The optimizer has a cosine annealing scheduling over 50 epochs, where each epoch contained 60,000 image triplets. Please refer to the Appendix D for more details.

Similar to current mainstream multi-modal object tracking frameworks (such as VIPT [25], SD-Strack [19], TENet [23]), which commonly adopt OStrack [3] frame-datasets pre-trained models for parameter fine-tuning or knowledge distillation, we also follow their standard configuration process. Specifically, we adapt our SpikeFET into a single-modal SpikeET model, pre-trained on frame-datasets such as COCO [40], LaSOT [41], TrackingNet [42], and GOT-10K [43] to fine-tune our SpikeFET network. For the single-modal SpikeET model, we remove one feature extraction branch and the spiking tracking head branch from SpikeFET, along with eliminating modality encoding. To balance speed and accuracy, we propose three variants with distinct architectures and parameters: SpikeFET-Base (Fig. 3) with larger parameter capacity, and SpikeFET-Tiny and SpikeET optimized for computational efficiency. Among these, only SpikeFET-Base undergoes frame-based dataset pre-training, while SpikeFET-Tiny and SpikeET adopt ImageNet-1K [44] to pre-train the backbone, consistent with OStrack [3]. To quantitatively evaluate the performance of tracking systems, we employ three core metrics: Area Under the Curve (AUC), Precision Rate (PR), and theoretical power consumption. For a detailed description of the theoretical power consumption calculation method, please refer to Appendix C.2.

4.2 Datasets

We evaluated our proposed SpikeFET using three large-scale frame-event single-object tracking datasets: FE108 [8], VisEvent [6], and COESOT [7]. All three datasets were captured using the DAVIS346 camera, which features a spatial resolution of 346×260 , a dynamic range of 120 dB, and a minimum latency of 20 μ s.

Specifically, The FE108 dataset [8] contains 108 synchronized frame-event sequences captured in indoor environments, totaling approximately 1.5 hours of data. It covers 21 distinct target objects and is divided into 76 training sequences and 32 testing sequences. The target bounding box annotations were generated using a Vicon motion capture system. The VisEvent [6] dataset originally collected 820 frame-event pairs, divided into 500 training sequences and 320 testing sequences. Following [13], we removed sequences with missing event data or misaligned timestamps, resulting in a refined dataset of 205 training sequences and 172 testing sequences. Multimodal networks and unimodal networks use 320 test sequences and 172 test sequences, respectively. COESOT [7], as a larger-scale benchmark, contains 578K frame-event pairs divided into 827 training sequences and 527 testing sequences. Collected across diverse indoor and outdoor scenarios, it covers 90 target categories annotated with 17 challenging visual attributes. All target bounding boxes were manually annotated.

For fair comparison, we trained SpikeFET using the full training set of VisEvent and evaluated it on the test set containing 320 sequences, while SpikeET was trained on a pruned version of the VisEvent training set and tested on a subset of 172 sequences. Note that on SpikeFET, we only use raw event frames from the dataset on VisEvent, not event temporal frames.

4.3 Main Results

We evaluated the performance of our proposed SpikeFET-Tiny and SpikeFET-Base on several benchmarks, including FE108 [8], VisEvent [6], and COESOT [7]. Meanwhile, for comparison with event-based trackers, we constructed a baseline model named SpikeET that using only event streams as input.

Table 1: Comparison with state-of-the-art trackers on FE108 [8], VisEvent [6], and COESOT [7]. We train and measure energy consumption on the VisEvent dataset. The best three results are shown in **red**, **blue** and **green** fonts. * indicates that frame-based trackers are extended to frame-event-based trackers through early fusion. † indicates models pre-trained on image-frame tracking datasets.

Methods	Architecture	Param (M)	Power (mJ)	FE108 [8]		VisEvent [6]		COESOT [7]	
				AUC(%)	PR(%)	AUC(%)	PR(%)	AUC(%)	PR(%)
ANN	DiMP50* [45]	-	-	57.1	85.1	47.8	67.0	58.9	67.1
	PrDiMP50* [46]	-	-	59.0	87.7	47.6	65.3	57.9	69.6
	SiamRCNN* [47]	-	-	-	-	49.9	65.9	60.9	71.0
	TrDiMP50* [20]	-	-	60.3	91.2	-	-	60.1	72.2
	TransT50* [2]	-	-	63.9	93.0	47.4	65.0	60.5	72.4
	ToMP101* [48]	-	-	61.8	91.1	-	-	59.9	67.2
	FENet [8]	-	262.2	63.1	91.8	-	-	-	-
	OSTrack [3]	92.52	262.29	-	-	53.4	69.5	59.0	70.7
	CEUTrack [7]	97.82	265.60	55.6	84.5	53.1	69.1	62.7	76.0
	HRCEUTrack [22]	97.82	239.79	-	-	-	-	63.2	71.9
	HRMonTrack [22]	100.20	-	68.5	96.2	-	-	-	-
	ViPT† [25]	93.36	134.93	65.2	92.1	59.2	75.8	65.7	73.9
	SDSTrack† [19]	107.80	719.10	65.8	92.6	59.7	76.7	63.7	71.7
ANN-SNN	MMHT [15]	22.80	160.93	63.0	93.6	55.1	73.3	65.8	74.0
SNN	SpikeFET-Tiny	29.33	18.36	68.5	96.5	56.8	73.5	64.0	77.9
	SpikeFET-Base†	105.48	102.61	68.7	97.0	59.0	75.3	68.5	81.7

Frame-Event Tracking As shown in Tab. 1, compared to several state-of-the-art trackers, our SpikeFET demonstrates significant advantages across most datasets. On the FE108 [8] dataset, SpikeFET-Tiny achieves an AUC score of 68.5%, matching the previous state-of-the-art HRMonTrack [22] while surpassing it by 0.3% in PR score and outperforming all other trackers by a substantial margin. Compared to the Tiny variant, the Base model further improves performance metrics, but there is a tendency towards overfitting. Notably, although the model size of the base model is comparable to the previous Base models, its theoretical energy consumption is significantly lower than equivalently scaled models. Additionally Compared with the latest SDSTrack [19] on ANN, our Tiny has improved its AUC score by 2.7% on the FE108 [8] dataset, with parameters only one-third of SDSTrack [19], while reducing power consumption by 39 times.

On the more complex COESOT [7] dataset, our SpikeFET-Base model achieves 2.7% and 7.7% improvements in AUC and PR scores respectively compared to the state-of-the-art MMHT [15], while the Tiny model also surpass the latest SDSTrack [19] by 0.3% in AUC. This demonstrates that our method exhibits stronger capabilities in handling large-scale complex scenarios.

On the VisEvent [6] dataset, our Tiny model outperforms the majority of trackers. However, for the Base model, our SpikeFET performs comparatively worse than ViPT [25] and SDSTrack [19]. We attribute this primarily to the fact that SpikeFET is particularly well-suited for handling datasets with challenging scenarios and large-scale datasets, such as FE108 [8] and COESOT [7]. In such scenarios, the inherent advantages of event cameras become evident, allowing SpikeFET to demonstrate exceptional performance. In contrast, ViPT [25] and SDSTrack [19] excel in less challenging scenarios, such as VisEvent [6].

At the same time, as shown in Tab. 1, it can be observed that SpikeFET-Base achieved increases of 0.2%, 2.2%, and 4.5% in AUC metrics compared to SpikeFET-Tiny on the FE108 [8], VisEvent [6], and COESOT [7] datasets (arranged in ascending order of data volume), with the performance gap progressively widening. Tab. 2 reveals that on FE108 [8]—a dataset featuring numerous challenging scenarios (e.g., overexposure, low illumination)—SpikeFET’s performance is fully leveraged: its AUC is only 4.1% lower than SpikeFET-Tiny, while outperforming all event-based tracking algorithms. In contrast, it underperforms by 17.4% on VisEvent [6]. This further illustrates the superiority of SpikeFET in challenging scenarios and complex datasets.

