2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Continuous-Time Optical Flow Estimation from Asynchronous Event-Frame Streams for Embedded Systems

Daolong Yang*[1], Hansheng Liang*[2], Haoyuan Liu[1], Chengcai Wang[1], Bin Xu[2], Kun Xu†[1] and Xilun Ding[1]

***Abstract*— Bioinspired event cameras, with their high temporal resolution, low power consumption, and inherent motion responsiveness, have been widely adopted for fundamental vision tasks in robotics, notably optical flow estimation. Recent studies have shown that incorporating complementary frame data can significantly enhance the performance of event-based optical flow estimation. However, two major challenges hinder the real-time deployment of such methods on robotic platforms: (1) the asynchronous nature of events and frames makes it difficult to generalize across varying input temporal offsets; and (2) reliance on computationally expensive correlation volume construction and iterative refinement results in high inference latency on embedded systems. To address these issues, we propose a novel method that takes asynchronous event and frame streams as input and predicts high-quality dense flow in a single forward pass. Our approach temporally encodes both intra- and inter-sensor features and efficiently integrates them into a lightweight correlation volume to enhance flow prediction. Experimental results on real-world scenes demonstrate that our method improves flow accuracy by up to 22% over state-of-the-art hybrid event-frame methods, while being 3× faster on embedded GPUs. Furthermore, our approach maintains strong performance and generalizes well across diverse frame-event temporal offsets, introducing a novel paradigm for fusing asynchronous frame and event streams for continuous-time optical flow estimation.**

I. Introduction

Optical flow estimation is a fundamental low-level vision task that captures per-pixel motion in image space and serves as the foundation for a wide range of downstream applications in computer vision [1] and robotics [2], [3]. In recent years, bio-inspired event cameras have gained increasing attention for optical flow estimation, as the events they capture are inherently aligned with motion cues due to their ability to asynchronously record per-pixel brightness changes with exceptionally high temporal resolution and dynamic range [4]. Moreover, their ultra-low power consumption makes event cameras ideal for deployment on resource-constrained embedded platforms [5]. Previous works have extended the well-established RAFT framework [6] for frame-based optical flow estimation to event cameras by constructing dense all-pairs correlation volumes [7]. This line of research has inspired several follow-up studies that further refine the correlation volume construction and enhance iterative update strategies [8]–[10]. More recently, some methods have combined events with complementary frame information to leverage the strengths of both modalities, leading to improved estimation accuracy [11], [12].

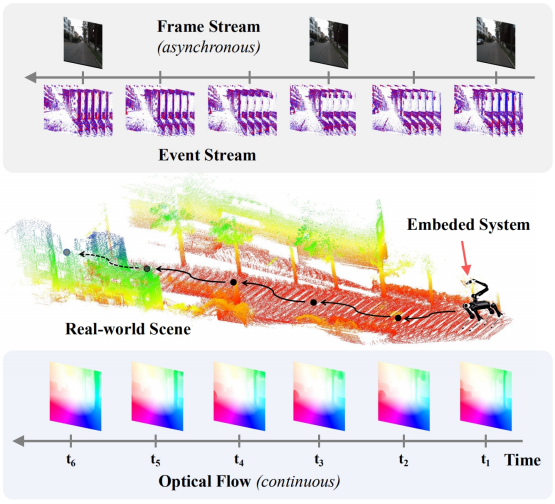

Fig. 1. Our method effectively fuses asynchronous frame and event streams to produce temporally continuous dense optical flow, enabling robust estimation as embedded systems navigate real-world scenes.

However, despite the high-quality optical flow achieved by state-of-the-art methods, several key challenges remain for deploying these approaches on embedded systems in real-world scenarios. First, the asynchronous nature of event and frame data poses significant challenges. Event cameras respond at microsecond resolution (up to 1 MHz), while frame cameras typically operate at much lower rates (10–20 Hz), which may be further limited or fluctuate during deployment due to bandwidth and power constraints [13]. Such a significant temporal imbalance makes it difficult for existing methods to effectively fuse information from both modalities. Second, real-time performance is constrained by limited computational resources on embedded platforms. Many prior approaches rely on high-end GPUs to construct dense correlation volumes and perform iterative refinement steps for accurate estimation. However, these methods are computationally intensive and require significant inference time, making real-time deployment on resource-constrained systems impractical.

This work was supported by the National Natural Science Foundation of China under Grant U22B2080, Grant T2121003, and Grant 52375003.

*Equal contribution. †Corresponding author.

[1]The authors are with the School of Mechanical Engineering and Automation, Beihang University, China. {YangDL, liuhaoyuan2020, cc_wang, xk007, xlding}@buaa.edu.cn.

[2]The authors are with the School of Mechanical and Vehicle Engineering, Beijing Institute of Technology, China. {hansheng.liang, bitxubin}@bit.edu.cn.

[1]*Full-resolution multimedia is available at:* https://www.youtube.com/watch?v=d2j2N70mXAs

TABLE I

COMPARISON OF STATE-OF-THE-ART OPTICAL FLOW METHODS.

Method	Input Modality		Temporal Continuity	Embedded Deployment
	Event	Frame		
RAFT [6]	–	synchronous	–	–
ERAFT [7]	✓	–	–	–
TMA [9]	✓	–	–	–
TID [14]	✓	–	–	✓
BFlow [12]	✓	synchronous	–	–
Ours	✓	**asynchronous**	✓	✓

* Temporal continuity refers to the modeling of dependencies between temporally adjacent inputs.

