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

Optical Flow Estimation using Speck Neuromorphic Hardware

Manupriya Singh[1], Dequan Ou[1], Jesse J. Hagenaars[1] and Guido C. H. E. de Croon[1]

***Abstract*— Neuromorphic hardware and spiking neural networks (SNNs) offer a bio-inspired path to low-latency, energy-efficient computation by emulating the brain's asynchronous, spike-based processing. This is particularly attractive for resource-constrained robots that are tightly limited in size, weight, and power. We propose a neuromorphic approach to real-time optical flow estimation tailored to the SynSense Speck system-on-chip, which integrates a Dynamic Vision Sensor (DVS) with a neuromorphic processor. Our inference architecture combines spiking and artificial neural layers in a hybrid SNN–ANN framework, enabling the use of Speck to perform regression for closed-loop drone control, an application not previously demonstrated on this chip. Despite its compact form factor, the system produces dense flow in real time and achieves stable indoor hover and forward flight using flow-based control. The hybrid pipeline runs ∼2× faster than an ANN-only baseline at identical power, highlighting the promise of neuromorphic sensing and processing for ultra-efficient autonomous flight in real-world scenarios. Code and data are available at: https://mavlab.tudelft.nl/speck-optical-flow**

I. Introduction

Micro Aerial Vehicles (MAVs) are increasingly used in applications such as industrial inspection [1], [2], search and rescue [3], and precision agriculture [4]. Accurate and energy-efficient perception is essential for enabling real-time autonomy of these resource-constrained platforms. Neuromorphic computing offers a promising solution by emulating the brain's event-driven processing paradigm [5], [6]. Unlike traditional clocked systems that process redundant information frame by frame, neuromorphic processors operate asynchronously using sparse, spike-based data, leading to lower power consumption and reduced latency [7]. Paired with event-based vision sensors [8], which output pixel-level brightness changes instead of full frames, neuromorphic systems avoid processing static, redundant visual information. However, most current event-camera pipelines—including ours—rely on temporal binning to convert asynchronous events into frame-like representations, and fully asynchronous end-to-end processing remains an open challenge.

A key perception task in this context is event-based optical flow estimation. Optical flow is foundational for ego-motion estimation [9], obstacle avoidance [10], and visual navigation [7], [11], [12]. While many systems rely on sparse [13] or frame-based [14] approximations, dense optical flow provides per-pixel motion estimates [15], [16], enabling finer control in complex environments. However, conventional dense flow algorithms are often too computationally expensive for MAVs, making the combination of event-based sensing and neuromorphic computing a compelling alternative.

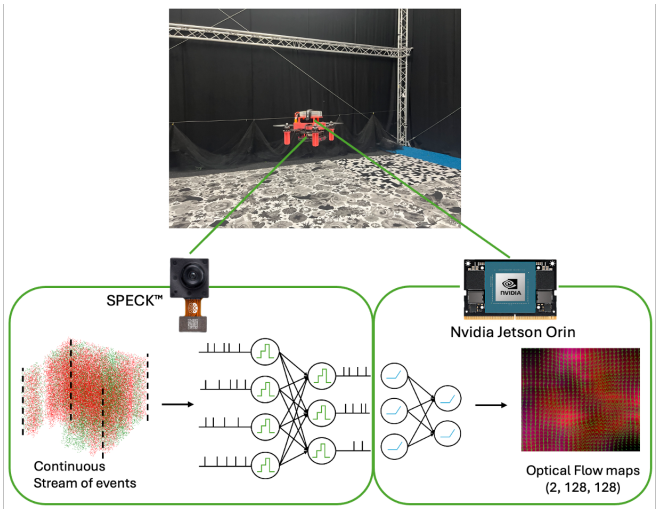


Fig. 1: Autonomous drone control driven by optical flows estimated using the Speck neuromorphic processor.

This work introduces a hardware-constrained approach for estimating dense optical flow from event-based data using the SynSense Speck2 neuromorphic chip [17]. Speck is a tiny, fully neuromorphic system-on-chip combining a DVS with a DYNAP-CNN processor, using Integrate-and-Fire (IAF) neurons for asynchronous, sparse, low-latency computation. Our architecture (Fig. 2) follows an encoder-memory-decoder structure inspired by IDNet [18], trained with a self-supervised contrast maximization loss [19], [20] and an iterative warping strategy [19]. The model is trained as a conventional ANN with ReLU activations, after which the encoder is converted into a spiking implementation via the Speck toolchain for deployment on the DynapCNN core.

The main contributions of this work are:

- To the best of our knowledge, the first demonstration of closed-loop control using dense optical flow on a drone with the Speck neuromorphic processor.
- Real-time inference of dense optical flow using a compact neuromorphic system, showcasing its suitability for embedded aerial robotics (Fig. 1).
- A hybrid SNN-ANN architecture integrating a spiking

*Part of this project was financed by the Dutch Research Council (NWO) under grant number NNWA.1292.19.298, and by the NWO VICI personal grant programme under grant number 20663.

[1]All authors are with the Micro Air Vehicle Lab of the Faculty of Aerospace Engineering, Delft University of Technology, 2629 HS Delft, The Netherlands M.Singh-5@tudelft.nl, D.Ou@tudelft.nl, J.J.Hagenaars@tudelft.nl, G.C.H.E.deCroon@tudelft.nl

encoder on Speck with artificial neural layers, preserving performance compared to a pure ANN counterpart while reducing latency.
- Quantitative evaluation of latency, demonstrating the viability of event-based spiking perception for onboard drone control.

II. Related Work

A. Event-Based Optical Flow

Event cameras have significantly advanced optical flow estimation by offering low latency, high dynamic range, and sparse output. Learning-based approaches have significantly advanced event-based optical flow. Zhu et al. introduced EV-FlowNet [15], the first self-supervised framework trained with photometric loss from conventional frames, and later extended this to unsupervised joint learning of flow, depth, and egomotion directly from events [21]. Gehrig et al. proposed E-RAFT [22], a dense optical flow model adapting recurrent all-pairs field transforms to event data, achieving state-of-the-art accuracy. Paredes-Vallés et al [23] revisited event representations by learning image reconstruction from events through photometric constancy, enabling more robust self-supervised pipelines.