Table 2: Comparison with state-of-the-art trackers on two event-based tracking benchmarks. The works in the table directly adopt the results of the SDTrack [16] report.

Methods	Architecture	Param (M)	Power (mJ)	FE108 [8]		VisEvent [6]	
				AUC(%)	PR(%)	AUC(%)	PR(%)
ANN	DiMP50 [45]	-	256.37	-	-	31.5	44.2
	PrDiMP50 [46]	-	258.37	-	-	32.2	46.9
	ATOM [49]	-	30.199	-	-	28.6	47.4
	SiamRPN [50]	-	203.88	-	-	24.7	38.4
	STARK [4]	28.23	58.88	57.4	89.2	34.1	46.8
	SimTrack [51]	88.64	93.84	56.7	88.3	34.6	47.6
	OSTrack [3]	92.52	98.90	54.6	87.1	32.7	46.4
	ARTrack [52]	202.56	174.80	56.6	88.5	33.0	43.8
	SeqTrack [53]	90.60	302.68	53.5	85.5	28.6	43.3
	HiT [54]	42.22	19.78	55.9	88.5	34.6	47.6
	GRM [55]	99.83	142.14	56.8	89.3	33.4	47.7
	HIPTTrack [56]	120.41	307.74	50.8	81.0	32.1	45.2
ANN-SNN	STNet [13]	20.55	-	58.5	89.6	35.0	50.3
	SNNTrack [14]	31.40	8.25	-	-	35.4	50.4
SNN	SDTrack [16]	107.26	30.52	59.9	91.5	37.4	51.5
	SpikeET	22.36	8.80	64.4	94.7	39.4	54.0

These results confirm the energy efficiency characteristics of SNNs and their potential for edge devices. Furthermore, due to the inherent compatibility between SNNs and event data, combined with the high-efficiency design of our SpikeFET network, SNNs demonstrate promising potential to surpass ANNs in frame-event tracking tasks.

Event-based Tracking As shown in Tab. 2, without using fusion networks, our SpikeET with RPM and STR strategy outperforms all event-based trackers on both datasets, achieving state-of-the-art performance. Specifically, on the FE108 [8] dataset, SpikeET with 22.36M parameters and ultra-low energy consumption improves AUC by 4.5% and PR score by 3.2% compared to the previous best tracker SDTrack [16], even surpassing most trackers listed in Tab. 1. On the VisEvent [6] dataset, SpikeET achieves 2.0% higher AUC and 2.5% better PR than SDTrack [16]. Notably, while consuming only 0.55mJ more energy than SNNTrack [14], our method delivers a 4.0% AUC improvement. These results effectively demonstrate the advantages of our proposed RPM and STR modules, while further validating the energy efficiency of SNNs and their potential for edge device deployment.

4.4 Ablation Study

Effectiveness of proposed components We analyze the effects of the proposed RPM, E_t , and STR. Our baseline model, similar to CEU-Track [7], processes six input images from both frame and event modalities with separate feature extraction. These features are concatenated before entering the fusion network, then separated prior to the prediction head and processed through decoupled tracking heads. As shown in Tab. 3, the baseline equipped with RPM (#3) achieves significant improvements. On FE108 [8] with numerous extreme scenarios, RPM substantially increases AUC by 34.7% and PR by 49.9% over the baseline, while on VisEvent [6], it improves AUC by 10.4% and PR by 11.9%. More specifically, illustrated in Fig. 6, without RPM method, the peak response in the output image fails to precisely indicate the target position, exhibiting a significant deviation. In contrast, when RPM is employed, the target area is accurately

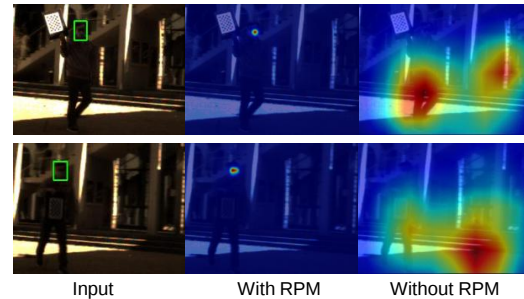


Figure 6: Visualization of position bias learnt in the model w/ and w/o RPM.

without RPM method, the peak response in the output image fails to precisely indicate the target position, exhibiting a significant deviation. In contrast, when RPM is employed, the target area is accurately

highlighted with high response activation. Quantitative analysis confirms that our model achieves enhanced tracking accuracy and robustness through RPM.

Furthermore, we conducted further experiments using one template frame and one search frame per modality, totaling four inputs. Without RPM, the six-input configuration (#baseline) yields only marginally higher performance than the four-input setup (#1) on the FE108 [8] dataset. Similarly, when RPM is employed, the six-input configuration (#3) also shows only a slight performance advantage over the four-input configuration (#2) on the FE108 [8] dataset. This conclusively demonstrates that the primary innovation driving baseline improvement is RPM’s effective mitigation of padding-induced degradation in translation invariance. In contrast, utilizing additional data provides minimal performance gains relative to RPM’s contribution. Additionally, the six-input setting allows for the further incorporation of STR.

Further additions of E_t (#4) and STR (#5) to #3 demonstrate effective enhancements, particularly with STR improving FE108 [8] AUC by 0.6%. This proves that while retaining the advantage of RPM in dynamically reorganizing spatial distributions, the STR strategy effectively suppresses matching deviations caused by feature spatial misalignment, significantly improving model performance. Finally, integrating both E_t and STR with RPM (#6) yields greater gains: 1.7% AUC and 1.5% PR improvements on FE108 [8], along with 1.3% AUC and 1.1% PR improvements on VisEvent [6].

Table 3: Ablation studies on RPM, E_t , and STR (Random Patchwork Module, Type Encodings, Spatio-Temporal Regularization).

#	Inputs	RPM	E_t	STR	FE108		VisEvent	
					AUC(%)	PR(%)	AUC(%)	PR(%)
Baseline	6	×	×	×	32.1	45.0	45.1	60.5
1	4	×	×	×	30.8	43.9	44.3	59.2
2	4	✓	×	×	66.1	93.9	54.5	71.6
3	6	✓	×	×	66.8	94.9	55.5	72.4
4	6	✓	✓	×	66.8	95.0	55.7	72.3
5	6	✓	×	✓	67.4	95.8	55.7	72.5
6	6	✓	✓	✓	68.5	96.5	56.8	73.5

Table 4: Ablation studies on the effect of the fusion methods.

#	Type	Method	FE108		VisEvent	
			AUC(%)	PR(%)	AUC(%)	PR(%)
1	Baseline	SpikeFET-Tiny	68.5	96.5	56.8	73.5
2	Modal	SpikeFT (Frame)	51.8	76.2	54.6	71.1
3		SpikeET (Event)	64.4	94.7	41.0	57.1
4	Backbone	Pre-Concat	67.5	95.9	55.6	72.3
5		Concat → Add	68.0	95.4	55.2	71.9
6		W/o modality Encodings	68.0	96.1	55.8	72.6
7		W/o Dual-Head	67.0	94.5	55.6	72.1
8	Tracking	W/o Response Loss	67.1	95.5	56.1	72.9
9		W/o Inference Fusion	67.9	95.6	56.1	72.8

Effectiveness of the fusion methods We evaluate the effectiveness of our fusion approach. As shown in entries #2 and #3 in Tab. 4, our frame-event fusion framework outperforms using either modality alone, demonstrating that our framework can effectively leverage the strengths of both modalities. The results from entry #4 reveal that concatenating the two modal images before feature extraction leads to performance degradation. The underlying reason may be that using independent parameters helps the model distinguish data types, enabling better learning of modality-specific features. In entry #5, we investigate an alternative approach of combining image features with event features via addition. This results in lower performance compared to our default method. Entry #6 demonstrates that modality-specific encoding more effectively differentiates modal features. Tracking experiments (entries #7, #8, and #9) confirm the necessity of our Decoupled Tracking Prediction With Similarity Fusion. Without this design, performance degrades to varying degrees. For more experiments, please refer to Appendix. F

Visualization results We compare the tracking results of SpikeFET-Base with the state-of-the-art tracker shown in Fig. 2. Despite facing challenges such as exposure and low light conditions, our tracker achieves robustness and excellent tracking accuracy. Additional visualization results are provided in the Appendix. A.

