

Article

Multi-Horizon 3D Position Prediction for IoT-Enabled UAVs: A Sensor-Enriched LSTM Benchmark in AirSim

Mohammad Alja'afreh ^{1,*}  and Ali Karime ²¹ Department of Communications Engineering, Princess Sumaya University for Technology, Amman 11941, Jordan² Department of Electrical and Computer Engineering, Royal Military College of Canada, Kingston, ON K7K 7B4, Canada; ali.karime@rmc-cmr.ca

* Correspondence: m.aljaafreh@psut.edu.jo

Highlights

What are the main findings?

- The sensor-enriched LSTM (LSTM-PVAQ) achieved the lowest 3D RMSE at every evaluated horizon—0.043 m at 0.1 s, 0.168 m at 0.5 s, and 0.371 m at 1.0 s. At the 1 s horizon, it reduced RMSE by 21.7% versus LSTM-PV, 13.1% versus GRU-PVAQ, 9.3% versus TCN-PVAQ, 16.8% versus Transformer-PVAQ, and 25.4% versus EKF-CA; all paired flight-level differences remained significant after Holm adjustment ($p < 0.001$).
- The cumulative feature ablation reduced 1 s RMSE from 0.612 m with position alone to 0.371 m with the full PVAQ input, with sequential conditional gains from velocity, acceleration treatment, and quaternion orientation. LSTM-PVAQ required 92.6 K parameters and 0.88 ms median batch-1 neural forward-pass time on a Raspberry Pi 5 CPU, while TCN-PVAQ provided a faster 0.54 ms model-inference alternative with a modest accuracy penalty.

What are the implications of the main finding?

- For the evaluated AirSim domain, incorporating inertial and attitude histories—especially gravity-resolved acceleration and quaternion orientation—improved short-horizon 3D prediction relative to position–velocity input. This result supports further evaluation of sensor-enriched predictive state estimation on physical UAVs rather than establishing universal real-world superiority.
- The measured accuracy–latency trade-off shows that the neural forward pass fits within the evaluated 50 Hz model-update interval: LSTM-PVAQ favors accuracy, whereas TCN-PVAQ offers lower model-inference latency. Collision-risk monitoring, route maintenance, link-quality forecasting, handover preparation, and prediction-assisted control are potential downstream applications that were not directly evaluated in this study.



Academic Editors: Salvatore Rosario Bassolillo, Vito Antonio Nardi and Giuseppe Tricomi

Received: 30 July 2026

Revised: 2 September 2026

Accepted: 4 September 2026

Published: 8 September 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

Abstract

Reliable short-term position forecasting may provide anticipatory state information for collision-risk assessment, communication management, and prediction-assisted control in Internet of Things (IoT)-enabled unmanned aerial vehicles (UAVs); these downstream functions are not evaluated directly here. This study reformulates UAV position prediction as a flight-wise, multi-horizon, three-dimensional forecasting problem and tests whether position, velocity, gravity-resolved acceleration, and quaternion-orientation histories improve predictive accuracy while measuring model-level edge-inference cost rather than end-to-end system latency. The dataset contains 3100 AirSim flights with high-rate kinematic, inertial, attitude, pressure, and magnetic-field measurements under variable horizontal wind. The reported generalization is flight-disjoint within one AirSim domain;

route/scenario disjointness and transfer to physical UAVs are not established. Signals are converted to a common navigation frame, gravity-resolved, low-pass filtered, resampled to 50 Hz, and partitioned by flight identifier before normalization and window construction. Each learned model receives 2 s of history and predicts the complete next 1 s trajectory, with errors evaluated at 0.1, 0.5, and 1.0 s. The sensor-enriched LSTM (LSTM-PVAQ) is compared under matched conditions with persistence, constant-velocity, constant-acceleration, extended Kalman filter, reduced-feature LSTM, GRU, temporal convolutional network (TCN), and compact Transformer baselines. LSTM-PVAQ achieved 3D RMSE values of 0.043, 0.168, and 0.371 m at 0.1, 0.5, and 1.0 s, respectively. At 1 s, its RMSE was 21.7% lower than LSTM-PV, 13.1% lower than GRU-PVAQ, 9.3% lower than TCN-PVAQ, and 16.8% lower than Transformer-PVAQ. Its one-second ADE and FDE were 0.216 and 0.339 m. On a Raspberry Pi 5 CPU using one FP32 thread and batch size one, median neural forward-pass latency was 0.88 ms, well below the 20 ms model-update interval. The results show that gravity-resolved inertial and orientation histories improve multi-horizon prediction, while TCN-PVAQ remains an attractive lower-latency alternative.

Keywords: unmanned aerial vehicle; multi-horizon trajectory prediction; three-dimensional position forecasting; long short-term memory; inertial measurement unit; AirSim; edge inference; Internet of Things

1. Introduction

Unmanned aerial vehicles (UAVs) increasingly operate as mobile sensing, computation, and communication nodes within Internet of Things (IoT) ecosystems. Their mobility enables adaptive coverage, rapid deployment, and access to remote or hazardous environments, but it also creates strict requirements for localization, motion anticipation, and timely decision-making [1–5]. Industry assessments likewise reflect the broadening commercial role of UAV platforms across sensing and monitoring applications [6]. A reliable short-horizon position predictor may provide an anticipatory state estimate to downstream collision monitoring, route maintenance, link-quality forecasting, handover preparation, or control modules. These are potential uses of the forecast rather than functions evaluated by the present benchmark.

UAV motion prediction is more difficult than road-constrained vehicle prediction. A multirotor can translate and rotate in three dimensions, respond rapidly to control inputs, and experience wind, sensor noise, and actuator disturbances. Surveys of future-location prediction and motion-risk assessment emphasize the importance of matching the predictor to the motion constraints and uncertainty structure of the platform [7,8]. Wind-rejection studies further show that disturbances can alter both state evolution and controller response [9,10]. Kinematic models and Kalman-filter variants remain attractive because they are interpretable and efficient, but their accuracy depends on process-model fidelity, covariance tuning, and noise assumptions [11–14]. Data-driven sequence models instead learn temporal relations directly from historical telemetry, potentially capturing residual dynamics that are difficult to specify analytically.

Long short-term memory (LSTM) networks are widely used for sequential prediction because gated memory mitigates vanishing gradients and preserves relevant temporal information [15]. Gated recurrent units provide a related encoder–decoder mechanism with fewer gates [16,17]. UAV studies have used recurrent networks to forecast ADS-B trajectories, predict local positions, and exploit velocity-enriched states [18–20]. However, the literature also includes technically different tasks such as full trajectory generation,

obstacle-motion forecasting, disturbance estimation, closed-loop navigation, action generation, and spoofing classification [21–27]. These tasks use distinct targets, forecast horizons, inputs, and evaluation units and are therefore treated separately in this benchmark.

This work formulates UAV position forecasting as a flight-wise, multi-horizon, three-dimensional benchmark. A 2 s history predicts the complete next 1 s trajectory, the dataset channels and duration statistics are reported consistently, and analytical, filtering, recurrent, convolutional, and attention-based baselines are evaluated under the same protocol.

The novelty is therefore methodological and empirical rather than a claim of a new recurrent cell: the contribution is the controlled sensor-enrichment study, common multi-horizon benchmark, flight-level inference, and edge-inference characterization under one protocol. The term “IoT-enabled UAV” denotes the intended deployment context in which an onboard or companion computer consumes flight-controller telemetry and can expose predicted states to local autonomy or networked services; the present experiments do not evaluate a radio link, handover algorithm, or IoT network stack.

The three reported horizons are application-oriented checkpoints within a single continuous 1 s forecast, rather than three separately trained prediction tasks. The 0.1 s point (five 50 Hz updates) tests immediate state-continuity prediction; 0.5 s (25 updates) represents a sub-second look-ahead relevant to detect-and-avoid and rapid replanning, for which UAV safety systems have used approximately 0.5 s trajectory-update intervals [28]; and 1.0 s (50 updates) represents a longer proactive interval relevant to planning and communication management, where recent UAV beam-management work explicitly evaluates 0.5–1.0 s prediction scales [29]. At the corpus’s nominal 3 m/s operating speed, these checkpoints correspond to approximately 0.3, 1.5, and 3.0 m of nominal travel. This range therefore spans immediate extrapolation through a practically meaningful one-second look-ahead while remaining distinct from long-horizon route generation.

Practical significance is interpreted separately from statistical significance. Millimeter-scale differences at 0.1 s are useful mainly as a near-term consistency check and should not be over-interpreted. At 1 s, however, the observed differences reach centimeters to decimeters (e.g., 0.103 m relative to LSTM-PV and 0.038 m relative to TCN-PVAQ), a scale that could matter to threshold-based collision-risk, route-maintenance, beam-management, or handover decisions if the predictor is integrated into such systems. These differences are not themselves certified safety margins; downstream operational benefit must be validated at system level.

Figure 1 summarizes the motivation and scope of the benchmark. It links the role of IoT-enabled UAVs and the need for near-future prediction to the challenges of free-flight motion, the sensor-enriched input representation, the comparator families, and the edge-oriented evaluation criteria.

The research question is: Does adding gravity-resolved acceleration and orientation history to a compact LSTM improve flight-wise, multi-horizon 3D UAV position prediction relative to simpler kinematic predictors and matched neural sequence models, while satisfying the study’s edge-inference latency constraint? Four hypotheses are evaluated:

- H1—feature value: LSTM-PVAQ yields lower per-flight 1 s 3D RMSE than LSTM-PV under the same split and training protocol.
- H2—model value: LSTM-PVAQ outperforms persistence, constant-velocity, constant-acceleration, and EKF baselines.
- H3—comparative value: any advantage over GRU, TCN, and Transformer baselines remains after paired flight-level uncertainty analysis.
- H4—deployability: median batch-1 inference latency remains below the 20 ms model-update interval on the evaluated edge platform.

Here, “edge-inference latency constraint” has a specific and limited meaning: the median batch-1 forward-pass time of the prediction model on the evaluated Raspberry Pi 5 CPU must be below the 20 ms interval between 50 Hz model updates. It is a model-computation budget, not an end-to-end control-loop or network-latency requirement; sensor acquisition, preprocessing, operating-system scheduling, communication, actuation, and tail latency are outside this timing measurement.

Motivation and Scope of Multi-Horizon 3D UAV Position Prediction

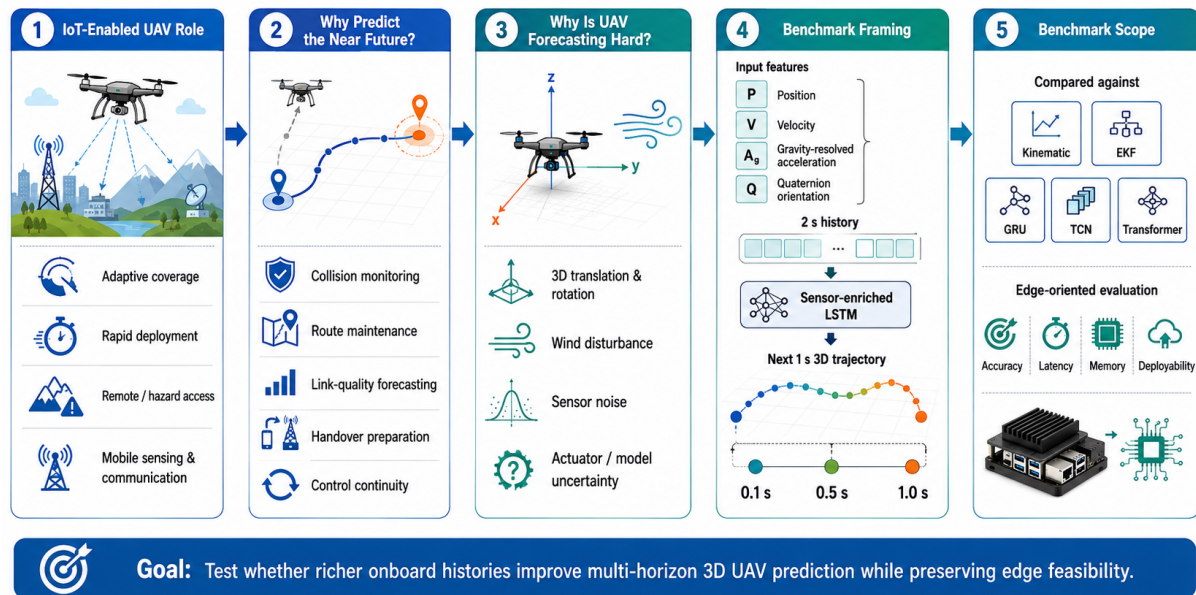


Figure 1. Motivation and scope of the multi-horizon 3D UAV position-prediction benchmark, including IoT-enabled use cases, forecasting challenges, the PVAQ input representation, comparator families, and edge-oriented evaluation criteria.

