Fast Survey: Relevancy of SSMs (Mamba-like) for Target Tracking

Literature context and positioning for an internal SSM-style neural filter toy study

Abstract

Selective state-space sequence models (SSMs), exemplified by Mamba, have renewed interest in tracking because they enable **streaming, linear-time** sequence inference under strict compute and latency constraints. In tracking pipelines, the most defensible role for such models is to provide a **learned motion prior** that improves prediction through occlusions, measurement gaps, or maneuvering regimes where simplistic priors (e.g., constant velocity) are mismatched. This survey briefly reviews representative baselines and adjacent learned-filter approaches, then positions the attached toy study as a controlled test of whether an SSM-inspired streaming block can learn turn-like priors and generalize to held-out trajectory compositions.

Date: January 12, 2026

1. Background and related work

1.1 Estimation under maneuvering and model mismatch

The interacting multiple model (IMM) filter formalizes maneuvering as switching dynamics and remains a standard baseline for abrupt-mode behavior; it provides a reference point for what strong performance can look like under mode changes and limited computational budget [1]. KalmanNet proposes a hybrid estimator that retains the Kalman filtering flow while learning, via a compact recurrent module, components associated with Kalman gain computation and mismatch handling; the authors report improved estimation under nonlinearities and model mismatch in their evaluated settings [2,3]. A key limitation is that performance is learned from data and therefore inherits training distribution constraints and any simulation-to-reality gap in the measurement and dynamics models [2,3].

1.2 Efficient sequence modeling as architectural inspiration (SSM/Mamba)

Mamba introduces selective state-space sequence modeling with linear-time scaling and hardware-aware recurrent inference; its relevance for tracking is the possibility of high-rate streaming inference under strict compute and latency constraints [4]. TrackingMamba illustrates how Mamba-like components can be packaged into real-time tracking pipelines in vision, emphasizing linear-growth compute; this is best interpreted as architectural inspiration for streaming models rather than evidence for sensor-grade probabilistic estimation, because sensing assumptions and evaluation regimes differ materially [5].

1.3 Positioning of the attached toy study

The attached internal report (2026-01-09) evaluates a narrow hypothesis consistent with the above literature: whether an SSM-inspired streaming block can learn motion priors (line plus turning primitives) and generalize to held-out combinations, with the primary stressors being intermittent measurements and mismatch between constant-velocity priors and turning motion. This is intentionally not framed as a general replacement for Kalman filtering, but as an exploration of when learned streaming priors may help and what failure modes (stability, hyperparameter sensitivity) emerge.

References

[1] H. A. P. Blom and Y. Bar-Shalom, "The Interacting Multiple Model Algorithm for Systems with Markovian Switching Coefficients," IEEE Transactions on Automatic Control, 33(8), 1988. DOI: 10.1109/9.1299. https://doi.org/10.1109/9.1299

[2] G. Revach, N. Shlezinger, R. J. G. van Sloun, and Y. C. Eldar, "KalmanNet: Data-Driven Kalman Filtering," ICASSP 2021. DOI: 10.1109/ICASSP39728.2021.9413750. https://doi.org/10.1109/ICASSP39728.2021.9413750

[3] G. Revach, N. Shlezinger, R. J. G. van Sloun, and Y. C. Eldar, "KalmanNet: Neural Network Aided Kalman Filtering for Partially Known Dynamics," arXiv:2107.10043, 2021. https://arxiv.org/abs/2107.10043

[4] A. Gu and T. Dao, "Mamba: Linear-Time Sequence Modeling with Selective State Spaces," arXiv:2312.00752, 2023. https://arxiv.org/abs/2312.00752

[5] Q. Wang et al., "TrackingMamba: Visual State Space Model for Object Tracking," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing (JSTARS), 2024. DOI: 10.1109/JSTARS.2024.3458938. https://doi.org/10.1109/JSTARS.2024.3458938

[6] H.-W. Huang et al., "MambaMOT: State-Space Model as Motion Predictor for Multi-Object Tracking," arXiv:2403.10826, 2024. https://arxiv.org/abs/2403.10826

SSM-Style Neural Filters for 2D Tracking: A Critical Toy Study

Internal exploration report - 2026-01-09

1. Goal and research question

This project was a hands-on exploration of whether state-space-model (SSM) inspired neural sequence models ("Mamba-like" streaming blocks) have practical potential for tracking problems. The focus was not to "beat Kalman" in general, but to test a more specific hypothesis: **can an SSM-style model learn motion priors from data (e.g., line and turning segments) and then generalize to new trajectories composed from those primitives, especially when measurements are unreliable or missing?**

2. Experimental setup

State: $x = [p_x, p_y, v_x, v_y]$ (4D). **Measurement:** $z = [p_x, p_y]$ (2D). **Time step:** dt = 0.1. All experiments are synthetic and 2D. Motion is generated by concatenating primitives: straight segments (constant heading) and constant-turn arcs (left/right) with configurable speed and turn rate.