5 Conclusion

In this article, we propose a full spiking neural network SpikeFET for unified frame-event object tracking. This network effectively integrates frame and event data in the spiking paradigm and achieves collaborative integration of convolutional local feature extraction and Transformer-based global modeling. Moreover, we design the RPM to overcome the degradation of translation invariance caused by convolution filling through random spatial reorganization, while preserving the residual structure. We also propose the STR strategy that overcomes the degradation of similarity metrics caused by asymmetric features by forcing spatiotemporal consistency between temporal template features in the latent space. The effectiveness and power consumption advantages of our SpikeFET have been demonstrated through extensive experiments on multiple benchmarks.

References

- [1] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *Computer vision–ECCV 2016 workshops: Amsterdam, the Netherlands, October 8–10 and 15–16, 2016, proceedings, part II 14*, pages 850–865. Springer, 2016.
- [2] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8126–8135, 2021.
- [3] Botao Ye, Hong Chang, Bingpeng Ma, Shiguang Shan, and Xilin Chen. Joint feature learning and relation modeling for tracking: A one-stream framework. In *European conference on computer vision*, pages 341–357. Springer, 2022.
- [4] Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. Learning spatio-temporal transformer for visual tracking. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10448–10457, 2021.
- [5] Jiqing Zhang, Kai Zhao, Bo Dong, Yingkai Fu, Yuxin Wang, Xin Yang, and Baocai Yin. Multi-domain collaborative feature representation for robust visual object tracking. *The Visual Computer*, 37(9):2671–2683, 2021.
- [6] Xiao Wang, Jianing Li, Lin Zhu, Zhipeng Zhang, Zhe Chen, Xin Li, Yaowei Wang, Yonghong Tian, and Feng Wu. Visevent: Reliable object tracking via collaboration of frame and event flows. *IEEE Transactions on Cybernetics*, 54(3):1997–2010, 2023.
- [7] Chuanming Tang, Xiao Wang, Ju Huang, Bo Jiang, Lin Zhu, Jianlin Zhang, Yaowei Wang, and Yonghong Tian. Revisiting color-event based tracking: A unified network, dataset, and metric. *arXiv preprint arXiv:2211.11010*, 2022.
- [8] Jiqing Zhang, Xin Yang, Yingkai Fu, Xiaopeng Wei, Baocai Yin, and Bo Dong. Object tracking by jointly exploiting frame and event domain. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13043–13052, 2021.
- [9] Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019.
- [10] Catherine D Schuman, Shruti R Kulkarni, Maryam Parsa, J Parker Mitchell, Prasanna Date, and Bill Kay. Opportunities for neuromorphic computing algorithms and applications. *Nature Computational Science*, 2(1):10–19, 2022.
- [11] Man Yao, Xuerui Qiu, Tianxiang Hu, Jiakui Hu, Yuhong Chou, Keyu Tian, Jianxing Liao, Luziwei Leng, Bo Xu, and Guoqi Li. Scaling spike-driven transformer with efficient spike firing approximation training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.
- [12] Weihao Yu, Chenyang Si, Pan Zhou, Mi Luo, Yichen Zhou, Jiashi Feng, Shuicheng Yan, and Xinchao Wang. Metaformer baselines for vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(2):896–912, 2023.
- [13] Jiqing Zhang, Bo Dong, Haiwei Zhang, Jianchuan Ding, Felix Heide, Baocai Yin, and Xin Yang. Spiking transformers for event-based single object tracking. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 8801–8810, 2022.
- [14] Jiqing Zhang, Malu Zhang, Yuanchen Wang, Qianhui Liu, Baocai Yin, Haizhou Li, and Xin Yang. Spiking neural networks with adaptive membrane time constant for event-based tracking. *IEEE Transactions on Image Processing*, 2025.
- [15] Hongze Sun, Rui Liu, Wuque Cai, Jun Wang, Yue Wang, Huajin Tang, Yan Cui, Dezhong Yao, and Daqing Guo. Reliable object tracking by multimodal hybrid feature extraction and transformer-based fusion. *Neural Networks*, 178:106493, 2024.
- [16] Yimeng Shan, Zhenbang Ren, Haodi Wu, Wenjie Wei, Rui-Jie Zhu, Shuai Wang, Dehao Zhang, Yichen Xiao, Jieyuan Zhang, Kexin Shi, et al. Sdtrack: A baseline for event-based tracking via spiking neural networks. *arXiv preprint arXiv:2503.08703*, 2025.
- [17] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4282–4291, 2019.

- [18] Zhipeng Zhang and Houwen Peng. Deeper and wider siamese networks for real-time visual tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4591–4600, 2019.
- [19] Xiaojun Hou, Jiazheng Xing, Yijie Qian, Yaowei Guo, Shuo Xin, Junhao Chen, Kai Tang, Mengmeng Wang, Zhengkai Jiang, Liang Liu, et al. Sdstrack: Self-distillation symmetric adapter learning for multi-modal visual object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26551–26561, 2024.
- [20] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1571–1580, 2021.
- [21] Chenxu Luo, Xiaodong Yang, and Alan Yuille. Exploring simple 3d multi-object tracking for autonomous driving. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10488–10497, 2021.
- [22] Zhiyu Zhu, Junhui Hou, and Dapeng Oliver Wu. Cross-modal orthogonal high-rank augmentation for rgb-event transformer-trackers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22045–22055, 2023.
- [23] Pengcheng Shao, Tianyang Xu, Zhangyong Tang, Linze Li, Xiao-Jun Wu, and Josef Kittler. Tenet: targetness entanglement incorporating with multi-scale pooling and mutually-guided fusion for rgb-e object tracking. *Neural Networks*, 183:106948, 2025.
- [24] Zongwei Wu, Jilai Zheng, Xiangxuan Ren, Florin-Alexandru Vasluianu, Chao Ma, Danda Pani Paudel, Luc Van Gool, and Radu Timofte. Single-model and any-modality for video object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19156–19166, 2024.
- [25] Jiawen Zhu, Simiao Lai, Xin Chen, Dong Wang, and Huchuan Lu. Visual prompt multi-modal tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9516–9526, 2023.
- [26] Lin Zhu, Xiao Wang, Yi Chang, Jianing Li, Tiejun Huang, and Yonghong Tian. Event-based video reconstruction via potential-assisted spiking neural network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3594–3604, 2022.
- [27] Jesse Hagenaars, Federico Paredes-Vallés, and Guido De Croon. Self-supervised learning of event-based optical flow with spiking neural networks. *Advances in Neural Information Processing Systems*, 34:7167–7179, 2021.
- [28] Jinye Qu, Zeyu Gao, Tielin Zhang, Yanfeng Lu, Huajin Tang, and Hong Qiao. Spiking neural network for ultralow-latency and high-accurate object detection. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.
- [29] Wenzhuo Liu, Shuiying Xiang, Tao Zhang, Yanan Han, Yahui Zhang, Xingxing Guo, Licun Yu, and Yue Hao. S4-kd: A single step spiking siamfc++ for object tracking with knowledge distillation. *Neural Networks*, 188:107478, 2025.
- [30] Yihao Luo, Min Xu, Caihong Yuan, Xiang Cao, Liangqi Zhang, Yan Xu, Tianjiang Wang, and Qi Feng. Siamspn: Siamese spiking neural networks for energy-efficient object tracking. In *International conference on artificial neural networks*, pages 182–194. Springer, 2021.
- [31] Shuiying Xiang, Tao Zhang, Shuqing Jiang, Yanan Han, Yahui Zhang, Xingxing Guo, Licun Yu, Yuechun Shi, and Yue Hao. Spiking siamfc++: Deep spiking neural network for object tracking. *Nonlinear Dynamics*, 112(10):8417–8429, 2024.
- [32] Jiahang Cao, Mingyuan Sun, Ziqing Wang, Hao Cheng, Qiang Zhang, Shibo Zhou, and Renjing Xu. Spiking neural network as adaptive event stream slicer. *arXiv preprint arXiv:2410.02249*, 2024.
- [33] Mingcheng Ji, Ziling Wang, Rui Yan, Qingjie Liu, Shu Xu, and Huajin Tang. Sctn: Event-based object tracking with energy-efficient deep convolutional spiking neural networks. *Frontiers in Neuroscience*, 17:1123698, 2023.
- [34] Liangwei Fan, Yulin Li, Hui Shen, Jian Li, and Dewen Hu. From dense to sparse: low-latency and speed-robust event-based object detection. *IEEE Transactions on Intelligent Vehicles*, 2024.
- [35] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