The contributions are:

1. a direct multi-output formulation that predicts the complete next 1 s 3D trajectory at 50 Hz from 2 s of history;
2. a consistent coordinate-frame and gravity-resolution procedure that distinguishes raw specific force from navigation-frame acceleration;
3. a controlled feature ablation from position-only through position–velocity–acceleration–quaternion inputs;
4. a matched benchmark spanning analytical, filtering, recurrent, convolutional, and attention-based models;
5. a flight-level statistical protocol with hierarchical bootstrap intervals and paired comparisons; and
6. an edge-inference evaluation combining accuracy, model size, latency, and memory measurements, with IoT/networking uses treated as downstream deployment scenarios rather than experimentally validated network functions.

The remainder of this paper is organized into a more compact structure. Section 2 reviews related work and clarifies task boundaries. Section 3 describes the AirSim corpus and preprocessing. Section 4 presents the prediction formulation, LSTM architecture, benchmark models, training diagnostics, and feature-ablation protocol. Section 5 defines accuracy, statistical, and edge-computing evaluation. Section 6 reports the results, Section 7 discusses practical interpretation and generalizability, and Section 8 concludes the paper.

2. Related Work and Task Boundaries

Figure 2 provides the organizing map for this section. The central block is the supervised ego-UAV position-forecasting problem evaluated in this study. Dashed connections denote technically related tasks whose targets and metrics are not directly comparable, whereas solid arrows show how a future-position sequence can serve as an input to planning, navigation, and control. Table 1 summarizes these task boundaries; the following subsections distinguish state prediction, cross-domain foundations, sequence models, and adjacent UAV tasks.

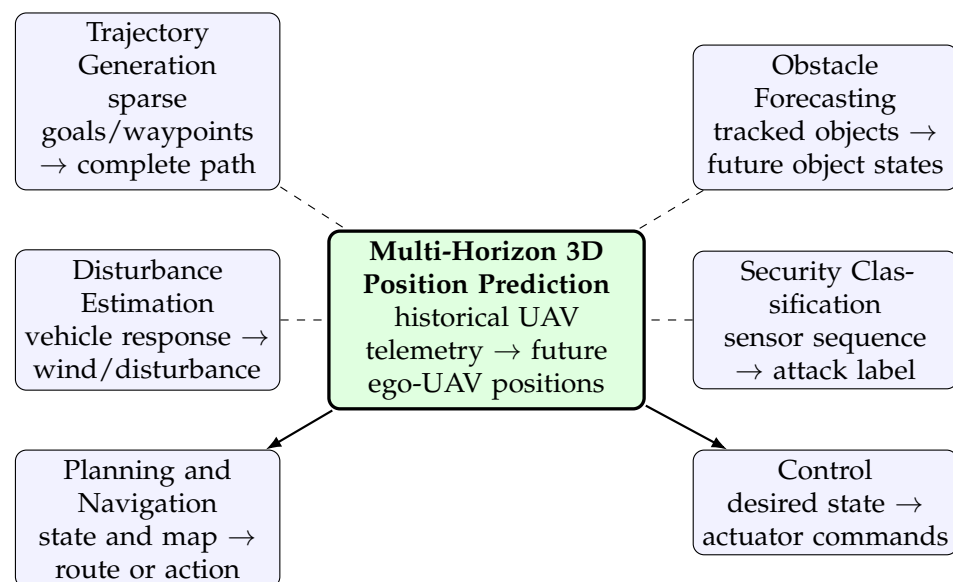


Figure 2. Task boundaries used in this article. Solid arrows indicate possible downstream use of position forecasts; dashed connections indicate related but technically distinct learning tasks.

2.1. Traditional and Learned UAV State Prediction

Traditional UAV prediction uses kinematic and dynamic models, Kalman filters, Gaussian processes, graph-based methods, and aerodynamic models [21]. Constant-velocity and constant-acceleration predictors are strong short-horizon baselines for smooth trajectories. Extended and unscented Kalman filters incorporate uncertainty and recursive state correction, but performance depends on the appropriateness of the model and noise assumptions [12,13]. Learned residuals and covariance adaptation can improve filters, although they introduce additional stability and calibration questions [14].

Zhang et al. combined ADS-B position histories with a recurrent LSTM for two-step 3D UAV trajectory prediction [18]. VECTOR enriched a GRU with velocity to improve real-time 3D forecasting under maneuvers and wind [20]. Jia et al. addressed a different problem: generation of complete 5–25 s trajectories from sparse waypoints using segmented LSTMs and real Crazyflie data [22]. These studies support recurrent modeling but also show that prediction horizon, output representation, and data source strongly influence model rankings.

Recent UAV trajectory-prediction studies further show that conclusions depend strongly on data provenance, maneuver diversity, sensing modality, and validation design. Hashemi et al. compared recurrent variants on 1820 simulated UAS-S4 trajectories and found that recurrent architecture choice affects prediction stability across horizons [30]. Yoon et al. evaluated a GRU-based framework on urban micro-UAV trajectory data and emphasized rapid directional changes and generalization as central challenges [31]. Ruseno and Lin moved beyond offline simulation by demonstrating real-time GRU trajectory prediction during DJI Phantom 4 flight tests using Broadcast Remote ID telemetry [32]. Tang

et al. evaluated a learning-based interacting-multiple-model predictor across 17 multirotor trajectory datasets and explicitly tested shifted trajectory distributions [33]. More recent studies have also examined multi-step GRU forecasting for autonomous site assessment [34] and short-term UAV prediction across different flight states, velocity levels, sampling frequencies, and prediction horizons [35]. Together, these works broaden the evidence base from single-regime recurrent prediction toward real-flight, multi-dataset, maneuver-aware, and distribution-shift evaluation, while also underscoring that performance measured in one simulator or operating envelope should not be assumed to transfer unchanged to other UAV platforms or flight regimes.

The distinction from these studies is not the use of an LSTM by itself. Zhang et al. forecast from ADS-B position histories, while VECTOR explicitly demonstrates the value of adding velocity to a recurrent predictor. Sensor-rich UAV work such as Shankar et al. uses inertial, attitude, and control telemetry to estimate wind rather than future ego-UAV position. Within the supporting literature considered here, the narrower unresolved question is whether gravity-resolved acceleration and quaternion attitude add measurable value to direct multi-output 3D ego-UAV forecasting when the data split, history/target, training objective, and comparator protocol are controlled. The present contribution addresses that question through a cumulative PVAQ ablation, matched recurrent/convolutional/attention baselines, flight-level uncertainty analysis, and model-level edge-inference measurements. It does not claim a novel LSTM cell or that inertial/attitude sensing itself is new.

2.2. Cross-Domain Foundations for Trajectory Prediction

Trajectory forecasting developed through location mining, trajectory-pattern discovery, and road-network clustering. Significant-location and pattern-mining methods predict future movement from recurring GPS traces, while road-network methods identify repeated spatial structures across users and routes [36–39]. More recent work uses geographically clustered deep estimators, convolutional models for pedestrian motion, recurrent models for vehicle trajectories, and vehicle tracks extracted from UAV imagery [40–43]. Interaction-aware graph-attention and Transformer models extend these ideas to multi-agent dependencies and social constraints [44–46]. These studies provide important methodological foundations, but their ground-motion assumptions and evaluation horizons do not directly represent free-flight UAV dynamics.

2.3. Alternative Neural Sequence Models

GRUs reduce the number of gates relative to LSTMs and often provide favorable accuracy–latency trade-offs [16,17,20]. Temporal convolutional networks (TCNs) use causal dilated convolutions and residual blocks to model long receptive fields with parallel computation [47]. Transformers use self-attention to capture long-range dependencies but can require more memory and computation, particularly for long input sequences [48]. Cross-domain benchmarking also indicates that recurrent and attention-based rankings depend on sequence length, noise, and the underlying data regime [49]. Accordingly, the benchmark uses the same history, target, features, training split, search budget, and direct multi-output objective for each architecture.

2.4. Adjacent but Non-Equivalent UAV Tasks

R-YOLO integrates object detection and recurrent forecasting to predict external obstacle motion near vertiports [23]; its target is not the ego-UAV state. Shankar et al. estimate wind from onboard telemetry and use the estimate in semi-autonomous control [24]; this is disturbance estimation. Layered-RQN learns collision-avoidance and target-acquisition actions in a dynamic environment [50]; this is closed-loop navigation. Dhiman et al. predict short-term PAV displacements and intervene using depth thresholds [25]; the system is eval-

uated by navigation success and collision rate. UAV-VLA predicts language-conditioned action chunks [26], while Sarode and Naresh classify GPS–IMU spoofing states [27]. These distinctions prevent invalid cross-task metric comparisons.

Table 1. Task boundaries within the supporting UAV/UAM literature.

Work	Task	Output	Relevance to This Study
Zhang et al. [18]	Ego-UAV trajectory prediction	Two future 3D points	Recurrent prediction from ADS-B position histories
Nacar et al. [20]	Ego-UAV trajectory prediction	Future 3D positions	Demonstrates the value of velocity-enriched recurrent input
Hashemi et al. [30]	UAS trajectory prediction	Future trajectory states	Compares LSTM variants on a dedicated UAS-S4 simulation corpus
Yoon et al. [31]	Ego-UAV trajectory prediction	Future UAV positions	Recent GRU benchmark emphasizing real-time accuracy and generalization
Ruseno and Lin [32]	Real-time UAV trajectory prediction	Future position from Remote ID telemetry	Physical DJI flight-test evidence and communication-latency context
Jia et al. [22]	Complete trajectory generation	5–25 s spatiotemporal path	Longer-horizon generation from sparse waypoints
Ywet et al. [23]	Obstacle forecasting	Future external-object tracks	Integrates detection and prediction, but not ego-state forecasting
Shankar et al. [24]	Disturbance estimation/control	Wind estimate and control aid	Motivates use of inertial and attitude evidence
Guo et al. [50]	Navigation policy learning	Discrete motion action	Closed-loop DRL, not supervised position prediction
Dhiman et al. [25]	PAV navigation/control	3D displacement and maneuver	Evaluates mission success and collision outcomes
Gu [26]	Semantic action generation	Velocity action chunks/tokens	Multimodal transformer for language-conditioned navigation
Sarode and Naresh [27]	Security classification	Five attack/normal classes	Temporal classification, not motion forecasting
This study	Ego-UAV multi-horizon prediction	50 future 3D displacements	Controlled sensor ablation and edge benchmark

3. Dataset and Preprocessing

3.1. AirSim Flight Corpus

The dataset contains 3100 AirSim flight sequences over a mountainous environment. Raw logging was reported at 1 kHz, with a nominal speed near 3 m/s and independently varied horizontal wind components in $[-7, 7]$ m/s. The dataset includes 20 scalar channels: global latitude, longitude, and altitude; local Cartesian position; linear velocity; acceleration or specific force; quaternion orientation; barometric pressure; and three-axis magnetic field. The primary PVAQ predictor uses 13 channels: local position, velocity, gravity-resolved acceleration, and quaternion orientation (Table 2).

The analyzed corpus totals 136.83 h across 3100 flights, corresponding to a mean duration of 158.90 s per flight. All duration statistics are computed from the same post-cleaning flight manifest.

Simulation provenance and flight-generation metadata. The archived dataset/manuscript package establishes the AirSim multirotor domain, one mountainous environment, flight identifiers, the nominal speed regime, and the horizontal-wind envelope, but it does not retain the exact vehicle preset, controller gains/control mode, waypoint or route-generation script, start-state seed policy, or route-family labels used when the 3100 flights were generated. AirSim itself supports multirotor position/velocity/path commands and configurable wind in the NED world frame [51]. Because these run-specific generation records are not recoverable from the current submission archive, we do not infer or invent them. This is now treated as a reproducibility limitation. In particular, the 70/15/15 split is strictly flight-ID-disjoint, but the absence of route/scenario labels means that route- or maneuver-family disjointness cannot be verified. Consequently, reported test

performance should be interpreted as unseen-flight, within-domain generalization rather than evidence of route-disjoint, map-disjoint, or vehicle-disjoint transfer.