While influential, these methods typically rely on large, image-inspired architectures that process events over extended temporal windows, effectively trading latency for accuracy. Latency can be reduced by shortening these windows and exploiting finer temporal structure. Subsequent works [19], [20] advanced in this direction, restructuring the training pipeline to minimize temporal encoding in the input and improve the convexity of the loss landscape, resulting in more stable training and high accuracy while preserving low-latency inference.

More recently, Wu et al. [18] introduced IDNet, a lightweight encoder-memory-decoder architecture specifically designed for event-based optical flow estimation. In contrast to earlier heavy-weight models, IDNet eliminates skip connections and utilizes convolutional recurrent layers such as ConvGRU, offering a compact yet effective solution.

However, none of the aforementioned approaches have been realized on neuromorphic processors, and most rely on architectures incompatible with resource-constrained platforms such as Speck. Nonetheless, they form the critical foundation on which efficient, and eventually neuromorphic-compatible event-based flow estimation is being built.

B. Control using Neuromorphic Hardware

In [24], an SNN was implemented on Loihi chip [25] to control a bench-mounted dual rotor, aligning its roll angle with a black-and-white disk placed in front of the camera. The network combined a visual Hough transform [26] for line orientation detection with a proportional–derivative (PD) controller that generated motor commands. Dupeyroux et al. [12] realized the first neuromorphic control of a flying drone, deployed an SNN-based landing controller on Intel's Loihi chip. The spiking neural network used only 35 neurons to compute thrust commands from ventral flow divergence, trained through evolution and validated in real-world flights. More recently, Paredes-Vallés et al. [7] demonstrated a fully neuromorphic pipeline for autonomous drone flight, where a spiking network estimated optical flow from raw event data and used it for closed-loop control. Deployed on Loihi chip, the system achieved stable hovering, landing, and lateral maneuvers at 200 Hz with only milliwatt-level power for network inference, showing the potential of fully neuromorphic vision–action loops.

While these studies underscore the potential of neuromorphic control systems, they rely on diverse sensing modalities and platforms such as Intel's Loihi, which provide substantially greater computational resources than Speck. For example, Loihi-2 integrates approximately 1M neurons and 128 neuromorphic cores per chip, compared to Speck's 0.32M neurons and 9 convolutional cores/layers [27], [28]. To date, none of these approaches have been demonstrated on Speck, which operates within a far more constrained computational budget, and with a more limited Integrate-And-Fire neural model.

C. Previous Speck Implementations

Previous implementations on the Speck chip have demonstrated its viability for real-time, on-device neuromorphic perception tasks under severe energy and computational constraints. The LENS system [29], which uses a spiking neural network on the Speck chip to perform large-scale visual place recognition on a hexapod robot. LENS achieved localization performance comparable to conventional Sum-of-Absolute-Differences (SAD) across traversals of up to 8 km, while consuming less than 8% of the energy. Caccavella et al. [30] presented the first on-chip spiking neural network for event-based face detection, achieving a mean average precision of $\sim$0.6 while consuming only $\sim$20 mW.

These works highlight Speck's strengths in pattern recognition, yet optical flow demands continuous temporal processing that may exceed the chip's minimal resources. We investigate whether Speck can sustain dense optical-flow estimation within a real perception–action loop. Prior event-based flow methods achieve high accuracy, but typically rely on heavyweight encoders, and no work has demonstrated closed-loop flight with Speck. We address this gap with a pipeline that produces dense flow on Speck for real-time autonomous control.

III. Methodology

This section describes how we turn asynchronous events into dense optical flow and use it for autonomous flight. We first define the input and learning signal, then detail the network and on-chip deployment, the drone hardware, and finally the closed-loop perception–and–control pipeline.

A. Input Representation and Self-Supervised Loss Function

The event camera on the Speck processor outputs asynchronous events $e_i = (x_i, y_i, t_i, p_i)$ from 128×128 DVS array, where $p_i \in \{+1, -1\}$ encodes ON/OFF polarity. Events