Training data: random sequences of primitives (typically 4-5 segments), with random v and omega per sequence. **Generalization test:** evaluate on a held-out primitive pattern (unseen exact sequence) composed from the same primitive set. Measurement corruption modes were added progressively: Gaussian noise, outliers, sparse sampling, and forced block dropouts.

Figure 1 shows the typical "train on primitives, test on held-out combination" baseline (Gaussian noise only).

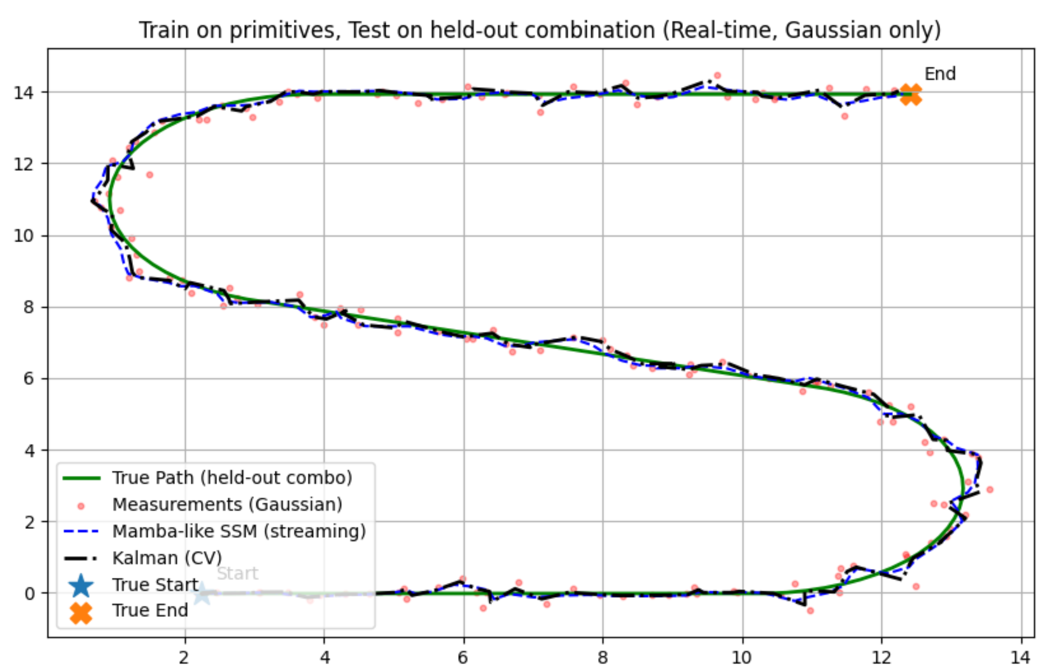

Figure 1. Train on motion primitives (line + arc), test on an unseen combination (Gaussian only).

3. Filters compared

3.1 Kalman baseline

A standard **linear Kalman filter** with a **constant-velocity (CV)** motion model. Prediction runs every frame; measurement update is applied only when a measurement exists. Process noise Q and measurement noise R are scalar-scaled identity matrices (simple tuning). This baseline is intentionally minimal; it is not IMM/CT/UKF.

3.2 Learned SSM-style streaming filter

A **streaming neural filter** built from two recurrent "Mamba-like" layers (simple gated state update) and trained end-to-end by backpropagation through time. To stay close to Kalman structure, the learned filter was implemented as:

Predict: start from current state x_t, apply CV propagation, and add a **learned residual dynamics** term (e.g., delta-v or a turn-rate proxy) that can run even when measurements are missing.
Update: when a measurement exists, compute innovation (z - predicted position) and apply a **learned correction** to the full state (similar role to Kalman gain, but learned and nonlinear).

Note: this is SSM-inspired (streaming state update), but it is not a faithful implementation of the full Mamba architecture with explicit A,B,C matrices. The intent was educational and pragmatic: keep real-time step() inference and test whether learned state dynamics + learned measurement update can help in tracking regimes where CV is mismatched.