The wind range is a controlled simulation stress envelope rather than a regulatory or hardware-limit claim. Component-wise horizontal wind up to 7 m/s is more than twice the nominal 3 m/s translational speed, so the disturbance is intentionally non-negligible relative to vehicle motion and can induce drift and controller compensation. The realized speed distribution in Figure 3 is therefore reported alongside a representative held-out horizontal position trace, allowing the reader to inspect the actual motion regime rather than infer it only from command settings. The present archived benchmark outputs were not stratified by commanded wind, so the study does not claim a causal wind–error response curve; wind variation is treated as a controlled source of operating-condition variability. The resulting corpus remains biased toward smooth flight near 3 m/s; this limitation is retained explicitly rather than treating the wind sweep as equivalent to broad maneuver diversity.

Table 2. Channel manifest for the AirSim dataset.

Group	Scalar Channels	Count	Primary Use
Global position	latitude, longitude, altitude	3	Raw geodetic metadata
Local position	x, y, z	3	Predictor history and target
Linear velocity	v_x, v_y, v_z	3	Predictor history
Acceleration/specific force	a_x, a_y, a_z	3	Predictor after frame conversion and gravity resolution
Orientation quaternion	q_w, q_x, q_y, q_z	4	Predictor after unit normalization and sign continuity
Barometer	pressure	1	Not used in the primary predictor
Magnetometer	m_x, m_y, m_z	3	Not used in the primary predictor
Total		20	13 predictors and 7 excluded raw channels

The 3100 sequences provide controlled within-domain heterogeneity: flight identifiers and wind realizations vary, while the simulator family, mountainous environment, sensing channels, preprocessing, and nominal speed regime are held consistent. Thus, the benchmark tests generalization to unseen flights from the same simulated domain, not external-domain generalization to a different map, vehicle, sensor suite, or aggressive flight regime. Table 3 makes this distinction explicit.

Table 3. Controlled consistency and heterogeneity of the AirSim corpus. The table distinguishes factors that vary across flights from factors that remain fixed within the benchmark domain.

Factor	Corpus Characterization	Interpretation for Generalization
Number of sequences	3100 flight identifiers	Supports flight-disjoint evaluation rather than random window splitting
Environment	One mountainous AirSim environment	Controls scene/domain variation; does not establish unseen-map generalization
Speed regime	Nominally near 3 m/s; realized magnitudes shown in Figure 3	Represents primarily smooth low-speed multirotor motion
Horizontal wind	$w_x, w_y \in [-7, 7]$ m/s varied independently	Introduces controlled disturbance diversity relative to the nominal motion scale
Sensors	Common 20-channel logged telemetry; 13-channel PVAQ subset used by the primary model	Holds the sensing interface consistent across flights
Partition	2170/465/465 train/validation/test flights	Flight-ID-disjoint; route/scenario-disjointness cannot be verified from retained metadata

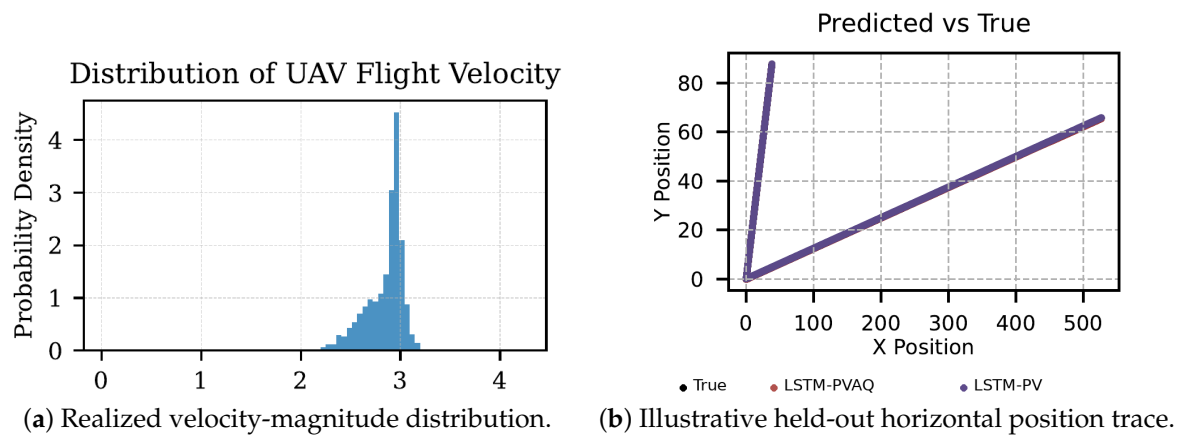


Figure 3. Dataset and test-trajectory characterization. Panel (a) shows the realized UAV velocity magnitudes across the 3100 flights rather than only the nominal command speed. Panel (b) shows an illustrative held-out x - y trajectory; the black curve is the true position trace, and the prediction curves are labeled explicitly as LSTM-PVAQ and LSTM-PV. The retained plotting artifact does not preserve the flight identifier or a predefined rule by which this trace was selected. Accordingly, it is presented only as an illustrative held-out example, not as a statistically selected “representative” case. The geometry provides an actual position curve for context and is consistent with the smooth-flight bias of the evaluated corpus.

3.2. Acceleration and Coordinate-Frame Processing

The mean acceleration magnitude near 9.81 m/s^2 indicates that the simulated IMU signal includes gravity. The acceleration channels are transformed to the navigation frame and gravity-resolved as

$$a_t^n = R(q_t)f_t^b + g^n, \quad (1)$$

where f_t^b is body-frame specific force, $R(q_t)$ rotates body to navigation coordinates, and g^n is the gravity vector. Quaternion sequences are normalized and made sign-continuous by flipping q_t when $q_t^\top q_{t-1} < 0$.

Because the IMU is simulated, its fidelity is characterized by a stochastic sensor model rather than by a single physical “accuracy” tolerance. AirSim constructs accelerometer and gyroscope outputs from simulator kinematics and adds white-noise and bias-drift processes [52,53]. In the documented default configuration, AngularRandomWalk is $0.3 \text{ deg}/\sqrt{\text{h}}$ (internally converted to $\text{rad}/\sqrt{\text{s}}$), GyroBiasStabilityTau is 500 s, GyroBiasStability is 4.6 deg/h (converted to rad/s), VelocityRandomWalk is 0.24 mg (converted internally using standard gravity), AccelBiasStabilityTau is 800 s, and AccelBiasStability is $36 \text{ }\mu\text{g}$ (converted to m/s^2) [53]. These are stochastic noise/bias-model parameters rather than a single meter- or degree-level hardware accuracy bound. The manuscript source package does not contain a run-specific `settings.json`; therefore, the documented AirSim model is reported as the reproducibility reference and any archived configuration override should supersede these defaults. This distinction avoids presenting an unverified hardware-equivalent accuracy figure.

Signal provenance and real-UAV interpretation. AirSim distinguishes simulator ground-truth kinematics, multirotor estimated kinematics, and explicit sensor outputs. Its API documentation states that multirotor state includes estimated position, orientation, linear/angular velocity, and acceleration; with the SimpleFlight firmware, estimated and ground-truth kinematics can coincide because no separate state estimator is modeled, whereas PX4 provides estimated kinematics [51]. The archived flight files used here contain local position, linear velocity, quaternion attitude, and gravity-bearing acceleration/specific-force channels together with simulated barometer and magnetometer channels. However, the retained archive does not record the exact AirSim API call or logging path that pop-

ulated the archived position, velocity, acceleration, and quaternion fields. We therefore cannot determine retrospectively whether each of these channels came from ground-truth kinematics, estimated multirotor state, or an explicit simulated sensor output. The expression “state/IMU-like telemetry” is descriptive of the archived variables and is not intended to imply a more specific AirSim signal source. The benchmark should consequently be interpreted as forecasting from simulator-generated telemetry, not as demonstrating the accuracy attainable from raw GNSS and IMU measurements on a physical aircraft. In a real deployment, analogous p , v , and q inputs would normally be supplied by the flight controller/state estimator, while acceleration would come from the calibrated inertial stream or estimator; the estimator uncertainty then becomes an additional source of error not captured by the present AirSim benchmark.

3.3. Sampling, Flight-Wise Splitting, and Normalization

The raw 1 kHz signals are low-pass filtered before decimation to 50 Hz. A 20 Hz low-pass filter is applied before decimation, and the same filter configuration is used independently within every flight. Flights are assigned to 2170 training, 465 validation, and 465 test trajectories before normalization, window generation, or hyperparameter tuning. Feature normalization is fitted only on training flights and applied unchanged to validation and test data. Windows never cross flight boundaries and advance by 10 samples (0.2 s), reducing near-duplicate examples.

The retained preprocessing record specifies the 20 Hz cutoff and the flight-wise filtering/resampling order but does not preserve the filter family, order, coefficients, or whether the original implementation used a causal one-pass filter or a forward–backward offline filter. We therefore no longer imply that this archival anti-aliasing stage has itself been demonstrated online. A deployment implementation must use a causal streaming filter (or a flight-controller/state-estimator output already bandwidth-limited to the required rate), preserve only past filter state, report its group delay, and then apply the fixed training-set normalization and rolling 100-sample history. The present Raspberry Pi timing excludes these operations (Figure 4; Table 4).

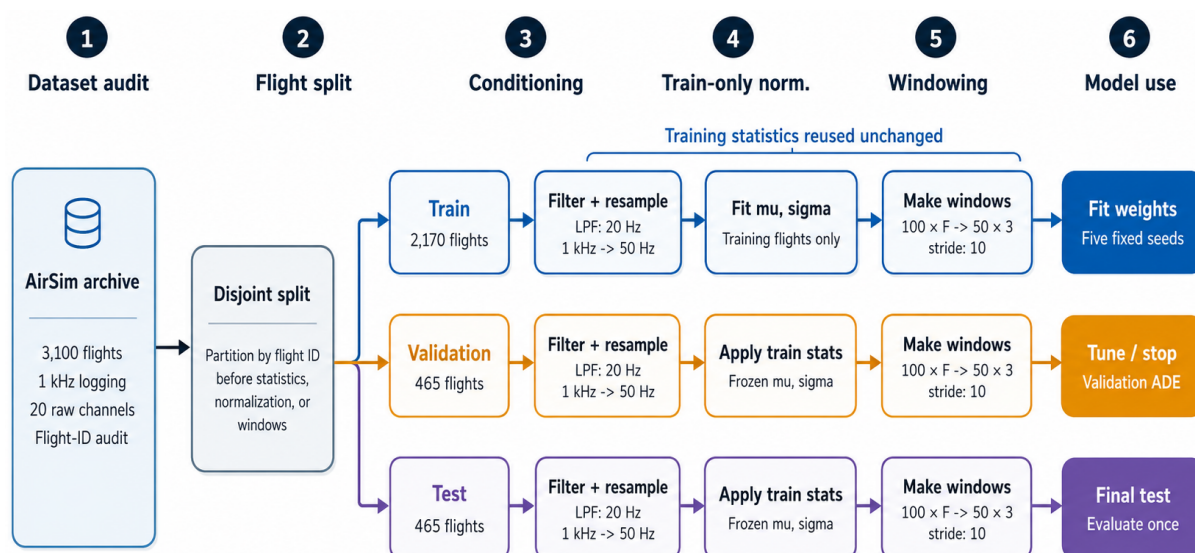


Figure 4. Flight-wise dataset preparation and preprocessing pipeline. Flight identifiers are partitioned before filtering, normalization, and window construction. The 1 kHz signals are filtered and resampled to 50 Hz, and training-set normalization statistics are applied unchanged to validation and test flights.

Table 4. Preprocessing and window-construction protocol.

Stage	Setting	Purpose
Native logging	1000 Hz	High-resolution source telemetry
Low-pass filter	20 Hz cutoff	Anti-aliasing before decimation; archived filter type/order/phase not retained
Model rate	50 Hz	20 ms update interval
Input history	2.0 s (100 steps)	Captures short-term motion trends
Forecast	1.0 s (50 positions)	Direct multi-output 3D trajectory
Reported horizons	0.1, 0.5, 1.0 s	Near-term continuity, sub-second detect/avoid, and one-second proactive look-ahead (5, 25, and 50 output steps)
Window stride	10 samples (0.2 s)	Reduces redundant overlap within each flight
Split	2170/465/465 flights	Flight-disjoint 70/15/15 partition
Normalization	Training statistics only	Consistent scaling across all partitions

The flight-ID partition is created before normalization and window construction, preventing window leakage across partitions. However, because the retained manifest has no route/waypoint-family or maneuver-regime labels, it cannot establish scenario-level independence among the three partitions. A future reproducibility release should include route/scenario identifiers and report an additional group-disjoint evaluation when those labels are available.