- [36] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baining Guo. Cswin transformer: A general vision transformer backbone with cross-shaped windows. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12124–12134, 2022.
- [37] Jiangning Zhang, Xiangtai Li, Jian Li, Liang Liu, Zhucun Xue, Boshen Zhang, Zhengkai Jiang, Tianxin Huang, Yabiao Wang, and Chengjie Wang. Rethinking mobile block for efficient attention-based models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1389–1400. IEEE Computer Society, 2023.
- [38] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European conference on computer vision (ECCV)*, pages 734–750, 2018.
- [39] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019.
- [40] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision—ECCV 2014: 13th European conference, zurich, Switzerland, September 6–12, 2014, proceedings, part v 13*, pages 740–755. Springer, 2014.
- [41] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. Lasot: A high-quality benchmark for large-scale single object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5374–5383, 2019.
- [42] Matthias Muller, Adel Bibi, Silvio Giancola, Salman Alsubaihi, and Bernard Ghanem. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *Proceedings of the European conference on computer vision (ECCV)*, pages 300–317, 2018.
- [43] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE transactions on pattern analysis and machine intelligence*, 43(5):1562–1577, 2019.
- [44] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
- [45] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6182–6191, 2019.
- [46] Martin Danelljan, Luc Van Gool, and Radu Timofte. Probabilistic regression for visual tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7183–7192, 2020.
- [47] Paul Voigtlaender, Jonathon Luiten, Philip HS Torr, and Bastian Leibe. Siam r-cnn: Visual tracking by re-detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6578–6588, 2020.
- [48] Christoph Mayer, Martin Danelljan, Goutam Bhat, Matthieu Paul, Danda Pani Paudel, Fisher Yu, and Luc Van Gool. Transforming model prediction for tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8731–8740, 2022.
- [49] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Atom: Accurate tracking by overlap maximization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4660–4669, 2019.
- [50] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8971–8980, 2018.
- [51] Boyu Chen, Peixia Li, Lei Bai, Lei Qiao, Qihong Shen, Bo Li, Weihao Gan, Wei Wu, and Wanli Ouyang. Backbone is all your need: A simplified architecture for visual object tracking. In *European conference on computer vision*, pages 375–392. Springer, 2022.
- [52] Xing Wei, Yifan Bai, Yongchao Zheng, Dahu Shi, and Yihong Gong. Autoregressive visual tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9697–9706, 2023.

- [53] Xin Chen, Houwen Peng, Dong Wang, Huchuan Lu, and Han Hu. Seqtrack: Sequence to sequence learning for visual object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14572–14581, 2023.
- [54] Ben Kang, Xin Chen, Dong Wang, Houwen Peng, and Huchuan Lu. Exploring lightweight hierarchical vision transformers for efficient visual tracking. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9612–9621, 2023.
- [55] Shen Yuan Gao, Chunlun Zhou, and Jun Zhang. Generalized relation modeling for transformer tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18686–18695, 2023.
- [56] Wenrui Cai, Qingjie Liu, and Yunhong Wang. Hiptrack: Visual tracking with historical prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19258–19267, 2024.
- [57] Wachirawat Ponghiran and Kaushik Roy. Spiking neural networks with improved inherent recurrence dynamics for sequential learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 8001–8008, 2022.
- [58] Guobin Shen, Dongcheng Zhao, Tenglong Li, Jindong Li, and Yi Zeng. Are conventional snns really efficient? a perspective from network quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27538–27547, 2024.
- [59] Zhenzhi Wu, Hehui Zhang, Yihan Lin, Guoqi Li, Meng Wang, and Ye Tang. Liaf-net: Leaky integrate and analog fire network for lightweight and efficient spatiotemporal information processing. *IEEE Transactions on Neural Networks and Learning Systems*, 33(11):6249–6262, 2021.
- [60] Nicholas FY Chen. Pseudo-labels for supervised learning on dynamic vision sensor data, applied to object detection under ego-motion. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 644–653, 2018.
- [61] Ana I Maqueda, Antonio Loquercio, Guillermo Gallego, Narciso García, and Davide Scaramuzza. Event-based vision meets deep learning on steering prediction for self-driving cars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5419–5427, 2018.
- [62] Xiao Wang, Shiao Wang, Chuanming Tang, Lin Zhu, Bo Jiang, Yonghong Tian, and Jin Tang. Event stream-based visual object tracking: A high-resolution benchmark dataset and a novel baseline. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19248–19257, 2024.
- [63] Mark Horowitz. 1.1 computing’s energy problem (and what we can do about it). In *2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC)*, pages 10–14. IEEE, 2014.
- [64] Man Yao, Jiakui Hu, Tianxiang Hu, Yifan Xu, Zhaokun Zhou, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer v2: Meta spiking neural network architecture inspiring the design of next-generation neuromorphic chips. *arXiv preprint arXiv:2404.03663*, 2024.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: We claim our contributions in the abstract (Lines 5-19) and introduction (Lines 60-73).

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: We discuss the limitations of our proposed work in Appendix I.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[Yes\]](#)

Justification: Our proposed method aims to design an effective structure Sec. 3 to improve the tracking performance. And we verify the effectiveness of our method through experiments. (Sec. 4, Appendix F).

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: Our contribution is a specific model. We describe the proposed methods in Sec. 3, and provide the implementation details in Sec. 4.1 and Appendix D.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We use publicly available data and provide the exact commands and environment required for running to reproduce the results in Appendix D. As a baseline, we will open source the code and provide it in supplementary materials.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide the training and test details in Sec. 3.4, Sec. 4.1 and Appendix D.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Although our RPM introduces randomness, the results are stable in several random seed experiments. Therefore, we used a fixed random seed for the experiment.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We provided the type of compute workers in Sec. 4.1 and the required theoretical power consumption in Sec. 4.3.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: The research conducted in the paper conforms with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: We discuss the potential positive societal impacts of the work on Appendix H.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Our work poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite the original paper that produced the code package or dataset.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: This paper does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: Our paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: Our paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: Our core method development does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

Appendix

A More Visualization Results

We present more results of SpikeFET and other methods on the COESOT [7] dataset here.