4. Training objective and optimization

Training is supervised: run the filter over a sequence and minimize position error to the known ground truth. The main loss was a **weighted SmoothL1** on position:

L_pos = mean_t(w_t * SmoothL1(p_hat_t - p_t)), where w_t is increased during missing-measurement frames to force the model to learn to "carry" motion through gaps.

Additional small regularizers were used in some runs: (1) a velocity consistency term from finite differences, and (2) (when using a turn-rate proxy) a smoothness penalty encouraging stable omega during gaps. Optimization used Adam (typical lr ~ 3e-3) for ~800-1200 epochs on small CPU batches.

Important practical finding: the learned filter can become numerically unstable (NaNs) when the scenario is made too hard (large gaps + strong turns + high learning rate). Clamping/tanh on residual outputs and modest regularization were needed.

5. Key experiments and results

Scenario	Meas. density	Forced gap	RMSE total (Learned / KF)	RMSE gap (Learned / KF)	MaxErr gap (Learned / K
Gaussian only (primitives -> held-out combos)	dense	none	0.182 / 0.208	n/a	n/a
Test-only outliers (no retraining)	dense	none	1.091 / 1.002	n/a	n/a
Sparse + long gap inside turn (K=5)	30 / 180	inside turn	1.887 / 3.524	4.181 / 5.584	6.030 / 10.171
Sparse + long gap inside turn (K=2)	76 / 180	inside turn	0.767 / 3.090	1.781 / 6.149	2.259 / 10.584

Interpretation: in the "Gaussian only" regime the learned filter is competitive but does not clearly dominate a well-tuned Kalman CV. In "test-only outliers" (no robustness training), both degrade similarly. The clearest advantage appears in the **forced long measurement gap occurring during a turning maneuver**: the learned model can maintain a plausible turn prior, while Kalman CV extrapolates the wrong motion and diverges.

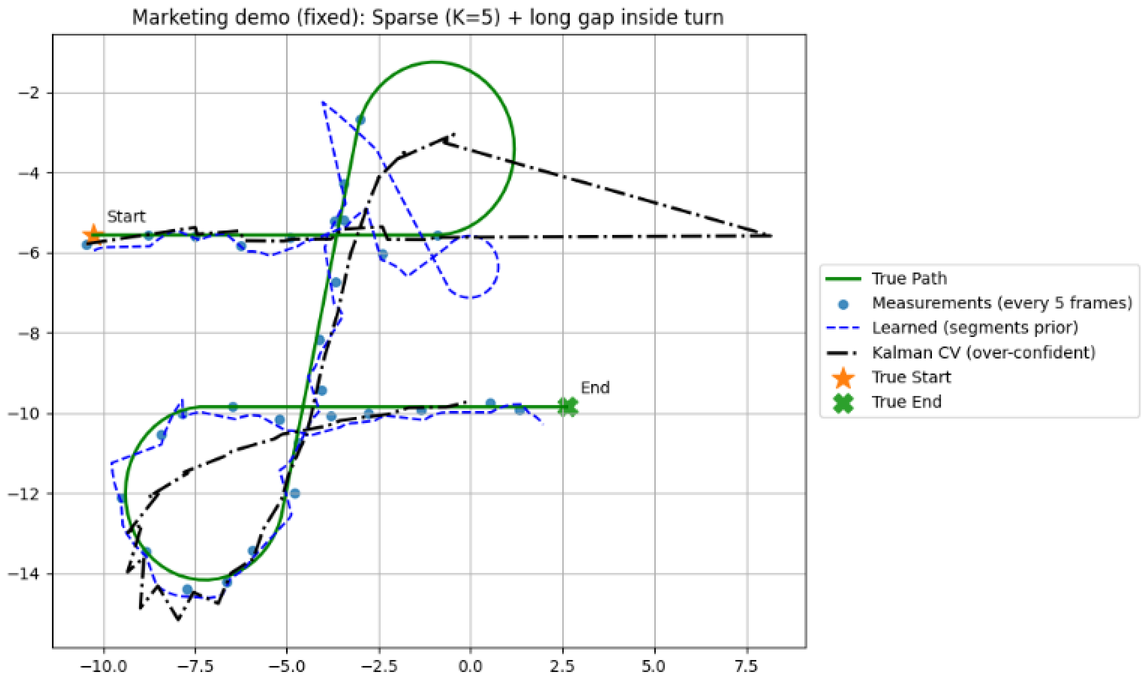

Figure 2. Sparse sampling (K=5) with a forced block dropout inside a turn.

