UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

CResMOT: Continuous RGB–Event Stateful Memory for Multi-Object Tracking

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE

By Evan McCulley

Norman, Oklahoma

2026

CResMOT: Continuous RGB–Event Stateful Memory for Multi-Object Tracking

A THESIS APPROVED FOR THE

SCHOOL OF AEROSPACE AND MECHANICAL ENGINEERING

BY THE COMMITTEE CONSISTING OF

Dr. Golnaz Habibi, Chair

Dr. Shuozhi Xu

Dr. Iman Ghamarian

© Copyright by Evan McCulley 2026

All Rights Reserved.

Acknowledgments

I would like to thank Dr. Golnaz Habibi for welcoming me into her lab and for supervising this work.

Training of the EVA event encoder was performed on H100 GPUs at the OU Supercomputing Center for Education & Research (OSCER), and I am grateful to OSCER for providing the compute resources that made this part of the thesis possible.

Finally, I would like to thank Daniel Perez, whose friendship over the past two years first got me into computer vision and robotics, and without whom I would not have ended up here.

Contents

List of Figures

List of Tables

Abstract

Recent advances in machine learning have made RGB-camera-only approaches increasingly viable for multi-object tracking. In the frame-based RGB setting considered here, however, detections arrive per frame at 20 Hz to 30 Hz and tracks are carried between frames by a motion model, so a track receives no new evidence between observations. Additional temporally resolved sensing is therefore needed if the tracker is to incorporate new measurements, rather than prediction alone, during those intervals. The usual remedy is a second sensor, but the established alternatives are themselves sampled. LiDAR, long-wave infrared, and radar all report at discrete instants tens of milliseconds apart. Continuous-time formulations model the motion between those instants, but they interpolate rather than observe. An event camera differs in kind rather than in rate. Each pixel reports independently when its log-luminance crosses a threshold, giving microsecond-timestamped output four to five orders of magnitude finer than the frame interval. It therefore supplies measurement *during* the interval, not only at its endpoints. This thesis investigates *CResMOT (Continuous RGB–Event Stateful Memory for Multi-Object Tracking)*, whose defining requirement is that local event-derived state persist across RGB intervals and condition persistent per-object memory rather than being rebuilt as a frame-rate event image. A frozen YOLOX-m detector supplies RGB detections, a recurrent EVA-derived encoder supplies event features, and ROI-local fusion forms the observation. Each track carries a matrix-valued latent state propagated between observations by a Neural Ordinary Differential Equation and corrected at matched observations by an RWKV-7-style structured jump. The study establishes RGB and Speed-Invariant Frame ByteTrack baselines

on DSEC-MOT, then evaluates detector-matched implementations of the event pathway. On a held-out sequence, an adapter acting on motion and covariance before association improves HOTA by 1.757, AssA by 4.421, and IDF1 by 2.574 over matched RGB ByteTrack. Memory added after candidate formation does not improve on it, and ground-truth oracles at that point are similarly limited. This locates the depth at which event state must enter. The complete architecture remains to be trained and evaluated as one system.

Chapter 1

Introduction

1.1 Motivation

Multi-object tracking (MOT) in automotive perception is the task of simultaneously detecting and maintaining persistent identity for every relevant traffic participant across a continuous sensor stream. The problem is foundational to autonomous driving, advanced driver assistance, and scene understanding. Recent advances in machine learning have made RGB-camera-only approaches increasingly viable for automotive multi-object tracking. In the frame-based RGB setting considered here, detections are produced per frame at 20 Hz to 30 Hz, and tracks are carried between those frames by a motion model rather than by measurement. Two limitations of that framing are particularly acute in automotive settings, and they are structurally independent: one is sensor-level, the other algorithmic.

The sensor-level limitation is that a frame-based camera delivers motion information only at multiples of the inter-frame interval $\Delta t_{\mathrm{RGB}} = 1/f_{\mathrm{RGB}}$. At 20 Hz this is 50 ms, over which a vehicle travelling at $50\,\mathrm{km\,h^{-1}}$ covers roughly 0.7 m unobserved. Separately, relative motion during each exposure smears the projected object, producing motion-blurred bounding boxes that degrade detector confidence. Automotive illumination conditions amplify the problem: rapid transitions between tunnel exits, oncoming headlights, and sun-glare exceed the 60 dB

to 70 dB dynamic range of a conventional CMOS sensor [1], and the latency of auto-exposure control leaves successive frames under- or over-exposed. Neither blur nor exposure latency is addressable by downstream algorithmic improvement; the information is not captured at the sensor.

The algorithmic limitation is orthogonal: classical MOT propagates track state between frames with a Kalman filter operating on a constant-velocity or constant-acceleration model [2], [3], [4]. Between observations, the track's predicted location evolves by linear extrapolation alone and receives no new evidence; when a high-rate auxiliary sensor is available, its information is discarded until the next frame boundary. The sensor's native temporal granularity does not propagate into the tracker's internal state.

The established remedy for the sensor-level limitation is a second modality, but the established alternatives share the property that matters here. Automotive LiDAR records sweeps at 10 Hz to 20 Hz, and is not necessarily the slower sensor: KITTI [5] and Waymo [6] sample at 10 Hz, while nuScenes [7] samples at 20 Hz against 12 Hz cameras. Long-wave infrared imagers, the usual answer to the illumination failures above, are themselves frame-based, and automotive thermal tracking is reported at 20 Hz [8]. Frequency-modulated continuous-wave radar transmits chirps grouped into frames separated by an inter-frame processing idle [9], giving frame periods tens of milliseconds long. Each of these delivers measurements at discrete instants tens of milliseconds apart, and continuous-time trajectory formulations, which fit a spline or interpolate between samples, describe the intervening motion rather than observe it. Adding a second sampled modality increases what is measured at the sampling instants; it does not convert the interval between them into observation, which is where the algorithmic limitation lives.

Event cameras address both limitations simultaneously, but at a structural level that only a tracker designed around their output can exploit. A Dynamic Vision Sensor (DVS) does not capture frames; each pixel independently monitors its local log-luminance and emits a microsecond-timestamped event whenever a threshold change is crossed [1]. The output

is a sparse, asynchronous event stream with three properties that distinguish it from any frame-based alternative: continuous-time output at microsecond resolution without a global frame clock; motion-proportional firing, as events arise from log-brightness change and fire densely in regions and at times where motion is occurring; and spatial locality, as each event carries the exact pixel coordinates at which it was emitted. Together, these properties define a continuous-time motion signal co-located with each object of interest, at a temporal resolution four to five orders of magnitude finer than the RGB frame interval, and a dynamic range of 120 dB [1].

An RGB–event multi-object tracker able to consume the event signal at its native resolution — without a preprocessing step that discretises events onto a fixed time grid — could obtain per-track motion information that is unavailable to an RGB-only pipeline or to an event–RGB pipeline whose event pathway has been collapsed to a frame-rate tensor. This thesis investigates such a tracker.

1.2 Problem Statement

Despite the availability of event cameras in automotive-scale benchmarks such as DSEC [10] and its tracking extensions [11], most RGB–event tracking pipelines reviewed in Chapter 2 first accumulate events into a frame-like tensor. Examples include event histograms, time surfaces, Speed-Invariant Frames [12], and the exposure-window event images used by FEMOTR [13]. The resulting representation is then consumed by a detector or tracker at the RGB rate. Within-window order is unavailable after this reduction. At the tracker level, a constant-velocity Kalman state likewise receives no direct event evidence between RGB observations.

This thesis addresses the following research question:

> *Can an RGB–event multi-object tracker consume the RGB and event streams at their native temporal rates — RGB at the frame rate, events asynchronously and without frame-rate discretisation — maintain a continuously evolving per-track*

latent state between observations, and deliver measurable identity-preservation improvements over a detector-matched ByteTrack baseline on DSEC-MOT?

The investigation is organised around three subordinate questions. First, on architecture: what is the form of a per-track latent state that evolves continuously between observations under event-stream conditioning, and how does it compose with the discrete-time components of a classical MOT pipeline (detection, association, lifecycle)? Second, on the data surface: which DSEC annotation artefact supports a detector-matched comparison, and what limitations follow from its construction? Third, on empirical evaluation: does the proposed architecture produce measurable tracking improvements against a well-characterised classical baseline, and do matched ablations isolate the contribution of the event pathway?

1.3 Contributions

This thesis combines existing components with new ones, and the two are separated here so that no borrowed part is presented as new. The novel contributions are stated first, followed by the integration and supporting work that made them testable.

Novel contributions

1. **A per-track continuous-time formulation for RGB–event tracking.** Chapters 4 and 5 formulate a track-local matrix state that evolves continuously between observations under event-stream conditioning and receives an RWKV-7-style structured correction at matched observations. The formulation unifies two operator families that the literature keeps apart: continuous-time latent dynamics, which give an exact elapsed-time update between irregularly spaced observations, and associative-memory writes, whose three RWKV-7 modes map onto the operations a per-object memory requires at an observation, namely forgetting contradicted content, revising an existing key–value association, and depositing new evidence. Applying that pair at *object* rather than scene granularity is

what makes identity a property of an explicit state rather than of a repeated matching decision, and Chapter 2 positions it against the closest prior work along six design axes.

2. **A persistent causal event-state requirement.** The event path must update local state at physical event times or causal event-derived emission times and carry that state across RGB intervals. RGB-clocked querying, association, and correction remain compatible with the requirement; an independently rebuilt static event bin does not satisfy it.

   This requirement is what gives the rest of the thesis its meaning, for three reasons. First, it converts the motivating claim of §1.2 into something testable. That argument is that a track receives no new evidence between observations and that high-rate sensor information is discarded until the next frame boundary. A system that rebuilds an event tensor at each frame changes the features the tracker sees without addressing that claim at all, and would satisfy a looser reading of "RGB–event tracking" while leaving the original limitation intact. Second, it is the property that separates this design from frame-synchronous fusion. Every RGB–event tracker reviewed in Chapter 2 pools events into an exposure-window representation before object memory sees them, after which their internal order is unavailable; without the requirement, the proposed system reduces to those methods with different features. Third, it makes the empirical result of the following item attributable. A finding about *where* recurrent state must enter a tracker is only interpretable if that state genuinely persisted across intervals, and the evaluations record zero event-only lifecycle violations, with RGB retaining sole authority over track creation, removal, and identity.

3. **A measured result on where recurrent state must enter a tracker.** Complete fitting-set experiments show that intervention after candidate formation is capacity-limited: non-deployable ground-truth oracles on already formed candidates gain less than 0.9 HOTA, and a learned persistent-slot scorer gains 0.031. An adapter acting

before association instead improves a held-out sequence by 1.757 HOTA, 4.421 AssA, and 2.574 IDF1. This establishes a property of tracking-by-detection that the RGB–event literature has not reported: the usable capacity of an event pathway is set by the depth at which it acts, and that capacity is largely spent before the association cost is reached. The result is measured on a frozen detector with matched controls, so it transfers to any system sharing that pipeline shape.

Integration of existing components

The evaluated systems compose a frozen YOLOX-m detector, a frozen EVA-derived recurrent event encoder, ROI-local fusion, and a ByteTrack shell into one detector-matched pipeline in which every arm shares a detection cache, an evaluator, and a sequence boundary. None of these components is new. What the integration provides is a substrate on which event-pathway variants become separable: every arm differs in exactly one place, so a measured difference is attributable to that place. The causal replay that advances event state between RGB observations, while RGB retains lifecycle authority, is what allows the requirement above to be enforced and audited rather than assumed.

Supporting work

Chapter 6 constructs a 53-sequence DSEC-derived annotation surface by combining the reviewed detection boxes of the SFNet release with available track identifiers, retaining per-row provenance and documenting unresolved classes and alignment limits. No public release combines manually reviewed boxes with track identity at this scale, and the surface makes per-class and per-sequence analysis possible where the canonical split does not. Chapter 7 establishes RGB-plus-ByteTrack and SIF-plus-ByteTrack baselines and the protocol under which the systems above are compared. These are reported as enabling work rather than as claims in their own right.

1.4 Scope and Non-Contributions

The EVA-derived encoder, YOLOX-m detector, ByteTrack shell, Neural ODE, cross-attention, and RWKV recurrence are not claimed as new individually. The thesis does not present a state-of-the-art comparison across all RGB–event benchmarks. The two-sequence adapter result is retrospective and single-seed, the calibrated-substrate experiments use a separate fitting substrate, and the frozen-EVA diagnostic does not attribute any gain to event order. The full method specified in Chapters 4 and 5 remains unevaluated as one system.

1.5 Thesis Outline

Chapter 2 surveys event representations, classical MOT, RGB–event tracking, and continuous-time sequence models. Chapter 3 establishes the technical background: event cameras, DSEC, SFNet, YOLOX, ByteTrack, recurrent sequence models, Neural ODEs, and MOT metrics. Chapters 4 and 5 specify the observation pipeline and the per-track recurrent tracker. Chapter 6 describes the 53-sequence DSEC-derived annotation surface. Chapter 7 reports the established baselines and the bounded implementation evidence available for CResMOT. Chapter 8 states the limitations and the custom-EVA direction left to future work. Chapter 9 concludes.

Chapter 2

Related Work

The literature reviewed here is organised around the difference between preserving event order and reducing events to a frame-rate representation. Section 2.1 states the RGB–event signal contrast. Section 2.2 reviews event representations and recurrent encoders. Sections 2.3–2.4 cover sparse event processing and classical tracking-by-detection. Sections 2.5–2.6 review event-based single- and multi-object tracking, including the FEMOT benchmark. Sections 2.7–2.8 review Neural ODEs and structured recurrent models. Section 2.9 identifies the works that most directly motivate the proposed design, and Section 2.10 summarises its intended position.

2.1 Asynchronous Events and Frame-Sampled RGB

The RGB frame and the event stream are not two noisier-or-cleaner versions of the same signal. They are the outputs of two different sensing principles that differ in their fundamental structure, and the architectural implications propagate all the way through the tracker.

An RGB frame is a time integral: each pixel reports the accumulated photon count over a globally shared exposure window, producing a dense $H \times W$ tensor at a fixed rate f_{RGB} (20 Hz on DSEC [10]). Temporal resolution is bounded by the inter-frame interval $\Delta t_{\mathrm{RGB}} = 1/f_{\mathrm{RGB}}$, below which no motion information exists in the signal. Spatial and temporal resolutions are

entangled by the global shutter: all pixels integrate over the same window, and no pixel can report information at finer temporal granularity than any other.

An event stream is a threshold-crossing process: each pixel independently emits a timestamped event whenever its log-luminance change exceeds a per-pixel threshold (Equation 3.1), yielding a sparse 4-tuple sequence (x_i, y_i, t_i, p_i) with t_i at microsecond precision. Temporal resolution is bounded only by the sensor's minimum event inter-arrival time — roughly 1 µs on the Prophesee Gen4 sensors used in DSEC — four orders of magnitude below the RGB inter-frame interval. Spatial and temporal resolutions are decoupled: a stationary region of the scene emits no events regardless of exposure, while a fast-moving region emits a dense temporal burst. The sensor spends bandwidth on information rather than on redundantly sampling static regions.

Three consequences of this contrast shape the CResMOT architecture and are referenced throughout the rest of this chapter. First, any operator that treats an event stream as if it were a frame — by accumulating events into a tensor on a shared time grid — incurs an information loss that is structural and cannot be recovered downstream, no matter how sophisticated the operator. The ceiling is set at preprocessing time. Second, an operator natively formulated on continuous time with a learned latent state is, in principle, able to receive events at their native timestamps and maintain a state that evolves at scene-appropriate rates without a preprocessing discretisation. Neural ODEs (§2.7) and linear recurrent architectures (§2.8) are the two operator families that satisfy this property. Third, RGB and events are complementary rather than substitutable: RGB provides dense appearance evidence at frame boundaries; events provide sparse motion evidence between them. The fusion question is where and how to couple them such that the event asynchrony is preserved through the coupling and not collapsed to the RGB rate.

2.2 Event Representations, Ordered by Asynchrony Preserved

Every event-processing architecture commits to a representation that reduces the raw stream to a form its operators can consume (Chapter 3, §3.1.1). Raw asynchronous lists retain the input directly. Recurrent latent encodings retain an ordered causal history, although the finite state remains a learned compression rather than a lossless copy of the stream. Voxel grids, the Event Spike Tensor [14], memory surfaces [15], and mixed-density stacks [16] preserve temporal order at bin granularity. Time surfaces and hierarchical time surfaces [17] retain per-pixel recency. Event frames, histograms, and Speed-Invariant Frames [12] remove within-window order.