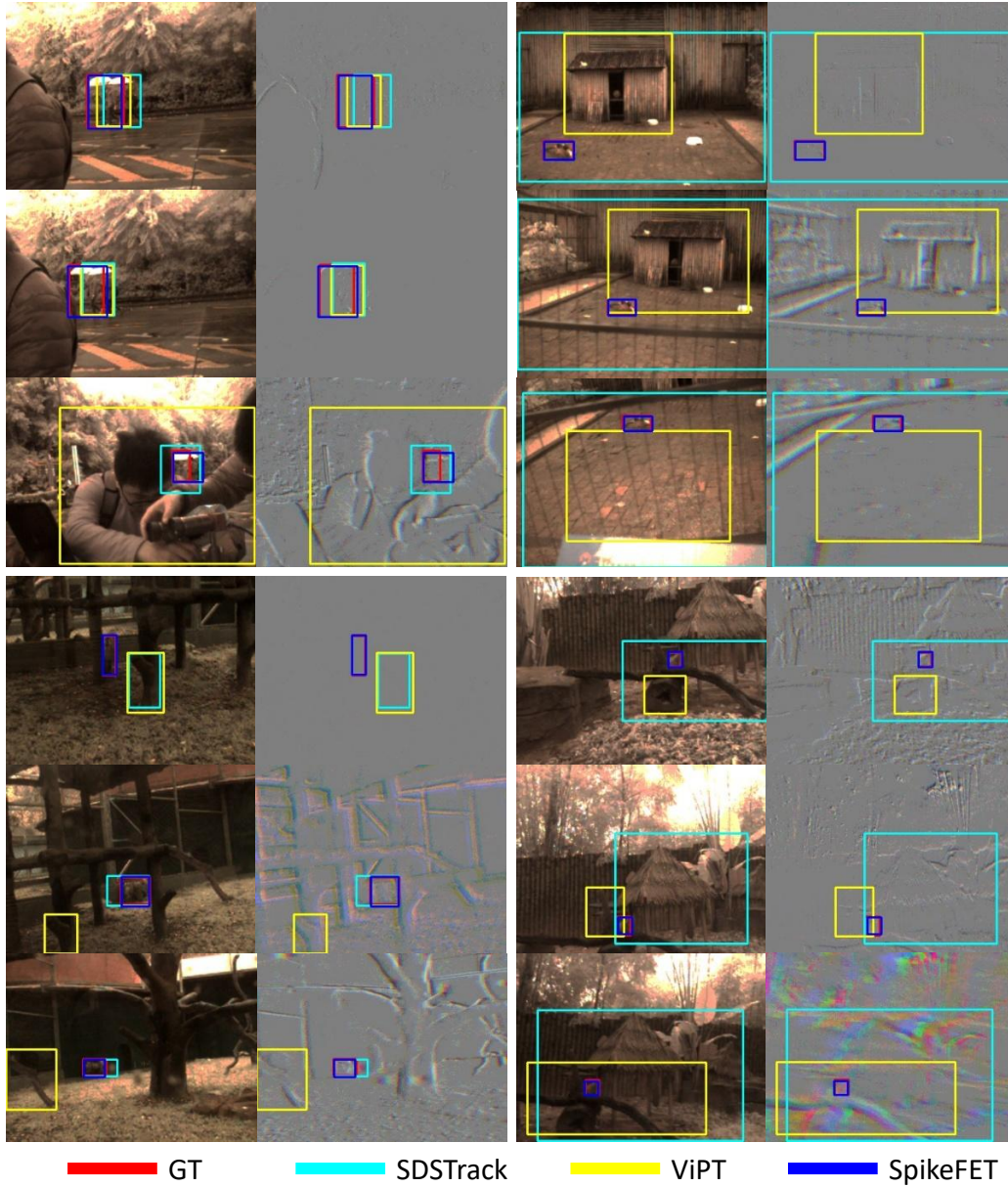


Figure 7: Partial visualization results on the COESOT [7] dataset.

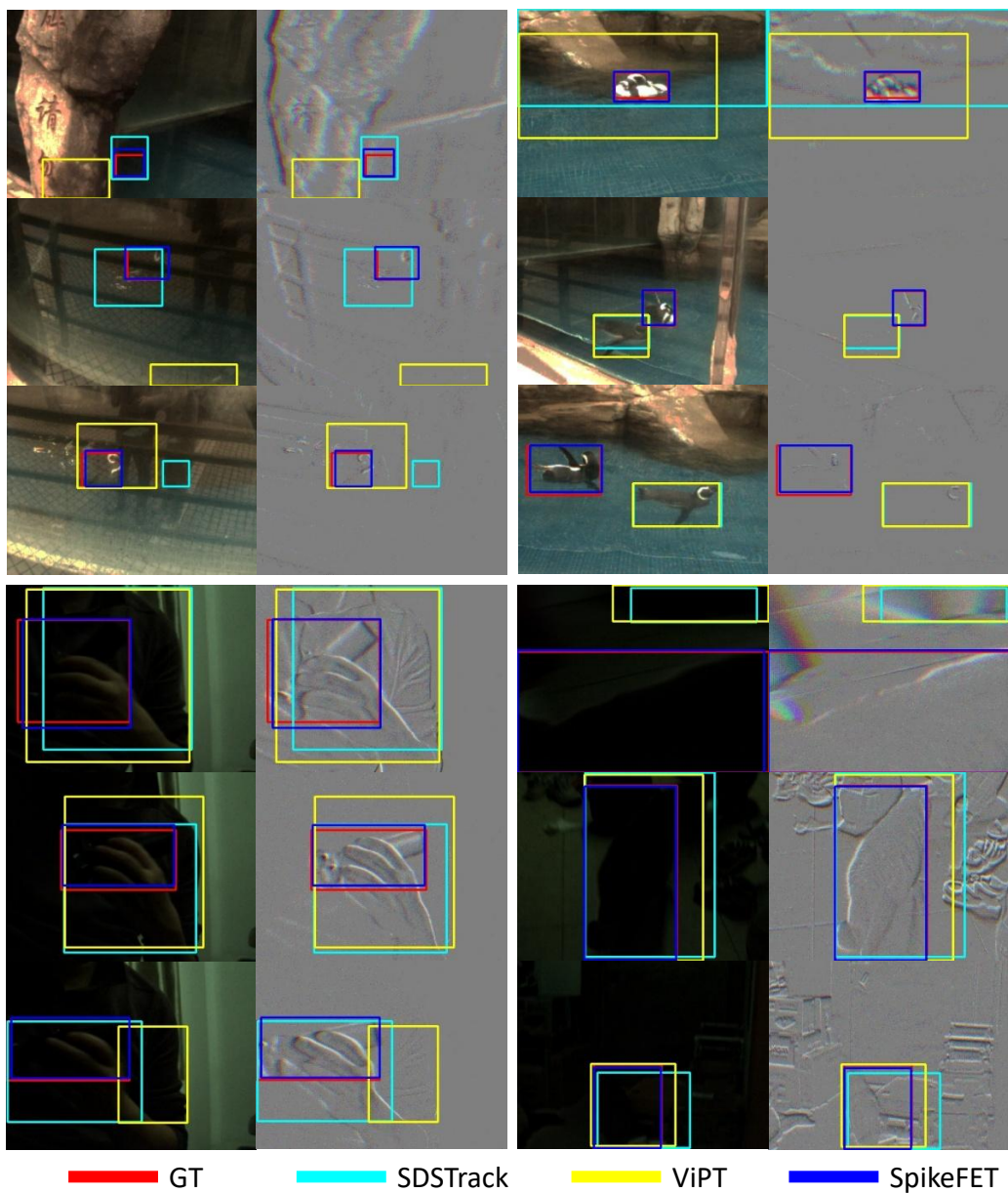


Figure 8: Partial visualization results on the COESOT [7] dataset.

B Preliminary

B.1 Spiking Neuron

SNNs derive biological plausibility from their event-driven operation. Conventional training faces a dilemma: enforcing strict 0,1 spikes induces gradient quantization errors, while mitigating errors via multi-bit spikes [57, 58] or continuous approximations [59] erodes event-driven efficiency. We introduce the Spike Firing Approximation (SFA) [11], a framework combining integer training with spike-driven inference, to optimize neuronal firing patterns.

We postulate that information transmission between spiking neurons is governed by spike firing rates. The firing rate of a spiking neuron is operationally defined as:

$$a^l = \frac{1}{T} \sum_{t=1}^T \mathbf{S}^l[t] \quad (16)$$

where a^l denotes the spike firing rate, and $\mathbf{S}^l[t]_T$ represents the spike train sequence at layer l over T discrete time steps.