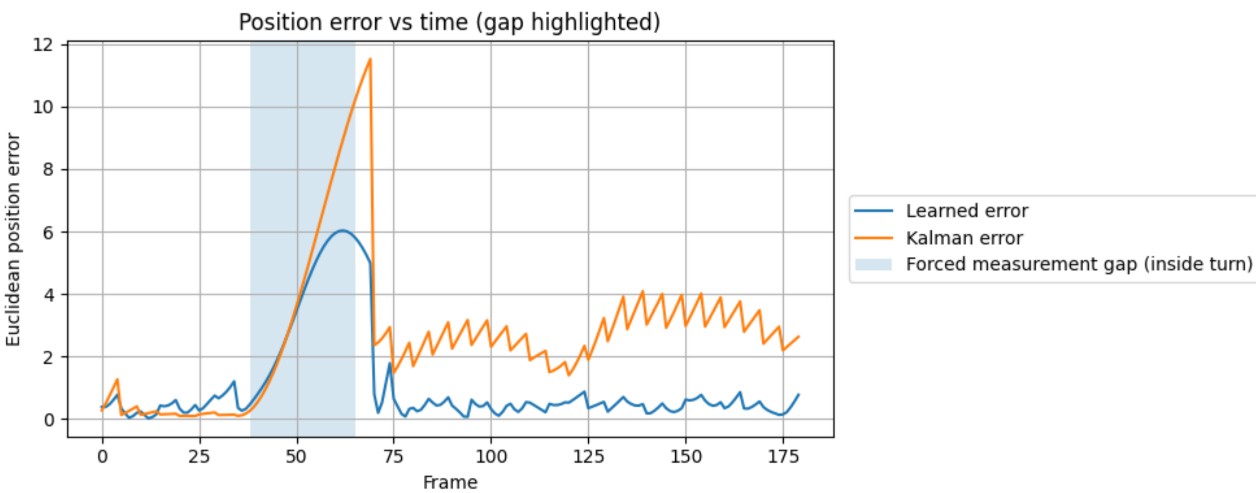

Figure 3. Position error vs time for Figure 2 (gap highlighted).

6. Focus result: K=2 (dense overall, but one long gap inside the turn)

With K=2 the tracker receives measurements frequently outside the forced gap. This makes the experiment visually clean: both methods look reasonable most of the time, and the gap inside a maneuver isolates the effect of the motion prior.