To address these challenges, we propose a novel optical flow estimation method that takes asynchronous event and frame streams as input and estimates dense optical flow in a single forward pass, enabling fast inference on embedded GPUs. To this end, we explicitly model temporally continuous features from asynchronous inputs and integrate them to construct a lightweight, motion-compensated correlation volume [14]. This design allows for accurate flow estimation with significantly reduced computational overhead. A direct comparison with state-of-the-art methods is presented in Table I. Our contributions can be summarized as follows:

- We propose an **Intra-Sensor Temporal Tracking** module and an **Inter-Sensor Temporal Tracking** module to encode asynchronous event and frame streams into temporally continuous features.
- We introduce two mechanisms—the **Temporal Initialization** block and the **Temporal Update** block—to incorporate temporally continuous features into the correlation volume construction through motion compensation.
- We conduct extensive experiments with both RGB and grayscale frame inputs under various frame-event temporal offsets on real-world datasets, including DSEC [15] and MVSEC [16]. Compared to state-of-the-art hybrid event-frame methods, our approach improves accuracy by $\mathbf{22}\%$ while running $\mathbf{3}\times$ faster on an NVIDIA Jetson Xavier NX. The source code will be released at: `https://github.com/YangDL-BEIHANG/EventFrameFlow`

II. RELATED WORK

Estimating optical flow from event cameras has been a longstanding and active research topic. Classical methods adapted frame-based techniques—such as gradient-based methods and plane fitting—to the event domain [17]. However, these methods typically produce sparse flow estimates, limited to regions where events are triggered, and struggle to generate dense flow fields. With the advent of deep learning and the availability of dedicated datasets [15], [16], dense optical flow estimation using only events has become feasible. One of the pioneering approaches, EV-FlowNet [18], introduced a self-supervised convolutional network that minimized the photometric reconstruction error of gray-level images synthesized from event streams. Later, E-RAFT [7] extended frame-based optical flow architectures to the event domain by introducing a 4D all-pairs correlation volume between two temporally adjacent event representations. The correlation volume is processed iteratively using recurrent updates to generate dense flow predictions. Building on E-RAFT, TMA [9] enhanced temporal modeling across event bins, reducing the number of required iterations while improving accuracy. More recently, E-FlowFormer [19] employed Transformer-based architectures to improve event representation and flow decoding. Furthermore, EEMFlow [8] extended the task beyond dense optical flow by additionally estimating mesh flow, offering a more flexible representation of motion dynamics. Despite the high quality of these methods, a common limitation lies in their computational and memory complexity. Specifically, correlation volume computation typically scales quadratically with spatial resolution, making it prohibitively expensive for real-time deployment on resource-constrained embedded systems.

Recent works have explored the integration of frame data as a complementary modality to enhance event-based optical flow estimation. RPEFlow [11] employs cross-attention to fuse frames and event point clouds, while BFlow [12] builds an additional correlation volume from consecutive frames to improve accuracy. While these methods effectively fuse frames and events, they rely on strict temporal alignment and struggle to generalize when the frame-event temporal offset deviates significantly from the training setup.

A promising line of work for embedded deployment is IDNet [14], which constructs the correlation volume via motion compensation, significantly reducing computational overhead while maintaining competitive accuracy. Its time-continuous variant, TID, introduces a warm-up strategy to improve temporal consistency. Inspired by this design, we integrate asynchronous frame inputs to enhance flow estimation and explicitly model temporal continuity across consecutive temporal inputs, going beyond simple warm-up strategies. This results in notable performance gains with negligible computational overhead, making our approach an ideal hybrid event-frame solution for continuous optical flow estimation on embedded systems.

III. METHODOLOGY

In contrast to state-of-the-art methods that treat optical flow as a time-discrete estimation problem and rely on computationally intensive iterative refinement, we propose modeling temporal dependencies from asynchronously triggered frames and event streams to enhance flow estimation. This design enables high-quality results in a single forward pass, significantly reducing computational overhead and achieving real-time performance on embedded systems. To this end, we propose to use **Asynchronous Temporal Feature Encoding** to encode temporal features from different sensors and timestamps. These temporal features are then integrated to support **Dense Optical Flow Estimation** through a Lightweight Correlation Module. An overview of our proposed pipeline is illustrated in Fig. 2.

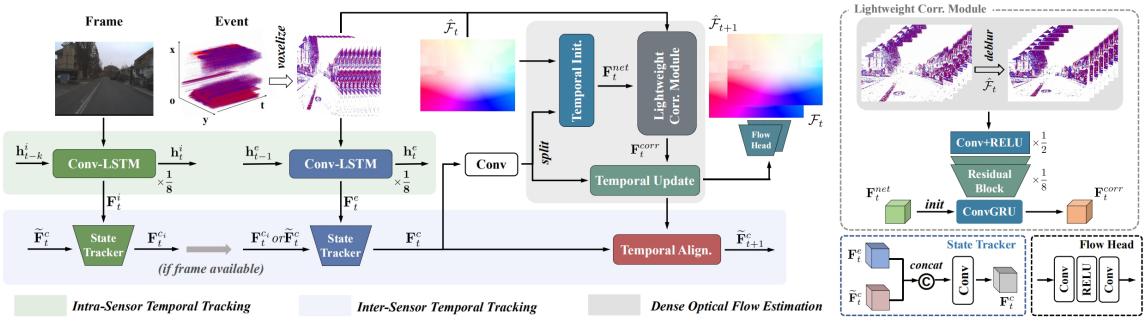

Fig. 2. An Overview of the Proposed Method. The overall architecture consists of three main components. In **Intra-Sensor Temporal Tracking** module (Section III-B(1)), each modality (event and frame) is independently encoded according to its triggered timestamp. These features are then asynchronously fused in the **Inter-Sensor Temporal Tracking** module (Section III-B(2)) to form temporal feature $\mathbf{F}_t^c$. Finally, the **Dense Optical Flow Estimation** module (Section III-C) takes the temporally encoded features, raw events, and the initial flow as inputs, and generates the final flow in a single forward pass.