The practical consequence for multi-object tracking is a preprocessing-time limit. Once events have been pooled into an exposure-window image, their internal order is no longer available to the tracker. CResMOT instead investigates a recurrent event pathway whose state can condition the tracker without first constructing an event image.

EVA and RWKV-7. EVA [18] applies recurrent linear attention to event streams and produces task-oriented event features from patch-local state. The evaluated checkpoint used in this thesis is an EVA-derived RWKV-7 configuration. RWKV-7 [19] adds a delta-rule rewrite to the matrix-state recurrence, alongside decay and outer-product writing. That structure motivates the observation jump in Chapter 5. The two states remain distinct: the encoder state is spatially indexed, whereas the tracker state belongs to an object track.

Neural Events. Neural Events [20] divides the sensor plane into patches and updates a local RWKV-7 state for every incoming event. A learned codebook converts this state into a discrete code, and a neural event is emitted only when the code changes. A time-surface decoder preserves local event history, while rate-alignment and latent-straightening losses limit unnecessary code changes. The paper evaluates detection and classification, not tracking.

The future custom-EVA design shares the use of local recurrence and change-triggered communication, but not the discrete codebook. Its selected state changes would be passed into persistent, RGB-grounded track memory for motion and association. Neural Events therefore supports the case for selective asynchronous propagation; it does not provide the scene-to-track mechanism proposed for CResMOT.

2.3 Sparse-Native Event Processing

One line of event-vision research has pursued the raw-stream ideal at inference time through sparse-native operators. These methods establish empirically that asynchronous event processing is tractable at automotive scale, and they define the dominant compute-reduction baselines against which any asynchronous architecture is measured.

2.3.1 AEGNN and the Graph-Based Asynchronous Paradigm

Asynchronous Event-based Graph Neural Networks (AEGNN) [21] treat events as nodes in a spatiotemporally evolving graph $G = (V, E)$ whose vertices carry per-event coordinates and polarity. A graph convolution of the form

$$h_v^{(l+1)} = \phi\left(h_v^{(l)}, \underset{u \in \mathcal{N}(v)}{\text{aggr}} \psi(h_u^{(l)}, h_v^{(l)}, e_{uv})\right) \tag{2.1}$$

computes updated node features from a learnable edge function ψ over local neighbours $\mathcal{N}(v)$. The key architectural property for asynchrony is that when a new event is inserted into the graph, only the activations in the local receptive field of the new node need to be recomputed; the rest of the graph's activations remain valid. The per-event update cost is therefore $\mathcal{O}(|\mathcal{N}|^L)$ for a depth-L network rather than the $\mathcal{O}(|V|\,|\mathcal{N}|^L)$ full-graph recomputation, yielding a large constant-factor reduction over dense methods at the cost of a practical cap on network depth.

2.3.2 DAGR: Asynchronous Graph Neural Networks Coupled with RGB

DAGR [22] is the most mature instantiation of the asynchronous-graph paradigm and one of the principal inspirations for CResMOT (§2.9). The system consists of two branches. An image branch — a conventional CNN detector — runs on the low-rate RGB stream and produces bounding-box detections at 20 Hz intervals. An event branch — an asynchronous graph neural network with the AEGNN recursive-update property — runs on the event stream between frame boundaries and is trained to produce detection updates that refine the image-branch predictions at event arrival times. Both branches predict detections in the same format; the event branch's training signal is a displacement correction over the image branch's output. The architecture is thus described by its authors as a network that "learns to track, detect and forget objects from the previous view" between RGB frames [22].

At deployment, DAGR switches from graph-level forward passes to an asynchronous mode in which events arrive one at a time and the recursive update rules at each GNN layer enforce that the output of the network for the incoming event equals the output of the augmented graph containing all previous events plus the new one. This yields compute reductions of approximately 4,800× relative to a dense forward pass over the equivalent temporal window, with detection accuracy on par with a dense image-plus-event pipeline on DSEC-Det. The authors quantify the latency-versus-bandwidth tradeoff as equivalent to "a 5,000-fps camera with the bandwidth of a 45-fps camera" [22], a two-orders-of-magnitude reaction-time improvement over an RGB-only pipeline at the same data rate.

Two properties of DAGR's position are directly relevant to CResMOT. First, DAGR demonstrates that commitment to asynchronous event consumption at automotive scale is empirically tractable — the dataset (DSEC-Det), the compute budget, and the latency regime are all shared with the present work. Second, DAGR is an object detector with between-frame refinement: it does not maintain persistent per-object memory across extended time horizons, does not perform identity association across frames, and is not evaluated at the

multi-object-tracking scale considered here. The architecture preserves asynchrony through the network but stops short of propagating it into a tracker. CResMOT pursues the same information-preservation commitment through a different operator family — recurrent linear attention with explicit per-track state — that is architecturally organised around persistent identity rather than per-frame detection refinement.

2.4 Classical Multi-Object Tracking

Online multi-object tracking has long been dominated by tracking-by-detection pipelines structured as a composition of loosely coupled modules: a detector, a motion model, an association solver, and a lifecycle manager. The modularity is structural: each component can be swapped independently, and the design space of classical MOT is largely the cross-product of choices at each of these four axes.

SORT [2] established the canonical instantiation: a constant-velocity Kalman motion model, Hungarian assignment [23] at $\mathcal{O}(n^3)$ over an IoU cost matrix, and a simple lifecycle machine. DeepSORT [3] added a CNN-extracted appearance embedding to mitigate identity switches under occlusion. ByteTrack [4] (§3.5) currently defines the classical reference: it retains a geometry-only association cost but adds a two-stage matching pipeline that rescues low-confidence detections corresponding to partial occlusions. BoT-SORT [24] combines ByteTrack's detection rescue with appearance features and adds camera-motion compensation; StrongSORT [25] revisits DeepSORT's re-identification embedding with contemporary backbones.

Relevance to CResMOT. The modular structure of classical MOT means that CResMOT's design does not require the simultaneous replacement of every component. The choice of Kalman filter as a motion model is a specific decision about how to predict between frames, not a structural commitment that would prevent mixing a learned motion model with a geometric association cost or a standard lifecycle machine. The key observation is

instead about what a Kalman filter does with the event signal: a frame-rate constant-velocity Kalman update consumes the event stream only through its effect on the RGB-rate detection output, and has no mechanism to incorporate the microsecond-scale motion evidence available between frames. An event camera's temporal resolution is structurally unused by a frame-rate Kalman filter. CResMOT's Neural ODE motion model is designed to close this gap while preserving the modular interface to association and lifecycle: it is a drop-in replacement for the motion block of a tracking-by-detection pipeline, not a monolithic reformulation of the tracker.

2.5 Event-Based Single-Object Tracking

Event-based single-object tracking (SOT) is an active research area whose methods merit a per-paper event-representation audit. The single-target assumption removes the birth/death lifecycle management, the persistent-identity problem across occlusions, and the multi-object association problem that together dominate MOT complexity. The methods are reviewed here as context for the event-representation taxonomy and for the operator families that have appeared on the RGB–event tracking task, not as direct baselines.

The dominant design pattern in RGB–event SOT is a one-stream transformer backbone shared across modalities. CEUTrack [26] introduced this template, concatenating RGB patch tokens with event-voxel tokens through a unified ViT backbone that performs feature extraction, cross-modal fusion, matching, and interactive learning in a single pass, trained on the COESOT dataset it also released. Zhu *et al.* [27] build on the CEUTrack baseline with plug-and-play training augmentations — a masked-modality strategy that randomly zeroes tokens from one modality to force cross-modal interaction, plus an orthogonal high-rank loss on the attention matrices to regularise the interaction pattern — showing consistent gains on COESOT and VisEvent benchmarks. SDSTrack [28] generalises the one-stream ViT pattern to a modality-agnostic tracker covering RGB+Depth, RGB+Thermal, and RGB+Event,

using lightweight symmetric adapters for parameter-efficient fine-tuning and a complementary masked-patch distillation objective that prevents the pre-trained backbone from collapsing onto the RGB modality. All three consume events as voxel-grid patches.

A complementary line of work replaces the quadratic-cost ViT backbone with a linear-complexity alternative. Mamba-FETrack [29] builds an RGB–event tracker on Vision Mamba (Vim) blocks: separate Vim backbones extract modality-specific features and a FusionMamba block performs cross-modal interaction, achieving $\mathcal{O}(nd)$-per-step cost in place of the $\mathcal{O}(n^2d)$ self-attention of the ViT-based trackers above. Mamba-FETrack V2 [30] extends this with a Prompt Generator that draws modality-specific learnable prompt vectors from a shared prompt pool; these prompts steer a unified FEMamba backbone through prompt-guided cross-modal feature extraction. Both variants consume events as voxel sequences. AMT-Track [31], introduced alongside the FELT long-term frame-event benchmark (1,044 videos, 1.9M RGB+event pairs), targets the orthogonal axis of long-range memory: a modern Hopfield retrieval layer [32] is inserted into a one-stream ViT tracker to aggregate multi-scale RGB and event-voxel tokens across extended time horizons, providing an explicit associative-memory mechanism for target re-identification. The event pathway remains voxel-based.

TAPFormer [33]. TAPFormer performs arbitrary-point tracking with transient asynchronous frame–event fusion. It is one of the few SOT methods operating at the raw-stream tier, and it targets keypoint-level point tracking rather than bounding-box object tracking.

The pattern across event-based SOT is clear: voxel-based event representations dominate, backbones are chosen from the ViT / Mamba / Hopfield-augmented transformer menu, and the few methods that preserve asynchrony target keypoint or feature-level tracking rather than object-level association. The methods reviewed in this subsection do not address persistent multi-object identity across a scene.

2.6 Event-Based Multi-Object Tracking

The published event-based MOT literature remains small, but now includes both DSEC-based work and a dedicated RGB–event benchmark.

SpikeMOT [11]. SpikeMOT is the event-only MOT method that introduces the 12-sequence DSEC-MOT annotation split used throughout this thesis. Events are processed through a spiking neural network operating on spike-train features in a Siamese matching architecture, with a parallel object detector providing frame-rate spatial updates. The spike-train representation preserves more temporal detail than a single event image but operates on a discretised time grid, and the method does not fuse RGB evidence. Its published results provide an external reference for the DSEC experiments.

AEMOT [34]. AEMOT uses Asynchronous Event Blobs (AEB) as per-object state objects propagated by a classical recursive estimator. Its event representation is the raw asynchronous stream, consumed by hand-designed blob trackers. AEMOT is evaluated on an insect-tracking benchmark and does not use RGB input — the method is architecturally precluded from the RGB–event setting.

FEMOT and FEMOTR [13]. FEMOT is a dedicated RGB–event MOT dataset containing 100 sequences, approximately 200,000 frames, 14,580 trajectories, and 420,000 bounding boxes. Its aligned RGB and event recordings cover people and vehicles, indoor and outdoor scenes, day and night, static and moving cameras, and fourteen challenge attributes. This breadth makes FEMOT the most direct external dataset for future evaluation of CResMOT.

The accompanying FEMOTR tracker forms a three-channel event image over each RGB exposure interval. RGB and event features are extracted at the corresponding frame time, combined through frequency-domain fusion, and decoded by a transformer using detection and propagated track queries. Its long-term track memory is updated from frame outputs by exponential averaging and attention.

FEMOTR is therefore a frame-synchronous RGB–event MOT method: the event stream is represented separately for each exposure interval and the tracking memory advances at frame times. The future custom-EVA design instead asks whether recurrent event state can update persistent object memory between RGB frames. The distinction is the representation and update schedule, rather than the choice of transformer or recurrent backbone. The FEMOT dataset is reserved for future external evaluation in Section 8.5.

The withdrawn ICLR 2024 RGB–Event MOT benchmark [35]. A single prior attempt at RGB–event multi-object tracking has appeared in the literature. An ICLR 2024 submission proposed a benchmark dataset of 12 RGB+event sequences (4K RGB, 1280×800 Celex5 events, static viewpoints, four classes) together with a baseline pipeline. The baseline used a three-bin event-frame accumulation passed through a conventional RGB backbone, fused with the RGB feature map through either unweighted averaging ("AVG") or a learnable mask multiplication ("MASK"), and then handed to an off-the-shelf RGB tracker. The submission reported a best MOTA of 43.6% with MASK fusion versus 28.8% for the best RGB-only tracker on the same dataset, and acknowledged in its own Data Format section that the voxelisation "might intrinsically reduce the high temporal resolution inherent to event data" [35].

The submission was withdrawn prior to publication. It is noted because its voxelised event frames and off-the-shelf tracker illustrate an earlier frame-synchronous design. FEMOT now provides a released external RGB–event benchmark, while the annotation effort in Chapter 6 remains useful for comparison with the existing DSEC experiments.

Summary. Existing methods cover event-only tracking, asynchronous specialised trackers, and frame-synchronous RGB–event MOT. CResMOT studies a different combination: recurrent event input, continuous per-track propagation, and a structured observation update. Whether this combination improves tracking remains an experimental question.

2.7 Neural Ordinary Differential Equations for Tracking

The Neural ODE [36] (Chapter 3, Equation 3.8) parameterises the evolution of a latent state $\mathbf{z}(t) \in \mathbb{R}^d$ by a learned vector field f_θ. The construction has four properties that a tracker for asynchronous sensors can productively exploit, independently of any specific prior architecture.

Time-translation invariance. The vector field f_θ depends on $\mathbf{z}$ and optionally on clock time t, but not on any fixed discretisation step. Training and inference can use different step schedules; the learned dynamics transfer without resampling.

Arbitrary query times. The integrator $\mathbf{z}(t_1) = \mathbf{z}(t_0) + \int_{t_0}^{t_1} f_\theta(\mathbf{z}, t)\, dt$ produces $\mathbf{z}$ at any t_1 without rounding to a grid. For MOT, this means that a track state can be queried at the timestamp of an incoming observation. The difficulty in the reviewed RGB–event trackers is not the Kalman filter itself, which can accommodate irregular observations, but the absence of event-derived observations between RGB frames.

Adaptive numerical integration. An adaptive solver selects its steps according to the requested tolerance and the local behaviour of the vector field. Its cost therefore depends on the interval, curvature, and stiffness of the learned dynamics rather than on a fixed frame-rate schedule. This flexibility is useful for event data, although it does not by itself guarantee low computational cost.

Continuous interpolation. A Neural ODE parameterises a continuous trajectory through observation points. This differs from a discrete recurrence whose update schedule is fixed in advance. The continuous formulation avoids choosing a universal unit step, but it does not guarantee that the learned state preserves all information in the observations.

These properties are general; they do not depend on any specific prior tracking architecture. The ODE–RNN family applies them to irregularly observed time series: ODE-RNN and Latent-ODE [37] combine a Neural ODE for inter-observation propagation with a GRU-style jump at each observation, producing a hybrid continuous–discrete recurrent architecture; GRU-ODE-Bayes [38] develops the same hybrid in a Bayesian filtering formulation with

explicit uncertainty propagation; Neural Controlled Differential Equations [39] generalise the Neural ODE to a driven form $d\mathbf{z} = f_\theta(\mathbf{z})\,dt + g_\theta(\mathbf{z})\,dX(t)$ where $X(t)$ is a continuous control signal. ContiFormer [40], reviewed in detail as an inspiration in §2.9, extends the Transformer attention operator to continuous time by evolving the query, key, and value projections along Neural ODEs between observations.

The works reviewed above do not apply this combination of continuous inter-observation dynamics and structured recurrent updates at per-object granularity to RGB–event multi-object tracking. They establish the vocabulary used in Chapter 5.

2.8 Structured State-Space Models

Structured state-space models (SSMs) and linear recurrent architectures are the second family of operators that admit continuous-time latent dynamics. In contrast to Neural ODEs, which expose the continuous-time integral explicitly and rely on a numerical solver at inference, SSMs discretise a linear ODE into a time-stepped recurrence that is closed-form computable and admits efficient parallel training. The theoretical expressivity of selective SSMs has been analysed within a Rough Path Theory framework by Muca Cirone *et al.* [41], who characterise the hidden state of a selective SSM as a low-dimensional projection of the *signature* of the input path and identify the gating (selectivity) mechanism as the architectural element responsible for multiplicative input–state interactions; this framework subsumes Mamba, GLA, Hawk/Griffin, and related variants and provides the theoretical grounding for the non-diagonal state transitions that appear in RWKV-7.