4. Prediction Models and Benchmark Protocol

4.1. Multi-Horizon 3D Formulation and LSTM

Let $\mathbf{p}_t, \mathbf{v}_t, \mathbf{a}_t \in \mathbb{R}^3$ denote local position, velocity, and gravity-resolved acceleration, and let $\mathbf{q}_t \in \mathbb{R}^4$ denote the unit quaternion. Historical positions are expressed relative to the final observed position:

$$\mathbf{s}_k = [(\mathbf{p}_k - \mathbf{p}_t), \mathbf{v}_k, \mathbf{a}_k, \mathbf{q}_k] \in \mathbb{R}^{13}, \quad k = t - T + 1, \dots, t. \quad (2)$$

The input and target are

$$\mathbf{X}_t = [\mathbf{s}_{t-T+1}, \dots, \mathbf{s}_t] \in \mathbb{R}^{T \times F}, \quad T = 100, \quad (3)$$

$$\mathbf{Y}_t = [\mathbf{p}_{t+1} - \mathbf{p}_t, \dots, \mathbf{p}_{t+H} - \mathbf{p}_t] \in \mathbb{R}^{H \times 3}, \quad H = 50. \quad (4)$$

The decoder predicts 150 displacement values and reshapes them into 50 future three-dimensional positions, yielding a direct and operationally meaningful multi-horizon forecast.

An LSTM updates its gates and memory as

$$\mathbf{f}_t = \sigma(\mathbf{W}_f[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f), \quad (5)$$

$$\mathbf{i}_t = \sigma(\mathbf{W}_i[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i), \quad (6)$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o), \quad (7)$$

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_c[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_c), \quad (8)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t, \quad (9)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t). \quad (10)$$

The final representation passes through an explicit dropout layer and a linear decoder $128 \rightarrow 150$. Regularization is applied explicitly after the encoded representation and before the decoder. The compact configuration is summarized in Table 5, and Figure 5 summarizes the complete input-to-forecast data flow and the evaluated 0.1, 0.5, and 1.0 s horizons.

Table 5. Compact LSTM configurations.

Component	Setting	Implementation Details
Input	$100 \times F$	$F = 6$ for LSTM-PV; $F = 13$ for LSTM-PVAQ
Encoder	One-layer LSTM, hidden size 128	Final hidden state
Regularization	Explicit dropout $p = 0.20$	Applied after the encoder
Decoder	Linear $128 \rightarrow 150$	Reshaped to 50×3 displacements
Loss	Mean squared 3D displacement	Normalized training and metric-space evaluation
Optimizer	Adam, initial 10^{-3}	Selected using validation ADE
Early stopping	Patience 15, maximum 200 epochs	Validation ADE criterion
Seeds	Five fixed seeds	Repeated training runs

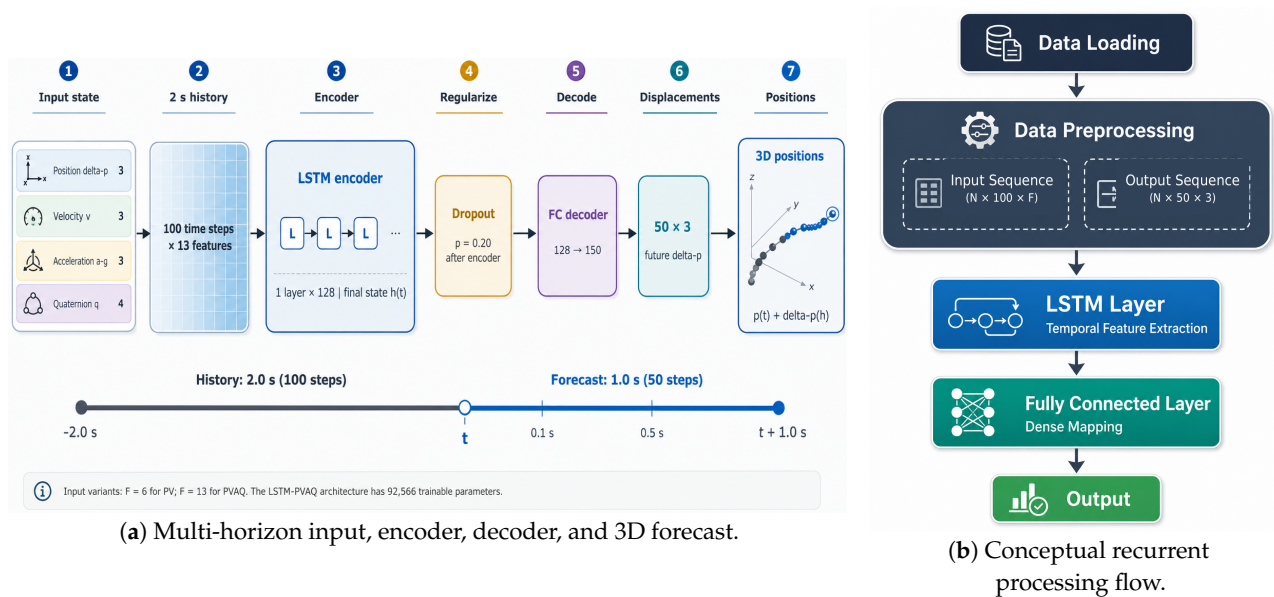


Figure 5. Sensor-enriched LSTM prediction architecture. The model consumes 2 s of history (100 steps) using either PV ($F = 6$) or PVAQ ($F = 13$) inputs, applies explicit post-encoder dropout, and directly decodes 50 future three-dimensional displacements spanning 1 s. The LSTM-PVAQ configuration contains 92,566 trainable parameters.

4.2. Benchmark Models, Training Fairness, and Diagnostics

The benchmark includes models that test distinct explanations for performance: no learned dynamics, simple kinematics, recursive filtering, gated recurrence, temporal convolution, and self-attention.

For the analytical baselines, persistence uses $\hat{p}_{t+\tau} = p_t$, constant velocity uses $\hat{p}_{t+\tau} = p_t + v_t\tau$, and constant acceleration uses $\hat{p}_{t+\tau} = p_t + v_t\tau + \frac{1}{2}a_t\tau^2$. The EKF-CA uses a nine-state $[p, v, a]$ constant-acceleration state representation with recursive prediction/correction. The current archive, however, does not retain the numerical process/measurement covariance matrices or their tuning record. Those values are required to reproduce the exact EKF numbers and are therefore explicitly identified below as missing implementation metadata rather than reconstructed after the fact.

The three simple analytical baselines are included as assumption tests rather than as competitive neural architectures. Persistence asks whether any motion model is needed at all; constant velocity (CV) asks whether the locally smooth trajectories can be explained by first-order kinematics; and constant acceleration (CA) asks whether the most recent acceleration is sufficient to extrapolate the trajectory. These assumptions are intentionally compatible with portions of the simulated corpus because the realized speed distribution is concentrated near 3 m/s and many segments are smooth. Their value is precisely to

determine how much of the apparent prediction problem can be solved without learning. Their progressively larger error with forecast horizon, reported later in Section 6, then quantifies where the smooth-motion assumptions begin to fail. Table 6 summarizes the benchmark roles.

Table 6. Benchmark models and their scientific roles.

Model	Implementation	Input	Scientific Role
Persistence	$\hat{p}_{t+h} = p_t$	Position	Persistence reference
Constant velocity	Analytical	Position and velocity	Smooth-flight analytical baseline
Constant acceleration	Analytical	Position, velocity, acceleration	Analytical acceleration baseline
EKF-CA	Nine-state recursive filter	Matched kinematic observations	Classical estimator/predictor
LSTM-PV	One-layer, 128 hidden units	Position and velocity (6)	Reduced-feature recurrent ablation
LSTM-PVAQ	One-layer, 128 hidden units	Position, velocity, acceleration, quaternion (13)	Sensor-enriched predictor
GRU-PVAQ	One-layer, matched budget	Same 13 features	Lower-gate-count recurrent comparison
TCN-PVAQ	Causal dilated residual blocks	Same 13 features	Parallel temporal model with long receptive field
Transformer-PVAQ	Compact encoder	Same 13 features and positional encoding	Attention-based comparison

All models use the identical flight split, filtered 50 Hz signals, 2 s history, 1 s target, stride, normalization, and test windows. Learned models predict all 50 future positions directly. Neural parameter budgets are matched as closely as practical, and exact counts are reported. Each model receives the same validation-search budget and seed set. Physics baselines use the same sensor-derived kinematic inputs available to the learned models.

For model selection, fairness was defined prospectively by common information access and common evaluation rules rather than by forcing identical internal architectures. Candidate learned configurations were fitted only on the 2170 training flights, ranked by validation ADE on the 465 validation flights, and evaluated on the 465 test flights only after selection. The same 2 s history, 1 s direct 50-step target, PVAQ channels (for the PVAQ models), normalization, loss family, five fixed training seeds, and validation endpoint were used across the learned comparators; early stopping used the validation criterion without test-set access. Parameter counts were kept in the same compact order of magnitude (74.3–118.5 K). However, because the exact architecture-specific search ranges and selected GRU/TCN/Transformer configuration files are not retained in the submission archive, the manuscript does not claim exhaustive or bitwise-reproducible hyperparameter equivalence. This limitation is reported explicitly in Table 7.

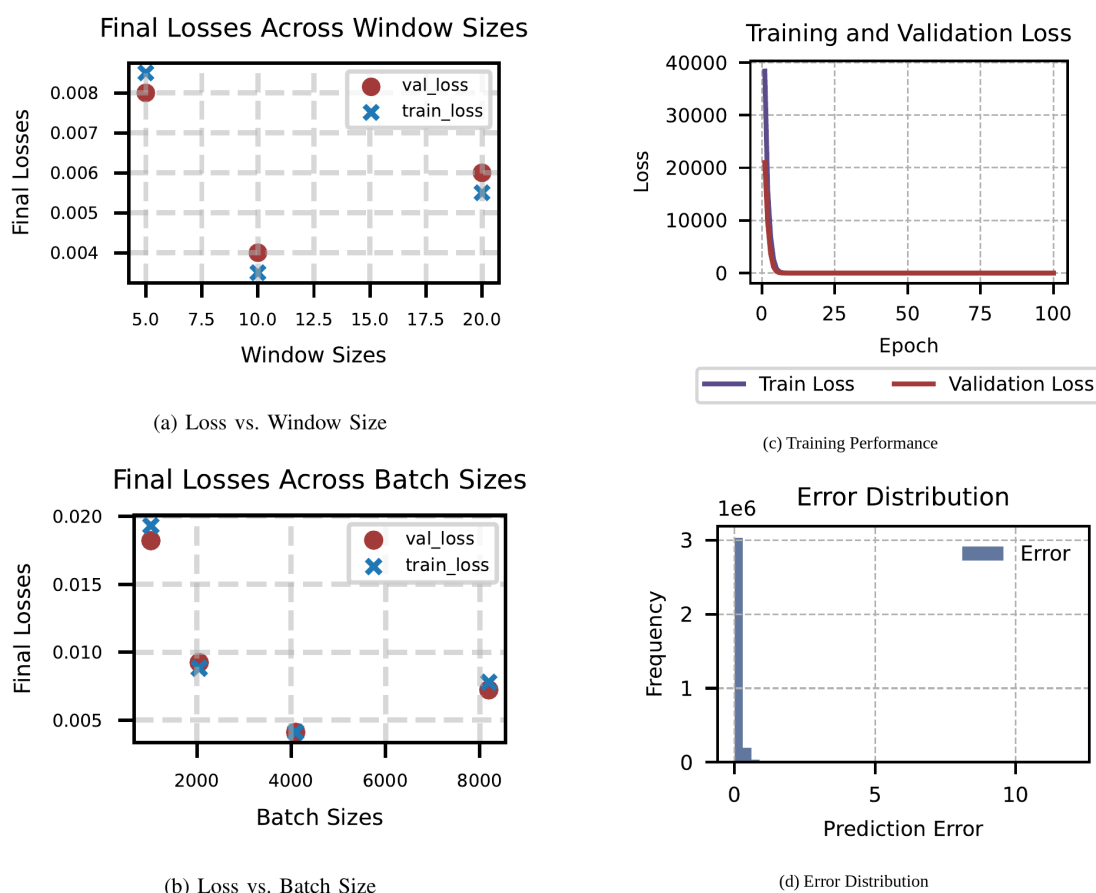
Table 7. Reproducibility status of the benchmark implementations in the retained experiment archive. Parameter counts are the measured counts reported in the benchmark; “not retained” denotes information that must be supplied by the original experiment configuration/code rather than inferred from the result table.