A. Asynchronous Temporal Feature Encoding

Fusion of frame and event modalities has been widely explored in various visual perception tasks to enhance the performance of single-modality systems, including dense optical flow estimation [12]. However, most existing methods rely on strict temporal alignment between events and frames, which contradicts the inherently asynchronous nature of these two modalities [13]. This introduces substantial difficulty in generalizing current state-of-the-art fusion methods. Recent work has shown promising progress in fusing asynchronously triggered events and frames for pose estimation [20]. Nevertheless, directly applying such approaches to optical flow estimation is non-trivial, as estimating dense motion fields involves significantly more degrees of freedom than pose estimation. Specifically, a camera pose can be represented by a 6-DoF transformation in $\mathrm{SE}(3)$, commonly parameterized as a translation vector and a quaternion, i.e., $[x, y, z, q_w, q_x, q_y, q_z] \in \mathbb{R}^{1\times 7}$. In contrast, optical flow aims to estimate a dense per-pixel motion field $\mathcal{F} \in \mathbb{R}^{H\times W\times 2}$, where each pixel is associated with a 2D displacement vector, leading to a substantially higher-dimensional output space.

Inspired by [20], we propose an Intra-Sensor Temporal Tracking module to model the temporal features within each individual sensor, and an Inter-Sensor Temporal Tracking module to capture the temporal correlations across different sensing modalities. To address the aforementioned issue, we design a Temporal Alignment Module that generates per-pixel displacement vectors to temporally align features, thereby providing richer motion cues for dense flow estimation. These modules jointly encode temporal features into the flow estimation process, effectively incorporating asynchronous information from different sensors into the motion representation.

1) Intra-Sensor Temporal Tracking: We utilize two sensor-specific Conv-LSTM modules to track the temporal features within each sensor stream. Each Conv-LSTM consists of a convolutional layer followed by a Long Short-Term Memory (LSTM), enabling joint modeling of spatial and temporal dynamics. To reduce computational load and memory consumption, the input features are first downsampled to $1/8$ of the original resolution via strided convolution before being passed to the LSTM:

$$\hat{\mathbf{F}}^i_{t_k} = \mathrm{Conv}^{\downarrow 8}_i(\mathbf{I}_{t_k}), \quad \mathbf{F}^i_{t_j} = \mathrm{LSTM}_i(\hat{\mathbf{F}}^i_{t_j}, h^i_{t_k-1}) \tag{1}$$

$$\hat{\mathbf{F}}^e_{t_j} = \mathrm{Conv}^{\downarrow 8}_e(\mathbf{V}_{t_j}), \quad \mathbf{F}^e_{t_j} = \mathrm{LSTM}_e(\hat{\mathbf{F}}^e_{t_j}, h^e_{t_j-1}) \tag{2}$$

2) Inter-Sensor Temporal Tracking: We introduce a Temporal Alignment block (TA) following the State Tracker block, instead of directly concatenating $\mathbf{F}^i$ with $\mathbf{F}^e$ for fusion as in [20]. This module aligns temporal features from previous timestamps based on the per-pixel motion cues provided by the Dense Optical Flow Estimation module (which will be discussed in detail in the next section). $\mathbf{F}^i$ is only updated upon receiving a new frame and serves as a latent state $\mathbf{F}^{c_i}$, while the temporal feature $\mathbf{F}^c$ passed to the flow estimation module is continuously updated using $\mathbf{F}^e$. This design reflects the fact that frames are captured at a much lower frequency than events, and that events inherently respond to motion. By decoupling the update mechanisms of the two modalities, this approach enables effective fusion of asynchronous features. The state update operation is carried out using the State Tracker block (ST), which concatenates the two features along the channel dimension and applies a simple convolution operation, as shown in the bottom right corner of Fig. 2. The overall tracking process can be formulated as:

$$\mathbf{F}^c_t = \begin{cases} \mathrm{ST}_e\left(\mathbf{F}^e_t \parallel \mathbf{F}^{c_i}_t\right), & \text{if frame available,} \\ \mathrm{ST}_e\left(\mathbf{F}^e_t \parallel \widetilde{\mathbf{F}}^c_t\right), & \text{otherwise.} \end{cases} \tag{3}$$

where

$$\mathbf{F}^{c_i}_t = \mathrm{ST}_i\left(\mathbf{F}^i_t \parallel \widetilde{\mathbf{F}}^c_t\right), \quad \widetilde{\mathbf{F}}^c_t = \mathrm{TA}\left(\mathbf{F}^c_{t-1}, \boldsymbol{O}_{t-1}\right). \tag{4}$$

B. Dense Optical Flow Estimation

Previous work [14] proposed a lightweight motion compensation-based event correlation volume estimation method, which eliminates the need to construct dense all-pairs volumes as required by earlier approaches [7], [9], thereby significantly reducing both computational cost and memory usage. However, achieving high-quality flow estimation with this method still relies on multiple refinement iterations, which increases inference time.