During training, we implement integer activation training by approximating a^l with integer values. This is achieved by replacing the temporal summation $\sum_{t=1}^T \mathbf{S}^l[t]$ with single-timestep integer-valued activations:

$$\mathbf{S}_T^l = \lfloor \text{clip}\{\mathbf{x}^l, 0, T\} \rfloor \quad (17)$$

where $\mathbf{S}_T^l$ denotes the integer activation within $[0, T]$, with T representing the maximum activation threshold. And $\text{clip}\{\mathbf{x}^l, 0, T\}$ constrains $\mathbf{x}^l$ to the interval $[0, T]$, followed by rounding to the nearest integer:

In inference, we perform spike activation inference:

$$\mathbf{S}_T^l = \sum_{t=1}^T \hat{\mathbf{S}}^l[t] \quad (18)$$

where $\hat{\mathbf{S}}^l[t]$ denotes the spike train sequence in SFA (Spike Firing Approximation) inference. The spatial input to spiking neurons in layer $(l+1)$ is computed as:

$$\mathbf{X}^{l+1} = W^{l+1} a^l = W^{l+1} \frac{1}{T} \mathbf{S}_T^l = \left(\frac{1}{T} \mathbf{W}^{l+1} \right) \sum_{t=1}^T \hat{\mathbf{S}}^l[t] \quad (19)$$

The spike sequence $\hat{\mathbf{S}}^l[t]$ consists solely of 0s and 1s, enabling all MAC (Multiply-Accumulate) operations to be converted into sparse AC (Accumulation Operations), thereby ensuring spike-driven computation during inference.

B.2 Input Representation

In this work, the input consists of two complementary modalities: frame-rate-fixed image frames I and a continuous event stream $E(x, y, t, p)$, where (x, y) denotes pixel coordinates, t represents timestamps, and p indicates polarity. To adapt to deep learning architectures, event streams are typically converted into image-like representations [34, 60, 61]. Compared with complex event representation methods in single-object tracking (such as event voxels [62] and GTP [16]), we employ a simple Time-Integrated Image of Events [34] to encode the event stream within an accumulated temporal window t . This approach maps events into a 3D tensor while preserving temporal information and polarity, ensuring compatibility with deep learning methods and facilitating fusion with existing image frames modalities. Specifically, the conversion process first partitions the event stream along the temporal axis into a set of B bins. The polarity values of events within each bin are accumulated and normalized

to the $[0, 255]$ range using an activation function. The transformation is formulated as follows:

$$t_i^* = \left\lceil B \cdot \frac{t_i - t_1}{t_{N_e} - t_1} \right\rceil \quad (20)$$

$$S(x, y, t) = \sigma \left(\sum_i p_i \delta(x - x_i) \delta(y - y_i) \delta(t - t_i^*) \right) \quad (21)$$

$$\sigma(a) = \frac{255}{1 + e^{-\frac{a}{2}}} \quad (22)$$

Where $\sigma(\cdot)$ and $\delta(\cdot)$ represent the activation function and Dirac delta function, respectively. t_{N_e} denotes the total number of events. The generated event time image $S \in \mathbb{R}^{B \times H \times W}$ and the corresponding image frames are processed using standard methods to extract template patches and search patches from both modalities. These patches are then jointly fed into the SpikeFET model.

C More Implementation Details

C.1 Response Loss

We adopt the weighted focal loss [38] for by constraining the similarity. Specifically, we denote the ground truth target center and the corresponding low-resolution equivalent as $\hat{p}$ and $\bar{p} = [\bar{p}_x, \bar{p}_y]$, respectively. The Gaussian kernel is then applied to generate the groundtruth heatmap: $\mathbf{P}_{xy} = \exp \left(-\frac{(x - \bar{p}_x)^2 + (y - \bar{p}_y)^2}{2\delta_p^2} \right)$, where δ_p represents the object size-adaptive standard deviation [38]. Thus, the Gaussian-Weighted Focal (GWF) loss function is formulated as:

$$\mathcal{L}_{\text{GWF}} = - \sum_{xy} \begin{cases} (1 - \mathbf{P}_{xy})^\alpha \log(\mathbf{P}_{xy}) & \text{if } \hat{\mathbf{P}}_{xy} = 1 \\ (1 - \hat{\mathbf{P}}_{xy})^\beta (\mathbf{P}_{xy})^\alpha \log(1 - \mathbf{P}_{xy}), & \text{otherwise} \end{cases} \quad (23)$$

where α and γ are hyperparameters and are set to 2 and 4, respectively, as suggested in [62]. The response maps of both modalities are normalized by dividing them by a temperature coefficient τ (empirically set to 2). The final loss function is expressed as: $\mathcal{L}_{\text{Res}} = \mathcal{L}_{\text{GWF}}(\mathbf{R}_F/\tau, \mathbf{R}_E/\tau)$

C.2 Metrics

When evaluating algorithms in the field of Spiking Neural Networks (SNNs), researchers often adopt theoretical power consumption estimation methods to simplify the analysis of hardware implementation details. Specifically, the energy cost of Artificial Neural Networks (ANNs) is calculated as FLOPs multiplied by E_{MAC} , while the energy cost of SNNs is determined by FLOPs multiplied by E_{AC} and the network spike firing rate. In 45 nm technology, the energy consumption for MAC (Multiply-ACcumulate) and AC (ACcumulate) operations is $E_{\text{MAC}} = 4.6pJ$ and $E_{\text{AC}} = 0.9pJ$, respectively [63]. For spiking-based convolutional or multilayer perceptron (MLP) layers, only the additional time step T and the spike firing rate per layer need to be considered. In this paper, we can calculate the peak emissivity of each layer, so the energy consumption of each layer is FLOPs multiplied by EAC multiplied by the peak emissivity of each layer. The subtle difference is that the network structure will affect the number of additions triggered by a single peak. For example, when using different convolution kernel sizes for matrix multiplication, the energy consumption of the same spiky tensor is different. we calculate the theoretical power consumption using the following formula [64]:

$$E_{\text{ANN}} = E_{\text{MAC}} \times (\text{FL}_{\text{Conv}} + \text{FL}_{\text{MLP}}) \quad (24)$$

$$E_{\text{SNN}} = T \times E_{\text{AC}} \times \left(\sum_{m=1}^M \mathbf{R}_C^{(m)} \times \text{FL}_{\text{Conv}}^{(m)} + \sum_{n=1}^N \mathbf{R}_M^{(n)} \times \text{FL}_{\text{MLP}}^{(n)} \right) + T \times E_{\text{MAC}} \times \text{FL}_{\text{Conv}} \quad (25)$$

where T denotes the time step length; $\mathbf{R}_C^{(m)}$ and $\mathbf{R}_M^{(n)}$ represent the spike firing rates of the m -th convolutional layer and the n -th fully connected (MLP) layer, respectively, defined as the ratio of non-zero elements in the spike tensor. $\text{FL}_{\text{Conv}}^{(m)}$ and $\text{FL}_{\text{MLP}}^{(n)}$ correspond to the FLOPs of their respective layers, and FL_{Conv} represents the FLOPs of the first and last convolutional layer in the tracking head.

More specifically, the FLOPs of the m -th Conv layer in ANNs are:

$$FL_{Conv} = (k_m)^2 \cdot h_m \cdot w_m \cdot c_{m-1} \cdot C_m \quad (26)$$

Where k_m is the kernel size, (h_m, w_m) is the output feature map size, c_{m-1} and c_m are the input and output channel numbers, respectively. The FLOPs of the n -th MLP layer in artificial neural networks is:

$$FL_{MLP} = i_n \cdot o_n \quad (27)$$

where i_n and o_n are the input and output dimensions of the MLP layer, respectively. In order to provide readers with a clear understanding of spiking firing rate, we have provided a detailed spiking firing rates of a SpikeFET model in Tab. 11.

D Detailed configuration and hyperparameters of SpikeFET models and training

On the Frame-Event tracking benchmark, we used two scales of SpikeFET in Tab. 5 and trained the model in our paper using the hyperparameters in Tab. 6.