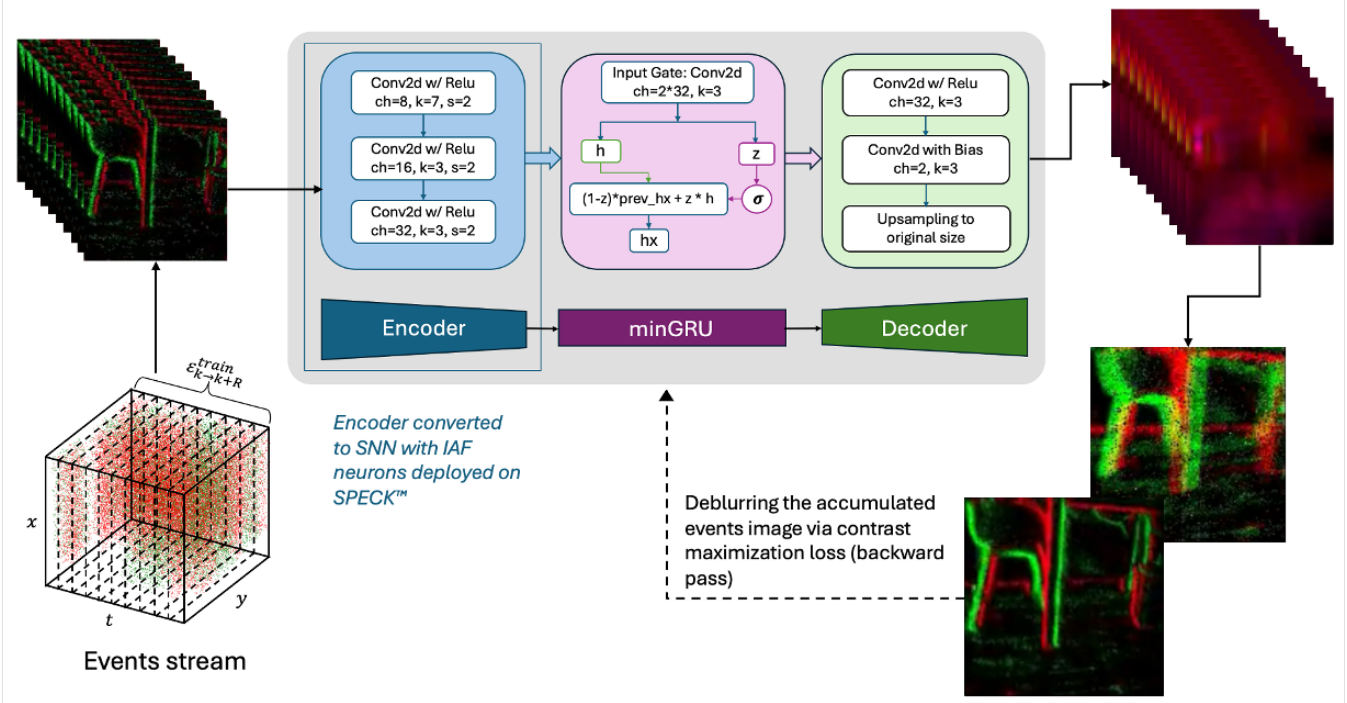


Fig. 2: Self-Supervised network training pipeline. The incoming event stream is first partitioned into time segments of 10 ms. These segments are fed sequentially into the network. For each segment, the model predicts a dense optical flow map (visualized using the HSV color model, with hue indicating motion direction and value indicating motion magnitude), assigning a motion vector to every input event. After processing 10 segments, a backward pass is performed using the contrast maximization loss to update the network weights.

fire only when a local brightness change exceeds a contrast threshold, yielding a sparse, high-temporal-resolution stream.

For training, we split the stream into non-overlapping 10 ms windows. Within each window, events are accumulated by polarity into a two-channel tensor of shape $(\text{Batch}, 2, H, W)$ (one channel per polarity). These windows are fed sequentially to the network. We apply no explicit temporal encoding; the model learns dynamics directly from the sequence under a self-supervised Contrast-Maximization (CM) loss [31], [32].

CM models these events as being primarily induced by motion: if an event is warped $e_k \mapsto e'_k = (t_{\text{ref}}, \mathbf{x}'_k, p_k)$ using an estimated motion increment $\Delta\mathbf{x}_k$, events triggered by the same moving edge tend to realign, which is expected to increase the contrast of the image of warped events (IWE).

We adopt the multi-reference CM framework of [19], estimating optical flow on thin time slices with a recurrent model. Concatenating per-slice flows $\{\mathbf{u}_i\}$ yields the cumulative warp $\Delta\mathbf{x}_k = \sum_i (\Delta t_i\, \mathbf{u}_i)(e_k)$, enabling iterative warps to neighboring slices; under this procedure, better flow estimates are associated with sharper IWEs at all references.

The CM objective averages, over all references, a per-reference, normalized timestamp contribution:

$$\mathcal{L}_{\text{CM}} = \frac{1}{T+1} \sum_{t_{\text{ref}}=0}^{T} \frac{\sum_k \bar{t}_k(t_{\text{ref}})\, \kappa(\mathbf{x}_k)}{\sum_k \kappa(\mathbf{x}_k)}, \qquad (1)$$

where $\bar{t}_k(t_{\text{ref}})$ denotes the (weighted) timestamp contribution of event e_k to the IWE at reference t_{ref}, and κ is a bilinear weighting kernel. To mitigate event collapse and boundary artifacts, we (i) scale by the number of pixels that received at least one warped event and (ii) mask events that leave the image domain at any point as done in [19], [20]. All training is performed offline on pre-recorded event data. The deployed system performs inference only; online learning during flight is not supported in the current implementation.

B. Network Architecture and On-Speck Deployment

We use a compact *encoder–memory–decoder* design tailored for streaming events and real-time dense flow. The full model (Fig. 2) has only 34.9k parameters and comprises:

- Encoder: A simple three-layer convolutional feedforward encoder, which reduces memory overhead and simplifies hardware implementation.
- Memory: a computationally efficient recurrent unit (minGRU) [33] that captures short-term dynamics with minimal complexity.
- Decoder: two convolutions plus upsampling to reconstruct dense flow at input resolution.

At present, only the *encoder* is deployed on the Speck chip, where we perform ANN→SNN conversion. The recurrent unit and decoder are executed off-chip on a Nvidia Jetson Orin NX. This restriction stems from current hardware limitations: Speck does not support gated recurrent units, and our attempts to substitute plain recurrent feedback

were unsuccessful. While Samna provides a layer-destination mechanism to emulate recurrence [34], in practice it failed to yield valid flow estimates in our trials, and, as noted by the developers, the mechanism can risk deadlocks. An alternative would be to run a fully spiking architecture off-chip on the Jetson using frameworks such as snnTorch [35], which support CUDA-accelerated recurrent spiking layers. However, our primary goal is to maximize on-chip processing on Speck and evaluate its viability for closed-loop control, rather than offloading spiking computation to a GPU. The current hybrid design serves as a stepping stone: the SNN encoder runs where neuromorphic hardware excels—in early, sparse event encoding on Speck—while the ANN memory and decoder handle temporal integration and dense regression on the Jetson, where stable gating dynamics and precise continuous outputs are required. As neuromorphic hardware matures to support recurrent architectures on-chip, the full pipeline could migrate to dedicated spiking hardware.