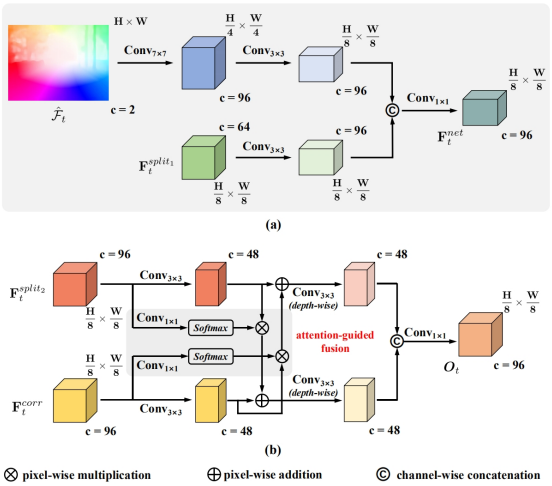

Fig. 3. (a) Detailed structure of the Temporal Initialization block (b) Detailed structure of the Temporal Update block

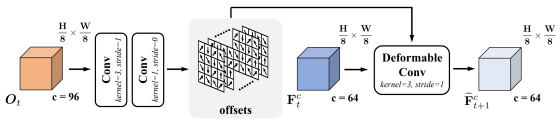

Fig. 4. Detailed structure of the Temporal Alignment block.

To achieve competitive performance within a single iteration and enable fast inference on embedded systems, we enhance the correlation volume construction proposed by [14] using temporally encoded features, motivated by two key insights. First, we introduce a Temporal Initialization block, which leverages temporal features and an initial flow estimate to pre-initialize the recurrent network memory, providing rich contextual information for correlation volume construction. Second, we introduce a Temporal Update block, which directly refines and enhances the correlation volume for improved flow estimation. The final aggregated feature is then used both for current flow estimation and temporal feature alignment.

1) Temporal Initialization Block: We aim to encode both the initial flow and temporal features for effective initialization. The initial optical flow $\hat{\mathcal{F}}_t$ is processed through consecutive strided convolutional layers to form the flow feature $\mathbf{F}_t^{\text{flow}}$. To initialize the correlation memory and guide refinement, the temporal feature $\mathbf{F}_t^c$ is split along the channel dimension into $\mathbf{F}_t^{\text{split}_1}$ and $\mathbf{F}_t^{\text{split}_2}$. Then, the correlation initialization feature $\mathbf{F}_t^{\text{net}}$ is constructed by directly concatenating these features followed by a simple convolutional layer. The architecture is illustrated in detail in Fig. 3(a).

2) Lightweight Correlation Module: The correlation volume is constructed by applying motion compensation to the event data [21]. The raw event voxel $\mathbf{V}_t$ is first warped using the initial flow $\hat{\mathcal{F}}_t$, followed by a convolutional layer and residual blocks for feature encoding. The encoded event feature $\mathbf{F}_t^V$ is then sequentially fed into a lightweight ConvGRU module to aggregate the temporal correlation volume $\mathbf{F}_t^{\text{corr}}$. The ConvGRU is initialized with the hidden state $\mathbf{F}_t^{\text{net}}$, which is generated by the Temporal Initialization block. A visualization of the module structure is provided in the top-right corner of Fig. 2.

3) Temporal Update Block: In this block, our goal is to efficiently enhance the correlation volume using temporal features, which is distinct from the objective of the Temporal Initialization block. Inspired by [22], we propose a lightweight attention-guided strategy for cross-feature fusion. Although the original pipeline leverages transformer modules to enhance the fused features, such designs incur substantial computational and memory costs. To mitigate this, we employ a per-channel cross-attention mechanism to guide the fusion, followed by a depth-wise convolution layer for refinement. This design is motivated by the insight that temporal features capture coarse motion patterns over extended sequences, whereas correlation volumes emphasize fine-grained motion over shorter temporal windows. As a result, the two types of features exhibit structurally compatible spatial patterns, making them suitable for attention-guided integration.

Given the temporal correlation volume $\mathbf{F}_t^{corr}$ and the split of temporal feature $\mathbf{F}_t^{split_2}$, we first reduce their channel dimensions to generate a more compact representation. A guided attention map is also generated from the raw features via a softmax activation, which facilitates the fusion between different feature representations. Each fused feature is subsequently processed by depthwise separable convolution, which decouples channel-wise filtering from spatial aggregation to achieve efficient feature extraction. Finally, the bidirectional context is fused to construct refined aggregated feature $\boldsymbol{O}_t$. An illustration of the structure is provided in Fig.3(b).

4) Temporal Alignment Module: We further leverage motion cues derived from the Dense Optical Flow Estimation module to temporally align the feature map $\mathbf{F}_t^c$ to the next timestamp, resulting in the aligned feature $\hat{\mathbf{F}}_{t+1}^c$. This alignment is achieved using deformable convolution [23], which generates pixel-wise spatial offsets to adaptively account for motion dynamics. Specifically, the aggregated motion features are first processed by two convolutional layers to predict the sampling offsets $\Delta\mathbf{p}_k$. The aligned feature is then computed as:

$$\hat{\mathbf{F}}_{t+1}^c(\boldsymbol{p}) = \sum_{k=1}^{n^2} w\left(\boldsymbol{p}_k\right) \cdot \mathbf{F}_t^c\left(\boldsymbol{p} + \boldsymbol{p}_k + \Delta\boldsymbol{p}_k\right), \tag{5}$$

where $\boldsymbol{p}_k \in \{(-1,-1),(-1,0),\ldots,(1,1)\}$ represents the k-th sampling offset in a standard convolution with kernel size 3×3, and $w(\cdot)$ denotes the learnable kernel weights. The overall structure is illustrated in Fig. 4.