Model	Retained Specification	Information Needed for Exact Reproduction
Persistence/CV/CA	Closed-form predictors stated in text	None beyond the common input preprocessing
EKF-CA	9-state $[p, v, a]$ CA formulation	Numerical initialization, Q , R , and tuning record not retained
LSTM-PV	1 layer, 128 hidden units, explicit dropout 0.20, linear $128 \rightarrow 150$ decoder, 89.0 K parameters	Common optimizer/early-stopping settings reported in Table 5
LSTM-PVAQ	Same LSTM/decoder, 13 features, 92.6 K parameters	Common optimizer/early-stopping settings reported in Table 5
GRU-PVAQ	One recurrent layer, direct 150-output decoder, 74.3 K measured parameters	Exact hidden width/dropout implementation not retained in submission archive
TCN-PVAQ	Causal dilated residual TCN, direct 150-output head, 96.8 K parameters	Channel widths, kernel size, dilation schedule, dropout not retained
Transformer-PVAQ	Compact encoder with positional encoding, direct 150-output head, 118.5 K parameters	d_{model} , heads, layers, feed-forward width, dropout not retained

Table 7 separates what is actually preserved in the submission archive from details that would be required for bitwise/implementation-level reproduction. This distinction is important: matched parameter counts and a common validation protocol support a

comparative benchmark, but they do not substitute for complete code/configuration disclosure. Accordingly, the public dataset release strengthens data access, while release of the generation, preprocessing, model, EKF, and statistical-analysis code remains necessary for full computational reproducibility.

Sensitivity analyses examined candidate input-window and batch-size settings, training convergence, and the distribution of held-out prediction errors. Figure 6 is diagnostic rather than confirmatory: the left panels compare final training and validation losses across the stored tuning candidates and show that extreme settings were not uniformly preferable, while the right panels show convergence behavior and the shape of held-out Euclidean errors. These plots are used to check optimization stability, train–validation separation, and outlier/skew behavior. Hyperparameter selection itself was based on validation ADE; held-out test errors were not used to tune the models.



Left: window-length and batch-size sensitivity. Right: training convergence and held-out error distribution.

Figure 6. Training diagnostics with unique panel labels. Panel (a): final training/validation loss versus candidate history-window size. Panel (b): final training/validation loss versus candidate batch size. Panel (c): training and validation loss versus epoch for the selected configuration. Panel (d): held-out Euclidean prediction-error distribution used only for post-training characterization. Hyperparameter selection used validation ADE; the held-out error distribution was not used for tuning.

Feature-ablation protocol. The ablation follows a cumulative sensor-enrichment sequence while keeping the prediction target, flight split, window construction, LSTM hidden size, decoder, optimizer family, early-stopping rule, and five-seed evaluation protocol fixed. The tested inputs are: (i) position only; (ii) position + velocity; (iii) position + velocity + raw gravity-bearing acceleration; (iv) position + velocity + gravity-resolved acceleration; and (v) the full PVAQ input after adding quaternion orientation. Normalization is fitted from the training flights for the channels present in each feature set. The primary ablation

endpoint is flight-level 1 s 3D RMSE, and each stage is compared with the immediately preceding cumulative feature set. This procedure isolates the incremental value of velocity, acceleration treatment, and attitude information without changing the forecasting task.

This is a sequential conditional ablation, not a factorial attribution analysis. Therefore, the quoted 7.0% gain after adding quaternion orientation is conditional on position, velocity, and gravity-resolved acceleration already being present, and the acceleration gains are conditional on the preceding feature set. The experiment does not identify an independent marginal effect for acceleration or orientation, nor their interaction. Combinations such as PV + Q and PV + gravity-resolved acceleration would be required to separate these effects and are proposed as a follow-up rather than inferred from the existing sequence.

5. Evaluation Protocol

5.1. Accuracy and Statistical Analysis

For test window i at horizon h , the Euclidean error is

$$e_{i,h} = \left\| \mathbf{p}_{i,t+h} - \hat{\mathbf{p}}_{i,t+h} \right\|_2. \quad (11)$$

Horizon-specific 3D RMSE, average displacement error (ADE), and final displacement error (FDE) are

$$\text{RMSE}_h = \sqrt{\frac{1}{N} \sum_{i=1}^N e_{i,h}^2} \quad (12)$$

$$\text{ADE} = \frac{1}{H} \sum_{h=1}^H \left\| \mathbf{p}_{t+h} - \hat{\mathbf{p}}_{t+h} \right\|_2, \quad (13)$$

$$\text{FDE} = \left\| \mathbf{p}_{t+H} - \hat{\mathbf{p}}_{t+H} \right\|_2. \quad (14)$$

All metrics are inverse-transformed and reported in meters. The archived benchmark output used for this revision stores the three-dimensional Euclidean error as the primary endpoint. The previous statement that axis-specific RMSE “accompanies the aggregate metric” was therefore removed because separate x -, y -, and z -axis aggregate curves are not present in the archived submission outputs and should not be back-calculated from 3D RMSE. The enlarged held-out trajectory and error-distribution panels provide direct qualitative characterization, while axis-resolved error curves require regeneration from the saved per-step predictions.

For a held-out flight f containing N_f valid windows and a learned-model training seed s , flight-level RMSE at horizon h is defined as

$$\text{RMSE}_{f,h}^{(s)} = \sqrt{\frac{1}{N_f} \sum_{i \in f} \left(e_{i,h}^{(s)} \right)^2}. \quad (15)$$

Thus, squared Euclidean window errors are averaged within a flight before the square root is taken; the manuscript no longer describes flight-level RMSE as an average of window-level RMSE values. Within a given seed, the equal-flight-weight summary is

$$\overline{\text{RMSE}}_h^{(s)} = \frac{1}{F} \sum_{f=1}^F \text{RMSE}_{f,h}^{(s)}, \quad F = 465, \quad (16)$$

so flights of different lengths contribute equally rather than being pooled by their numbers of windows. This makes the flight, not the heavily overlapping window, the primary evaluation unit.

Each learned model was trained with five fixed seeds. The retained statistical export, however, contains only the final archived point summaries and inferential outputs; it does not preserve the five seed-specific intermediate summaries or code needed to establish whether the historical single point estimate was formed by averaging seed-specific equal-flight-weight summaries or by an equivalent seed-pooling step before export. We therefore do not introduce an unsupported seed-aggregation formula after the fact. The reported point estimates in the result tables are retained archival summaries; Equations (15) and (16) specify the recoverable window-to-flight and equal-flight-weight aggregation, while exact regeneration of the final seed combination requires the original per-flight/per-seed predictions or analysis script.

Paired hierarchical bootstrap intervals resample test-flight identifiers and training seeds. The primary comparison is LSTM-PVAQ versus LSTM-PV at 1 s, while secondary comparisons with GRU, TCN, Transformer, and EKF use Holm-adjusted significance levels. Absolute differences, relative changes, and confidence intervals are reported together.

For paired model comparisons, the same resampled flight identifiers must be used for both members of a model pair so that within-flight dependence is preserved; seed resampling forms the second hierarchy for learned-model variability. The retained statistical result exports contain the reported confidence intervals and Holm-adjusted p -values but do not retain the bootstrap replicate count, random-number seed, resampling index arrays, or the exact conversion of the paired bootstrap distribution to raw p -values. We therefore do not invent these implementation details. Exact inferential reproduction requires release or regeneration of the statistical-analysis script from the per-flight/per-seed predictions. This limitation does not change the reported archived point estimates, but it limits independent reproduction of the seed aggregation, intervals, and adjusted p -values.

5.2. Edge-Oriented IoT Evaluation

Batch-1 inference was benchmarked on a Raspberry Pi 5 CPU using one thread, FP32 precision, and a $100 \times F$ input history. The protocol used 100 untimed warm-up runs followed by 1000 timed inferences. Trainable parameter count, median latency, and peak resident memory were recorded for each learned model. Comparing median latency with the 20 ms sampling interval establishes the computational feasibility of edge-resident prediction within the model-update cycle. This comparison assesses the neural forward pass only; it does not claim that the complete sensing–preprocessing–networking–actuation loop meets a 20 ms end-to-end deadline.

The intended deployment architecture is an onboard companion-computer arrangement rather than a remote-cloud assumption. A flight controller/state estimator supplies position, velocity, attitude, and inertial/state telemetry to a Raspberry-Pi-class companion computer over a local interface; PX4, for example, documents companion-computer connections over serial or Ethernet using MAVLink or uXRCE-DDS, and lists Raspberry Pi among suitable companion computers [54]. The companion maintains the rolling history, applies the deployment-equivalent causal preprocessing/normalization, runs the predictor, and can pass the forecast to local planning/control software or forward selected predictions to networked IoT services. This architecture is illustrative of the intended use; it was not integrated and timed end-to-end in the present experiment. Table 8 summarizes the timing scope.

Table 8. Scope of the Raspberry Pi timing experiment. Only the neural forward pass was measured; the other stages are necessary components of an operational streaming pipeline and require separate latency characterization.

Pipeline Stage	Included in Reported Latency?	Deployment Implication
Sensor acquisition/flight-controller state estimation	No	Depends on avionics and estimator update rate
Telemetry transfer to companion computer	No	Local serial/Ethernet/MAVLink/uXRCE-DDS delay must be measured
Frame conversion and gravity handling	No	Can be implemented causally from current/past samples
Low-pass/resampling	No	Requires a specified causal streaming filter and group-delay accounting
Normalization and 100-sample history maintenance	No	Fixed training statistics; rolling buffer can be updated online
Neural forward pass	Yes	0.88 ms median for LSTM-PVAQ under the stated Raspberry Pi 5 protocol
Downstream networking, planning, or actuation	No	Application-specific and not evaluated here

6. Results

6.1. Multi-Horizon Three-Dimensional Accuracy

Table 9 reports flight-level 3D RMSE at the three forecast horizons. Persistence degraded rapidly as the horizon increased, reaching 2.963 m at 1 s. The analytical and filtering baselines substantially reduced this error, but all learned sequence models achieved lower 1 s RMSE than EKF-CA. LSTM-PVAQ produced the lowest error at every horizon: 0.043 m at 0.1 s, 0.168 m at 0.5 s, and 0.371 m at 1.0 s. TCN-PVAQ was the strongest alternative at 1 s with 0.409 m, followed by GRU-PVAQ with 0.427 m. Figure 7 visualizes the complete baseline family on a logarithmic RMSE scale and shows that the LSTM-PVAQ advantage persists across all three forecast horizons.

Table 9. Flight-level 3D RMSE in meters; brackets show 95% hierarchical bootstrap confidence intervals. Lower is better.

Model	0.1 s RMSE	0.5 s RMSE	1.0 s RMSE
Persistence	0.298 [0.290, 0.306]	1.489 [1.451, 1.528]	2.963 [2.877, 3.051]
Constant velocity	0.071 [0.068, 0.074]	0.284 [0.272, 0.296]	0.621 [0.590, 0.653]
Constant acceleration	0.066 [0.063, 0.069]	0.258 [0.246, 0.270]	0.523 [0.498, 0.550]
EKF-CA	0.059 [0.056, 0.062]	0.232 [0.221, 0.244]	0.497 [0.472, 0.523]
LSTM-PV	0.052 [0.049, 0.055]	0.211 [0.200, 0.222]	0.474 [0.452, 0.497]
GRU-PVAQ	0.049 [0.046, 0.052]	0.194 [0.184, 0.204]	0.427 [0.408, 0.447]
TCN-PVAQ	0.047 [0.044, 0.050]	0.187 [0.177, 0.197]	0.409 [0.391, 0.428]
Transformer-PVAQ	0.054 [0.051, 0.057]	0.206 [0.196, 0.217]	0.446 [0.425, 0.468]
LSTM-PVAQ	0.043 [0.040, 0.046]	0.168 [0.159, 0.177]	0.371 [0.354, 0.389]

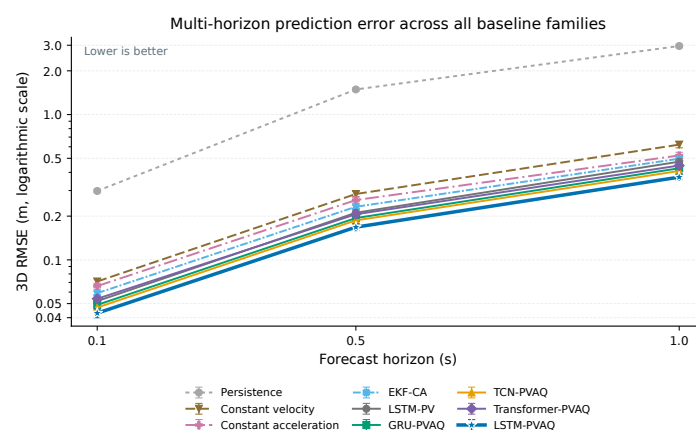


Figure 7. Multi-horizon three-dimensional RMSE across analytical, filtering, recurrent, convolutional, and attention-based baseline families. The logarithmic vertical axis keeps the persistence baseline and the more accurate learned models visible on the same scale. Lower is better.