S4 [42] and Mamba [43] instantiate the pattern with the core SSM recurrence

$$\mathbf{x}_{k+1} = \bar{A}\,\mathbf{x}_k + \bar{B}\,u_k, \qquad y_k = C\,\mathbf{x}_k, \tag{2.2}$$

where $\bar{A}, \bar{B}$ are discretisations of an underlying continuous-time linear ODE. Time complexity is $\mathcal{O}(nd)$ and memory $\mathcal{O}(d)$ per step, compared to $\mathcal{O}(n^2 d)$ time and $\mathcal{O}(n^2)$ memory for softmax

attention [44]. The limits of diagonal-transition SSMs (Mamba included) on state-tracking problems — and the corresponding TC^0 upper bound on transformer expressivity under log-precision assumptions — are established by Merrill and Sabharwal [45] and Merrill *et al.* [46]; these results together motivate the non-diagonal, input-dependent state transitions of RWKV-7.

The RWKV family [19], [47], [48] pursues the same linear-time recurrent goal through a different derivation, from a reformulation of attention as a weighted moving average over a matrix-valued state. The three published generations relevant to this thesis are RWKV-4 [47] (channel-wise time decay over scalar state), RWKV-5/6 [48] (matrix-valued state with data-dependent decay), and RWKV-7 [19] (matrix-valued state with a bilinear delta-rule rewrite that places the architecture outside the TC^0 expressivity class that bounds log-precision softmax-attention transformers [45]). The implementation detail relevant to CResMOT's per-track memory update is the specific form of the RWKV-7 matrix-state write (Equation 3.7); its architectural role in the tracker is developed in Chapter 5.

The operational relationship between Neural ODEs and RWKV-lineage operators is complementary: the Neural ODE specifies what happens between observations (continuous-time state evolution under a learned vector field); the RWKV jump specifies what happens at an observation (a structured matrix-state write).

2.9 Principal Inspirations and a Cross-Domain Parallel

Four published works serve as the principal intellectual inspirations for CResMOT, each contributing a distinct piece of its design. A fifth body of work, on 4D LiDAR point-cloud tracking, is noted as a cross-domain parallel rather than a direct inspiration.

DAGR [22]. DAGR is covered in detail in §2.3. It contributes the empirical demonstration that asynchronous event consumption at automotive scale and automotive-grade latency is tractable, and that a two-branch decomposition — a CNN on frames and an asynchronous

network on events — is a viable architectural pattern for RGB–event perception.

StreamingFlow [49]. StreamingFlow addresses bird's-eye-view (BEV) occupancy forecasting under asynchronous multi-sensor input (camera, LiDAR, radar) and maintains a learned BEV latent that evolves continuously between observations under a Neural ODE, receiving discrete GRU-style updates at each observation. The task differs from MOT — occupancy forecasting predicts future scene states at the grid level, not persistent per-object identity across frames — and the latent granularity is correspondingly different: StreamingFlow maintains a single scene-level latent; CResMOT maintains one matrix-valued latent per currently tracked object. Despite this task mismatch, StreamingFlow is the closest published instantiation of an ODE–RNN hybrid at automotive scale, and supplies CResMOT with the pattern of treating the ODE–RNN as a persistent world-state latent receiving asynchronous multi-sensor observations. The divergences are (i) scene-level versus per-track granularity, (ii) forecasting versus association as training signal, (iii) GRU-style versus RWKV-7-style observation-time write.

ContiFormer [40]. ContiFormer extends the Transformer attention operator to continuous time by evolving the query, key, and value projections along Neural ODEs between observations, enabling attention evaluation at arbitrary query times:

$$\mathrm{Attn}(t) = \mathrm{softmax}\left(\frac{Q(t)K(t)^\top}{\sqrt{d_k}}\right)V(t), \tag{2.3}$$

$$\frac{dQ}{dt} = f_\theta^Q(Q,t), \qquad \frac{dK}{dt} = f_\theta^K(K,t), \qquad \frac{dV}{dt} = f_\theta^V(V,t). \tag{2.4}$$

The authors show that several specialised irregular-time-series Transformers are recovered as special cases of this continuous-time attention operator under specific choices of $f_\theta^{Q,K,V}$. Its contribution to CResMOT is the architectural template of continuous-time latent evolution of attention projections coupled to discrete observation-time updates, applied here at per-track rather than per-sequence granularity.

TimeTracker [50]. TimeTracker (Liu *et al.*, CVPR 2025) addresses event-based video frame interpolation with non-linear motion by continuous point tracking of image regions through events. Its Scene-Aware Region Segmentation (SARS) module divides the RGB frame into patches; its Continuous Trajectory-guided Motion Estimation (CTME) module tracks the continuous-time motion of each patch through the event stream; a frame refinement step produces the interpolated frame. The core methodological finding is that "object motion is continuous in space, [so] tracking local regions over continuous time enables more accurate identification of spatiotemporal feature correlations" [50]. Its contribution to CResMOT is the principle of treating a per-object local region as the unit of continuous-time tracking through events, with RGB providing the appearance-level region template and events driving the between-frame trajectory. CResMOT differs in task (MOT versus VFI), granularity (object-level versus patch-level), and state (learned matrix-valued memory versus an explicit trajectory), but shares the core inductive bias that per-object continuous-time motion is the right primitive for event-based perception.

A cross-domain parallel: 4D LiDAR point-cloud tracking. Events and LiDAR returns are structurally related signals: both produce sparse, asynchronous samples — (x, y, t, p) for events and (x, y, z, t) for LiDAR — that require tolerance to irregular sampling and persistence across sparse observations. Recent 4D LiDAR methods adopt similar operator families: Mamba4D [51] uses SSM backbones over 4D LiDAR sequences, while Xu *et al.* [52] model continuous motion for 3D point-cloud single-object tracking with a memory-bank attention pattern. This parallel motivates, but does not validate, the use of continuous dynamics and recurrent memory for sparse event data.

2.10 Positioning

Table 2.1 compares CResMOT with the closest related work along six design axes. The table is descriptive rather than a novelty claim: the proposed combination still requires the

Table 2.1: Position of CResMOT against closest prior work along six design axes.

Method	Task	Event repr. (tier)	Motion model	Per-track mem.	Obs. update	End-to-end async
ByteTrack [4]	MOT	—	Kalman (CV)	none	Kalman innov.	no
SFNet+ByteTrack [12]	MOT	SIF (image-frame)	Kalman (CV)	none	Kalman innov.	no
SpikeMOT [11]	MOT	Spike trains (SNN)	SNN features	none	Siamese match	partial
AEMOT [34]	MOT (event-only)	Raw async	Recursive est.	per-blob	Classical filter	yes
[35] (withdrawn)	MOT	Event frame (3-bin)	Off-the-shelf MOT	none	ByteTrack innov.	no
FEMOTR [13]	MOT	Event image (frame)	Track queries	vector EMA	Transformer update	no
DAGR [22]	Det.	Raw async (graph)	—	—	GNN msg.-passing	yes
Neural Events [20]	Det./cls.	Raw to code change	—	patch state	code emission	yes
StreamingFlow [49]	Forecast	Various	Neural ODE	scene BEV	GRU update	yes
TimeTracker [50]	VFI	Raw async	Continuous traj.	per-patch	Transformer attn.	yes
CResMOT (proposed)	MOT	Recurrent EVA state	Neural ODE	Matrix state	RWKV-7 jump	intended

controlled evaluation specified in Chapter 7.

Chapters 4 and 5 develop the architecture that instantiates this position.

Chapter 3

Background

This chapter establishes the technical context on which the remainder of the thesis depends. Section 3.1 introduces the event camera and the representations used to consume its output, including several representations not used by this thesis that are referenced in the literature review of Chapter 2. Section 3.2 describes the DSEC dataset and the two distinct downstream annotation efforts that this thesis builds against: DSEC-Detection (uzh-rpg) and the Spike-MOT DSEC-MOT split. Section 3.3 summarises the SFNet paper and its Speed-Invariant Frame representation, which provide the detector recipe used in this work. Section 3.4 covers YOLOX; Section 3.5 covers ByteTrack; Section 3.6 reviews attention and linear-recurrent sequence models; Section 3.7 reviews Neural ODEs; Section 3.8 defines the MOT metrics used throughout the Experiments chapter.

3.1 Event Cameras

Event cameras, also called Dynamic Vision Sensors (DVS), operate on a per-pixel asynchronous principle whose behaviour and consequences are surveyed in detail by Gallego *et al.* [1]. Each pixel at coordinates (x, y) continuously monitors its local log-luminance $L(x, y, t) = \log I(x, y, t)$ and emits an event whenever the change since its last-fired event exceeds a

per-pixel contrast threshold C:

$$\left|L(x, y, t) - L(x, y, t - \Delta t)\right| \;\geq\; C. \tag{3.1}$$

Each event is a tuple

$$e_i \;=\; (x_i,\; y_i,\; t_i,\; p_i) \;\in\; \mathbb{Z}^2 \times \mathbb{R} \times \{-1, +1\}, \tag{3.2}$$

where t_i carries microsecond-scale precision and p_i encodes the sign of the log-luminance change. Events from different pixels are emitted independently, producing a sparse asynchronous stream rather than a fixed-rate tensor.

Three consequences of this operating principle are central to downstream design. First, the temporal resolution of the event stream is roughly four orders of magnitude finer than the inter-frame interval of a 20 Hz RGB camera, which eliminates frame-rate motion aliasing [1]. Second, because each pixel operates on a log-luminance baseline rather than an integrated global exposure, the sensor's dynamic range is on the order of 120 dB, approximately three orders of magnitude larger than a typical CMOS sensor [1]. Third, events are threshold-crossing instants rather than time integrals, so fast motion produces dense event sequences rather than blurred spatial extent; the localisation of each event remains exact to the emitting pixel.

3.1.1 Event Representations

Deep learning on events requires a conversion from the raw stream to a tensor. The landscape of representations is richer than the subset used in this thesis and is surveyed here in full, because several of the representations not used by this work are nonetheless referenced in Chapter 2 when characterising prior trackers.

Event frames and histograms. A window of events is aggregated into a two-dimensional count or binary histogram, optionally per-polarity. The representation is the lossiest in common use: within-window temporal order is destroyed.

Time surfaces. Each pixel stores the timestamp of its most recent event, often with exponential decay, yielding a dense representation

$$T(x, y) \;=\; \exp\Big(-(t - t_{\text{last}}(x, y))/\tau\Big) \tag{3.3}$$

that retains per-pixel recency but requires a decay constant τ adapted to scene speed.

Hierarchical time surfaces. HATS [17] averages time surfaces over local neighbourhoods at multiple scales, producing a representation more robust to event noise than the raw time surface, at a fixed spatial cost.

Voxel grids. A window of length ΔT is divided into B equal-width temporal bins, and each event is tri-linearly interpolated in both space and time into the grid

$$V(x, y, b) \;=\; \sum_{i:t_i \in \Delta T} p_i \cdot \kappa_{xy}(x_i, y_i \mid x, y) \cdot \kappa_t\left(\frac{(B-1)(t_i - t_0)}{\Delta T} \,\middle|\, b\right), \tag{3.4}$$

where κ_{xy} and κ_t are the bilinear and linear interpolation kernels. Voxel grids preserve temporal order at bin granularity while yielding a tensor of shape (B, H, W) compatible with $\mathcal{O}(BHW)$-cost convolutional networks [1].

Event Spike Tensor (EST). EST [14] replaces the hand-chosen kernels of the voxel grid with learnable per-channel kernels applied to the four event dimensions, and can be viewed as a voxel grid whose spatiotemporal interpolation is trained jointly with the downstream task.

Memory surfaces and matrix-LSTM stacks. Memory surfaces [15] maintain per-pixel recurrent state over time, producing a dense image-like tensor while retaining longer temporal

history than single-frame time surfaces. The representation loses per-event ordering but preserves multi-event context per pixel.

Mixed-density event stacks. Mixed-Density Event Stacks (MDES) [16] allocate bins adaptively by event density rather than by time interval, alleviating the fixed-duration limitation of voxel grids at the cost of a data-dependent bin structure.

Ordered event representations. Zubić *et al.* [53] show that the choice of representation parameters (number of bins, temporal window, polarity encoding) has as much effect on downstream accuracy as the choice of representation family, and propose a ranking-based method for selecting good representations per task.

Speed-Invariant Frames (SIF). The SFNet paper [12] introduces SIF as an event-frame variant with an adaptive accumulation window whose length is chosen to equalise event count across scene-speed regimes. SIF preserves structural coherence under diverse motion but doubly collapses time: event counts are flattened as in standard frames, and the chosen window length is itself a scalar summary of the underlying temporal distribution.

Raw streams and recurrent latent encodings. At the opposite extreme from event frames, a network can consume the event list $\{(x_i, y_i, t_i, p_i)\}$ directly, either through sparse-native operators (point clouds [21], asynchronous graph networks [22]) or through a recurrent architecture whose hidden state summarises temporal history without preprocessing-time accumulation. EVA [18] instantiates the recurrent-encoder variant using linear-attention blocks that carry state across successive event chunks. The event pathway investigated in this thesis uses a frozen EVA-derived checkpoint of this form. The chunk length is an execution choice; it is not treated as a replacement for event timestamps.

3.2 The DSEC Dataset and Its Annotation Extensions

The DSEC dataset [10] provides synchronised stereo event and RGB recordings collected across Zurich and Interlaken, Switzerland. Two Prophesee Gen4 event cameras record at 640×480 resolution; two FLIR Blackfly S RGB cameras record at 1440×1080 native, rectified and cropped to 640×480 for the release. Approximately one hour of driving spans urban, suburban, and highway conditions. DSEC as originally released contains no object annotations; three separate downstream annotation efforts extend it.

DSEC-Detection (uzh-rpg). The authors of DSEC extended the dataset with object-detection annotations for the left-camera view in support of the DAGR low-latency detection system [22]. The extension, DSEC-Det, is maintained in the `uzh-rpg/dsec-det` repository and covers 53 of the 60 DSEC sequences with bounding-box annotations for eight traffic classes (CAR, PEDESTRIAN, RIDER, MOTORCYCLE, BICYCLE, TRUCK, BUS, TRAIN) across 63,931 rectified frames. The DSEC-Det release does not document a per-box manual verification step; the additional manual review applied by the SFNet paper (below) is the source of the lower-noise annotation surface used in this thesis.

SFNet manually reviewed annotations. The SFNet paper [12] builds on DSEC-Det and produces a 53-sequence eight-class detection set whose bounding boxes have been manually reviewed. SFNet reports a set of 208,000+ labels across 63,931 frames, substantially matching the DSEC-Det sequence coverage but with reduced annotation noise due to the human review step. Throughout this thesis, the term *SFNet annotations* refers to this manually reviewed detection set; whenever *DSEC-Det* is referenced without further qualification, the unreviewed uzh-rpg release is meant.

SpikeMOT DSEC-MOT (12-sequence tracking split). SpikeMOT [11] releases multi-object-tracking annotations for 12 DSEC sequences, adding persistent track identities and a seven-class scheme (with PEDESTRIAN and RIDER merged into PERSON). SpikeMOT's

annotations are produced against the original DSEC release rather than against DSEC-Det and therefore have no formal dependency on the uzh-rpg DSEC-Det detection set. The 12-sequence split is the canonical DSEC-MOT benchmark and is the principal evaluation surface used in this thesis.

The annotation surface used in this thesis. The custom 53-sequence MOT annotation used by this thesis (Chapter 6) combines (i) the SFNet manually reviewed bounding boxes and (ii) available DSEC-derived track identifiers. It retains provenance for the source rows and leaves unresolved identity cases explicit. Its construction and limitations are the subject of Chapter 6.

RGB frame alignment between annotation surfaces. The three annotation surfaces (DSEC-Det, SFNet, SpikeMOT's DSEC-MOT) are defined over slightly different rectified RGB frames. SpikeMOT's DSEC-MOT uses the rectified 640×480 RGB frames from the original DSEC release. The SFNet training pipeline applies an additional geometric transformation during its data-loading stage, producing an RGB image whose difference from the DSEC-MOT rectified RGB is empirically on the order of a sub-pixel to single-pixel offset at matched positions. Cross-evaluation between SFNet-trained detectors and SpikeMOT-annotated MOT ground truth is conducted under the working assumption that these frames are co-registered to within sub-pixel tolerance; the residual alignment error is reported as a caveat where relevant.

3.3 SFNet and Speed-Invariant Frames

The SFNet paper [12] (Structure-aware Fusion Network) is the current reference event–RGB fusion detector on DSEC-Det. Two design decisions in the paper are directly targeted by the baselines and methods in this thesis: the Speed-Invariant Frame event representation, and the Adaptive Feature Complement fusion pathway.

Speed-Invariant Frames (SIF). As introduced in §3.1.1, SIF converts the asynchronous event stream into a two-dimensional image-like tensor by adaptively accumulating events over a variable-length window whose duration is chosen to equalise the event count across scene-speed regimes. The transform is lossy in two structurally distinct senses. First, microsecond-scale timestamps are collapsed into a single (H, W) tensor, discarding within-window ordering; voxel grids retain this information at bin granularity, and raw-stream encoders retain it at event granularity. Second, although the histogram can be updated incrementally, the released representation is defined only at the selected window boundary. It therefore does not expose the intermediate event order to the tracker. Retaining that information is the design target of the CResMOT observation encoder (Chapter 4).