Although this approach introduces additional computational overhead compared to directly using the predicted optical flow for alignment, it enables learning task-specific offsets, which significantly improves the flexibility of the alignment process and leads to notable performance gains.

TABLE II
QUANTITATIVE COMPARISON WITH STATE-OF-THE-ART METHODS ON DSEC AND MVSEC DATASETS

Method	DSEC					MVSEC					Model Size	GMAC	Runtime (E-GPU)
	EPE	AE	1PE	2PE	3PE	EPE	AE	1PE	2PE	3PE			
Frame-only													
RAFT [6]	0.83	**3.05**	18.94	6.26	3.40	0.72	12.61	22.20	4.98	1.44	5.26M	84.11	272ms
GMFlow [24]	1.75	5.38	44.81	20.29	12.07	1.16	21.89	42.17	15.94	6.61	4.68M	115.93	505ms
Event-only													
E-RAFT [7]	0.93	4.28	23.54	7.89	4.11	0.57	10.67	13.70	**2.16**	**0.68**	5.33M	92.78	276ms
TMA [9]	0.91	4.26	22.66	7.17	3.65	0.68	12.46	18.47	3.91	1.66	6.88M	159.09	539ms
TID [14]	0.83	4.33	21.67	5.87	2.77	0.53	10.03	11.32	2.66	0.99	**1.88M**	62	**213ms**
Hybrid Event-Frame													
BFlow [12]	0.88	3.68	22.39	7.20	3.63	0.66	11.96	19.92	3.32	0.81	6.59M	192.55	708ms
Ours	**0.74**	3.92	**17.22**	**4.66**	**2.29**	**0.51**	**9.69**	**10.64**	2.51	0.95	2.06M	**61.25**	215ms

C. Implementation Details and Loss Function

Following the approach in [7], [21], we utilize two lightweight Flow Heads (structure shown in the lower right corner of Fig. 2) to estimate both the current optical flow $\mathcal{F}_t$ and the initial flow $\hat{\mathcal{F}}_{t+1}$ for the subsequent timestamp. For the loss function, we use the L1 loss between the ground truth flow and both the current and next-timestamp flow estimates. To facilitate effective LSTM activation, we predict a flow sequence of length T, applying L1 loss to intermediate outputs with geometric weighting to emphasize later predictions.

$$\mathcal{L}_{loss} = \sum_{t=0}^{T} \gamma^{T-t} \left(\left\| \mathcal{F}_t^{\text{gt}} - \mathcal{F}_t \right\|_1 + \left\| \mathcal{F}_{t+1}^{\text{gt}} - \hat{\mathcal{F}}_{t+1} \right\|_1 \right) \quad (6)$$

IV. EXPERIMENTS

A. Datasets and evaluation setup

1) Datasets: To evaluate the performance of our proposed method in real-world scenarios with varying input resolutions, qualities, and camera motion patterns, we select two distinct datasets following prior work: DSEC [15] and MVSEC [16]. DSEC contains driving sequences with large pixel displacements under both daytime and nighttime conditions, captured using a high-resolution event camera (640×480) and RGB frames from a global shutter camera. In contrast, MVSE is recorded using a low-resolution event camera (346×260) and grayscale frames, featuring relatively small motion ranges. Additional details on the experimental setup are provided in the supplementary materials due to page limitations and will be made publicly available along with our code.

2) Baselines: To validate the performance of our method, we compare it against three categories of baselines: (i) Frame-only: We adopt RAFT [6] as a representative frame-based approach. We also include GMFlow [24], which performs a single forward iteration, similar to the inference strategy employed by our method. (ii) Event-only: We include three state-of-the-art methods—E-RAFT [7], TMA [9], and TID [14]. (iii) Hybrid Event-Frame: We select BFlow [12] as a representative hybrid method. All baselines, except TID, employ iterative refinement in their original design. Since our method is designed for real-time deployment on embedded systems, we constrain all baseline methods to a single iteration during evaluation to ensure fair comparison under fast inference settings.

3) Metrics: We report the following performance metrics for optical flow evaluation: EPE (endpoint error, measured as L2 distance in pixels), AE (angular error in degrees), and nPE (percentage of pixels with a flow magnitude error greater than n pixels). We also report model size in millions of parameters and GMACs (multiply-and-accumulate operations during inference, measured in billions). To assess runtime performance, we benchmark all algorithms on an NVIDIA Jetson Xavier NX embedded GPU (E-GPU), a platform commonly used in mobile robotics. All accuracy metrics, GMACs, and runtime values for iterative approaches are reported under single forward inference, which may deviate from their best performance reported in the original publications.

B. Performance Evaluation

1) Accuracy Comparison: We report the accuracy of each method in Table II. The primary difference between the two datasets lies in the input quality, including resolution and frame format. Our method achieves consistently strong performance across both scenarios.

DSEC. From the results on the DSEC dataset, we observe that most state-of-the-art methods relying on complex correlation volume construction and iterative refinement strategies do not demonstrate clear advantages when constrained to a single iteration in our evaluation setup. In contrast, our method achieves the best performance on most metrics. Specifically, it outperforms the RAFT by 12%, and surpasses event-only approaches by 10–20%. Although our approach adopts the same correlation volume construction as TID, it achieves better performance due to the proposed temporal feature modeling strategy. Compared to BFlow, our method achieves a 15% improvement, demonstrating its ability to more effectively fuse the complementary strengths of frames