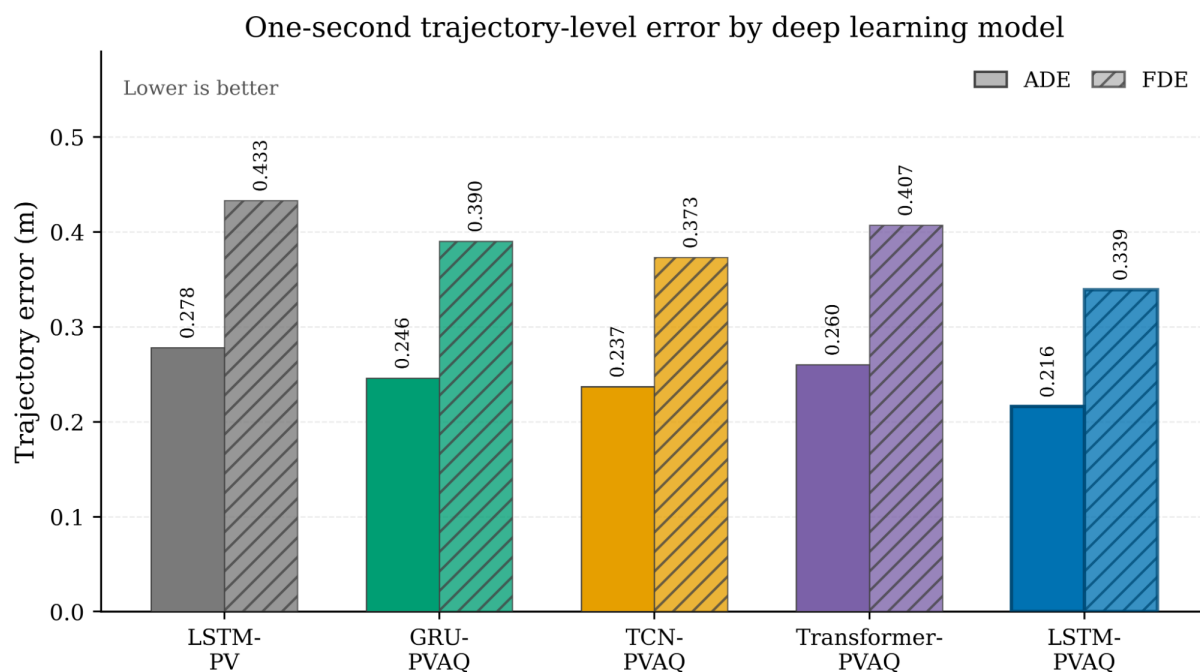
The absolute scale is important for practical interpretation. At 0.1 s, LSTM-PVAQ improves on TCN-PVAQ by 0.004 m (4 mm) and on LSTM-PV by 0.009 m (9 mm); relative to approximately 0.3 m of nominal travel in 0.1 s, these are modest differences. At 1 s, the gaps widen to 0.038 m (3.8 cm) versus TCN-PVAQ and 0.103 m (10.3 cm) versus LSTM-PV, corresponding to approximately 1.3% and 3.4% of the nominal 3 m one-second travel distance. These differences can be operationally relevant when a downstream module applies spatial thresholds or uncertainty margins, but the present benchmark does not equate RMSE improvements with a certified collision-avoidance or communication-safety margin.

6.2. Trajectory-Level Accuracy and Edge Efficiency

The one-second trajectory-level metrics in Table 10 show the same ordering. LSTM-PVAQ achieved the lowest ADE (0.216 m) and FDE (0.339 m). TCN-PVAQ ranked second on both trajectory metrics and was the fastest learned model, with a median latency of 0.54 ms. LSTM-PVAQ required 92.6 K parameters, 14.2 MB peak RAM, and 0.88 ms median latency. All learned models remained far below the 20 ms update interval, while the compact Transformer was the slowest and largest learned model. See Figures 8 and 9.

Table 10. One-second trajectory accuracy and edge-computing measurements for the learned predictors. Latency was measured on a Raspberry Pi 5 CPU with one FP32 thread, batch size one, 100 warm-up runs, and 1000 timed inferences.

Model	ADE (m)	FDE (m)	Params (K)	Median Latency (ms)	Peak RAM (MB)
LSTM-PV	0.278	0.433	89.0	0.84	13.9
GRU-PVAQ	0.246	0.390	74.3	0.72	13.2
TCN-PVAQ	0.237	0.373	96.8	0.54	15.8
Transformer-PVAQ	0.260	0.407	118.5	1.41	18.7
LSTM-PVAQ	0.216	0.339	92.6	0.88	14.2



PV: position + velocity; PVAQ: position + velocity + gravity-resolved acceleration + quaternion.

Figure 8. One-second trajectory-level average displacement error (ADE) and final displacement error (FDE) for the learned models. Lower is better.

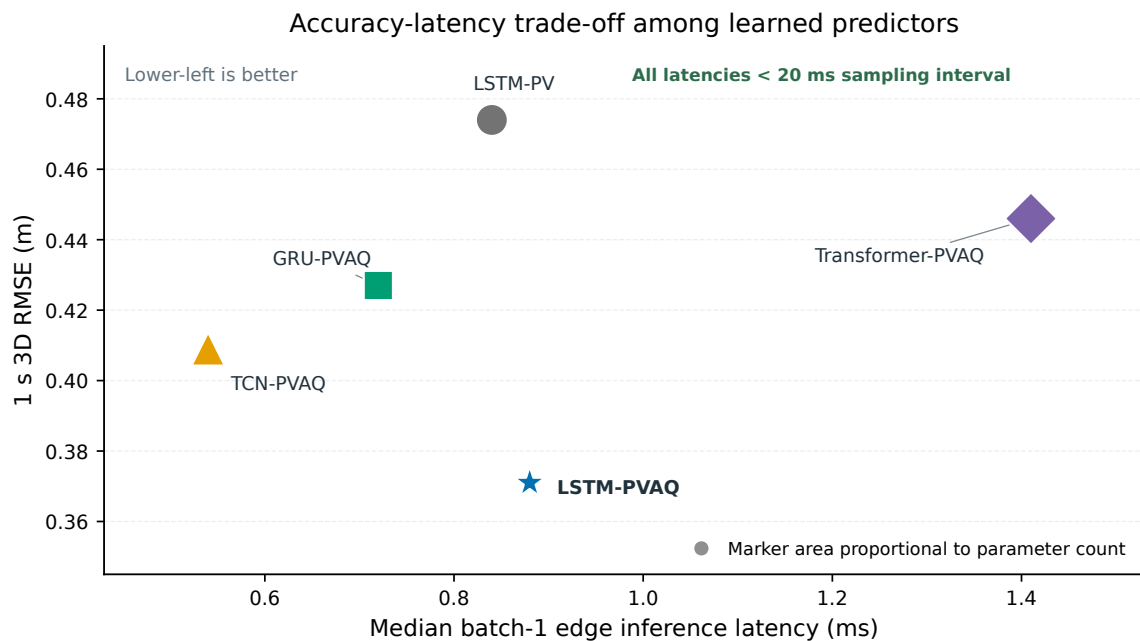


Figure 9. Accuracy–latency trade-off among learned predictors. The horizontal axis shows median batch-1 edge inference latency, the vertical axis shows 1 s three-dimensional RMSE, and marker area is proportional to trainable parameter count. Model identity is labeled directly; no color bar is used. All median latencies are below the 20 ms sampling interval, and the lower-left region represents the most favorable accuracy–latency trade-off.

6.3. Feature Ablation

Using the cumulative ablation procedure defined in Section 4.2, Table 11 quantifies sequential conditional gains as sensor groups are added to the LSTM; it does not identify independent marginal feature effects. Position-only input produced 0.612 m RMSE at 1 s. Adding velocity reduced error by 22.5%. Raw gravity-bearing acceleration produced a further 7.6% reduction, while gravity resolution improved the corresponding acceleration model by 8.9%. Quaternion orientation reduced RMSE by another 7.0%, yielding the full LSTM-PVAQ result of 0.371 m. Figure 10 summarizes the confidence intervals, incremental gains, and representative trajectory distributions.

Table 11. Sequential feature ablation for the LSTM at the 1 s horizon. Brackets show 95% confidence intervals.

Feature Set	<i>F</i>	1 s RMSE (m)	Gain vs. Prior	Observed Effect
Position only	3	0.612 [0.584, 0.642]	–	Position-history baseline
Position + velocity	6	0.474 [0.452, 0.497]	22.5%	Substantial gain from velocity
+ raw gravity-bearing acceleration	9	0.438 [0.417, 0.460]	7.6%	Additional short-term motion information
+ gravity-resolved acceleration	9	0.399 [0.380, 0.419]	8.9%	Gain from gravity resolution
+ quaternion orientation	13	0.371 [0.354, 0.389]	7.0%	Complete sensor-enriched input

6.4. Paired Flight-Level Comparisons

The paired analysis in Table 12 confirms that the LSTM-PVAQ advantage was consistent across held-out flights. Its 1 s RMSE was 0.103 m lower than LSTM-PV, 0.056 m lower than GRU-PVAQ, 0.038 m lower than TCN-PVAQ, 0.075 m lower than Transformer-PVAQ, and 0.126 m lower than EKF-CA. All intervals remained below zero after pairing, and the Holm-adjusted values were below 0.001. See Figure 11.

Table 12. Paired flight-level comparisons at the 1 s horizon. Define $\Delta = \text{RMSE}(\text{LSTM-PVAQ}) - \text{RMSE}(\text{comparator})$; negative values favor LSTM-PVAQ.

Comparison	Δ RMSE (m)	95% CI	Relative Change	Holm-Adjusted p
LSTM-PVAQ vs. LSTM-PV	−0.103	[−0.121, −0.085]	21.7% lower	<0.001
LSTM-PVAQ vs. GRU-PVAQ	−0.056	[−0.071, −0.040]	13.1% lower	<0.001
LSTM-PVAQ vs. TCN-PVAQ	−0.038	[−0.051, −0.024]	9.3% lower	<0.001
LSTM-PVAQ vs. Transformer-PVAQ	−0.075	[−0.092, −0.058]	16.8% lower	<0.001
LSTM-PVAQ vs. EKF-CA	−0.126	[−0.148, −0.104]	25.4% lower	<0.001

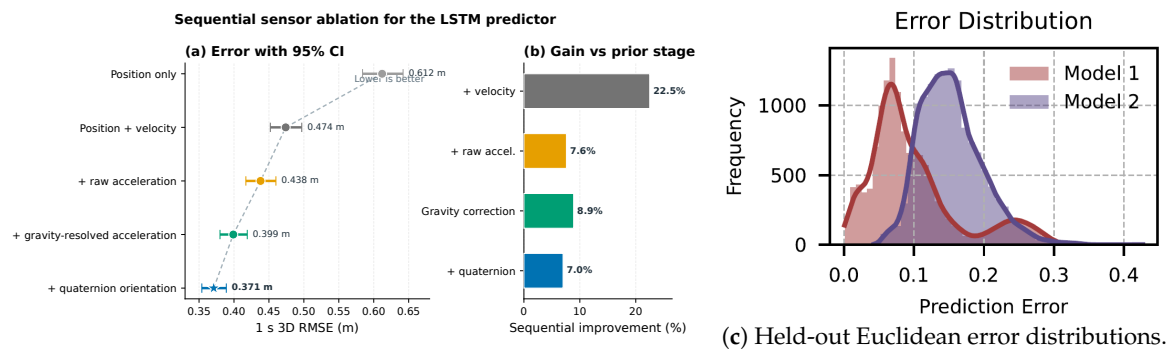


Figure 10. Feature-enrichment and held-out error characterization. Panels (a) and (b) report 1 s three-dimensional RMSE with 95% confidence intervals and sequential relative improvements, respectively. Panel (c) enlarges the held-out Euclidean error distributions for LSTM-PVAQ and LSTM-PV; the LSTM-PVAQ distribution is shifted toward lower errors. The corresponding illustrative true/predicted x - y trace is shown earlier in Figure 3; as noted there, the retained plotting artifact does not preserve a formal trajectory-selection rule, so the example is not used as inferential evidence.