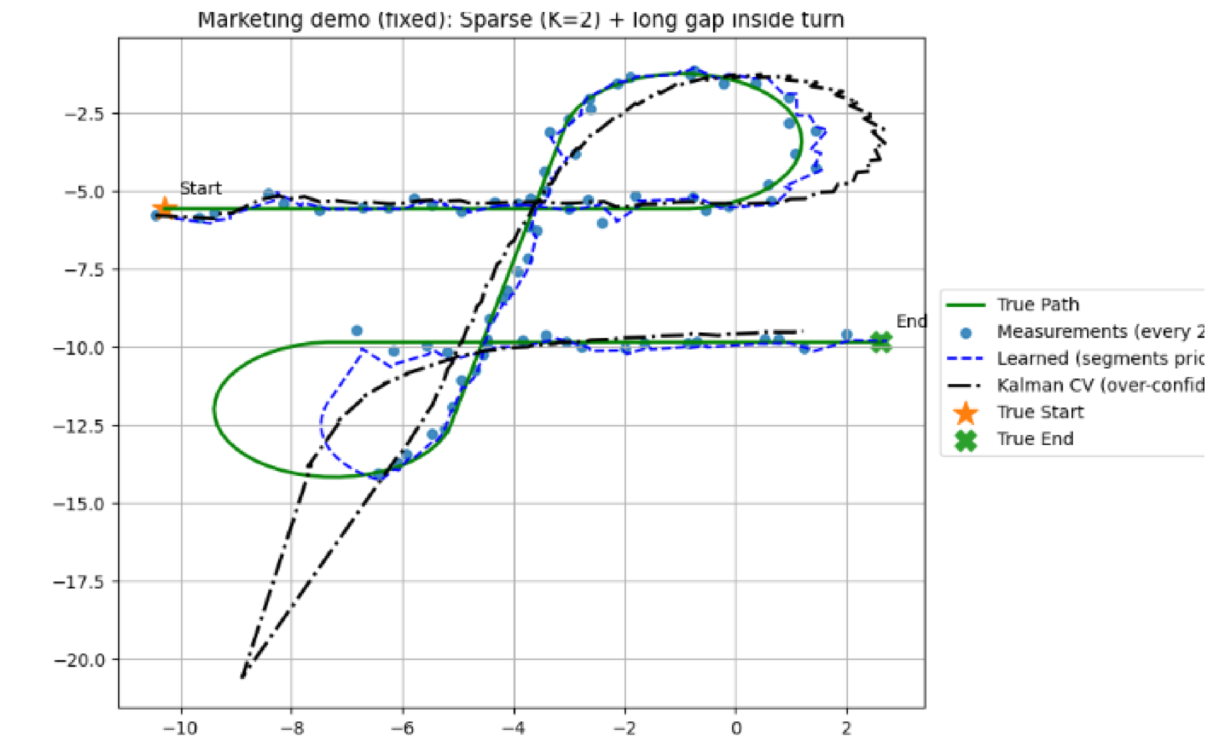

Figure 4. K=2 overall measurement rate, but a long forced gap inside the first turn.

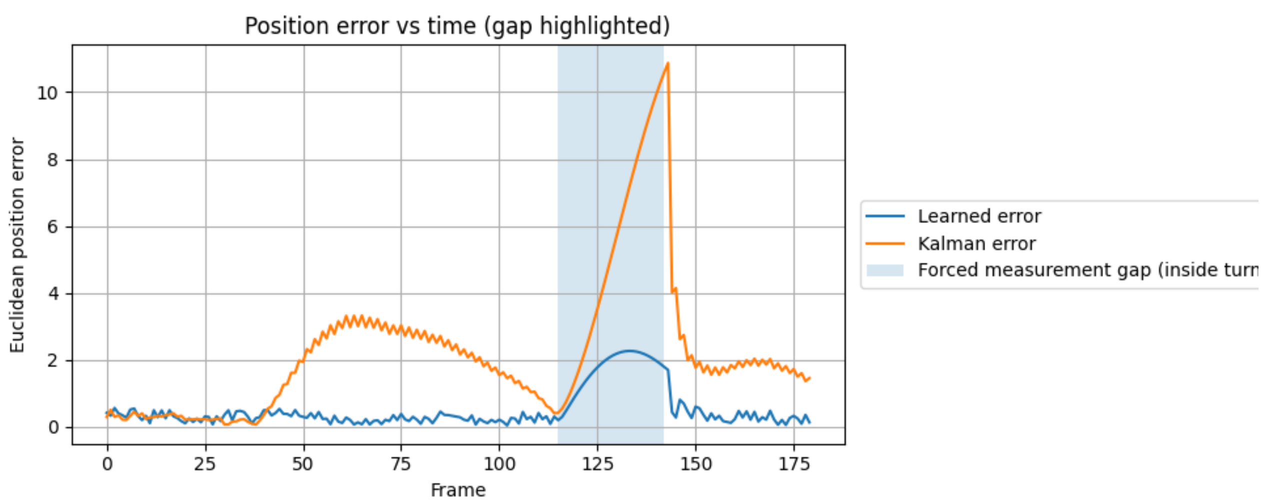

Figure 5. Error vs time for Figure 4: learned filter rises but stays bounded; Kalman CV spikes strongly.

7. Conclusions (critical and realistic)

Do SSM-style networks have potential for tracking? Yes, but the potential is situational. In this toy setting, the strongest evidence is that a learned streaming model can encode a **motion prior** (e.g., turning behavior) that helps during measurement gaps where a simplistic CV model is wrong. This aligns with recent MOT literature that replaces/augments Kalman motion prediction with learned SSM modules.

What did the learned model actually learn here? Primarily a nonlinear measurement update and a residual dynamics term (turn tendency) conditioned on recent history. It did not discover new physics; it learned a data-driven maneuver model from the primitive distribution.

Limitations: (1) The learned model is not a full Mamba implementation with explicit A,B,C matrices; it is an SSM-inspired recurrent block. (2) Results depend heavily on training distribution and hyperparameters; instability (NaNs) can occur without output clamping. (3) The Kalman baseline is intentionally simple; stronger Kalman variants (CT/IMM/adaptive Q) would likely close much of the gap. (4) This is a synthetic single-target 2D toy problem; conclusions do not directly transfer to real sensors without additional work.

Practical takeaway: SSM-style learned filters are best viewed as **learned motion models** that can plug into a classical pipeline (or operate alongside it), especially for regimes with model mismatch and intermittent observations. For well-modeled regimes with frequent measurements, a tuned Kalman filter remains hard to beat on simplicity, stability, and interpretability.

Reference (motivation only): arXiv:2403.10826v2 (MambaMOT) discusses using an SSM module as a motion predictor in multi-object tracking.