Table 5: Configuration of backbone models for different SpikeFET (considering only single branch)

stage	# Tokens	Layer Specification			Tiny	Base
1	$\frac{H}{2} \times \frac{W}{2}$	Downsampling		Dim	32	64
		ConvFormer Spike Block	SepSpikeConv	MLP ratio	2	
			Channel Conv	Conv ratio	4	
			# Blocks		1	
2	$\frac{H}{4} \times \frac{W}{4}$	Downsampling		Dim	64	128
		ConvFormer Spike Block	SepSpikeConv	MLP ratio	2	
			Channel Conv	Conv ratio	4	
			# Blocks		1	
3	$\frac{H}{8} \times \frac{W}{8}$	Downsampling		Dim	128	256
		ConvFormer Spike Block	SepSpikeConv	MLP ratio	2	
			Channel Conv	Conv ratio	4	
			# Blocks		2	
4	$\frac{H}{16} \times \frac{W}{16}$	Downsampling		Dim	256	512
		TransFormer Spike Block	SepSpikeConv	MLP ratio	2	
			CSWin-SSA	Gamma ratio	4	
			Channel MLP	MLP ratio	4	
			# Blocks		6	9
5	$\frac{H}{16} \times \frac{W}{16}$	Downsampling		Dim	320	
		TransFormer Spike Block	SepSpikeConv	MLP ratio	2	
			CSWin-SSA	Gamma ratio	4	
			Channel MLP	MLP ratio	4	
			# Blocks		2	3

Table 6: Hyper-parameters for training on SpikeFET

Hyper-parameter	Finetune		Directly Training	
Traning	Dual-modal	Single-modal	Single-modal	
Model size	Tiny	Base	Tiny	Base
Timestemp	4	8	4	8
Epochs	50	50	50	300
Batch size	32	16	80	40
Optimizer			ADAMW	
Learning rate	6e-4	7.5e-5	6e-4	7.5e-4
Learning rate decay			Cosine	
Warmup eopchs	5	5	5	20
Weight decay			0.05	

E CSWin-SSA

CSWin Transformer [36] is an efficient and effective Transformer based general visual task backbone, which we have extended to pulse based Cross-Shaped Windows Spiking Self-Attention (CSWin-SSA). The CSWin-SSA first generates key ($\mathbf{K}_S$), query ($\mathbf{Q}_S$), and value ($\mathbf{V}_S$) vectors by linearly transforming the input $\mathbf{U}$. Then, it performs a γ -fold channel expansion on $\mathbf{V}_S$ to enhance representation. Next, the CSWinSSA operator applies cross-modal self-attention to establish dynamic associations between template and search region features, enabling adaptive extraction of target features. CSWinSSA is achieved by performing self-attention within the horizontal and vertical parallel stripes that form a cross-shaped window. According to the multihead self-attention mechanism, the input feature $\mathbf{U} \in R^{T \times H \times W \times C}$ will be first linearly projected to K heads, and then each head will perform local self-attention within either the horizontal or vertical stripes. Specifically, the CSWin-SSA module can be formulated as:

$$\mathbf{Q}_S = \text{SN}(\text{Linear}(\mathbf{U})), \mathbf{K}_S = \text{SN}(\text{Linear}(\mathbf{U})), \mathbf{V}_S = \text{SN}(\text{Linear}_\gamma(\mathbf{U})) \quad (28)$$

$$\mathbf{Q}_S = [\mathbf{Q}_S^1, \mathbf{Q}_S^2, \dots, \mathbf{Q}_S^M], \mathbf{K}_S = [\mathbf{K}_S^1, \mathbf{K}_S^2, \dots, \mathbf{K}_S^M], \mathbf{V}_S = [\mathbf{V}_S^1, \mathbf{V}_S^2, \dots, \mathbf{V}_S^M] \quad (29)$$

$$\mathbf{Y}_k^i = \text{SSA}(\mathbf{Q}_S^{ik}, \mathbf{K}_S^{ik}, \mathbf{V}_S^{ik}) \quad (30)$$

$$\text{H-SSA}_k(\mathbf{X}) = [\mathbf{Y}_k^1, \mathbf{Y}_k^2, \dots, \mathbf{Y}_k^M] \quad (31)$$

$$\text{SSA}(\mathbf{Q}_S, \mathbf{K}_S, \mathbf{V}_S) = \text{SN}(\mathbf{Q}_S \mathbf{K}_S^\top \mathbf{V}_S * \text{scale}) \quad (32)$$

where $\mathbf{Q}_S^i \in R^{(sw \times W) \times C}$ and $M = H/sw$, $i = 1, \dots, M$. The vertical stripes self-attention can be similarly derived, and its output for k^{th} head is denoted as $\text{V-SSA}_k(\mathbf{X})$, $\mathbf{K}$, $\mathbf{V}$ are the same. To address the challenge of large values generated by matrix multiplication, a scaling factor (scale) is introduced to regulate the magnitude of the results.

Assuming that the natural image has no directional deviation, we divide the K heads into two parallel groups on average (each group has $K/2$ heads, K is usually an even number). The first group of heads performs horizontal stripe self attention, while the second group of heads performs vertical stripe self attention. Finally, the outputs of these two parallel groups will be reconnected together.

$$\mathbf{U}' = \text{Linear}_{\frac{1}{\gamma}}(\text{Concat}(\text{head}_1, \dots, \text{head}_K)) \quad (33)$$

$$\text{head}_k = \begin{cases} \text{H-SSA}_k(\mathbf{X}), & k = 1, \dots, K/2 \\ \text{V-SSA}_k(\mathbf{X}), & k = K/2 + 1, \dots, K \end{cases} \quad (34)$$

F Experiments of the hyper-parameters

In the proposed SpikeFET framework, the hyper-parameters involved are $\mathcal{L}_{\text{res}}$ weight α and $\mathcal{L}_{\text{sim}}$ weight β , for which we only present ablation studies around the optimal parameters.

The ablation study of α We introduced the ablation study of α in Tab. 7. When α is greater than 1, the fusion response map of the two modalities overly depends on the worst modality, leading to a decrease in metrics, whereas when α is less than 1, the response maps of image frames and event frames cannot be adequately aligned. Therefore, we select $\alpha = 1$.

Table 7: The ablation study of α .

	$\alpha = 0.5$	$\alpha = 1$	$\alpha = 2$
AUC(%)	56.3	56.8	55.8

The ablation study of β The ablation study of β is shown in Tab. 8. When $\beta = 0.5$, we achieved the best performance. When β is too large, excessive emphasis on consistency between two template frames can lead to different features tending towards consistency, resulting in errors. When β is too small, it cannot fully make the two template frames consistent in time and space, and the effect is not significant enough.

Table 8: The ablation study of β .

	$\beta = 0.1$	$\beta = 0.5$	$\beta = 1$
AUC(%)	56.4	56.8	56.2

G Runtime and FLOPs

Here we present the specific values of MAC (Multiply-ACcumulate) and AC (ACcumulate), along with a runtime comparison between SpikeFET/SpikeET and other methods on an RTX 4090 GPU for reference. Results for SpikeFET and SpikeET are shown in Tab. 9 and Tab. 10, respectively.

Table 9: Comparison of MAC, AC, and runtime between SpikeFET and other methods.

Methods	Architecture	Speed (FPS)	MAC/AC	FE108 [8]		VisEvent [6]		COESOT [7]	
				AUC(%)	PR(%)	AUC(%)	PR(%)	AUC(%)	PR(%)
ANN	DiMP50* [45]	-	-	57.1	85.1	47.8	67.0	58.9	67.1
	PrDiMP50* [46]	-	-	59.0	87.7	47.6	65.3	57.9	69.6
	SiamRCNN* [47]	-	-	-	-	49.9	65.9	60.9	71.0
	TrDiMP50* [20]	-	-	60.3	91.2	-	-	60.1	72.2
	TransT50* [2]	-	-	63.9	93.0	47.4	65.0	60.5	72.4
	ToMP101* [48]	-	-	61.8	91.1	-	-	59.9	67.2
	FENet [8]	92.45	57	63.1	91.8	-	-	-	-
	OTrack [3]	113.29	57.02	-	-	53.4	69.5	59.0	70.7
	CEUTrack [7]	116.53	57.74	55.6	84.5	53.1	69.1	62.7	76.0
	HRCEUTrack [22]	123.38	52.13	-	-	-	-	63.2	71.9
	HRMonTrack [22]	-	-	68.5	96.2	-	-	-	-
	ViPT [†] [25]	135	29.33	65.2	92.1	59.2	75.8	65.7	73.9
ANN-SNN	SDSTrack [†] [19]	42.03	156.33	65.8	92.6	59.7	76.7	63.7	71.7
	MMHT [15]	85.48	34.98	63.0	93.6	55.1	73.3	65.8	74.0
SNN	SpikeFET-Tiny	106	30.34	68.5	96.5	56.8	73.5	64.0	77.9
	SpikeFET-Base[†]	49.26	121.24	68.7	97.0	59.0	75.3	68.5	81.7

Table 10: Comparison of MAC, AC, and runtime between SpikeET and other methods.