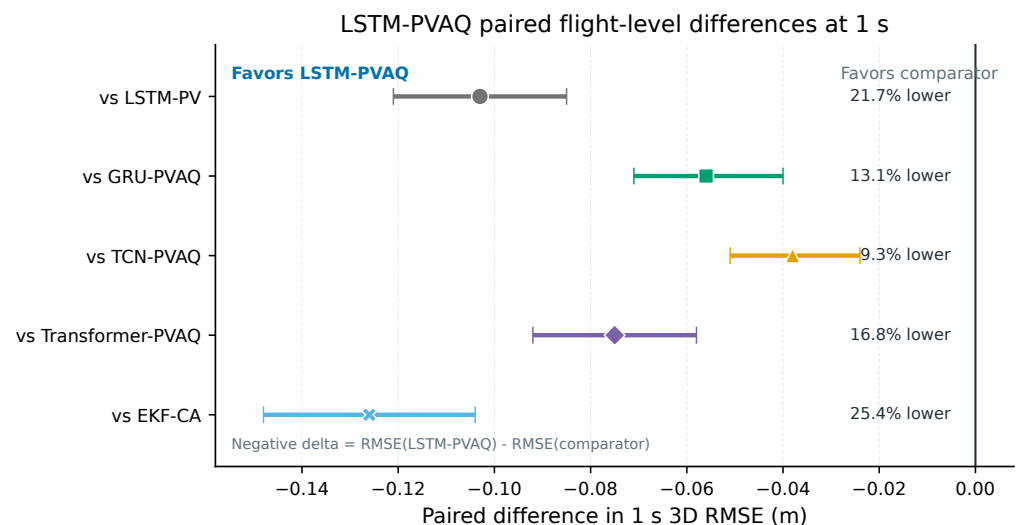


Figure 11. Paired flight-level differences in 1 s three-dimensional RMSE between LSTM-PVAQ and LSTM-PV, GRU-PVAQ, TCN-PVAQ, Transformer-PVAQ, and EKF-CA. Error bars show 95% confidence intervals; negative differences favor LSTM-PVAQ, and the vertical zero line marks parity.

7. Discussion

7.1. Interpretation of the Predictive Gains

The results support the primary feature hypothesis. Relative to LSTM-PV, the full LSTM-PVAQ reduced 1 s RMSE by 21.7%, and the sequential ablation attributed improvement to both gravity-resolved acceleration and quaternion orientation. The advantage increased with forecast horizon, which is consistent with the added channels becoming more informative as simple positional persistence and constant-velocity assumptions lose

accuracy. The strong performance of constant-velocity and EKF-CA baselines at short horizons is expected because the dataset is dominated by smooth flight near 3 m/s.

The ablation also suggests a physically plausible interpretation, while not proving a causal feature mechanism. Velocity provides the first temporal derivative of position, and acceleration provides higher-order information about changes in motion that can precede visible position divergence over a short forecast horizon. Gravity resolution is particularly informative because it separates translational acceleration from the gravity component expressed through vehicle attitude. Importantly, the transition from raw gravity-bearing acceleration to gravity-resolved acceleration keeps the feature dimension fixed at nine yet reduces 1 s RMSE from 0.438 to 0.399 m; this same-dimensional comparison supports the value of physically meaningful preprocessing rather than an explanation based only on adding more input channels. Quaternion orientation then supplies the body-to-navigation attitude context needed to interpret inertial motion consistently during rotation. Its additional 7.0% conditional gain may reflect both genuine attitude information and the greater input dimensionality. Because the ablation is sequential rather than factorial, the present experiment cannot fully separate semantic/physical information content from generic additional-covariate effects or quantify acceleration–orientation interactions independently.

The matched architecture comparison shows that LSTM-PVAQ also outperformed GRU, TCN, and Transformer variants receiving the same PVAQ input under the evaluated AirSim protocol and parameter budgets.

The Transformer-PVAQ result should not be interpreted as evidence that Transformers are intrinsically inferior for UAV trajectory prediction. Parameter count is only a capacity proxy and does not guarantee better generalization. In this benchmark, the input is a relatively short 100-step history dominated by smooth local dynamics, for which recurrent gating and causal convolution provide strong temporal inductive biases. The compact Transformer also had to learn positional and attention structure from the same 3100-flight domain without pretraining, and the matched search budget was designed for fairness rather than exhaustive Transformer-specific optimization. In contrast, recent multirotor work has reported benefits from Transformer-based sequence estimation on more diverse and distribution-shifted trajectory collections [33]. Together with broader sequence-model benchmarks [49], this supports a protocol-specific explanation: the additional Transformer parameters were not automatically useful under the present data regime, sequence length, and tuning budget.

7.2. Accuracy Versus Deployability

LSTM-PVAQ provided the best accuracy, but TCN-PVAQ offered the strongest speed–accuracy compromise: its 1 s RMSE was only 0.038 m higher while its median latency was 0.54 ms rather than 0.88 ms. GRU-PVAQ used the fewest parameters and the least peak RAM among the learned PVAQ models, while the compact Transformer was both slower and less accurate in this protocol. All learned models completed the measured forward pass within a small fraction of the 20 ms update interval, supporting model-level edge inference for the evaluated batch-1 workload but not establishing end-to-end real-time pipeline latency. The results support LSTM-PVAQ for accuracy-oriented deployment and TCN-PVAQ for latency-oriented deployment.

7.3. Limitations and Future Work

The evaluation is based on AirSim flights from one mountainous environment, with trajectories concentrated near a nominal speed of 3 m/s. Performance on physical UAVs, aggressive turns, vertical maneuvers, payload changes, turbulence, and different vehicle

platforms remains to be established. Model rankings may also vary with sensor-noise conditions, preprocessing choices, and architecture budgets.

Accordingly, four transfer dimensions remain untested: environmental transfer (new maps/obstacle layouts), dynamic transfer (different speed and maneuver envelopes), sensing transfer (different noise, estimator delay, and calibration), and platform transfer (different multirotor dynamics, payloads, and flight controllers). The reported numerical errors should therefore be treated as benchmark evidence for unseen flights within the retained AirSim domain, not as expected error bounds for arbitrary UAV deployments.

Accordingly, the demonstrated generalization is flight-disjoint within-domain generalization: test windows come only from 465 flights that were excluded before normalization and window generation, but those flights still share the same simulator domain and broad operating regime as training. The study does not establish zero-shot transfer to unseen maps, aircraft, sensor calibrations, or real flight. For deployment, the coordinate-frame convention, gravity removal, sensor scaling, and temporal rate must be reproduced or recalibrated, and model performance should be revalidated under the target vehicle's maneuver envelope. This distinction is consistent with recent real/multi-source UAV trajectory studies showing that model rankings can change under shifted trajectory distributions [33].

A disturbance- or maneuver-stratified test would be particularly informative because acceleration and attitude cues should be most valuable when constant-velocity assumptions break down. The present archive does not retain a synchronized per-window wind label, turning-rate summary, or acceleration-bin result table tied to the saved model predictions, so a post hoc wind/turning-rate performance curve cannot be regenerated reliably from the submission outputs alone. We therefore avoid claiming that the PVAQ gain is concentrated in high-wind or high-maneuver flights. A future analysis should prespecify bins (for example by wind magnitude, $\|a\|$, yaw rate, or curvature), maintain flight-level grouping within each stratum, and report stratum-specific uncertainty.

Different flight conditions can affect this predictor through distinct mechanisms. Higher translational speeds increase the distance traveled within a fixed 0.1–1.0 s horizon and can amplify the effect of controller lag or model mismatch; a network trained mainly near 3 m/s may therefore extrapolate poorly at substantially higher speeds. Aggressive turns and rapid attitude changes increase yaw/pitch/roll rates and curvature, making orientation and acceleration histories potentially more informative, but they also move the input sequence farther from the smooth-motion regime represented in the training corpus. Strong vertical climbs or descents change the coupling between thrust, gravity resolution, and the z-axis dynamics, so performance observed in predominantly horizontal flight cannot be assumed to hold for sustained vertical maneuvers. Gusts and turbulence can introduce higher-frequency accelerations and estimator noise; under such conditions, the 50 Hz model rate and 20 Hz anti-aliasing choice may require revalidation because rapid disturbance content can be attenuated or aliased differently. Payload, battery state, airframe, and flight-controller changes alter mass, inertia, actuation limits, and closed-loop response, changing the mapping from the same recent PVAQ history to future displacement. These mechanisms explain why broader validation should vary speed, maneuver class, wind/turbulence, payload, controller, and vehicle rather than simply add more flights from the same scenario. Recent real-flight and multi-dataset studies likewise show the value of testing across physical flights, multiple trajectory distributions, and explicit maneuver/velocity conditions [32,33,35].

Scalability should therefore be interpreted in two separate senses. Computationally, the present predictor is compact, and its batch-1 neural forward pass fits comfortably within the measured Raspberry-Pi-class budget; increasing the number of stored training flights primarily affects offline training and data management. Operational scalability,

however, is not established by this timing result: deploying across heterogeneous UAV fleets would require platform-specific state-interface validation, normalization/calibration checks, and preferably scenario-disjoint or leave-one-platform/environment-out testing. The current model is also a single-ego-UAV predictor and does not model inter-UAV interactions, so scalability to dense multi-UAV traffic is a separate research problem rather than a demonstrated property of the benchmark.

The edge comparison reports median model-inference latency on one platform; preprocessing cost, state-estimator delay, telemetry transport, tail latency, and energy consumption remain outside the present scope. The archived experiment package also lacks some run-specific generation metadata, exact non-LSTM comparator configurations, EKF covariance settings, filter coefficients, and bootstrap-resampling details; these omissions limit exact computational reproduction and motivate a code/configuration release in addition to the now-public dataset. The present trajectory and error plots characterize horizontal geometry and Euclidean error but do not replace a regenerated axis-resolved $x/y/z$ error analysis; that analysis should be produced from stored per-step predictions in a future reproducibility update rather than inferred from aggregate 3D RMSE. Future evaluation will include physical UAV logs, multiple environments and maneuver regimes, sensor degradation, axis-resolved error diagnostics, and closed-loop control or networking experiments.

8. Conclusions

This study reformulated UAV position prediction as a flight-wise, multi-horizon 3D benchmark and evaluated a compact sensor-enriched LSTM against analytical, filtering, recurrent, convolutional, and attention-based alternatives. Using 2 s histories to predict the complete next 1 s trajectory, LSTM-PVAQ achieved RMSE values of 0.043, 0.168, and 0.371 m at 0.1, 0.5, and 1.0 s. Its one-second RMSE was 21.7% lower than LSTM-PV and 9.3% lower than TCN-PVAQ, the strongest alternative learned model. The cumulative ablation showed sequential conditional gains when velocity, gravity-resolved acceleration, and quaternion orientation were added; it does not establish independent causal effects for each feature group.

The model required 92.6 K parameters and a median of 0.88 ms per batch-1 neural forward pass on a Raspberry Pi 5 CPU, demonstrating model-level computational compatibility with the 20 ms update interval. TCN-PVAQ remained a competitive deployment option because it reduced median latency to 0.54 ms with a modest accuracy penalty. Within the evaluated AirSim conditions, the results establish a strong accuracy–efficiency profile for sensor-enriched multi-horizon prediction, while route/scenario transfer, physical-sensor robustness, and end-to-end IoT/network timing remain unvalidated. Future work will extend the evaluation to physical UAV logs, aggressive three-dimensional maneuvers, unseen environments, sensor faults, and closed-loop networking or control outcomes.