In the ANN-to-SNN conversion, ReLU activations in ANN are replaced by IAF neurons [36] with membrane dynamics defined as

$$\tau \dot{v} = -v_{\text{leak}} + R\,(I_{\text{syn}} + I_{\text{bias}}) \tag{2}$$

with v_{leak} being the constant leak. Because it is constant, it can be joined with the regular bias, such that the equation v will be simplified to (3);

$$\tau \dot{v} = R\left(I_{\text{syn}} + \tilde{I}_{\text{bias}}\right) \tag{3}$$

with

$$\tilde{I}_{\text{bias}} = v_{leak}/R \tag{4}$$

with constant membrane resistance R (set to unity to match units) and time constant τ. In practice we omit bias terms to keep spike integration simple and stable.

After the encoder, output spikes are collected by a `samna` spike-collection node [37] every 10 ms and converted into frame-like tensors for the off-chip minGRU and decoder.

C. Drone Architecture

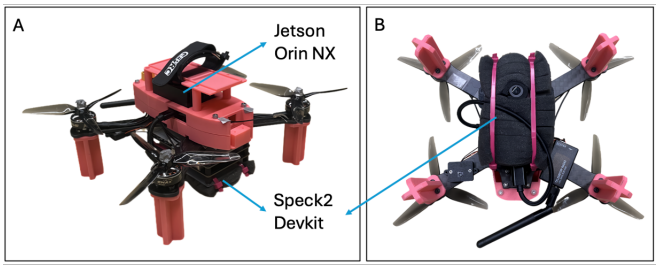


Fig. 3: Figure A shows the isometric view of the drone containing an Nvidia Jetson Orin NX and a Speck facing downwards. Figure B shows the Speck platform.

The drone platform employed in this work features a dual-computer architecture (Fig. 3) that enables autonomous flight and real-time onboard perception. Low-level flight control is managed by a Kakute H7 flight controller running PX4 firmware, which handles attitude stabilization and thrust regulation. High-level processing tasks, including event-based optical flow inference (in conjunction with the Speck) and control decision-making, are executed on an onboard Nvidia Jetson Orin NX companion computer.

Communication between the companion computer and the flight controller is established using ROS 2. Control commands are transmitted via PX4's `attitude_setpoint` interface, allowing the system to issue desired pitch and roll values for real-time flight control.

D. Closed-Loop Perception–and–Control on the Drone

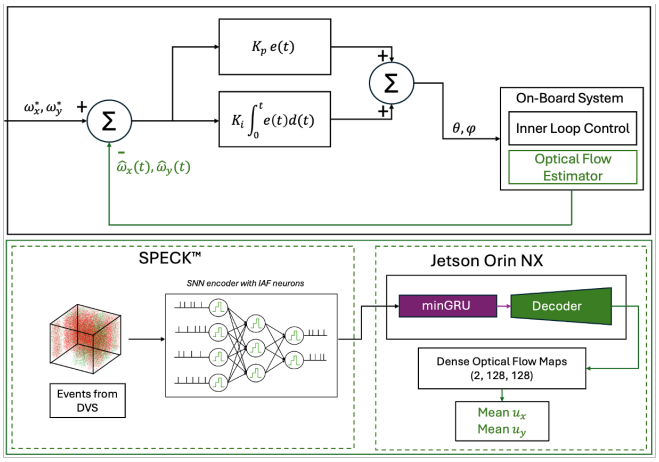


Fig. 4: Overview of the closed-loop perception and control system. **Top:** Outer-loop control architecture for drone attitude control. The velocities with the asterisk are the commanded velocities. **Bottom:** Hybrid optical–flow estimator. Events are encoded on Speck by an SNN encoder and processed on the Jetson Orin NX by a minGRU and decoder to yield dense flow maps $(2, 128, 128)$. Mean flow provides $\hat{\omega}_x, \hat{\omega}_y$.

In a minimal-sensing setup, drones can control their egomotion purely with optical flow [38]. Hence, we close the outer loop based on optical flows rather than reconstructing the full state. Lateral body motion driven by roll/pitch manifests as mean horizontal optical flows.

The green block in Fig. 4 (bottom) details the hybrid optical–flow estimator. A DVS integrated on Speck streams events continuously into an on-chip SNN encoder (DynapCNN with non-leaky IAF neurons), producing events which after binning in 10ms windows on Jetson Orin NX, create compact feature maps of size $(32, 16, 16)$. These are forwarded to a lightweight recurrent unit which aggregates temporal context, and a two-layer convolutional decoder upsamples to dense flow fields of shape $(2, 128, 128)$.

For onboard use, we compute the spatial mean of the decoder's final dense flow output of shape $(2, 128, 128)$, averaging each of the two flow components (u and v) across the 128×128 spatial grid to yield a single (u, v) motion vector per frame.

Figure 4 (top) depicts the outer–loop PI controller that generates attitude commands. The desired angular rates

ω_x^*, ω_y^* are compared to the optical–flow–based estimates $\hat{\omega}_x(t), \hat{\omega}_y(t)$, yielding errors $e_i(t) = \omega_i^* - \hat{\omega}_i(t)$ for $i \in \{x, y\}$. These errors are processed by proportional and integral terms ($K_p e(t)$ and $K_i \int_0^t e(\tau)\, d\tau$), whose sum is converted into roll and pitch set-points (ϕ, θ).

IV. Experiments

We evaluate our approach along four fronts: (i) flow quality via the RSAT alignment metric on Speck and UZH-FPV data across different ANN architectures; (ii) real-time viability by benchmarking end-to-end throughput and latency against an ANN-only baseline; (iii) consistency with a commercial optical-flow sensor; and (iv) practical utility through closed-loop forward-flight and hovering experiments driven solely by optical-flow cues.

A. Validation on Speck Data: RSAT Loss

To quantify how much the predicted flow actually *sharpens* the event alignment, we use the RSAT loss introduced in [20]. This metric is suitable when ground truth is unavailable and lets us compare different architectures under identical input streams. For a batch of events $\mathcal{E} = \{(x_i, y_i, t_i)\}_{i=1}^N$ and flow maps $\mathbf{u}$, let $\mathcal{L}_{\text{CM}}(\mathcal{E}, \mathbf{u})$ denote the CM loss (Sec. III-A). RSAT is defined as the ratio between the (blur) cost after warping with the estimated flow and the cost when using zero flow:

$$\text{RSAT} = \frac{\mathcal{L}_{\text{CM}}(\mathcal{E}, \mathbf{u})}{\mathcal{L}_{\text{CM}}(\mathcal{E}, \mathbf{0}) + \epsilon}, \qquad \epsilon \approx 10^{-9}. \tag{5}$$

A value of 1 means the flow did not improve sharpness over the unwarped case; values < 1 indicate better temporal alignment (sharper IWEs).