Three-branch fusion and detector controls. The SFNet paper fuses the SIF tensor with the RGB image through the Adaptive Feature Complement Module (AFCM), which propagates a global illumination cue through the RGB feature map before concatenation-plus-convolution fusion at three levels of a YOLOX-style feature pyramid. The detection head is unchanged. The RGB and SIF baselines in this thesis use the same YOLOX-m architecture and training pool but have separately trained weights. They are therefore detector-family controls, not identical-detector ablations. The proposed CResMOT-versus-ByteTrack comparison must instead reuse one fixed checkpoint and detection cache.

3.4 YOLOX

YOLOX [54] is an anchor-free single-stage object detector. Its properties relevant to this thesis are three. First, it separates classification and box regression into a decoupled detection head. Second, it uses SimOTA as a dynamic label-assignment strategy during training. Third, it exposes a Path Aggregation Feature Pyramid Network (PAFPN) at three spatial scales $\{p_3, p_4, p_5\}$ with resolutions (60×80), (30×40), (15×20) on a 640×480 input. CResMOT uses these features as its RGB observation input.

3.5 ByteTrack

ByteTrack [4] is the classical online-tracking reference used in this thesis: it is paired with the baseline detector and compared with the ByteTrack variant reported by SpikeMOT [11]. Each track maintains a Kalman filter with a constant-velocity motion model in image space,

$$\mathbf{x}_{k+1} \;=\; F\mathbf{x}_k + \mathbf{w}_k, \qquad \mathbf{z}_k \;=\; H\mathbf{x}_k + \mathbf{v}_k, \tag{3.5}$$

with state $\mathbf{x}_k = (x, y, s, r, \dot{x}, \dot{y}, \dot{s})$ encoding box centre, scale, aspect ratio, and their time derivatives. At each frame, detections are partitioned by score into $\mathcal{D}_{\text{high}}$ (score $> \tau_{\text{high}}$) and $\mathcal{D}_{\text{low}}$ (score $\in [\tau_{\text{low}}, \tau_{\text{high}}]$). The first matching stage associates $\mathcal{D}_{\text{high}}$ with predicted tracks by Hungarian assignment [23] over the IoU cost at $\mathcal{O}(n^3)$; the second stage associates $\mathcal{D}_{\text{low}}$ with tracks unmatched after stage one, on the hypothesis that low-confidence detections correspond to partially occluded real objects. Tracks unmatched after both stages enter a lost state and are retained for T_{die} frames before removal. ByteTrack is training-free. Chapter 7 reports the original 12-sequence hyperparameter sweep and the post-hoc training-only reaggregation.

3.6 Attention and Linear-Recurrent Sequence Models

Cross-attention. Cross-attention [44] computes a content-dependent aggregation of a context sequence (K, V) under queries Q:

$$\text{Attention}(Q, K, V) \;=\; \text{softmax}\!\left(\frac{QK^\top}{\sqrt{d_k}}\right)V, \qquad Q \in \mathbb{R}^{n\times d_k},\; K, V \in \mathbb{R}^{m\times d_k}. \tag{3.6}$$

Time complexity is $\mathcal{O}(nmd_k)$ and memory is $\mathcal{O}(nm)$. CResMOT applies Equation 3.6 to ROI-pooled RGB and event tokens (Chapter 4).

Linear attention and RWKV. The quadratic cost of softmax attention in the sequence length motivates a family of linear-time alternatives in which the attention operator is

rewritten as a recurrence over a running matrix-valued state. The RWKV architectures [19], [47], [48] instantiate this pattern and are the substrate for both the frozen event encoder used in this work and the structured observation-jump operator introduced in Chapter 5. In the most recent variant [19], the per-head token-mixing recurrence operates on a matrix state $S_t \in \mathbb{R}^{d_v \times d_k}$ through a bilinear update,

$$S_t \;=\; S_{t-1} \cdot \mathrm{diag}(w_t) \;+\; \Big(S_{t-1} \cdot a_t\Big) \cdot b_t^{\top} \;+\; v_t \cdot k_t^{\top}, \tag{3.7}$$

with projections w_t, a_t, b_t, v_t, k_t produced from the current token embedding. Time complexity is $\mathcal{O}(n d_v d_k)$ and per-step memory is $\mathcal{O}(d_v d_k)$. The architectural role of the three terms in the tracker's per-track memory update is developed in Chapter 5.

3.7 Neural Ordinary Differential Equations

Neural ODEs [36] are a continuous-time generalisation of residual networks in which the hidden state evolves according to a learned vector field:

$$\frac{d\mathbf{z}(t)}{dt} \;=\; f_\theta\Big(\mathbf{z}(t), t\Big), \qquad \mathbf{z}(t_1) \;=\; \mathbf{z}(t_0) + \int_{t_0}^{t_1} f_\theta(\mathbf{z}(t), t)\, dt, \tag{3.8}$$

with integration performed by a numerical solver such as RK4 or adaptive Dormand–Prince, and gradients obtained either by backpropagation through the solver or by the adjoint method. The architecture decouples the depth of computation from the number of discrete layers and permits evaluation at arbitrary continuous times. Chapter 2, §2.7 reviews its use in tracking.

3.8 Multi-Object-Tracking Metrics

All tracking experiments in this thesis report the standard CLEAR-MOT and HOTA metric families [55], [56], computed via the reference implementations `py-motmetrics` [57] and

`TrackEval` [58]. Matching between predicted and ground-truth boxes is performed at IoU ≥ 0.5 per frame using the Hungarian algorithm.

MOTA and IDF1. The Multi-Object Tracking Accuracy (MOTA) [55] combines three error modes into a single detection-oriented score,

$$\text{MOTA} \;=\; 1 - \frac{\text{FN} + \text{FP} + \text{IDSW}}{\text{GT}}, \tag{3.9}$$

where FN and FP are per-frame false-negative and false-positive counts, IDSW is the number of ID switches between consecutively matched tracks, and GT is the total number of ground-truth boxes. MOTA is dominated by detection quality and is annotation-density-sensitive: denser ground truth inflates FN relative to the same tracker predictions. IDF1 [59] measures identity preservation as the F1 score over identity-matched true positives,

$$\text{IDF1} \;=\; \frac{2\,\text{IDTP}}{2\,\text{IDTP} + \text{IDFP} + \text{IDFN}}, \tag{3.10}$$

where IDTP counts predictions whose assigned track identity matches ground truth.

HOTA. HOTA [56] is the current de facto metric for MOT and is the headline metric reported by SpikeMOT. It decomposes tracking performance into a detection score DetA and an association score AssA evaluated over all IoU thresholds $\alpha \in [0.05, 0.95]$,

$$\text{HOTA} \;=\; \frac{1}{|\mathcal{A}|} \sum_{\alpha \in \mathcal{A}} \sqrt{\text{DetA}(\alpha) \cdot \text{AssA}(\alpha)}, \tag{3.11}$$

with $\mathcal{A} = \{0.05, 0.10, \ldots, 0.95\}$. $\text{DetA}(\alpha)$ measures detection quality — the fraction of predicted boxes that match a ground-truth box at IoU $\geq \alpha$, independent of identity. $\text{AssA}(\alpha)$ measures identity consistency — given a correct box match, how often the predicted track ID stays bound to the same ground-truth object across frames. HOTA is the geometric mean of the two, averaged over α, so both components must be high for a high HOTA score.

Compared to MOTA, HOTA weights identity preservation more heavily and is less sensitive to annotation-density differences between surfaces.

Chapter 2 surveys the MOT literature itself, positioning CResMOT against classical, event-based single- and multi-object trackers, and the continuous-time sequence modelling families that constitute its principal intellectual inspirations (DAGR, StreamingFlow, ContiFormer, and TimeTracker).

Chapter 4

CResMOT Method I: Observation Pipeline

This chapter specifies the first of the two CResMOT method components: an observation pipeline that consumes synchronised RGB frames and a recurrent representation of the raw event stream, then produces a bundle of cross-modally fused tokens for each detection. Section 4.1 states the architectural decomposition. Section 4.2 describes the two upstream substrates — a YOLOX-m RGB detector based on the SFNet recipe [12], [54], and a frozen EVA-derived RWKV-7 event encoder [18], [19]. Section 4.3 specifies the ROI-local observation encoder. Section 4.4 states the interface to the tracker core in Chapter 5. The chapter describes the proposed full interface; Chapter 7 distinguishes the portions that have been correctness-checked from those that remain unevaluated.

4.1 Overall Architecture

The CResMOT pipeline is organised around three distinct inductive biases, each used where its structural assumptions are justified. Continuous-time latent dynamics are used for propagating per-track state between observations (Chapter 5, §5.2). Structured matrix-state jump updates are used for assimilating discrete new evidence into per-track state

at observation times (Chapter 5, §5.3). Transformer-style token attention is used for the cross-modal observation encoder specified in this chapter and for the learned association scorer (Chapter 5, §5.5).

At an RGB-frame boundary t_k, the event pathway first advances the recurrent encoder over events arriving in $[t_{k-1}, t_k]$. A decoded event representation $\mathbf{E}_k$ is then made available to the observation encoder. In parallel, the frozen YOLOX-m detector processes the RGB frame $I_k \in \mathbb{R}^{3\times H\times W}$ and produces detections $\mathcal{D}_k = \{(b_j, s_j, c_j)\}$ together with PAFPN features $\{\mathbf{P}_k^{(3)}, \mathbf{P}_k^{(4)}, \mathbf{P}_k^{(5)}\}$. For each detection b_j, the proposed observation encoder extracts event and RGB ROIs, applies bidirectional cross-attention, and produces three factored tokens (§4.3).

Between RGB frames, the recurrent state advances with the event stream and is read at the causal query times defined in Chapter 5, §5.2.2.

4.2 Frozen Upstream Substrates

The proposed CResMOT comparison freezes both upstream substrates. This isolates the tracker-side question and avoids attributing encoder fine-tuning to the recurrent tracker. The historical RGB and SIF baselines use separate detector weights; the final CResMOT-versus-ByteTrack comparison must instead fix one YOLOX-m checkpoint and its detections. The frozen EVA-derived checkpoint is a control and starting point; a task-trained event encoder is considered separately in Section 8.5.

4.2.1 YOLOX-m RGB detector

The RGB detector is a DSEC-specialised YOLOX-m checkpoint trained from the public COCO-pretrained YOLOX-m weights [54] with the SFNet data recipe [12]. This establishes common pretraining provenance, not identity with either separately trained baseline checkpoint. Before a controlled CResMOT evaluation, one checkpoint and its detections must be selected and reused by both CResMOT and ByteTrack. The detector is anchor-free with a

decoupled classification and regression head; at inference it produces, for each 640×480 frame, bounding boxes, confidence scores, and class predictions after non-maximum suppression at IoU 0.45.

The PAFPN feature pyramid is exposed as an additional output. At the 640×480 input resolution, the three pyramid levels have spatial dimensions $\mathbf{P}^{(3)} \in \mathbb{R}^{192\times60\times80}$ (stride 8), $\mathbf{P}^{(4)} \in \mathbb{R}^{384\times30\times40}$ (stride 16), and $\mathbf{P}^{(5)} \in \mathbb{R}^{768\times15\times20}$ (stride 32). Each level is projected to a shared $C_f = 64$ channel dimension by a 1×1 convolution with BatchNorm and SiLU, producing $\tilde{\mathbf{P}}^{(l)} \in \mathbb{R}^{64\times H_l\times W_l}$ for $l \in \{3, 4, 5\}$. The projections are the only trainable parameters introduced on the RGB side; the YOLOX backbone and PAFPN are frozen.

4.2.2 EVA-derived RWKV-7 event encoder

The evaluated event pathway uses an EVA-derived checkpoint [18] configured with RWKV-7 token mixing [19]. The checkpoint consumes the ordered event tuples $e_i = (x_i, y_i, t_i, p_i)$ and carries recurrent state across calls. The current configuration uses 16-pixel spatial patches, 24 recurrent heads of width 16, and execution chunks of 512 events. The chunk size sets the execution batch size; the timestamps and order within the stream remain part of the input.

For each head, the RWKV-7 recurrence updates a matrix-valued state $\mathbf{S}_t \in \mathbb{R}^{d_v\times d_k}$ through

$$\mathbf{S}_t \;=\; \mathbf{S}_{t-1} \cdot \mathrm{diag}(\mathbf{w}_t) \;+\; \Big(\mathbf{S}_{t-1} \cdot \mathbf{a}_t\Big) \cdot \mathbf{b}_t^{\top} \;+\; \mathbf{v}_t \cdot \mathbf{k}_t^{\top} \tag{4.1}$$

(Chapter 3, Equation 3.7). The tracker-core jump in Chapter 5 uses the same algebraic form but maintains a separate per-object state with separate projections.

For the proposed observation interface, $\mathbf{E}(t)$ denotes the decoded spatial event feature map at time t. It is projected to a shared $C_f = 64$ fusion dimension by a 1×1 convolution, BatchNorm, and SiLU. The recurrent encoder body remains frozen in the present configuration.

Streaming passes the returned recurrent state to the next event call instead of reinitialising

it at the call boundary.

4.3 Cross-Modal Observation Encoder

The observation encoder consumes, for each detected bounding box b_j, the corresponding RGB PAFPN features and the event latent feature map, and produces three factored output tokens destined for distinct roles in the tracker core. The factorisation is motivated by the separation of concerns in the tracker: event features serve both as discrete observation evidence (for the RWKV-7 jump, Chapter 5, §5.3) and as continuous-time conditioning for the ODE drift (Chapter 5, §5.2), and the two roles benefit from being produced as separate summaries rather than collapsed into a single vector.

4.3.1 ROI pooling across modalities

For each detection box b_j, the encoder extracts a 7×7 ROI-aligned patch from the projected event latent $\tilde{\mathbf{E}}(t_k)$, producing an event feature patch $\mathbf{f}_j^e \in \mathbb{R}^{64\times7\times7}$ flattened to a sequence of $N_e = 49$ event tokens of dimension 64. Three RGB patches are extracted at 4×4 spatial resolution from the projected PAFPN levels $\{\tilde{\mathbf{P}}^{(3)}, \tilde{\mathbf{P}}^{(4)}, \tilde{\mathbf{P}}^{(5)}\}$, flattened and concatenated to produce $N_r = 48$ RGB tokens of dimension 64 (16 tokens per scale). The asymmetric spatial resolution (7×7 for events versus $4 \times 4 \times 3$ for RGB) reflects the different information profiles of the two modalities: events carry fine-grained spatiotemporal structure within a single resolution tier, while RGB carries appearance evidence across a scale hierarchy.

Both token sequences are linearly projected to the fusion dimension $d_{\text{fus}} = 128$ and augmented with learned positional embeddings: 2D sinusoidal positions within the ROI for event tokens, concatenated 2D-within-scale plus scale-identity embeddings for RGB tokens.

4.3.2 Bidirectional cross-modal fusion

The event and RGB token sequences are fused by two identical bidirectional cross-attention blocks. Each block performs, in sequence, the following operations on the event tokens $\mathbf{X}^e \in \mathbb{R}^{N_e \times d_{\text{fus}}}$ and RGB tokens $\mathbf{X}^r \in \mathbb{R}^{N_r \times d_{\text{fus}}}$:

$$\mathbf{X}^e \leftarrow \mathbf{X}^e + \text{MHA}(\mathbf{X}^e, \mathbf{X}^e, \mathbf{X}^e) \tag{4.2}$$

$$\mathbf{X}^r \leftarrow \mathbf{X}^r + \text{MHA}(\mathbf{X}^r, \mathbf{X}^r, \mathbf{X}^r) \tag{4.3}$$

$$\mathbf{X}^e \leftarrow \mathbf{X}^e + \text{MHA}(\mathbf{X}^e, \mathbf{X}^r, \mathbf{X}^r) \tag{4.4}$$

$$\mathbf{X}^r \leftarrow \mathbf{X}^r + \text{MHA}(\mathbf{X}^r, \mathbf{X}^e, \mathbf{X}^e) \tag{4.5}$$

$$\mathbf{X}^{e,r} \leftarrow \mathbf{X}^{e,r} + \text{FFN}(\mathbf{X}^{e,r}) \tag{4.6}$$

where $\text{MHA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\mathbf{Q}\mathbf{K}^\top / \sqrt{d_{\text{fus}}/H})\mathbf{V}$ is multi-head attention with $H = 4$ heads (Chapter 3, Equation 3.6), and FFN is a two-layer feedforward network with GELU activation and hidden dimension $4 \cdot d_{\text{fus}}$. Each sub-operation is preceded by LayerNorm. The two blocks are stacked with identical structure but independent parameters, producing refined token sequences $\tilde{\mathbf{X}}^e$ and $\tilde{\mathbf{X}}^r$ whose representations have been informed by the complementary modality through two rounds of bidirectional cross-attention.