Methods	Architecture	Speed (FPS)	MAC/AC	FE108 [8]		VisEvent [6]	
				AUC(%)	PR(%)	AUC(%)	PR(%)
ANN	DiMP50 [45]	71.93	55.73	-	-	31.5	44.2
	PrDiMP50 [46]	71.82	55.73	-	-	32.2	46.9
	ATOM [49]	121.11	44.321	-	-	28.6	47.4
	SiamRPN [50]	518.55	6.57	-	-	24.7	38.4
	STARK [4]	172.19	12.8	57.4	89.2	34.1	46.8
	SimTrack [51]	266.25	20.4	56.7	88.3	34.6	47.6
	OTrack [3]	204.38	21.5	54.6	87.1	32.7	46.4
	ARTrack [52]	97.47	38	56.6	88.5	33.0	43.8
	SeqTrack [53]	113.09	65.8	53.5	85.5	28.6	43.3
	HiT [54]	437	4.3	55.9	88.5	34.6	47.6
	GRM [55]	153.18	30.9	56.8	89.3	33.4	47.7
	HIPTrack [56]	110.23	66.9	50.8	81.0	32.1	45.2
ANN-SNN	STNet [13]	163.40	-	58.5	89.6	35.0	50.3
	SNNTrack [14]	-	9.17	-	-	35.4	50.4
SNN	SDTrack [16]	-	30.52	66.56	91.5	37.4	51.5
	SpikeET	185.87	9.78	64.4	94.7	39.4	54.0

H Impact Statement

This paper proposes a fully spike-based frame-event tracking framework, and demonstrates the effectiveness and necessity of SNNs in target tracking through extensive experimental validation.

This work provides a novel approach and establishes a baseline for achieving low-power, high-performance, and robust visual object tracking, aiming to inspire further research and development in energy-efficient and high-performance tracking systems. At the same time, it provides a new paradigm for developing low-power edge vision computing, demonstrating potential applications in real-time perception fields such as autonomous driving, and promoting the engineering process of neural morphological computing technology. However, the misuse of Object tracking technology can have a negative impact on personal privacy.

I Limitations

Although SpikeFET has achieved commendable accuracy and robustness in frame-event tracking through SNNs, its sparse characteristic results in insufficient pre-training on tracking datasets, leading to suboptimal performance on VisEvent [6] compared to state-of-the-art trackers. Meanwhile, deploying the tracker on edge computing platforms (e.g., brain-inspired chips or neuromorphic chips) could further exploit SNNs' low-power characteristics. These challenges will constitute critical research directions for future exploration.

Table 11: Layer spiking firing rates of model SpikeFET-Tiny on VisEvent.

			Image	Event
	Downsampling	Conv	1	1
Stage 1	ConvFormer Spike Block	PWConv1	0.3645	0.2723
		SepSpikeConv DWConv	0.3314	0.3330
		PWConv2	0.3526	0.3703
		Channel Conv Conv1	0.4293	0.3656
		Conv2	0.0740	0.0771
Stage 2	ConvFormer Spike Block	Downsampling Conv	0.3203	0.2694
		PWConv1	0.1791	0.1616
		SepSpikeConv DWConv	0.2381	0.2243
		PWConv2	0.1712	0.1702
		Channel Conv Conv1	0.2680	0.2442
Stage 3	ConvFormer Spike Block	Conv2	0.0371	0.0302
		Downsampling Conv	0.2245	0.2031
		PWConv1	0.1876	0.1719
		SepSpikeConv DWConv	0.2484	0.2379
		PWConv2	0.1300	0.1248
	ConvFormer Spike Block	Channel Conv Conv1	0.1818	0.1721
		Conv2	0.0197	0.0175
		PWConv1	0.2642	0.2465
		SepSpikeConv DWConv	0.1486	0.1412
		PWConv2	0.1160	0.1102
stage4	TransFormer Spike Block1	Channel Conv Conv1	0.1960	0.1874
		Conv2	0.0133	0.0116
		Downsampling Conv	0.2335	0.2050
		SepSpikeConv Conv-1/2/3	0.1408	
		Q_S	0.2372	
		K_S	0.1077	
		CSWin-SSA V_S	0.1851	
		$Q_S(K_S^T V_S)$	0.5644	
		Linear	0.7385	
		Channel MLP Linear 1	0.2194	
		Linear 2	0.0156	
	TransFormer Spike Block1	SepSpikeConv Conv-1/2/3	0.1811	
		Q_S	0.1901	
		K_S	0.0584	
		CSWin-SSA V_S	0.0939	
		$Q_S(K_S^T V_S)$	0.1180	
		Linear	0.5973	
		Channel MLP Linear 1	0.2867	
		Linear 2	0.0150	
	TransFormer Spike Block2	SepSpikeConv Conv-1/2/3	0.1782	
		Q_S	0.1834	
		K_S	0.0460	
		CSWin-SSA V_S	0.1147	
		$Q_S(K_S^T V_S)$	0.1160	
		Linear	0.5349	
		Channel MLP Linear 1	0.2977	
		Linear 2	0.0140	

Continued on next page

Table 11 – continued from previous page

		Image	Event
	SepSpikeConv Conv-1/2/3		0.1816
		Q_S	0.1633
		K_S	0.0532
		V_S	0.0909
		$Q_S(K_S^T V_S)$	0.0962
		Linear	0.4846
		Linear 1	0.2922
	Channel MLP	Linear 2	0.0103
	SepSpikeConv Conv-1/2/3		0.1733
		Q_S	0.1925
		K_S	0.0422
		V_S	0.0808
		$Q_S(K_S^T V_S)$	0.0676
		Linear	0.3879
		Linear 1	0.2574
	Channel MLP	Linear 2	0.0116
	SepSpikeConv Conv-1/2/3		0.1558
		Q_S	0.1886
		K_S	0.0532
		V_S	0.1221
		$Q_S(K_S^T V_S)$	0.1294
		Linear	0.5527
		Linear 1	0.1854
	Channel MLP	Linear 2	0.0146
stage5	Downsampling	Conv	0.1564
	SepSpikeConv Conv-1/2/3		0.1000
		Q_S	0.1951
		K_S	0.0369
		V_S	0.0487
		$Q_S(K_S^T V_S)$	0.0449
		Linear	0.3142
		Linear 1	0.2651
	Channel MLP	Linear 2	0.0032
	SepSpikeConv Conv-1/2/3		0.0900
		Q_S	0.1978
		K_S	0.0165
		V_S	0.0073
		$Q_S(K_S^T V_S)$	0.0148
		Linear	0.1635
		Linear 1	0.1314
	Channel MLP	Linear 2	0.0054
Tracking	Ctr	Conv1	0.1519 0.1511
		Conv2	0.0648 0.0665
		Conv3	0.0800 0.0752
		Conv4	0.1429 0.1439
		Conv5	1 1
	Offset	Conv1	0.1519 0.1511
		Conv2	0.0818 0.0865
		Conv3	0.0912 0.0844
		Conv4	0.1273 0.1213

Continued on next page

Table 11 – continued from previous page

		Image	Event
	Conv5	1	1
	Conv1	0.1511	0.1511
	Conv2	0.0818	0.0865
	Conv3	0.1126	0.1112
	Conv4	0.1526	0.1531
	Conv5	1	1