We report RSAT on two Speck collected sequences: a *simple* patterned planar motion and a *complex* room scene and one of the sequences from UZH-FPV dataset left for validation. This loss is tested on pure ANN pipelines.

TABLE I: RSAT (↓ better) on Speck and UZH-FPV data.

Model	Simple	Complex	Indoor_forward_11_davis
Simple Enc. + minGRU (32ch)	0.80	0.85	0.89
Simple Enc. + minGRU (64ch)	0.79	0.84	0.87
Simple Enc. + Recurrency (64ch)	0.76	0.81	0.87
Simple Enc. + GRU (32ch)	0.76	0.82	0.86
Simple Enc. + GRU (64ch)	0.76	0.815	**0.84**
Original (ResNet + GRU)	**0.75**	**0.80**	0.86

As shown in Table I, the simplified minGRU 32 channels variant remains competitive: although its RSAT is slightly higher than the larger GRU-based models, it still achieves substantial sharpening (RSAT $\ll 1$) on all three sequences.

This supports our claim that a simplified encoder–memory design can deliver accuracy comparable to heavier ResNet-style encoders while offering markedly lower complexity and latency, thereby meeting the efficiency targets of our Speck deployment.

B. Flow Visualization and Endpoint-Error Comparison: Speck+ANN vs. Pure ANN

In this experiment, we compare the ANN-only and Speck encoder + ANN (memory & decoder) outputs, to evaluate the difference of using spiking front-end (Fig. 5).

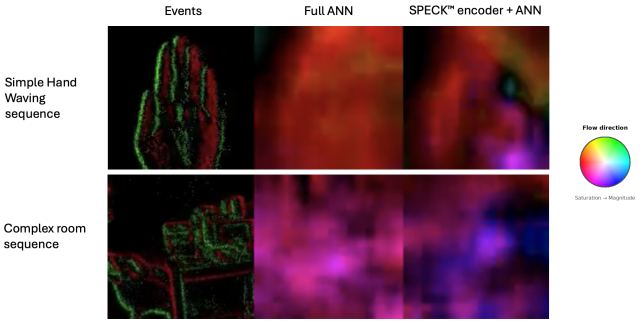


Fig. 5: HSV visualization of representative flow maps for *simple* and *complex* sequences. Hue encodes flow direction; saturation encode magnitude.

In the absence of ground truth, we quantify agreement between models using the Average Endpoint Error (AEE), computed per pixel as

$$\text{AEE} = \frac{1}{|\Omega|} \sum_{(i,j)\in\Omega} \sqrt{(u_1 - u_2)^2 + (v_1 - v_2)^2},$$

where (u, v) are the horizontal/vertical flow components and Ω is the set of valid pixels in the 128×128 maps. As a sanity baseline, we also compare the hybrid output to a random flow field with components sampled uniformly from $(-1, 1)$; as expected, the AEE against random increases with the random field's range.

TABLE II: Average Endpoint Error (AEE) comparing Hybrid outputs to ANN and Random fields.

Sequence	ANN vs. Hybrid	Random vs. Hybrid
Simple hand-waving	**0.83**	1.33
Complex room	**2.04**	2.32

Table II reports AEE for two sequences. On the *simple* hand-waving sequence, the hybrid output is substantially closer to the full ANN than to random (AEE 0.83 vs. 1.33). On the *complex* room sequence, the hybrid remains closer to the full ANN (AEE 2.04 vs. 2.32), though the margin is smaller. This may be because random flows sampled in $(-1, 1)$ overlap with the true motion range, inflating the apparent similarity. Overall, the results indicate that the neuromorphic front end preserves the global flow structure learned by the ANN; differences between hybrid and full ANN are small relative to a naive baseline and grow only modestly with scene complexity.

C. Live Inference: Optical Flow Evaluation

To validate real-time performance, we conducted live inference with the hybrid Speck+ANN pipeline by translating

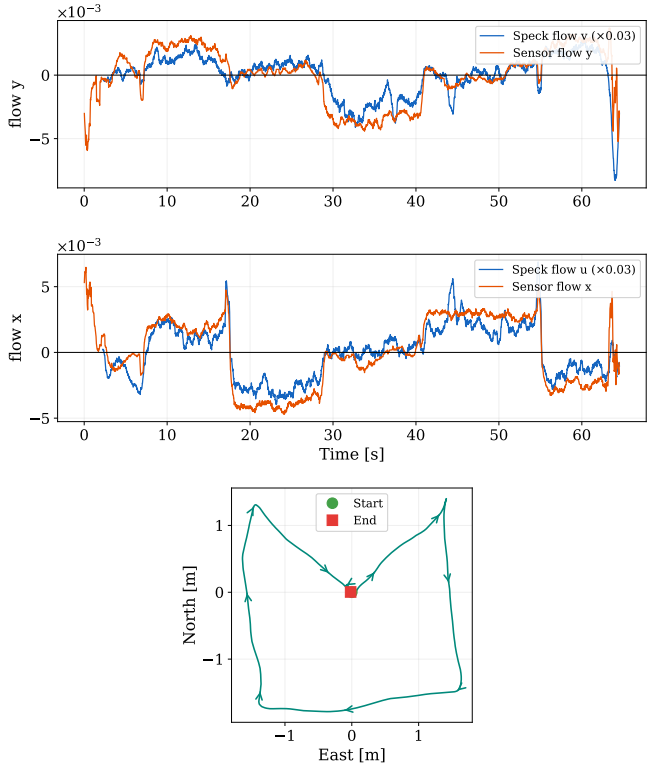


Fig. 6: Live optical flow experiment. **Top:** Mean Speck flow over time in comparison to flow recorded by the MTF-01P flow sensor as the system followed a square path while mainteining constant heading (clockwise, starting at the center). **B:** Corresponding trajectory.

the system along a square trajectory while remaining constant heading over a patterned surface (Fig. 6). From each dense flow map, we computed the spatial mean of the horizontal and vertical components (u, v) to obtain two flow time series. To properly compare with the estimated flow from Speck (referred to as Speck flow u, v), we also mounted a mature off-the-shelf flow sensor, the MicroAir MTF-01P, to serve as 'ground truth' (referred to as Sensor flow x, y). We multiplied the Speck flow by a scaling factor of 0.03 to better compare the two. Furthermore, for better visualization, we applied a 500 ms moving average to both signals to remove noise.