Compute cost per ROI is $\mathcal{O}(N_e^2 d_{\text{fus}} + N_r^2 d_{\text{fus}} + N_e N_r d_{\text{fus}})$ for each of the two blocks.

4.3.3 Query-based readouts

Three learned query vectors $\mathbf{q}^{\text{evt-part}}$, $\mathbf{q}^{\text{evt-ctrl}}$, $\mathbf{q}^{\text{rgb-part}} \in \mathbb{R}^{d_{\text{fus}}}$ read from the fused token sequences via cross-attention to produce the factored per-detection summaries:

$$\mathbf{o}_j^{\text{evt-part}} = \mathbf{W}^{\text{evt-part}} \cdot \text{MHA}(\mathbf{q}^{\text{evt-part}}, \tilde{\mathbf{X}}_j^e, \tilde{\mathbf{X}}_j^e) \tag{4.7}$$

$$\mathbf{o}_j^{\text{evt-ctrl}} = \mathbf{W}^{\text{evt-ctrl}} \cdot \text{MHA}(\mathbf{q}^{\text{evt-ctrl}}, \tilde{\mathbf{X}}_j^e, \tilde{\mathbf{X}}_j^e) \tag{4.8}$$

$$\mathbf{o}_j^{\text{rgb-part}} = \mathbf{W}^{\text{rgb-part}} \cdot \text{MHA}(\mathbf{q}^{\text{rgb-part}}, \tilde{\mathbf{X}}_j^r, \tilde{\mathbf{X}}_j^r) \tag{4.9}$$

where $\mathbf{W}^{(\cdot)} \in \mathbb{R}^{d_o \times d_{\text{fus}}}$ are learned output projections and $d_o = 64$ is the output dimension consumed by the tracker core. The three outputs have distinct intended roles. The *event-part* token $\mathbf{o}_j^{\text{evt-part}}$ is the discrete observation-time event evidence, consumed by the RWKV-7 jump projections in Chapter 5 (§5.3). The *event-control* token $\mathbf{o}_j^{\text{evt-ctrl}}$ is the continuous-time event conditioning, consumed by the ODE drift field in Chapter 5 (§5.2). The *RGB-part* token $\mathbf{o}_j^{\text{rgb-part}}$ is the discrete observation-time appearance evidence, also consumed by the jump projections.

At an inter-frame query time $t \in (t_{k-1}, t_k)$, the event-only path applies event self-attention and the event-part and event-control queries to $\tilde{\mathbf{X}}^e(t)$, bypassing cross-modal attention.

4.4 Interface to the Tracker Core

The proposed output of the observation pipeline at each RGB frame t_k is a bundle of per-detection tokens:

$$\mathcal{O}_k = \left\{ \left(b_j, s_j, c_j, \mathbf{o}_j^{\text{evt-part}}, \mathbf{o}_j^{\text{evt-ctrl}}, \mathbf{o}_j^{\text{rgb-part}} \right) \right\}_{j=1}^{|\mathcal{D}_k|} \tag{4.10}$$

comprising, for each detection, the bounding box b_j, confidence s_j, class c_j, and three factored fusion tokens. Between RGB frames, the tracker may also request an event-only conditioning token $\mathbf{o}_i^{\text{evt-ctrl}}(t)$ at an active track's ROI. Chapter 5 develops the tracker core against this interface.

Chapter 5

CResMOT Method II: Tracker Core

Chapter 4 specified an observation pipeline in which a frozen YOLOX-m detector [12], [54] supplies detections and RGB features, an EVA-derived recurrent encoder [18], [19] supplies event features, and ROI-local fusion produces three factored tokens ($\mathbf{o}^{\text{evt-part}}, \mathbf{o}^{\text{evt-ctrl}}, \mathbf{o}^{\text{rgb-part}}$). This chapter specifies the proposed tracker core. Its principal state is a matrix-valued memory $\mathbf{Z}_k(t)$ for each track. The state is propagated between observations by a Neural ODE (§5.2) and corrected at a matched observation by an RWKV-7-style jump (§5.3). Motion, appearance, and association outputs are read from the same state (§5.4). The association core and lifecycle machine complete the proposed tracker. Chapter 7 reports the narrower implementation evidence currently available; it does not treat this complete design as validated.

5.1 Per-Track Latent State

In the proposed core, each active or lost track k carries a persistent state $\mathbf{Z}_k(t) \in \mathbb{R}^{H\times N\times N}$ with $H = 8$ heads and $N = 16$. The state is matrix-valued for two reasons. First, the RWKV-7 recurrence [19] acts on latent key and value directions through column decay, delta-rule rewrite, and outer-product write (§5.3). Second, a task-specific query $\mathbf{q}^{(\ell)} \in \mathbb{R}^N$ can read from the same state through $\mathbf{Z}_k\mathbf{q}^{(\ell)}$. These 16×16 axes are latent key/value dimensions, not spatial image coordinates.

No separate motion latent, appearance latent, or association latent is maintained per track. All three are readouts of $\mathbf{Z}_k$ via distinct learned queries; observation-time updates consolidate into a single structured write to $\mathbf{Z}_k$. This differs from earlier DETR-family trackers [60], [61], [62] that maintain separate track-query embeddings per task and update them additively.

Design-space alternative: global latent field. Per-track state is one of at least two architecturally coherent choices at this position in the design space, and the alternative merits explicit mention because an earlier iteration of this work explored it before settling on the per-track formulation. A *global latent field* architecture maintains a single scene-level latent state $\mathbf{Z}^{\text{scene}}(t)$ indexed over spatial coordinates or a shared set of scene-level slot positions, with per-track readouts performed by spatial queries into the shared state. This is the pattern used by StreamingFlow [49] for BEV occupancy forecasting, where the task is to predict future scene states at the grid level rather than to maintain persistent identity. The global-field alternative has the attraction of allowing cross-track reasoning to emerge directly in the shared state's dynamics: interactions between nearby tracks (e.g. occlusions, crossings) are implicit in the evolution of the scene-level latent. The cost is that persistent identity becomes an association problem against a continuously evolving anonymous field, rather than an attribute of an explicit per-object state object; the identity decomposition must be recovered from the field rather than maintained within it. The per-track choice in this thesis trades the implicit cross-track reasoning of a global field for the explicit identity substrate that an MOT task demands. A coupled per-track formulation (in which per-track states evolve under a drift field that depends on neighbouring tracks' states) sits intermediate between the two and is identified as future work in Chapter 8.

5.2 Neural ODE Drift Between Observations

Between two matched observation times t_k and t_{k+1}, the proposed per-track state evolves according to

$$\frac{d\mathbf{Z}_k(\tau)}{d\tau} \;=\; -\,\mathbf{Z}_k(\tau)\cdot \mathrm{diag}\Big(\boldsymbol{\lambda}(\mathbf{Z}_k(\tau), \boldsymbol{\phi}_k(\tau))\Big) \tag{5.1}$$

where $\boldsymbol{\lambda}(\mathbf{Z}, \boldsymbol{\phi}) \in \mathbb{R}^{H\times N}_{\geq 0}$ is a learned per-column decay-rate field that depends on a lightweight summary of the current state $\mathbf{Z}$ and a per-track conditioning vector $\boldsymbol{\phi}_k$. The drift in (5.1) is linear in $\mathbf{Z}$ at each instant and separable per column of each head: each column of each head decays independently at its own learned rate.

5.2.1 Conditioning of the drift field

The conditioning vector $\boldsymbol{\phi}_k(\tau)$ contains three terms. The *per-track event rate* $r_k(\tau)$ measures event activity inside the current ROI over a short interval and is clipped to $[0, 10]$. It is an activity cue rather than an unambiguous motion measurement because contrast, ego-motion, and sensor noise also affect the rate. The *event-control feature* $\mathbf{o}_k^{\text{evt-ctrl}}(\tau)$ is a learned ROI summary of the decoded recurrent event features (Chapter 4, §4.3.3). The *time features* $\boldsymbol{\phi}_t(\Delta\tau)$ contain normalised elapsed time, a sinusoidal embedding, and $\log(1+\Delta\tau)$. The full conditioning vector is $\boldsymbol{\phi}_k(\tau) = [\, r_k(\tau); \mathbf{o}_k^{\text{evt-ctrl}}(\tau); \boldsymbol{\phi}_t(\Delta\tau)\,]$.

5.2.2 Causal inter-frame schedule

The RGB interval is divided into causal subintervals

$$t_k = q_0 < q_1 < \cdots < q_M = t_{k+1}. \tag{5.2}$$

Interior times q_m are produced when the recurrent event encoder completes an execution chunk; any remaining events are consumed at the RGB boundary. At q_m, the event encoder has processed only events with timestamps no later than q_m. The tracker reads $\mathbf{o}_k^{\text{evt-ctrl}}(q_m)$

at the current predicted ROI, recomputes $\boldsymbol{\lambda}$, and holds that rate over $[q_m, q_{m+1}]$. A track queried inside the subinterval is propagated with the most recent causal rate.

The 512-event setting is an execution batch size rather than a temporal representation; a partial batch is consumed at the RGB boundary. Event input between RGB observations modulates the retention of the existing track state through $\boldsymbol{\lambda}$. It does not deposit a new key–value pair into $\mathbf{Z}_k$; new content enters through the matched-observation jump in §5.3.

5.2.3 Closed-form integration between observations

For one causal subinterval $[\tau_0, \tau_1] = [q_m, q_{m+1}]$, Equation 5.1 admits a closed-form solution per column:

$$\mathbf{Z}_k(\tau_1) \;=\; \mathbf{Z}_k(\tau_0) \cdot \operatorname{diag}\Big(\exp(-\boldsymbol{\lambda} \cdot \Delta\tau)\Big), \qquad \Delta\tau = \tau_1 - \tau_0. \tag{5.3}$$

The factor $\exp(-\boldsymbol{\lambda}\Delta\tau) \in (0, 1]$ is applied per column of each head. For a DSEC inter-frame interval $\Delta\tau = 50\,\mathrm{ms}$ and rates $\lambda \in [10^{-2}, 10^{1}]\,\mathrm{s}^{-1}$, the preservation factor spans approximately $[0.61, 0.9995]$, allowing both rapid and slow forgetting.

The proposed implementation evaluates $\boldsymbol{\lambda}$ once per causal subinterval. A higher-order scheme could instead re-evaluate the rate within the subinterval.

5.2.4 Uniform treatment of active and lost tracks

Active and lost tracks use the same drift. A lost track's state continues to decay while it remains eligible for revival. The present design does not couple the ODE states of neighbouring tracks; cross-track reasoning is left to the association core at observation times.

5.2.5 Relationship to control-theoretic state-space models

The drift–jump decomposition developed above has a close formal resemblance to the continuous-time linear state-space models studied in control theory. A linear time-invariant

(LTI) plant with a sampled observation correction is conventionally written as

$$\dot{\mathbf{x}}(t) = \mathbf{A}\,\mathbf{x}(t) + \mathbf{B}\,\mathbf{u}(t), \qquad \mathbf{x}(t_k^+) = \mathbf{x}(t_k^-) + \mathbf{K}_k\Big(\mathbf{y}_k - \mathbf{C}\,\mathbf{x}(t_k^-)\Big), \tag{5.4}$$

with continuous-time plant dynamics driven by a control input $\mathbf{u}$ between observations, and an observer correction (a Kalman innovation, in the stochastic case) applied at discrete observation instants t_k. The mathematical shape of Equation 5.1 together with the RWKV-7 jump of §5.3 is the same: a continuous-time drift of the latent between observations, with state-dependent dynamics parameterised by $\boldsymbol{\lambda}(\mathbf{Z}, \boldsymbol{\phi})$ in place of the LTI system matrix $\mathbf{A}$, and a structured correction at each observation that mixes the pre-jump state with freshly observed content. Three differences from the classical control-theoretic form are worth noting. First, the drift field $\boldsymbol{\lambda}$ is learned and state-dependent, making the per-track drift a nonlinear (but still linear-in-$\mathbf{Z}$) dynamical system whose coefficients depend on its own state. Second, the observation correction is bilinear in the state (the delta-rule term of Equation 5.5 is second-order in $\mathbf{Z}_k$ and its projections), placing the full update outside the LTI regime proper and into the bilinear-control family studied in nonlinear systems theory. Third, the correction is structured as the three write modes of an associative memory rather than as a projection onto an innovation residual, which reflects the downstream task: the tracker is maintaining an identity slot rather than estimating an unobserved plant state. These correspondences are noted for intellectual context and as a motivation for continued study of the architecture from a systems-theoretic perspective; they do not materially affect the experimental methodology of Chapter 7.

5.3 RWKV-7-Style Observation Jump

At a matched observation time t_k, the proposed correction applies the RWKV-7 matrix-state step [19] once to that track:

$$\mathbf{Z}_k(t_k) \;=\; \mathbf{Z}_k(t_k^-) \cdot \mathbf{D}_k \;+\; \Big(\mathbf{Z}_k(t_k^-) \cdot \mathbf{a}_k\Big) \cdot \mathbf{b}_k^\top \;+\; \mathbf{v}_k \cdot \mathbf{k}_k^\top \tag{5.5}$$

where $\mathbf{Z}_k(t_k^-)$ denotes the pre-jump state after the drift (§5.2) has propagated $\mathbf{Z}_k$ up to t_k, and $\mathbf{D}_k = \mathrm{diag}(\exp(-\exp(\mathbf{w}_k)))$ is a bounded column-wise decay. The three terms of (5.5) are the three structurally distinct update modes of the RWKV-7 step and each has an interpretable role in an associative per-track memory.

Column-wise decay ($\mathbf{Z}_k \cdot \mathbf{D}_k$). The first term applies a bounded discrete decay to each column of the already-drifted state, allowing the observation to dampen key-value slots that the new evidence contradicts. The double-exponential parameterisation $\mathbf{D}_k = \mathrm{diag}(\exp(-\exp(\mathbf{w}_k)))$ ensures that the decay factor per column lies in $(0, 1]$ for any $\mathbf{w}_k \in \mathbb{R}^{H\times N}$, and the projected scalar $\mathbf{w}_k$ can be bounded away from $-\infty$ during training to keep the discrete decay floor above a stable numerical threshold.

In-place delta-rule rewrite ($(\mathbf{Z}_k \cdot \mathbf{a}_k) \cdot \mathbf{b}_k^\top$). The second term reads the state along key direction $\mathbf{a}_k \in \mathbb{R}^N$ (producing a vector of per-head values) and rewrites the result along value direction $\mathbf{b}_k \in \mathbb{R}^N$. This is the delta-rule operation that updates an existing key–value association to reflect the new observation, rather than merely decaying the old content and additively accumulating new content. The projections $\mathbf{a}_k$ and $\mathbf{b}_k$ are L2-normalised before the outer product to bound the magnitude of the rewrite term independently of projection scale.

Outer-product write ($\mathbf{v}_k \cdot \mathbf{k}_k^\top$). The third term deposits a fresh key–value pair into the state: store value $\mathbf{v}_k \in \mathbb{R}^{H\times N}$ at key direction $\mathbf{k}_k \in \mathbb{R}^N$. This is the raw associative write

that introduces new content into the state not currently represented.

5.3.1 Jump projection heads

The six projections $(\mathbf{w}_k, \mathbf{a}_k, \mathbf{b}_k, \mathbf{v}_k, \mathbf{k}_k, \mathbf{r}_k)$ are produced by a small feed-forward head from the concatenated observation tokens at the matched detection:

$$(\mathbf{w}_k, \mathbf{a}_k, \mathbf{b}_k, \mathbf{v}_k, \mathbf{k}_k, \mathbf{r}_k) \;=\; \mathrm{JumpHead}\Big(\big[\mathbf{o}_k^{\text{evt-part}};\, \mathbf{o}_k^{\text{evt-ctrl}};\, \mathbf{o}_k^{\text{rgb-part}}\big]\Big) \tag{5.6}$$

where JumpHead is a two-layer MLP with GELU activation. The receptance projection $\mathbf{r}_k$ is not used in the jump itself but is retained for the readout stage (§5.4) in line with the standard RWKV-7 formulation.

Both modalities enter through the concatenated observation tokens, allowing the learned projections to re-weight their contributions.