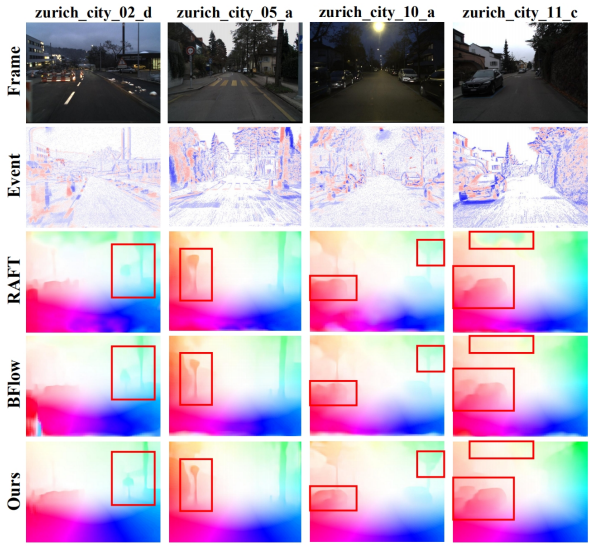

Fig. 5. Qualitative visualization of flow estimation results from frame-only (RAFT [6]), hybrid event-frame (BFlow [12]), and our method using a single forward inference. Frame-based methods struggle in regions with drastic lighting variations (e.g., sky or dark areas), whereas our method produces consistent, structure-aligned flow with clear boundaries.

and events. GMFlow exhibits strong performance on synthetic datasets; however, it shows limited generalization to real-world scenarios with complex lighting conditions, such as nighttime or shadowed environments typical in the DSEC dataset. We also provide qualitative visualizations under both daytime and nighttime conditions in Fig. 5.

AE is the only metric where our method shows slightly inferior performance. We attribute this to the fact that constructing correlation volumes from high-resolution frames can enhance directional flow estimation, which explains the generally higher AE observed in event-only approaches. Nevertheless, even without frame correlation volumes, our method still outperforms all event-based baselines in terms of AE, highlighting its effectiveness in extracting directional cues from frame features.

MVSEC. Due to the use of grayscale images in MVSEC, both RAFT and BFlow experience significant performance degradation. However, our method still demonstrates robust performance, outperforming RAFT and BFlow by 29% and 22%, respectively. Furthermore, our approach consistently surpasses all event-based baselines. These results surprisingly reveal that even with low-resolution grayscale frames, our method can effectively extract temporal features from both modalities and assist accurate flow estimation.

2) Performance under High-Dynamic Motion: We further evaluate the flow estimation performance on the *indoor_flying* sequence in MVSEC, captured from a UAV operating within an indoor motion capture system. Compared to the forward-facing driving scenarios in Table II, this sequence exhibits more complex and abruptly changing optical flow due to the high agility of UAV motion. We include EV-FlowNet [18], an unsupervised baseline, and evaluate BFlow with one (1 itr.) and five (5 itrs.) refinement iterations, with the latter corresponding to its best-performing configuration. All methods are trained and evaluated on the *indoor_flying_3* sequence with different time intervals. Results are summarized in Table III. Our method consistently outperforms most baselines under such drastic motion, demonstrating its potential for deployment on high-agility robotic platforms. Although it slightly underperforms BFlow (5 itrs.) by 3% in EPE and 1% in 1PE, we consider this trade-off acceptable and discuss it further in the next section. A qualitative visualization is provided in Fig. 6

TABLE III

QUANTITATIVE RESULTS ON INDOOR_FLYING SEQUENCE

Method	Indoor-Flying (MVSEC)				
	EPE	AE	1PE	2PE	3PE
GMFlow [24]	2.11	50.27	69.12	37.91	20.96
EV-FlowNet [18]	2.29	56.83	75.78	46.07	27.27
BFlow [12] (*1 itr.*)	1.00	19.51	37.83	9.38	2.93
BFlow [12] (*5 itrs.*)	0.91	17.37	32.53	8.01	2.50
Ours	0.94	19.44	32.79	8.32	2.63

TABLE IV

EVALUATION UNDER VARIOUS INPUT TEMPORAL OFFSETS (Δt)

Metric	Ours (Event + Frame)					RAFT (Frame)	
	$\Delta t = 1$	$\Delta t = 2$	$\Delta t = 3$	$\Delta t = 4$	None	$\Delta t = 1$	$\Delta t = 3$
EPE	0.74	0.74	0.76	0.77	0.82	2.31	4.79
AE	3.92	3.92	4.03	4.09	4.43	6.24	12.74
1PE	17.22	17.79	17.55	19.05	21.51	39.34	64.86

3) Efficiency Comparison: Our method contains only $2.06M$ parameters and achieves the lowest GMAC, along with fast inference on the embedded GPU. It lags only $2ms$ behind TID (the slight difference with [14] is attributed to different versions of the NVIDIA Jetson Xavier NX). Despite encoding additional frame features, we achieve slightly lower GMAC than TID thanks to our lightweight module design (e.g., vanilla and depth-wise convolutions with compact intermediate feature representations). Compared to BFlow under the same evaluation setting, our method is $3\times$ faster and requires only one third of the parameters and GMACs. Although BFlow achieves slightly better accuracy with multiple refinements, its inference time (946 ms at a resolution of 480×640) is nearly $5\times$ longer than ours, making it less suitable for embedded deployment. These results indicate that our hybrid design achieves a strong balance between efficiency and effectiveness, making it well-suited for real-time deployment on embedded platforms.

C. Performance Evaluation with Asynchronous Inputs

Our core contribution lies not only in estimating optical flow under fixed frame-event temporal offsets but also in effectively fusing asynchronous event and frame streams with arbitrary temporal gaps. In this section, we evaluate the robustness of our method under varying temporal offsets between frames and events. Specifically, we evaluate the model with different frame sampling intervals while keeping

Fig. 6. Qualitative visualization of flow estimation on the *indoor_flying* sequence. From left to right: reference frame, ground truth, stacked events, EVFlow [18], BFlow [12], and Ours. For better visualization, all estimated flow maps are masked to show only regions where events are triggered.