Agreement was quantified using normalized root mean square error (NRMSE) and Pearson correlation coefficient. After amplitude scaling (factor ≈ 0.027), results showed strong correspondence on both axes: u—NRMSE $= 11.4\%$, $r = 0.88$; v—NRMSE $= 13.1\%$, $r = 0.80$. Together these results confirm that the Speck-derived optical flow captures the correct motion dynamics, closely tracking the commercial sensor across the full rectangular trajectory. Please note that commercial optical flow sensors only output a global horizontal and vertical flow vector, whereas our pipeline produces dense optical flow.

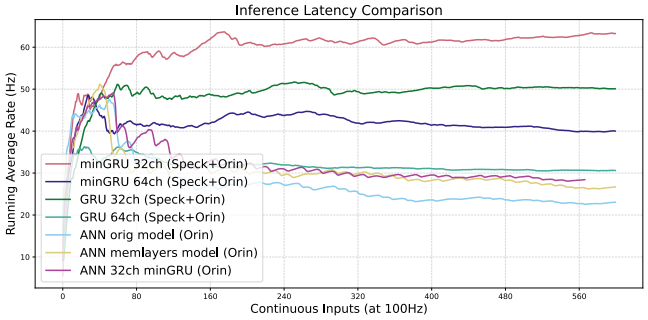


Fig. 7: Average inference frequency f over time for the six architectures compared in RSAT. The hybrid Speck+ANN pipeline (minGRU, 32 channels, Speck+Orin) sustained the highest average rate, approximately $2\times$ that of the pure ANN baseline (minGRU, 32 channels, Orin). All flows were measured on comparable continuous live event input streams.

D. Latency Comparison

To assess the runtime gain of the hybrid pipeline (Spiking encoder on-Speck, minGRU + decoder on-Orin), we measured the average end-to-end inference frequency on comparable inputs for seven models (Fig. 7). Frequency is defined as $f = 1/\Delta t$, where Δt is the interval between consecutive dense flow estimates. While the camera is stationary no events are generated, so no model produces flow. Once motion begins, the curves align briefly, with latency differences only becoming apparent later because the running-average frequency requires a sufficient history of outputs to reveal them. On average, the hybrid model (minGRU 32ch, Speck+Orin) sustains an inference rate approximately $2\times$ that of the pure ANN baseline (Orin) and consistently outperforms the GRU variants (Speck+Orin), confirming that a spiking front end on Speck yields a clear latency reduction.

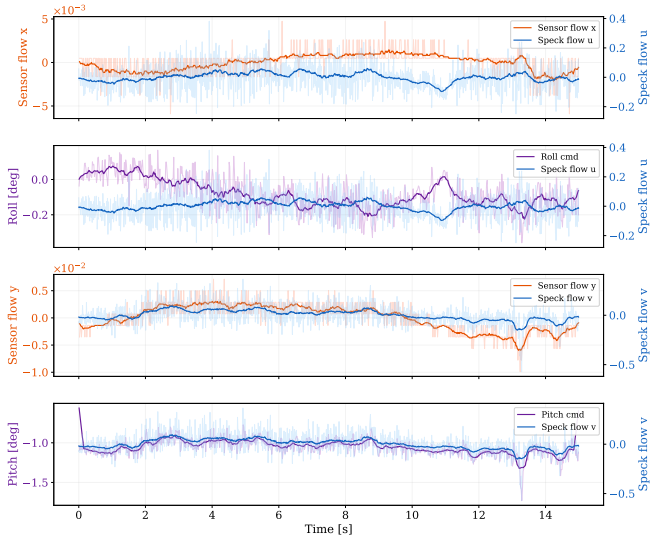
E. Flight Validation: Flow-Only Control

We validated the flow-only controller in flight by commanding the MAV to fly forward and to hover (Fig.8), using only the spatially averaged image-plane flows (u, v) as feedback. The vehicle took off in *Position* mode to ensure a safe initial attitude hold; after stabilization we switched to *Auto* mode, in which an outer-loop PI controller acts directly on the Speck flow observables (Fig.4): errors are $e_u = -u$ and $e_v = -v$; proportional plus integral terms generate roll and pitch commands from (e_u, e_v), while yaw rate is held at zero. Figure 9 depicts both flights in *Auto* mode, comparing the Speck flow observables with the co-mounted MTF-01P sensor flow and the corresponding attitude commands.

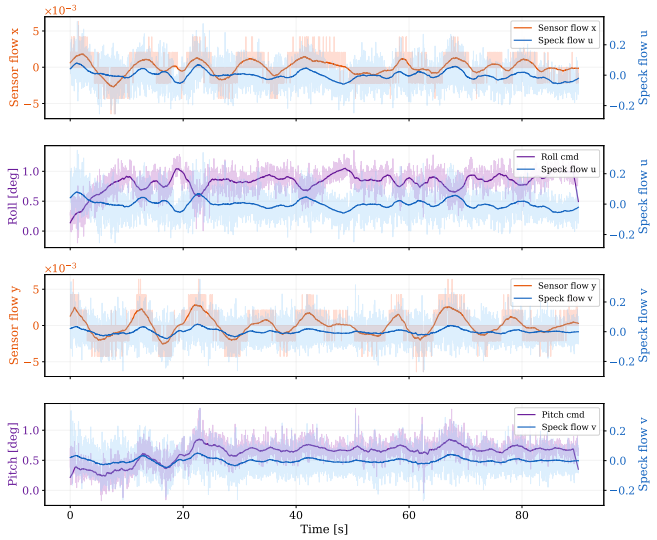
During the forward flight (Fig. 9a), a sustained pitch command of $\sim -1.1,^\circ$ drives the vehicle forward, and the longitudinal Speck flow v tracks the sensor flow closely ($r = 0.94$, NRMSE $= 13.6\%$). The lateral axis sees little motion, as expected for straight-line flight.