5.4 Readout Heads

Per-track outputs are query-based readouts of $\mathbf{Z}_k$. Separate learned queries are used for the box, appearance, and association tasks. For task index ℓ at time t, the readout is

$$\mathbf{y}_k^{(\ell)}(t) \;=\; \mathbf{Z}_k(t) \cdot \mathbf{q}^{(\ell)} \;\in\; \mathbb{R}^{H \times N}. \tag{5.7}$$

The flattened HN-dimensional readout is then passed through a task-specific head to produce the final output.

The *box readout* head decodes the motion readout $\mathbf{y}_k^{(\text{box})}$ to a normalised bounding box $(c_x, c_y, w, h) \in [0,1]^4$ via a stack of residual fusion blocks (LayerNorm, linear, GELU, residual add). The decoded box is rescaled to pixel coordinates on the image plane and serves as the tracker's predicted track position at the query time.

The *appearance readout* head projects $\mathbf{y}_k^{(\text{app})}$ through a two-layer MLP with L2 normalisa-

tion on the output, producing a unit-norm appearance embedding of dimension $d_{\text{app}} = 128$. The appearance embedding is consumed by the association core (§5.5) and by the appearance-alignment training objective (§5.8).

The *association readout* head projects $\mathbf{y}_k^{(\text{assoc})}$ through a two-layer MLP to produce a d_{assoc}-dimensional association feature consumed as the track-side input to the association core.

5.5 Pairwise Association Core

The association core scores track–detection pairs for the matching stage. The core is shared between the active-stage matcher (scoring tracked tracks against current detections), the second-stage rescuer (scoring low-confidence detections against first-stage-unmatched tracks), and the lost-track linker (scoring lost tracks against candidate births). The three stages differ only in their input pools; the scoring function is the same.

The input to the core is a track-side feature $\mathbf{u}_k^{\text{trk}} = [\,\mathbf{y}_k^{(\text{assoc})};\, \mathbf{y}_k^{(\text{app})};\, \text{PE}(\mathbf{b}_k^{\text{proj}})\,]$ (the association readout, appearance readout, and positional encoding of the track's ODE-projected box) and a detection-side feature $\mathbf{u}_j^{\text{det}} = [\,\mathbf{o}_j^{\text{rgb-part}};\, \mathbf{o}_j^{\text{evt-part}};\, \text{PE}(\mathbf{b}_j)\,]$ (the observation tokens and positional encoding of the detection box).

A cross-attention refinement block exchanges information bidirectionally between the track- and detection-side feature sets, producing refined features $\tilde{\mathbf{u}}_k^{\text{trk}}$ and $\tilde{\mathbf{u}}_j^{\text{det}}$. The affinity score is then computed as

$$a_{k,j} \;=\; \text{MLP}\Big(\big[\,\tilde{\mathbf{u}}_k^{\text{trk}};\, \tilde{\mathbf{u}}_j^{\text{det}};\, \mathbf{g}_{k,j}\,\big]\Big) \tag{5.8}$$

where $\mathbf{g}_{k,j}$ is a pairwise geometric feature (IoU of the track's projected box with the detection, box-centre distance, log-aspect-ratio ratio, log-scale ratio). A parallel birth-versus-no-match head produces an auxiliary scalar a_j^{birth} from the detection-side feature alone, scoring whether an unmatched detection should spawn a new track.

5.6 Lifecycle State Machine and Two-Stage Matching

The tracker's runtime composes the learned components into a classical ByteTrack-style two-stage matching flow [4] with an explicit lifecycle state machine. Each track at any given frame is in one of three states. A *tracked* track was matched to a detection in the most recent frame. A *lost* track has been unmatched for fewer than T_{die} frames; its state is still propagated forward and remains eligible for revival. A *removed* track has been unmatched for longer than T_{die} frames and is permanently discarded. A separate *unconfirmed* substate applies to newly spawned tracks until they are matched in a second consecutive frame.

The matching procedure at each RGB frame t_k runs as follows. Detections are split by confidence into a high-confidence set $\mathcal{D}_{\text{high}} = \{b_j : s_j > \theta_{\text{high}}\}$ and a low-confidence set $\mathcal{D}_{\text{low}} = \{b_j : s_j \in [\theta_{\text{low}}, \theta_{\text{high}}]\}$. All active and lost tracks are projected by the ODE (§5.2) from their last observation time to t_k and decoded to boxes $\mathbf{b}_k^{\text{proj}}$ via the box readout (§5.4). In the first matching stage, $\mathcal{D}_{\text{high}}$ is matched against projected tracks by Hungarian assignment [23] over the association-core affinity $a_{k,j}$ gated by an IoU floor; matched pairs trigger the RWKV-7 jump update (§5.3) on each matched track's $\mathbf{Z}_k$. In the second matching stage, $\mathcal{D}_{\text{low}}$ is matched against first-stage-unmatched tracks by pure IoU distance on the ByteTrack rationale that low-confidence detections are less reliable inputs to a learned scorer. First-stage-unmatched detections become candidate births. Before new track IDs are assigned, the lost-track pool is scanned by the association core for revival candidates: a lost track whose affinity with a candidate-birth detection exceeds a revival threshold is revived with its original ID preserved. Only candidates that fail the revival check become genuinely new tracks. The staging order — first-stage match, second-stage rescue, revival scan, new-track spawn — is important because revival must pre-empt a fresh ID assignment to preserve identity across occlusion gaps.

Table 5.1: Relationship between the method specification and the configurations evaluated in Chapter 7. "Evaluated" denotes a complete tracker rollout on the stated substrate.

System	Implemented interface	Evidence status
Complete method	ROI-local RGB–event fusion, per-track Neural-ODE propagation, RWKV-7 observation jump, joint association core	Specified only; not trained or rolled out as one system
Event-motion adapter	Persistent causal event field; affine box correction and covariance applied before ByteTrack association	Complete retrospective two-sequence rollout; modest gain
Later association memory	Per-track state read after candidate formation, as a cost residual or an identity-slot scorer	Complete rollouts on both substrates; rejected or too small to promote

5.7 Implemented Reductions of the Proposed Tracker

The complete tracker specified above was not the model used for any full-sequence result in Chapter 7. Several narrower reductions were implemented, and their boundaries matter.

The event-motion adapter is a wrapper around an otherwise fixed ByteTrack shell. Its recurrent event field advances causally and yields an affine box correction and covariance that alter association geometry; ByteTrack retains Kalman and lifecycle authority. The later systems instead read per-track state after candidate formation, either as a bounded cost residual or as an explicit per-track recurrent state feeding an already formed assignment surface. None implements the Neural ODE, ROI-local bidirectional fusion, RWKV-7 jump, and joint association core together. The fitting-set oracles are diagnostics rather than tracker variants.

5.8 Training Objectives

The training procedure below belongs to the complete proposed method. It specifies how the complete system would be trained; the evaluations in Chapter 7 use the narrower configurations described above. It jointly optimises the cross-modal observation encoder, drift field, jump projection head, readout heads, association core, and box decoder on detector-driven DSEC rollouts. The frozen YOLOX-m and EVA-derived encoders provide observations,

while per-track state is carried across the rollout.

The *core supervision* terms are binary cross-entropy losses on the association core's affinity scores for ground-truth-matched and ground-truth-unmatched pairs across the three association contexts (active-stage $\mathcal{L}_{\text{ab}}$, second-stage $\mathcal{L}_{\text{bc}}$, and lost-track $\mathcal{L}_{\text{link}}$), plus cross-entropy on the birth head's candidate-choice decision ($\mathcal{L}_{\text{birth}}$). These four terms carry fixed unit weights and provide the primary supervision.

The *auxiliary supervision* terms shape the ODE and the appearance embedding. The ODE-carry loss $\mathcal{L}_{\text{ode-carry}}$ regularises the ODE-projected motion readout at observation times against the expected target implied by the next ground-truth box, preventing the jump from absorbing arbitrary drift behaviour by corrective overshoot. The ODE-box loss $\mathcal{L}_{\text{ode-box}}$ supervises the box decoded from the ODE-projected state directly against the ground-truth box. The async-supervision loss $\mathcal{L}_{\text{async}}$ applies Huber loss to the box decoded from the propagated state at a sub-frame time, using an interpolated ground-truth box; this encourages smooth evolution across the interval rather than delta-like jumps at the endpoints. The appearance-alignment loss $\mathcal{L}_{\text{app}}$ is a cosine-similarity contrastive loss on the appearance readout, pulling same-identity embeddings together and orthogonalising different-identity embeddings.

The specified optimiser is AdamW [63] with learning rate 10^{-4}, weight decay 10^{-4}, linear warmup, cosine decay, and gradient clipping at norm 1.0. Kendall-style uncertainty weights [64] are available for the auxiliary losses, with clamps to prevent a required term from being assigned negligible weight. For causal streaming, the EVA recurrent state is exposed at each call boundary and passed to the next event chunk.

Chapter 6

53-Sequence MOT Annotation Surface

This thesis retains two annotation surfaces on DSEC: the canonical 12-sequence DSEC-MOT split released with SpikeMOT [11], and a larger 53-sequence annotation built for this work. Chapter 7 reports the canonical split and a 12-sequence SFNet-label subset; it reports no aggregate over all 53 sequences. The larger surface is not a new dataset. It merges two existing artefacts, the manually reviewed boxes of the SFNet release [12] and the track identifiers of the DSEC-Det extension [22], onto a shared frame surface with per-row provenance. No public release combines them: DSEC-Det supplies identity without documented per-box verification, and SFNet supplies verified boxes without identifiers (Chapter 3, §3.2).

6.1 Two-Lane Annotation Contract

The merger produces two parallel surfaces. **Lane A (projected runtime,** 298,292 **rows)** carries DSEC track identifiers through the SFNet training geometry. **Lane B (enriched manual-box,** 208,094 **rows)** preserves the original SFNet box geometry; 207,849 rows have non-sentinel identifiers and 245 carry a manual-review sentinel. Each Lane B row records its transfer method, confidence, matched Lane A tracklet, and margin. Some Lane B identities come from ReID or synthetic tracking, so Lane B is an enriched analysis surface rather than native DSEC identity ground truth. The lanes share sequences and timestamps but not box

geometry.

6.2 Coverage, Caveats, and Alignment

Coverage increased from 169,494 accepted correspondences (81.5%) after geometric matching to 188,750 assigned rows (90.7%) after tracklet-based transfer. Provenance-labelled ReID and synthetic-track passes then reduced the manual-review set to 245 rows (0.12%). These are successive stages of enrichment, not alternative estimates of coverage.

One structural finding requires disclosure. Lane A contains no native identities for the TRAIN class, covering 2,307 annotations in training and 727 in test. Lane B retains these boxes, most with provenance-labelled ReID or synthetic identifiers and 15 with manual-review sentinels. Detection evaluation remains available, but MOT analysis must distinguish inferred Lane-B identities from native DSEC tracks. The pooled 12-sequence SFNet analysis in Chapter 7 uses a separate truth surface with limited TRAIN identities.

The SFNet training pipeline applies a further geometric transformation during data loading. Its output differs from the DSEC-MOT rectified RGB by a sub-pixel to single-pixel offset at matched positions. Cross-evaluation between SFNet-trained detectors and SpikeMOT ground truth therefore assumes sub-pixel co-registration; no pixel-identity claim is made.

6.3 Sequence Inventory

The annotation covers urban, suburban, and highway scenes with DSEC-Det's eight classes (CAR, PEDESTRIAN, RIDER, MOTORCYCLE, BICYCLE, TRUCK, BUS, TRAIN). Lane B contains boxes on 55,490 unique RGB frames (38,975 training and 16,515 evaluation); Lane A contains 53,686 (37,918 and 15,768). The split used here has 39 training and 14 evaluation sequences; the separate 41/12 count describes source records, not the experimental split. For comparison with SpikeMOT's seven-class scheme, PEDESTRIAN and RIDER merge into PERSON. The 12-sequence head-to-head uses SpikeMOT annotations directly; the paired internal analysis

uses the separately labelled 12-sequence SFNet surface.

Chapter 7

Experiments

This chapter reports the detector and ByteTrack baselines on DSEC, followed by experiments on the recurrent event pathway and tracker integration. Section 7.1 specifies the datasets, metrics, and training conditions. Sections 7.2–7.5 report detector, MOT, per-class, and hyperparameter results. Section 7.6 separates the retrospective two-sequence comparison from the later complete fitting substrate. Section 7.7 measures the ceiling of interventions applied after candidate formation. Section 7.8 reports the earlier frozen-EVA recurrence check and matched short-window causal diagnostic.

In the baseline sections, the **designated test split** refers to the two canonical DSEC-MOT test sequences `interlaken_00_d` and `zurich_city_00_b`. The original ByteTrack sweep and later architecture work exposed these sequences, so all later two-sequence numbers are retrospective comparisons, not held-out results. The calibrated-substrate experiments use the other 10 sequences as a complete fitting surface and are not numerically interchangeable with the earlier two-sequence protocol. Section 7.8 reports a separate 100-frame diagnostic on `zurich_city_00_b`; it is not a final controlled evaluation.

7.1 Experimental Setup

Datasets and splits. The canonical 12-sequence DSEC-MOT split released with SpikeMOT [11] supplies the external head-to-head comparison. The per-class analysis uses a 12-sequence SFNet-label subset of the annotation lineage described in Chapter 6; no aggregate result over the complete 53-sequence surface is reported. The canonical test sequences are `interlaken_00_d` and `zurich_city_00_b`; the other 10 sequences form the training pool.

Metrics. Tracking metrics are computed by `TrackEval` [58] and `py-motmetrics` [57] at IoU ≥ 0.5 per frame. The headline metrics reported are HOTA [56] and its two components DetA and AssA, together with MOTA [55] and IDF1 [59]. Evaluation is conducted under the class-agnostic protocol used by SpikeMOT's Table 5 unless stated otherwise; per-class breakdowns are reported separately in §7.4. Detection metrics are the COCO AP_{50}, $AP_{50:95}$, and AP_{75} at class-agnostic granularity for cross-tracker comparability, plus standard 8-class COCO mAP for internal detection-quality reporting.

Hardware and training recipe. The RGB detector (YOLOX-m) is fine-tuned on two NVIDIA RTX 2080 Super GPUs with DistributedDataParallel. The RGB-only and SFNet event-fused detectors share the YOLOX-m backbone with COCO pre-training; their DSEC training follows the SFNet recipe [12] for 50 rather than 80 epochs and uses the 10-sequence DSEC-MOT training pool. The frozen event diagnostic uses an epoch-90 EVA-derived RWKV-7 checkpoint. All 164 inference tensors load exactly, the training-only loss tensor is omitted, and the encoder has no trainable parameters. The checkpoint's training duration is reported as provenance and is not used as evidence of downstream tracking quality.

Tracker-stage hyperparameters. ByteTrack's scalar hyperparameters are swept as described in §7.5. The original pooled sweep includes all 12 sequences; a post-hoc reaggregation over the 10 training sequences selects the same settings. The losses in Chapter 5 remain the intended specification for the complete CResMOT tracker. The implementation experiments

instead evaluate the narrower systems in Table 5.1. Within each evaluation, detector outputs, evaluator, sequence boundaries, and ByteTrack substrate are fixed, and scores are compared only against controls sharing them.

7.2 Detection Results

Table 7.1 reports class-agnostic detection on the designated test split for the two detector configurations (RGB-only YOLOX-m, and SFNet with Speed-Invariant Frames) cross-evaluated against both annotation surfaces. The RGB-only detector achieves $AP_{50} = 80.52$ against SFNet labels and 80.13 against DSEC-MOT labels; SFNet with SIF achieves 73.73 and 78.49 respectively.

The per-class breakdown (Table 7.2) confirms the pattern the SFNet paper predicts. SIF gains on classes with high apparent motion and loses on appearance-dominant ones. The MOTORCYCLE class gains $+20.1\,AP_{50:95}$ from SIF, the largest single per-class gain observed; BUS loses $-10.3\,AP_{50:95}$ under SIF, which matches SFNet's own reported failure mode on slow appearance-dominant classes. The CAR and PEDESTRIAN classes, which together carry most of the annotation volume, lose small amounts under SIF. This motion-vs-appearance split is the primary lens through which the per-class MOT results in §7.4 should be read.

Table 7.1: Class-agnostic detection on the designated DSEC-MOT test split (mean across `interlaken_00_d` and `zurich_city_00_b`).

Detector	Labels	AP_{50}	$AP_{50:95}$	AP_{75}
RGB	SFNet	80.52	53.57	58.55
RGB	DSEC-MOT	80.13	50.39	56.48
SIF	SFNet	73.73	48.43	51.77
SIF	DSEC-MOT	78.49	50.40	55.00

Table 7.2: Per-class detection $AP_{50:95}$ on the two designated SFNet validation sequences (epoch 50). Classes with positive Δ favour the event-fused detector.