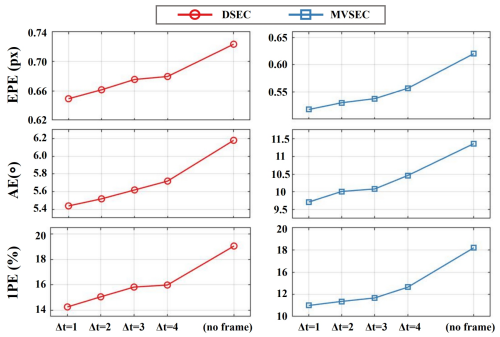

Fig. 7. Accuracy evaluation in event-free regions under varying frame temporal offsets. "no frame" denotes that our method operates solely on event data without any image frames.

the training configuration fixed. This setup allows us to assess the model's generalization ability to real-world scenarios where input streams arrive at irregular or asynchronous rates.

We first evaluate the performance of our method under different temporal offsets, specifically $\Delta t = 1, 2, 3, 4$, as well as in the extreme case where no frame input is provided ($None$). To better understand the impact of asynchronous inputs, we compare our results with the frame-only method, RAFT. The results are summarized in Table IV. Our method exhibits only a modest performance drop as the frame interval increases—approximately 4% at $\Delta t = 4$ and around 10% when no frame input is provided. In contrast, RAFT shows a significant degradation in performance, with its EPE increasing by almost $3\times$ even when the frame interval is enlarged to $\Delta t = 1$. These results demonstrate the robustness of our method to asynchronous inputs and its ability to effectively leverage temporal information from both events and frames under challenging conditions.

Since the correlation volume in our method is primarily constructed from event bins, we further evaluate the impact of varying frame intervals on performance in regions where **no events are triggered**. We report results using both RGB frames (DSEC) and grayscale frames (MVSEC) in Fig. 7. In particular, incorporating a frame input with $\Delta t = 1$ instead of relying solely on events leads to improvement, with an average reduction of 13% in EPE and AE, and a 28% in 1PE within event-free regions. These results demonstrate that frame features can effectively compensate for event-free regions in dense optical flow estimation.

TABLE V
ABLATION STUDY

Method	EPE	AE	1PE	2PE	3PE
full	0.75	3.70	17.92	4.83	2.32
w/o Temporal Init.	0.76	3.76	18.36	4.97	2.40
w/o Temporal Update	0.77	3.78	18.74	5.05	2.42
w/o Temporal Align.	0.79	3.86	20.29	5.46	2.54
align w/ optical flow	0.82	4.14	21.45	5.53	2.55

TABLE VI
ABLATION ON TRAINING STRATEGIES WITH DIFFERENT Δt SETTINGS

Training Strategy	$\Delta t = 1$			$\Delta t = 3$		
	EPE	AE	1PE	EPE	AE	1PE
$\Delta t = 1$	0.511	9.69	10.64	0.544	10.24	12.02
$\Delta t = 2$	0.528	9.98	11.33	0.523	9.81	11.37
$\Delta t = 3$	0.529	9.96	11.45	0.523	9.82	11.43

D. Ablation Study

1) Network Components: To further validate the contributions of each component in our method, we conduct an ablation study by training several model variants on a subset of the DSEC dataset for efficiency.

Temporal Feature Integration: To validate the effect of temporally encoded features from asynchronous inputs on the accuracy of optical flow estimation, we evaluate two variants: (1) **w/o Temporal Init.**: The Lightweight Correlation Module is initialized solely using the warm start technique, as in [7], [14], without incorporating our proposed Temporal Initialization block. (2) **w/o Temporal Update**: Dense flow estimation and temporal alignment are performed using only the correlation volume, without the refinement introduced by our Temporal Update block. We report the accuracy of these two variants in Table V. When either type of temporal feature integration is removed, all evaluation metrics exhibit performance degradation. These results confirm the effectiveness of our proposed integration of temporal features in improving the accuracy of optical flow estimation.

Temporal Feature Alignment: We propose a Temporal Alignment block that aligns temporal features to the next time step, enhancing the temporal continuity of the overall network. To evaluate its effectiveness, we design a variant trained without this block, where inter-sensor updates are performed solely based on sensor-specific features (**w/o Temporal Align.**). As shown in Table V, removing the Temporal Alignment block leads to a noticeable performance drop–approximately 5% in EPE and around 13% in both 1PE and 2PE–highlighting its crucial role in maintaining temporal continuity.

In addition, we also attempt to align temporal features by warping them using the estimated optical flow (**align w/ optical flow**). However, the performance is significantly inferior compared to our proposed method. We attribute this to the limited flexibility of flow warping, whereas learning flexible, task-specific offsets through deformable convolutions plays

a critical role in effective temporal alignment.

2) Training Strategy: In this section, we explore how the choice of temporal offset during training affects the overall performance. To achieve this, we train two model variants with fixed time offsets of 2 and 3 on the MVSEC Driving sequence. We then evaluate all models on time offsets of 1 and 3 to assess their ability to learn short- and long-term temporal dependencies. The results are summarized in Table VI. We find that training the model with a smaller Δt better captures short-term dependencies but performs worse in modeling long-term dependencies. However, this effect does not persist when the training offset increases from 2 to 3. Overall, the training strategy does not significantly affect the model's performance, with all metrics fluctuating within a range of $3\%-6\%$. Nevertheless, we believe that designing a more effective training strategy for sequential learning could further reduce training time and lead to better results, which we consider a promising direction for future work.