For the 90-second hover (Fig. 9b), the controller regulates both flow channels toward zero. Mean body-frame velocities remain small ($\bar{v}_{\text{fwd}} = -0.005$m/s, $\bar{v}_{\text{rgt}} = -0.031$m/s), con-

Fig. 8: Time-lapse composite of the MAV controlled using only onboard event-based optical flow (*Auto* mode). The safety rope was kept slackened (not tight) during flight. **Top**: forward flight. **Bottom**: 90-second hovering test.



(a) Forward flight (∼15 s in *Auto* mode): Speck flow (u, v) vs. sensor flow (x, y) and roll/pitch commands.



(b) Hover (∼90 s in *Auto* mode): Speck flow (u, v) vs. sensor flow (x, y) and roll/pitch commands.

Fig. 9: Flight tests in *Auto* mode using Speck-based optical flow control. Each subfigure shows four rows: sensor flow vs. Speck flow for the lateral (u/x) and longitudinal (v/y) axes, together with the corresponding roll and pitch commands. Light traces show raw signals; solid lines are smoothed with a moving average (with time windows of 300 ms and 2 s.

firming stable station-keeping. Speck-to-sensor agreement is $r = 0.80$ (v axis) and NRMSE $= 14.6\%$–16.5%. Small but persistent attitude trims are visible (roll $\approx 0.8,^{\circ}$, pitch $\approx 0.6,^{\circ}$); because the controller regulates optical flow rather than body angles, the PI integrator automatically supplies the constant roll/pitch needed to cancel fixed biases and drive $\bar{u}, \bar{v} \to 0$.

V. Discussion

We showed that a hybrid architecture—Speck as a spiking front-end encoder with an off-chip minGRU decoder—can produce dense optical flow suitable for closed-loop MAV control. Replacing a 64-channel ConvGRU with a 32-channel minGRU and streamlining the encoder for neuromorphic constraints retained most of the accuracy of heavier baselines, demonstrating that high-quality flow is achievable in compact form.

The main practical gain is latency: the hybrid pipeline (minGRU 32ch, Speck+Orin) sustained roughly $2\times$ higher inference rates than a pure ANN baseline (Orin) with no measurable change in power consumption, confirming the value of offloading early processing to event-driven hardware. For context, FastFlowNet [39] (1.37 M parameters) reaches only 5.7 FPS on a Jetson TX2 at 1024×436, and NeuFlow-v2 [40] attains 20 FPS on a Jetson Orin Nano at 512×384. Both operate at higher resolutions than our 128×128 input, complicating direct comparison, yet they illustrate the computational burden dense flow imposes on embedded platforms. Closed-loop experiments further confirmed stable lateral-motion tracking during hover despite noise from the limited spatial resolution.

Limitations remain. Only the encoder runs on Speck; the recurrent unit and decoder execute on the Jetson because on-chip recurrence is not yet supported, making the hybrid design the only viable deployment strategy under current hardware constraints.

VI. Conclusion

We demonstrated the first dense event-based optical flow pipeline to partially run on the Speck neuromorphic chip and close the loop on a flying platform. A Speck-resident spiking encoder paired with a lightweight minGRU–decoder stack delivered competitive accuracy and robust control, sustaining $\sim 2\times$ the inference rate of a purely ANN baseline. These results highlight that with co-designed architectures, dense, real-time perception–action loops are feasible on severely constrained neuromorphic hardware, paving the way for fully neuromorphic autonomous flight.

References

[1] Myssar Jabbar Hammood Al-Battbootti, Iuliana Marin, Nicolae Goga, and Ramona Popa. Oil and gas pipeline monitoring during covid-19 pandemic via unmanned aerial vehicle, 2021.

[2] Aleixo Cambeiro Barreiro, Clemens Seibold, Anna Hilsmann, and Peter Eisert. Automated damage inspection of power transmission towers from uav images. In *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2022) - Volume 5: VISAPP*, pages 382–389. INSTICC, SciTePress, 2022.

[3] Jiaping Xiao, Phumrapee Pisutsin, and Mir Feroskhan. Collaborative target search with a visual drone swarm: An adaptive curriculum embedded multistage reinforcement learning approach. *IEEE Transactions on Neural Networks and Learning Systems*, 36(1):313–327, 2025.

[4] E. C. Nunes. Employing drones in agriculture: An exploration of various drone types and key advantages, 2023.

[5] Catherine D. Schuman, Thomas E. Potok, Robert M. Patton, J. Douglas Birdwell, Mark E. Dean, Garrett S. Rose, and James S. Plank. A survey of neuromorphic computing and neural networks in hardware, 2017.

[6] O. Richter, Y. Xing, M. De Marchi, C. Nielsen, M. Katsimpris, R. Cattaneo, Y. Ren, Y. Hu, Q. Liu, S. Sheik, T. Demirci, and N. Qiao. Speck: A smart event-based vision sensor with a low latency 327k neuron convolutional neural network processing pipeline. Technical report, SynSense AG, Switzerland; SynSense PR China; Bio-Inspired Circuits and Systems Lab, University of Groningen, Netherlands, 2023.

[7] Federico Paredes-Vallés, Jesse Hagenaars, Julien Dupeyroux, Stein Stroobants, Yingfu Xu, and Guido de Croon. Fully neuromorphic vision and control for autonomous drone flight, 2023.