Class	RGB	SIF	Δ (SIF − RGB)
car	69.97	66.57	−3.40
pedestrian	42.66	35.70	−6.96
cyclist	5.05	4.84	≈ 0
motorcycle	15.88	**35.94**	**+20.06**
bicycle	1.06	1.14	≈ 0
truck	13.79	14.09	≈ 0
bus	15.57	5.27	**−10.30**
train	*no ground truth in the validation pair*		

7.3 MOT Results and SpikeMOT Head-to-Head

Table 7.3 reports aggregate class-agnostic MOT metrics on the designated test split across the four baseline configurations. The strongest baseline is **RGB + ByteTrack** under DSEC-MOT labels: HOTA 57.77, DetA 52.74, AssA 63.87, MOTA 53.25, IDF1 73.48. Under SFNet labels the same configuration gives HOTA 57.96, MOTA 57.10, IDF1 72.02. The MOTA gap reflects annotation density (Chapter 3, §3.8). SIF + ByteTrack trails RGB + ByteTrack on aggregate MOT despite its per-class detection wins (Table 7.2). Per-class gains do not reach the pooled mean because that mean is car-dominated; §7.4 re-examines this.

Table 7.4 places the strongest baseline against the published SpikeMOT numbers. The RGB + ByteTrack configuration exceeds SpikeMOT on four of five headline metrics: HOTA +5.27, DetA +3.24, AssA +8.17, IDF1 +10.58, with MOTA trailing by −1.45. It also exceeds SpikeMOT's own ByteTrack reference (which SpikeMOT reports as an ablation baseline) on all five metrics by margins ranging from +5.15 (MOTA) to +18.98 (IDF1).

Table 7.3: Class-agnostic MOT on the designated test split, four baseline configurations. Bold = strongest baseline per column.

Tracker	Labels	HOTA	DetA	AssA	MOTA	IDF1
RGB + ByteTrack	SFNet	**57.96**	52.01	**65.45**	**57.10**	72.02
RGB + ByteTrack	DSEC-MOT	57.77	**52.74**	63.87	53.25	**73.48**
SIF + ByteTrack	SFNet	55.58	49.15	63.77	53.52	68.74
SIF + ByteTrack	DSEC-MOT	55.84	50.23	62.74	41.43	70.44

Table 7.4: Head-to-head against SpikeMOT on the canonical DSEC-MOT labels and the same two test sequences.

Method	HOTA	DetA	AssA	MOTA	IDF1
SpikeMOT paper [11]	52.5	49.5	55.7	**54.7**	62.9
SpikeMOT's ByteTrack reference [11]	44.9	42.3	47.9	48.1	54.5
RGB + ByteTrack (this thesis)	**57.77**	**52.74**	**63.87**	53.25	**73.48**
Δ vs. SpikeMOT paper	+5.27	+3.24	+8.17	−1.45	+10.58
Δ vs. their ByteTrack	+12.87	+10.44	+15.97	+5.15	+18.98

This comparison requires a methodological caveat on the detector-stage difference. SpikeMOT's ByteTrack reference uses a YOLOv5 detector whose RGB pre-training configuration is not disclosed in the published paper. This thesis begins from the COCO-pre-trained YOLOX-m checkpoint published with the original YOLOX release [54], the same pre-step SFNet uses before its DSEC-Det fine-tune (Chapter 4, §4.2.1). Both sides therefore start from a pre-trained RGB detector. What differs is the detector family, the disclosure status of its pre-training, and the specialisation recipe. The comparison conflates tracker quality, detector quality, and that recipe, while controlling only for annotation regime. Table 7.4 therefore shows that a classical RGB-plus-ByteTrack pipeline with disclosed provenance exceeds SpikeMOT's published numbers. It exercises no CResMOT contribution. CResMOT must instead be compared with RGB + ByteTrack under the same frozen detector; §7.8 provides only a short frozen-EVA diagnostic and not that required comparison.

Table 7.5: Per-class MOT IDF1 on the 12-sequence SFNet surface at match_thresh=0.7. Bold Δ = SIF improves by at least +0.04 IDF1 over RGB.

Class	RGB IDF1	SIF IDF1	Δ	n_seqs
car	0.840	0.838	≈ 0	12
pedestrian	0.481	0.479	≈ 0	11
cyclist	0.425	0.399	-0.026	8
motorcycle	0.579	**0.620**	**+0.041**	6
bicycle	0.258	**0.365**	**+0.107**	8
truck	0.638	0.659	+0.021	6
bus	0.799	0.772	-0.027	5
train	0.765	0.763	≈ 0	2
Macro mean (class-balanced)	0.603	0.617	+0.014	—
Pooled mean (car-dominated)	*tied (car dominates)*			—

7.4 Per-Class MOT Analysis

Aggregate class-agnostic metrics reflect the pooled distribution of annotations, which on DSEC-MOT is dominated by the CAR class. A macro-averaged per-class view reveals a different picture than the pooled mean suggests. Table 7.5 reports per-class IDF1 for the RGB + ByteTrack and SIF + ByteTrack configurations under SFNet labels at match threshold 0.7.

On this pooled 12-sequence surface, macro-averaged per-class IDF1 favours SIF by 0.014. The largest differences are for BICYCLE (+0.107) and MOTORCYCLE (+0.041). Motorcycle also has a substantial SIF detection gain, whereas bicycle detection is approximately unchanged. Pooled IDF1 is approximately tied because the CAR class dominates the annotation volume. Because this analysis includes training sequences, it is descriptive and is not an independent test gain.

7.5 ByteTrack Hyperparameter Sweep

The sweep varies one ByteTrack axis at a time from the default tuple $(\tau_{\text{track}}, \tau_{\text{match}}, A_{\min}, \tau_{\text{det}}) = (0.5, 0.8, 100, 0.25)$. The ranges are $\tau_{\text{track}} \in \{0.3, 0.4, 0.5, 0.6, 0.7\}$, $\tau_{\text{match}} \in \{0.7, 0.8, 0.9\}$, $A_{\min} \in \{50, 100, 200\}$ pixels, and $\tau_{\text{det}} \in \{0.15, 0.25, 0.35\}$. For each detector/label pair, 14

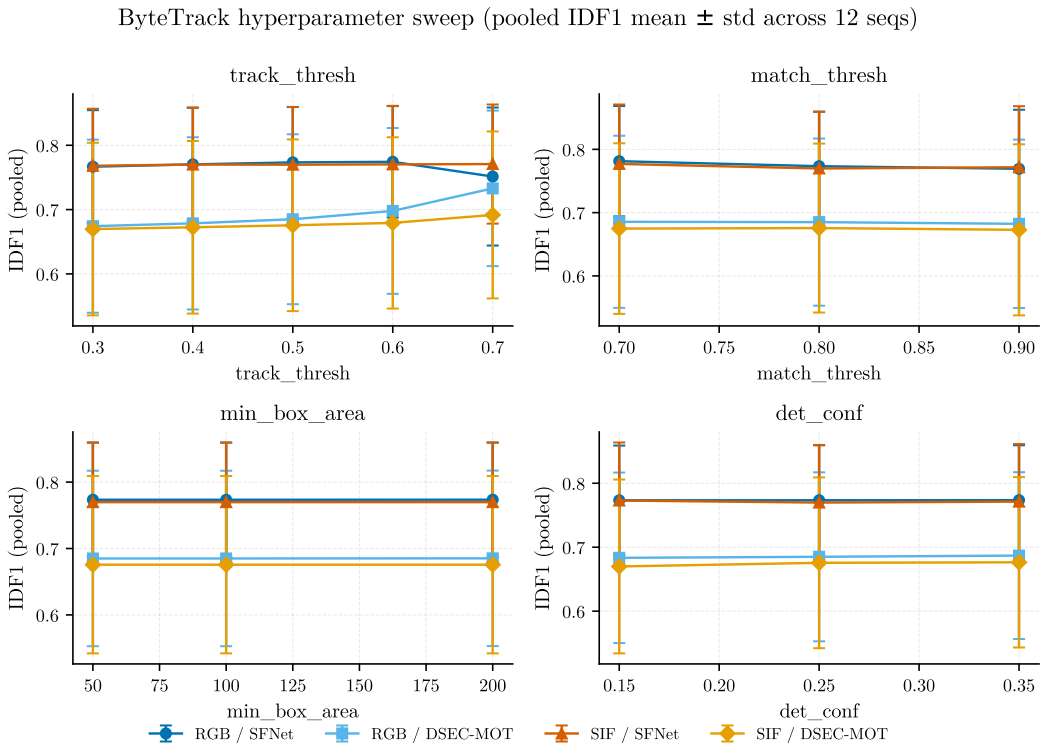

Figure 7.1: Pooled ByteTrack sweep across 12 sequences, varying one axis at a time from $(0.5, 0.8, 100, 0.25)$. The SFNet-labelled pairs select $\tau_{\text{match}} = 0.7$; the DSEC-MOT-labelled pairs select $\tau_{\text{track}} = 0.7$. A train-only reaggregation selects the same settings.

configurations are evaluated on 12 sequences, giving 168 sequence-level runs per pair and 672 across all four pairs. The pooled aggregation selects $\tau_{\text{match}} = 0.7$ for both SFNet-labelled pairs and $\tau_{\text{track}} = 0.7$ for both DSEC-MOT-labelled pairs, with the remaining values at their defaults.

Because the pooled aggregation includes the two designated test sequences, the original selection is not strictly held out. Reaggregating the existing per-sequence results over the 10 training sequences alone selects the same four operating points. The reported predictions and test metrics are therefore unchanged under either aggregation. Track buffer is fixed at 30 frames and tracking NMS IoU at 0.45. Figure 7.1 summarises the pooled sweep.

Table 7.6: Held-out same-detector evaluation on `zurich_city_01_d`, a sequence excluded from field fitting and policy selection. One authorised run, no tuning.

Policy	HOTA	DetA	AssA	IDF1	MOTA
RGB ByteTrack	44.857	38.786	52.397	57.741	41.307
Event-motion adapter	**46.615**	38.661	**56.818**	**60.315**	40.966
Δ	+1.757	−0.125	+4.421	+2.574	−0.341

Table 7.7: Same-detector two-sequence comparison. These sequences were exposed during development; this is not held-out evidence.

Policy	HOTA	DetA	AssA	IDF1	MOTA
RGB ByteTrack	56.907	50.579	64.664	69.942	57.515
Event-motion adapter	57.123	50.569	65.154	70.495	57.588
Association-memory extension	56.819	50.567	64.488	69.853	57.581

7.6 Detector-Matched CResMOT Implementation Evidence

The event-motion adapter is the strongest same-detector CResMOT result. It uses a persistent causal event field to produce a direct box correction and covariance before association. It is evaluated twice: once on a held-out sequence that no development pass had seen, and once on the two designated sequences, which were exposed during development. The association-memory extension adds a compact per-track memory after this motion interface.

On the held-out sequence the adapter gains 1.757 HOTA, 4.421 AssA, and 2.574 IDF1 over matched RGB ByteTrack, with zero event-only lifecycle violations, at a cost of 0.341 MOTA. This is the cleanest measurement of the event pathway in this thesis: one authorised run, no tuning, and a sequence held out of both field fitting and policy selection. Table 7.7 then reports the same policy on the two designated sequences.

There the adapter exceeds the matched RGB row by 0.216 HOTA, 0.489 AssA, and 0.553 IDF1. The smaller margin reflects the surface: `interlaken_00_d` begins at AssA 84.303 for RGB-only and supplies half the comparison, so little association error remains there for any method to repair. The association-memory extension changes assignments and returns 0.304

HOTA, 0.665 AssA, and 0.642 IDF1 relative to the adapter, which places the later interface rather than the memory itself. Neither table isolates event order or exercises the complete method.

A second evaluation uses a calibrated 10-sequence fitting substrate. Its stock ByteTrack baseline is 73.281 HOTA, 76.284 DetA, 70.833 AssA, 83.941 IDF1, and 87.529 MOTA. A persistent identity-slot scorer carries one per-track RWKV state per RGB-created slot. Cached event observations update that state before the RGB boundary; accepted RGB assignments correct the same state afterward. Events cannot create or remove slots. After one complete fitting pass, this model changes HOTA by +0.031, AssA by +0.060, IDF1 by +0.061, and MOTA by +0.039. This is a complete causal rollout; its margin did not meet the threshold set for a held-out run.

Two reweighted training variants attempted to repair the severe action imbalance in that scorer. Both preserved existing bindings well (98–99%) and recovered 14–21% of required rebindings, without selecting a required new slot. Both were stopped at their ranking gates, before tracker rollout.

7.7 Late Association Headroom

The the fitting-set oracles provide the cleanest bound on the late interface because they use ground-truth information unavailable at deployment. All act only after ByteTrack has already formed its candidate surface.

Table 7.8: Non-deployable fitting-set oracle deltas on the calibrated substrate. These measure late association headroom.

Oracle intervention	ΔHOTA	ΔAssA	ΔIDF1	ΔMOTA
Bounded candidate edge	+0.327	+0.656	+0.658	+0.012
All labelled bounded edges	+0.346	+0.754	+0.752	−0.146
Hard identity gate	+0.897	+2.001	+2.050	−0.454

Even the hard identity gate remains below one HOTA point and reduces MOTA. The available margin after candidate formation is therefore small, whatever scores those edges. A census on a separate fitting artifact agrees, finding 174 re-rankable rows among 31,095 supervised high-stage decisions. The consequence for architecture is taken up in Chapter 8.

7.8 Frozen-EVA and Tracker Diagnostics

Frozen-EVA recurrence check. The direct recurrence diagnostic uses the same 927,102 accepted events in each intervention arm over the first three RGB intervals of `zurich_city_01_d`, with 12 aligned targets used to verify the observation alignment. The ordered recurrent state remains finite and changes in every interval. Reset, zero-state, timestamp-shuffled, spatially shifted, and matched-random inputs all change the resulting features relative to the ordered arm. This shows that the checkpoint is event-responsive and that the interventions reach the recurrence. It is not a predictive test: no readout was fitted and no target loss or IoU was measured.

Matched tracker causal diagnostic. The tracker-facing diagnostic uses the first 100 frames of `zurich_city_00_b`, containing 436 ground-truth rows and 64,807,006 source events. Seven conditions share the RGB detections, source event slices, selected patch queries, model checkpoints, and evaluator. Event proposals are disabled, and the queried patches are fixed from the detector boxes rather than from tracker state. The predeclared gate requires ordered EVA to exceed every control by at least 2.0 MOTA points without reducing HOTA or IDF1.

The gate fails. Ordered EVA, temporal shuffle, and history-only produce byte-identical

Table 7.9: Input-matched 100-frame frozen-EVA causal diagnostic on `zurich_city_00_b`. This is not a complete-sequence or CResMOT-versus-ByteTrack comparison.

Condition	HOTA	DetA	AssA	MOTA	IDF1
Ordered EVA	61.92	52.56	73.84	51.38	66.92
Reset each frame	62.61	52.79	75.13	50.69	67.01
Zero state	62.61	52.79	75.13	50.69	67.01
Temporal shuffle	61.92	52.56	73.84	51.38	66.92
Spatial shift	61.92	52.64	73.73	51.61	66.84
Random matched state	62.32	52.45	74.92	50.69	67.27
History only	61.92	52.56	73.84	51.38	66.92

prediction files; reset-each-frame and zero-state are also identical. Spatial shift is 0.23 MOTA points above ordered EVA, while the random matched state has the highest IDF1. Ordered EVA exceeds reset and zero state by only 0.69 MOTA points and has lower HOTA and IDF1. The recurrent state therefore reaches the tracker, but the readout extracts no selective advantage from event order, boundary state, or spatial placement.

The diagnostic covers 100 frames, one seed, one exposed sequence, and an earlier checkpoint. It carries no matched ByteTrack run and does not instantiate the complete method, so its scope is event responsiveness at the tracker boundary rather than tracking quality. The adapter evaluation in Section 7.6 supplies the complete-sequence detector-matched comparison that this diagnostic does not.

7.9 Summary

RGB + ByteTrack leads SpikeMOT on four of five headline metrics, under a comparison that also changes the detector. Within the event pathway, the arm acting before association gains on both the held-out sequence and the designated pair; arms acting after candidate formation do not, and ground-truth oracles at that point stay below one HOTA point. The frozen-EVA diagnostic isolates no benefit from event order. The complete tracker remains to be evaluated as one system.

Chapter 8

Discussion

Three questions must be kept apart: whether recurrent event code executes correctly, whether it changes a tracker, and whether that change improves MOT. Only the first is settled cleanly. The remainder of this chapter reads the second and third against where in the tracker the event state was allowed to act.