V. Conclusion

In this work, we propose a novel method for generating continuous-time dense optical flow from asynchronous event and frame streams using a single forward pass with fast inference on embedded GPUs. To achieve this, we explicitly model temporal features from asynchronous inputs and integrate them into a lightweight correlation volume to boost overall performance. Experiments conducted in real-world scenes demonstrate that our method outperforms existing frame-based, event-based, and hybrid approaches in terms of both accuracy and inference efficiency on embedded systems. Moreover, our model exhibits strong generalization across different frame-event temporal offsets, despite being trained with a fixed offset. These results highlight the potential of our approach for deployment on resource-constrained embedded platforms for continuous-time optical flow estimation, particularly in environments characterized by irregular timing of events and frames, such as underground exploration and planetary missions.

References

[1] H. Rebecq, G. Gallego, E. Mueggler, and D. Scaramuzza, "Emvs: Event-based multi-view stereo—3d reconstruction with an event camera in real-time," *International Journal of Computer Vision*, vol. 126, no. 12, pp. 1394–1414, 2018.

[2] Y. Zhou, G. Gallego, X. Lu, S. Liu, and S. Shen, "Event-based motion segmentation with spatio-temporal graph cuts," *IEEE transactions on neural networks and learning systems*, vol. 34, no. 8, pp. 4868–4880, 2021.

[3] D. Yang, X. Zhang, H. Liu, H. Wu, C. Wang, K. Xu, and X. Ding, "Dense mapping from sparse visual odometry: a lightweight uncertainty-guaranteed depth completion method," *Frontiers in Robotics and AI*, vol. 12, p. 1644230, 2025.

[4] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis *et al.*, "Event-based vision: A survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 1, pp. 154–180, 2020.

[5] S. Zhu, Z. Tang, M. Yang, E. Learned-Miller, and D. Kim, "Event camera-based visual odometry for dynamic motion tracking of a legged robot using adaptive time surface," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 3475–3482.

[6] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 2020, pp. 402–419.

[7] M. Gehrig, M. Millhäusler, D. Gehrig, and D. Scaramuzza, "E-raft: Dense optical flow from event cameras," in *2021 International Conference on 3D Vision (3DV)*. IEEE, 2021, pp. 197–206.

[8] X. Luo, A. Luo, Z. Wang, C. Lin, B. Zeng, and S. Liu, "Efficient meshflow and optical flow estimation from event cameras," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 19 198–19 207.

[9] H. Liu, G. Chen, S. Qu, Y. Zhang, Z. Li, A. Knoll, and C. Jiang, "Tma: Temporal motion aggregation for event-based optical flow," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 9685–9694.

[10] F. Hamann, Z. Wang, I. Asmanis, K. Chaney, G. Gallego, and K. Daniilidis, "Motion-prior contrast maximization for dense continuous-time motion estimation," in *European Conference on Computer Vision*. Springer, 2024, pp. 18–37.

[11] Z. Wan, Y. Mao, J. Zhang, and Y. Dai, "Rpeflow: Multimodal fusion of rgb-pointcloud-event for joint optical flow and scene flow estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 10 030–10 040.

[12] M. Gehrig, M. Muglikar, and D. Scaramuzza, "Dense continuous-time optical flow from event cameras," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 7, pp. 4736–4746, 2024.

[13] F. Mahlknecht, D. Gehrig, J. Nash, F. M. Rockenbauer, B. Morrell, J. Delaune, and D. Scaramuzza, "Exploring event camera-based odometry for planetary robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8651–8658, 2022.

[14] Y. Wu, F. Paredes-Vallés, and G. C. De Croon, "Lightweight event-based optical flow estimation via iterative deblurring," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 14 708–14 715.

[15] M. Gehrig, W. Aarents, D. Gehrig, and D. Scaramuzza, "Dsec: A stereo event camera dataset for driving scenarios," *IEEE Robotics and Automation Letters*, 2021.

[16] A. Z. Zhu, D. Thakur, T. Özaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The multivehicle stereo event camera dataset: An event camera dataset for 3d perception," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2032–2039, 2018.

[17] R. Benosman, C. Clercq, X. Lagorce, S.-H. Ieng, and C. Bartolozzi, "Event-based visual flow," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 2, pp. 407–417, 2013.

[18] A. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "Ev-flownet: Self-supervised optical flow estimation for event-based cameras," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[19] Y. Li, Z. Huang, S. Chen, X. Shi, H. Li, H. Bao, Z. Cui, and G. Zhang, "Blinkflow: A dataset to push the limits of event-based optical flow estimation," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 3881–3888.

[20] R. Pellerito, M. Cannici, D. Gehrig, J. Belhadj, O. Dubois-Matra, M. Casasco, and D. Scaramuzza, "Deep visual odometry with events and frames," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 8966–8973.

[21] G. Gallego, H. Rebecq, and D. Scaramuzza, "A unifying contrast maximization framework for event cameras, with applications to motion, depth, and optical flow estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3867–3876.

[22] X. Wang, K. Chen, W. Yang, L. Yu, Y. Xing, and H. Yu, "Fe-detr: Keypoint detection and tracking in low-quality image frames with events," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 14 638–14 644.

[23] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 764–773.

[24] H. Xu, J. Zhang, J. Cai, H. Rezatofighi, and D. Tao, "Gmflow: Learning optical flow via global matching," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 8121–8130.