[8] Christoph Posch, Teresa Serrano-Gotarredona, Bernabe Linares-Barranco, and Tobi Delbruck. Retinomorphic event-based vision sensors: Bioinspired cameras with spiking output. *Proceedings of the IEEE*, 102(10):1470–1484, 2014.

[9] Liren Yang. Ego-motion estimation based on fusion of images and events, 2022.

[10] Raoul Dinaux, Nikhil Wessendorp, Julien Dupeyroux, and Guido C. H. E. de Croon. Faith: Fast iterative half-plane focus of expansion estimation using optic flow. *IEEE Robotics and Automation Letters*, 6(4):7627–7634, 2021.

[11] Jesse J. Hagenaars, Federico Paredes-Vallés, Sander M. Bohté, and Guido C. H. E. de Croon. Evolved neuromorphic control for high speed divergence-based landings of mavs. *IEEE Robotics and Automation Letters*, 5(4):6239–6246, 2020.

[12] Julien Dupeyroux, Jesse J. Hagenaars, Federico Paredes-Vallés, and Guido C. H. E. de Croon. Neuromorphic control for optic-flow-based landing of mavs using the loihi processor. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 96–102, 2021.

[13] Yannick Schnider, Stanislaw Wozniak, Mathias Gehrig, Jules Lecomte, Axel von Arnim, Luca Benini, Davide Scaramuzza, and Angeliki Pantazi. Neuromorphic optical flow and real-time implementation with event cameras, 2023.

[14] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI'81)*, pages 674–679, Vancouver, Canada, August 1981. Morgan Kaufmann.

[15] Alex Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Ev-flownet: Self-supervised optical flow estimation for event-based cameras. In *Robotics: Science and Systems XIV*, RSS2018. Robotics: Science and Systems Foundation, June 2018.

[16] Mathias Gehrig, Mario Millhäusler, Daniel Gehrig, and Davide Scaramuzza. E-raft: Dense optical flow from event cameras, 2021.

[17] SynSense. SPECK: Event-driven neuromorphic SoC. `https://www.synsense.ai/products/speck-2/`, 2024. Accessed: Jul. 27, 2025.

[18] Yilun Wu, Federico Paredes-Vallés, and Guido C. H. E. de Croon. Lightweight event-based optical flow estimation via iterative deblurring. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14708–14715, 2024.

[19] Federico Paredes-Vallés, Kirk Y. W. Scheper, Christophe De Wagter, and Guido C. H. E. de Croon. Taming contrast maximization for learning sequential, low-latency, event-based optical flow, 2023.

[20] Jesse Hagenaars, Federico Paredes-Vallés, and Guido de Croon. Self-supervised learning of event-based optical flow with spiking neural networks, 2021.

[21] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised event-based learning of optical flow, depth, and egomotion, 2018.

[22] Mathias Gehrig, Mario Millhäusler, Daniel Gehrig, and Davide Scaramuzza. E-raft: Dense optical flow from event cameras, 2021.

[23] F. Paredes-Vallés and G. C. H. E. de Croon. Back to event basics: Self-supervised learning of image reconstruction for event cameras via photometric constancy, 2021.

[24] Antonio Vitale, Alpha Renner, Celine Nauer, Davide Scaramuzza, and Yulia Sandamirskaya. Event-driven vision and control for uavs on a neuromorphic chip, 2021.

[25] Intel Corporation. Loihi 2: A new generation of neuromorphic computing, 2025. Online; accessed 2025-09-04.

[26] D. H. Ballard. Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, January 1981.

[27] Intel Laboratories. Taking neuromorphic computing to the next level with loihi 2 technology brief, 2025. Accessed: Aug. 14, 2025.

[28] SynSense. SPECK™ Overview — Sinabs Documentation (version 3.0.2), 2025. Accessed: Aug. 14, 2025.

[29] Adam D. Hines, Michael Milford, and Tobias Fischer. A compact neuromorphic system for ultra–energy-efficient, on-device robot localization. *Science Robotics*, 10(103), June 2025.

[30] Caterina Caccavella, Federico Paredes-Vallés, Marco Cannici, and Lyes Khacef. Low-power event-based face detection with asynchronous neuromorphic hardware. *arXiv preprint arXiv:2312.14261*, December 2023.

[31] Guillermo Gallego, Henri Rebecq, and Davide Scaramuzza. A unifying contrast maximization framework for event cameras, with applications to motion, depth, and optical flow estimation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, page 3867–3876. IEEE, June 2018.

[32] Guillermo Gallego, Mathias Gehrig, and Davide Scaramuzza. Focus is all you need: Loss functions for event-based vision. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 12272–12281. IEEE, June 2019.

[33] Leo Feng, Frederick Tung, Mohamed Osama Ahmed, Yoshua Bengio, and Hossein Hajimirsadeghi. Were rnns all we needed?, 2024.

[34] SynSense. Speck2e cnn layers, 2025. Samna Documentation, ver. 0.47.1. Accessed: Sep. 4, 2025.

[35] Jason K Eshraghian, Max Ward, Emre Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D Lu. Training spiking neural networks using lessons from deep learning. *Proceedings of the IEEE*, 111(9), 2023.

[36] Sinabs. Neuron models, 2024. Accessed: Jul. 16, 2025.

[37] SynSense AG. Samna 0.47.1 documentation — SpikeCollectionNode, 2025. Accessed: Aug. 22, 2025.

[38] G.C.H.E. de Croon, J.J.G. Dupeyroux, C. De Wagter, et al. Accommodating unobservability to control flight attitude with optic flow. *Nature*, 610:485–490, 2022.

[39] Lingtong Kong, Chunhua Shen, and Jie Yang. Fastflownet: A lightweight network for fast optical flow estimation, 2021.

[40] Zhiyong Zhang, Aniket Gupta, Huaizu Jiang, and Hanumant Singh. Neuflow v2: Push high-efficiency optical flow to the limit, 2025.