8.1 What the Results Establish

The event-motion adapter is the strongest same-detector result. On a held-out sequence its persistent event field improves HOTA by 1.757, AssA by 4.421, and IDF1 by 2.574 over matched RGB ByteTrack; on the two designated sequences, which were exposed during development, the same policy gains 0.216 HOTA. Both are single-seed. The difference between them tracks how much association error each surface leaves available.

Memory placed after that interface adds nothing to it. The association-memory extension loses 0.304 HOTA relative to the adapter. On the separate 10-sequence substrate, the identity-slot scorer produces a complete causal rollout but gains only 0.031 HOTA, and its reweighted variants fail their action-ranking gates. The fitting-set oracles then bound the interface itself: ground truth applied to already formed candidate edges gains at most 0.897 HOTA and reduces MOTA. These results locate the placement rather than the memory.

Recurrent track memory as a family remains open.

8.2 Frozen EVA and Attribution

The frozen EVA-derived checkpoint loads exactly, remains finite, and carries state across event chunks. In the tracker-facing diagnostic, ordered events, temporal shuffle, and history-only input produce identical predictions. The checkpoint is recurrent and event-responsive, but the tested readout does not isolate useful event order. Neither that diagnostic nor the adapter proves the gain is caused by ordered recurrence. The method chapters use EVA as a representation substrate; the epoch checkpoint and the later compact fields are different artifacts and must not be merged into one trained model.

8.3 Where Recurrent State Must Enter

The adapter acts before association and gains. The later scorers act after candidate formation and do not, and ground truth at that same point is similarly limited. The next architecture should therefore use one shared all-class mechanism in the following order:

$$\begin{aligned} \text{causal event state} &\rightarrow \text{persistent per-track RGB–event state} \\ &\rightarrow \text{pre-association box, covariance, and continuity} \\ &\rightarrow \text{candidate gating and low-score recovery} \\ &\rightarrow \text{joint RGB–event compatibility} \\ &\rightarrow \text{global association and RGB correction.} \end{aligned} \tag{8.1}$$

The state must change prediction, uncertainty, and candidate reachability before it contributes to assignment. With detector proposals fixed after non-maximum suppression, the expected gains are in AssA and IDF1. A material DetA or MOTA change would require event evidence to act on the proposals.

8.4 Threats to Validity

Four threats bound the claims above. Protocol mixing: the two-sequence and calibrated substrates are calibrated differently and their absolute values are not comparable. Sequence exposure: the two-sequence result is retrospective and the calibrated-substrate evidence is fitting-only. Attribution: the adapter lacks a complete event-off, memory-reset, and temporal intervention set on the same rollout. Scope: the Neural-ODE propagation, ROI-local fusion, RWKV-7 observation jump, and joint association core of Chapters 4 and 5 were never trained and evaluated together.

8.5 Future Work

The immediate work is a compact early-integration implementation that keeps persistent per-track state and changes pre-association motion and candidate construction before global matching. Its first comparison must use one detector cache and one protocol for RGB-only, the event-motion adapter, and the new system. FEMOT remains a useful external benchmark, but changing datasets does not repair an interface; DSEC should first show that the revised system is executable and improves on its same-substrate controls.

Chapter 9

Conclusion

This thesis investigated CResMOT, an RGB–event tracking architecture in which local event-derived state persists across RGB intervals and updates tracking before the next image arrives. Chapters 4 and 5 specify a complete tracker with recurrent event encoding, track-local fusion, continuous per-track propagation, and an RWKV-7-style correction. That system remains to be trained as one architecture, and the implementation work established what it must do first.

Implementation rather than design supplied the main finding. On a held-out sequence the event-motion adapter improves HOTA by 1.757, AssA by 4.421, and IDF1 by 2.574 over matched RGB ByteTrack, altering predicted motion and covariance before association. On the two designated sequences the same policy gains 0.216 HOTA, against a surface already at AssA 84.303 on half its extent. Every later system read per-track state after candidate formation, and none improved on the adapter; fitting-set oracles at that point stay below one HOTA point even with ground-truth identity. What these experiments establish is the depth at which event state must enter the tracker.

That determines how the design should be integrated. Its components were built as separable stages, and the late stages sit past the point where event evidence still has room to act. A revised implementation must run one shared mechanism: causal event state feeds

persistent per-track state, which sets pre-association box, covariance, and continuity, which governs candidate gating and low-score recovery before global assignment. Memory entering at the final cost matrix arrives after that room has closed.

FEMOT is the surface on which that design should be tested. Its 100 sequences and 14,580 trajectories span day and night, indoor and outdoor, and static and moving cameras. Its aligned recordings remove the co-registration assumption qualifying every DSEC result here, and its attribute labels make the prediction of Chapter 1 directly testable: event conditioning should help most under fast motion and low illumination.

FEMOTR sharpens that test by contrast. It builds an event image over each exposure interval and advances its track memory at frame times, so every stage runs on the RGB clock. CResMOT asks whether between-frame structure carries information that schedule discards. Running both under one detection regime would separate event *content* from event *timing*, which neither the FEMOT release nor this thesis has measured.

What is new here should be stated plainly against what is borrowed. The frozen detector, the event encoder, the ByteTrack shell, and the RWKV and Neural-ODE operators are all established, and this thesis integrates rather than invents them. What is new is the per-track continuous-time formulation of Chapters 4 and 5, the requirement that event-derived state persist across RGB intervals, and the measured result locating where that state must enter a tracking-by-detection pipeline. The held-out improvement demonstrates the third directly. The first is established as a formulation, with its inter-observation update, its structured observation write, and its interface to association and lifecycle each specified in full; what remains is to train and evaluate it as one system. An effectiveness claim for the complete design requires one same-substrate evaluation of the integrated early interface, followed by the FEMOT comparison above.

References

[1] G. Gallego et al., "Event-based vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 154–180, 2022. DOI: 10.1109/TPAMI.2020.3008413.

[2] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *IEEE International Conference on Image Processing (ICIP)*, 2016.

[3] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *IEEE International Conference on Image Processing (ICIP)*, 2017.

[4] Y. Zhang et al., "Bytetrack: Multi-object tracking by associating every detection box," in *European Conference on Computer Vision (ECCV)*, 2022.

[5] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[6] P. Sun et al., "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 2446–2454.

[7] H. Caesar et al., "Nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 621–11 631.

[8] A. S. Bhadoriya, V. Vegamoor, and S. Rathinam, "Vehicle detection and tracking using thermal cameras in adverse visibility conditions," *Sensors*, vol. 22, no. 12, p. 4567, 2022. DOI: 10.3390/s22124567.

[9] S. M. Patole, M. Torlak, D. Wang, and M. Ali, "Automotive radars: A review of signal processing techniques," *IEEE Signal Processing Magazine*, vol. 34, no. 2, pp. 22–35, 2017.

[10] M. Gehrig, W. Aarents, D. Gehrig, and D. Scaramuzza, "Dsec: A stereo event camera dataset for driving scenarios," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4947–4954, 2021. DOI: 10.1109/LRA.2021.3068942.

[11] S. Wang, Z. Wang, C. Li, X. Qi, and H. K.-H. So, "SpikeMOT: Event-based multi-object tracking with sparse motion features," *IEEE Access*, vol. 13, pp. 214–230, 2025, Originally arXiv:2309.16987 (2023). DOI: 10.1109/ACCESS.2024.3523411.

[12] Z. Liu, N. Yang, Y. Wang, Y. Yang, H. Zhu, and X. Ma, "Enhancing traffic object detection in variable illumination with rgb-event fusion," *arXiv preprint arXiv:2311.00436*, 2024, Author list [VERIFY] against paper v2 PDF. Published IEEE T-ITS 2024.

[13] S. Wang et al., "FEMOT: Multi-object tracking using frame and event cameras," *arXiv preprint arXiv:2606.14094*, 2026. DOI: 10.48550/arXiv.2606.14094. [Online]. Available: https://arxiv.org/abs/2606.14094.

[14] D. Gehrig, A. Loquercio, K. G. Derpanis, and D. Scaramuzza, "End-to-end learning of representations for asynchronous event-based data," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, Event Spike Tensor (EST) representation, 2019.

[15] M. Cannici, M. Ciccone, A. Romanoni, and M. Matteucci, "A differentiable recurrent surface for asynchronous event-based data," in *European Conference on Computer Vision (ECCV)*, Matrix-LSTM / memory surfaces, 2020.

[16] Y. Nam, M. Mostafavi, K.-J. Yoon, and J. Choi, "Stereo depth from events cameras: Concentrate and focus on the future," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Mixed-Density Event Stacks (MDES). [VERIFY] – exact paper introducing MDES may differ., 2022.

[17] A. Sironi, M. Brambilla, N. Bourdis, X. Lagorce, and R. Benosman, "HATS: Histograms of averaged time surfaces for robust event-based object classification," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[18] H. Hao, N. Zubić, W. He, Z. Sui, D. Scaramuzza, and W. Wang, "Maximizing asynchronicity in event-based neural networks," *arXiv preprint arXiv:2505.11165*, 2025, EVA event encoder.

[19] B. Peng et al., "Rwkv-7 "goose" with expressive dynamic state evolution," *arXiv preprint arXiv:2503.14456*, 2025.

[20] R. Pellerito, D. Gehrig, S. Shiba, and D. Scaramuzza, "Neural events: Discrete asynchronous autoencoders for event-based vision," *arXiv preprint arXiv:2606.19835*, 2026. DOI: 10.48550/arXiv.2606.19835. [Online]. Available: https://arxiv.org/abs/2606.19835.

[21] S. Schaefer, D. Gehrig, and D. Scaramuzza, "AEGNN: Asynchronous event-based graph neural networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[22] D. Gehrig and D. Scaramuzza, "Low-latency automotive vision with event cameras," *Nature*, vol. 629, no. 8014, pp. 1034–1040, 2024, The DAGR system. DOI: 10.1038/s41586-024-07409-w.

[23] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1–2, pp. 83–97, 1955.

[24] N. Aharon, R. Orfaig, and B.-Z. Bobrovsky, "BoT-SORT: Robust associations multi-pedestrian tracking," *arXiv preprint arXiv:2206.14651*, 2022.

[25] Y. Du et al., "StrongSORT: Make DeepSORT great again," *IEEE Transactions on Multimedia*, 2023, [VERIFY] exact volume/issue.

[26] C. Tang et al., "Revisiting color-event based tracking: A unified network, dataset, and metric," *arXiv preprint arXiv:2211.11010*, 2022, Original CEUTrack tracker and the COESOT benchmark. [VERIFY] full author list.

[27] Z. Zhu, J. Hou, and D. O. Wu, "Cross-modal orthogonal high-rank augmentation for RGB-event transformer-trackers," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 22 045–22 055.

[28] X. Hou et al., "SDSTrack: Self-distillation symmetric adapter learning for multi-modal visual object tracking," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 26 551–26 561.

[29] J. Huang, S. Wang, S. Wang, Z. Wu, X. Wang, and B. Jiang, "Mamba-FETrack: Frame-event tracking via state space model," in *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, arXiv:2404.18174, 2024.

[30] S. Wang et al., "Mamba-FETrack V2: Revisiting state space model for frame-event based visual object tracking," *arXiv preprint arXiv:2506.23783*, 2025.

[31] X. Wang, X. Lou, S. Wang, J. Huang, L. Chen, and B. Jiang, "Long-term visual object tracking with event cameras: An associative memory augmented tracker and a benchmark dataset (AMTTrack)," *arXiv preprint arXiv:2403.05839*, 2024, Introduces the FELT long-term frame–event SOT benchmark.

[32] H. Ramsauer et al., "Hopfield networks is all you need," in *International Conference on Learning Representations (ICLR)*, 2021.

[33] "TAPFormer: Robust arbitrary point tracking via transient asynchronous fusion of frames and events," *arXiv preprint arXiv:2603.04989*, 2026, [VERIFY] authors and exact title.

[34] O. Apps, "Asynchronous multi-object tracking with an event camera (AEMOT)," in *IEEE International Conference on Robotics and Automation (ICRA)*, arXiv:2505.08126. [VERIFY] full authors., 2025.

[35] Anonymous (withdrawn submission), "RGB-Event MOT: A cross-modal benchmark for multi-object tracking," *OpenReview*, 2024, Withdrawn ICLR 2024 submission; code and dataset not released. OpenReview id: Z1Em654CSE.

[36] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[37] Y. Rubanova, R. T. Q. Chen, and D. Duvenaud, "Latent ODEs for irregularly-sampled time series," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[38] E. De Brouwer, J. Simm, A. Arany, and Y. Moreau, "GRU-ODE-Bayes: Continuous modeling of sporadically-observed time series," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[39] P. Kidger, J. Morrill, J. Foster, and T. Lyons, "Neural controlled differential equations for irregular time series," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[40] Y. Chen, K. Ren, Y. Wang, Y. Fang, W. Sun, and D. Li, "ContiFormer: Continuous-time transformer for irregular time series modeling," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[41] N. Muca Cirone, A. Orvieto, B. Walker, C. Salvi, and T. Lyons, "Theoretical foundations of deep selective state-space models," in *Advances in Neural Information Processing Systems (NeurIPS)*, arXiv:2402.19047. Rough Path Theory framework for selective SSMs; [VERIFY] full author list vs. final camera-ready., 2024.

[42] A. Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces," in *International Conference on Learning Representations (ICLR)*, 2022.

[43] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*, 2023.

[44] A. Vaswani et al., "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[45] W. Merrill and A. Sabharwal, "The parallelism tradeoff: Limitations of log-precision transformers," *Transactions of the Association for Computational Linguistics*, vol. 11, pp. 531–545, 2023, Establishes that log-precision transformers can be simulated by uniform TC^0 threshold circuits. DOI: 10.1162/tacl_a_00562.

[46] W. Merrill, J. Petty, and A. Sabharwal, "The illusion of state in state-space models," in *International Conference on Machine Learning (ICML)*, Limits of diagonal-transition SSMs on state-tracking tasks., 2024.

[47] B. Peng et al., "RWKV: Reinventing RNNs for the transformer era," *arXiv preprint arXiv:2305.13048*, 2023, Findings of EMNLP 2023.

[48] B. Peng et al., "Eagle and finch: RWKV with matrix-valued states and dynamic recurrence," *arXiv preprint arXiv:2404.05892*, 2024, RWKV-5 (Eagle) and RWKV-6 (Finch).

[49] Y. Shi et al., "StreamingFlow: Streaming occupancy forecasting with asynchronous multi-modal data streams via neural ordinary differential equations," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Author list [VERIFY]., 2024.

[50] H. Liu et al., "TimeTracker: Event-based continuous point tracking for video frame interpolation with non-linear motion," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, arXiv:2505.03116., 2025, pp. 17 649–17 659.

[51] O. Liu, "Mamba4D: Efficient 4d point cloud video understanding with disentangled spatial-temporal state space models," *arXiv preprint arXiv:2405.14338*, 2024, [VERIFY] full author list.

[52] O. Xu, "Modeling continuous motion for 3d point cloud object tracking," *arXiv preprint arXiv:2303.07605*, 2023, [VERIFY] full author list.

[53] N. Zubić, D. Gehrig, M. Gehrig, and D. Scaramuzza, "From chaos comes order: Ordering event representations for object recognition and detection," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[54] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," *arXiv preprint arXiv:2107.08430*, 2021.

[55] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The CLEAR MOT metrics," *EURASIP Journal on Image and Video Processing*, vol. 2008, pp. 1–10, 2008.

[56] J. Luiten et al., "HOTA: A higher order metric for evaluating multi-object tracking," *International Journal of Computer Vision*, vol. 129, pp. 548–578, 2021. DOI: 10.1007/s11263-020-01375-2.

[57] J. Breiter et al., *py-motmetrics: The MOT metrics library*, https://github.com/cheind/py-motmetrics, 2016.

[58] J. Luiten and A. Hoffhues, *TrackEval*, https://github.com/JonathonLuiten/TrackEval, MOT evaluation library, 2020.

[59] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking," in *European Conference on Computer Vision Workshops (ECCVW)*, 2016.

[60] F. Zeng, B. Dong, Y. Zhang, T. Wang, X. Zhang, and Y. Wei, "MOTR: End-to-end multiple-object tracking with transformer," in *European Conference on Computer Vision (ECCV)*, 2022.

[61] Y. Zhang, T. Wang, and X. Zhang, "MOTRv2: Bootstrapping end-to-end multi-object tracking by pretrained object detectors," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[62] R. Gao and L. Wang, "MeMOTR: Long-term memory-augmented transformer for multi-object tracking," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[63] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations (ICLR)*, 2019.

[64] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.