Author Contributions: Conceptualization, M.A. and A.K.; methodology, M.A. and A.K.; software, M.A. and A.K.; validation, M.A. and A.K.; formal analysis, M.A. and A.K.; investigation, M.A. and A.K.; resources, M.A. and A.K.; data curation, M.A. and A.K.; writing—original draft preparation, M.A. and A.K.; writing—review and editing, M.A. and A.K.; visualization, M.A. and A.K.; supervision, M.A. and A.K.; project administration, M.A. and A.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset supporting this study is publicly available as Dataset Version 1.0 (September 2026) through Google Drive at <https://drive.google.com/drive/folders/1LDBI4PIeq4zmwggqel1TIBZuW1f48BDHj?usp=sharing>, accessed on 7 September 2026. The release contains the AirSim flight data analyzed in this study, comprising 3100 flight sequences and the 20 recorded scalar channels summarized in Table 2; 13 PVAQ channels are used by the primary predictor. Google Drive was selected as the hosting platform because the complete high-frequency dataset is very large, and it allows the full release to remain publicly accessible through a single shared location. The Google Drive release represents the dataset version associated with this manuscript. No DOI-backed repository record is currently claimed for Dataset Version 1.0. To preserve traceability, any future corrected or reorganized release will receive a new version identifier and an updated manifest/changelog rather than silently replacing Version 1.0. The manuscript and accompanying manifest provide the flight counts, duration statistics, channel organization, split sizes, preprocessing order, window protocol, version information, and access conditions needed to interpret the release. The corresponding author can provide clarification regarding file organization and preprocessing.

Acknowledgments: The authors acknowledge the RMC that hosts the AirSim simulation environment.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

Abbreviations used in this manuscript:

ADE	Average Displacement Error
ADS-B	Automatic Dependent Surveillance–Broadcast
EKF	Extended Kalman Filter
FDE	Final Displacement Error
GRU	Gated Recurrent Unit
IMU	Inertial Measurement Unit
IoT	Internet of Things
LSTM	Long Short-Term Memory
PVAQ	Position, Velocity, Acceleration, and Quaternion
RMSE	Root Mean Squared Error
TCN	Temporal Convolutional Network
UAV	Unmanned Aerial Vehicle
VLA	Vision–Language–Action

References

1. Alotaibi, B. A Review of Resilient IoT Systems: Trends, Challenges, and Future Directions. *Appl. Sci.* **2026**, *16*, 2079. [CrossRef]
2. Al-Ahmed, S.A.; Ahmed, T.; Zhu, Y.; Malaolu, O.O.; Shakir, M.Z. UAV-Enabled IoT Networks: Architecture, Opportunities, and Challenges. In *Wireless Networks and Industrial IoT: Applications, Challenges and Enablers*; Springer Nature: Cham, Switzerland, 2020; pp. 263–288. [CrossRef]
3. Hoque, M.A.; Hossain, M.; Noor, S.; Islam, S.R.; Hasan, R. IoTaaS: Drone-Based Internet of Things as a Service Framework for Smart Cities. *IEEE Internet Things J.* **2022**, *9*, 12425–12439. [CrossRef]
4. Wei, Z.; Zhu, M.; Zhang, N.; Wang, L.; Zou, Y.; Meng, Z.; Wu, H.; Feng, Z. UAV-Assisted Data Collection for Internet of Things: A Survey. *IEEE Internet Things J.* **2022**, *9*, 15460–15483. [CrossRef]
5. Mozaffari, M.; Saad, W.; Bennis, M.; Debbah, M. Mobile Unmanned Aerial Vehicles (UAVs) for Energy-Efficient Internet of Things Communications. *IEEE Trans. Wirel. Commun.* **2017**, *16*, 7574–7589. [CrossRef]
6. Lindqvist, T. UAV Drone Industry Statistics: Market Data Report 2026. Online Report, 2026. Available online: <https://gitnux.org/uav-drone-industry-statistics/> (accessed on 16 March 2026).
7. Georgiou, H.; Karagiorgou, S.; Kontoulis, Y.; Pelekis, N.; Petrou, P.; Scarlatti, D.; Theodoridis, Y. Moving Objects Analytics: Survey on Future Location and Trajectory Prediction Methods. *arXiv* **2018**, arXiv:1807.04639. [CrossRef]
8. Lefèvre, S.; Vazquez, D.; Laugier, C. A Survey on Motion Prediction and Risk Assessment for Intelligent Vehicles. *ROBOMECH J.* **2014**, *1*, 1. [CrossRef]
9. Benevides, J.R.; Paiva, M.A.; Simplicio, P.V.; Inoue, R.S.; Terra, M.H. Disturbance Observer-Based Robust Control of a Quadrotor Subject to Parametric Uncertainties and Wind Disturbance. *IEEE Access* **2022**, *10*, 7554–7565. [CrossRef]

10. Bannwarth, J.; Chen, Z.; Stol, K.; MacDonald, B. Disturbance Accommodation Control for Wind Rejection of a Quadcopter. In Proceedings of the 2016 International Conference on Unmanned Aircraft Systems (ICUAS), Arlington, VA, USA, 7–10 June 2016; pp. 695–701. [\[CrossRef\]](#)
11. Thrun, S. Probabilistic Robotics. *Commun. ACM* **2002**, *45*, 52–57. [\[CrossRef\]](#)
12. Wan, E.A.; Van Der Merwe, R. The Unscented Kalman Filter for Nonlinear Estimation. In Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium, Lake Louise, AB, Canada, 1–4 October 2000; pp. 153–158. [\[CrossRef\]](#)
13. Tao, L.; Watanabe, Y.; Yamada, S.; Takada, H. Comparative Evaluation of Kalman Filters and Motion Models in Vehicular State Estimation and Path Prediction. *J. Navig.* **2021**, *74*, 1142–1160. [\[CrossRef\]](#)
14. Bai, Y.; Yan, B.; Zhou, C.; Su, T.; Jin, X. State of the Art on State Estimation: Kalman Filter Driven by Machine Learning. *Annu. Rev. Control* **2023**, *56*, 100909. [\[CrossRef\]](#)
15. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [\[CrossRef\]](#) [\[PubMed\]](#)
16. Cho, K.; van Merriënboer, B.; Bahdanau, D.; Bengio, Y. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. In Proceedings of the Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014; pp. 103–111. [\[CrossRef\]](#)
17. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv* **2014**, arXiv:1412.3555. [\[CrossRef\]](#)
18. Zhang, Y.; Jia, Z.; Dong, C.; Liu, Y.; Zhang, L.; Wu, Q. Recurrent LSTM-Based UAV Trajectory Prediction with ADS-B Information. In Proceedings of the 2022 IEEE Global Communications Conference (GLOBECOM), Rio de Janeiro, Brazil, 4–8 December 2022; pp. 1–6. [\[CrossRef\]](#)
19. Zhu, R.; Yang, Z.; Chen, J.; Li, N.; Zhang, Z.; Song, Y. Short-Term Trajectory Prediction for Small Scale Drones at Low-Altitude Airspace. In Proceedings of the 2022 IEEE 4th International Conference on Civil Aviation Safety and Information Technology (ICCASIT), Dali, China, 12–14 October 2022; pp. 514–519. [\[CrossRef\]](#)
20. Nacar, O.; Abdelkader, M.; Ghouti, L.; Gabr, K.; Al-Batati, A.; Koubaa, A. VECTOR: Velocity-Enhanced GRU Neural Network for Real-Time 3D UAV Trajectory Prediction. *Drones* **2025**, *9*, 8. [\[CrossRef\]](#)
21. Shukla, P.; Shukla, S.; Singh, A.K. Trajectory-Prediction Techniques for Unmanned Aerial Vehicles (UAVs): A Comprehensive Survey. *IEEE Commun. Surv. Tutor.* **2025**, *27*, 1867–1910. [\[CrossRef\]](#)
22. Jia, D.; Kua, J.; Liu, X. A Lightweight LSTM Model for Flight Trajectory Prediction in Autonomous UAVs. *Future Internet* **2026**, *18*, 4. [\[CrossRef\]](#)
23. Ywet, N.L.; Maw, A.A.; Lee, J.W. R-YOLO: Enhancing Takeoff/Landing Safety in UAM Vertiports with Deep Learning Model. *IEEE Access* **2025**, *13*, 89045–89058. [\[CrossRef\]](#)
24. Shankar, R.S.; Siotia, V.; Panicker, R.O. Drone Flight Dataset and Lightweight LSTM-Based Wind Estimation for Semi-Autonomous Quadcopter Control. *IEEE Access* **2025**, *13*, 203057–203076. [\[CrossRef\]](#)
25. Dhiman, P.; Ambade, A.; Agrawal, K.; Banda, G. Deep Learning-Based Autonomous Navigation for PAVs in Urban Airspaces via Synthetic Dataset Generation Framework. *IEEE Access* **2026**, *14*, 99750–99763. [\[CrossRef\]](#)
26. Gu, H. UAV-VLA: Multimodal Vision–Language–Action Pretraining for Autonomous Drone Navigation. *Zenodo* **2026**, preprint. Available online: <https://zenodo.org/records/20313605> (accessed on 16 March 2026).
27. Sarode, A.; Naresh, K. Hybrid GPS–IMU Spoofing Detection for UAVs Using Deep Learning: A Multi-Class LSTM Approach. In Proceedings of the 2025 1st International Conference on Advancement in Futuristic Technologies (ICAFT), Belagavi, India, 19–21 December 2025; pp. 1–6. [\[CrossRef\]](#)
28. Aldao, E.; González-de Santos, L.M.; González-Jorge, H. LiDAR Based Detect and Avoid System for UAV Navigation in UAM Corridors. *Drones* **2022**, *6*, 185. [\[CrossRef\]](#)
29. Cheng, Q.; Wu, W.; Zhao, Z. Uncertainty-Calibrated UAV Trajectory Prediction for Beam Management in UAV-Assisted ISAC Scenarios. *Drones* **2026**, *10*, 434. [\[CrossRef\]](#)
30. Hashemi, S.M.; Botez, R.M.; Ghazi, G. Bidirectional Long Short-Term Memory Development for Aircraft Trajectory Prediction Applications to the UAS-S4 Ehécatl. *Aerospace* **2024**, *11*, 625. [\[CrossRef\]](#)
31. Yoon, S.; Jang, D.; Yoon, H.; Park, T.; Lee, K. GRU-Based Deep Learning Framework for Real-Time, Accurate, and Scalable UAV Trajectory Prediction. *Drones* **2025**, *9*, 142. [\[CrossRef\]](#)
32. Ruseno, N.; Lin, C.Y. Real-Time UAV Trajectory Prediction for UTM Surveillance Using Machine Learning. *Unmanned Syst.* **2025**, *13*, 505–519. [\[CrossRef\]](#)
33. Tang, R.; Ng, K.K.H.; Li, L.; Yang, Z. A Learning-Based Interacting Multiple Model Filter for Trajectory Prediction of Small Multirotor Drones Considering Differential Sequences. *Transp. Res. Part C Emerg. Technol.* **2025**, *174*, 105115. [\[CrossRef\]](#)
34. Kebede, Y.B.; Yang, M.D.; Shikur, H.D.; Tseng, H.H. Real-Time UAV Flight Path Prediction Using GRU Networks for Autonomous Site Assessment. *Drones* **2026**, *10*, 56. [\[CrossRef\]](#)

35. Wang, X.; Li, W.; Chen, Z.; Xue, L.; Wu, Z.; Ding, X.; Cao, L. Low-Speed UAV Trajectory Prediction under Short-Term Data Loss: A Preliminary Feasibility Study for Vehicle-Mounted Electro-Optical Tracking. *Sci. Rep.* **2026**. [CrossRef] [PubMed]
36. Ashbrook, D.; Starner, T. Using GPS to Learn Significant Locations and Predict Movement Across Multiple Users. *Person. Ubiquitous Comput.* **2003**, *7*, 275–286. [CrossRef]
37. Monreale, A.; Pinelli, F.; Trasarti, R.; Giannotti, F. WhereNext: A Location Predictor on Trajectory Pattern Mining. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*; ACM: New York, NY, USA, 2009; pp. 637–646. [CrossRef]
38. Won, J.I.; Kim, S.W.; Baek, J.H.; Lee, J. Trajectory Clustering in Road Network Environment. In *Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Data Mining*, Nashville, TN, USA, 30 March–2 April 2009; pp. 299–305. [CrossRef]
39. Roh, G.P.; Hwang, S.W. NNCluster: An Efficient Clustering Algorithm for Road Network Trajectories. In *Proceedings of the Database Systems for Advanced Applications*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 47–61. [CrossRef]
40. Shrivastava, A.; Verma, J.P.V.; Jain, S.; Garg, S. A Deep Learning Based Approach for Trajectory Estimation Using Geographically Clustered Data. *SN Appl. Sci.* **2021**, *3*, 597. [CrossRef]
41. Zamboni, S.; Kefato, Z.T.; Girdzijauskas, S.; Norén, C.; Dal Col, L. Pedestrian Trajectory Prediction with Convolutional Neural Networks. *Pattern Recognit.* **2022**, *121*, 108252. [CrossRef]
42. Jiang, H.; Chang, L.; Li, Q.; Chen, D. Trajectory Prediction of Vehicles Based on Deep Learning. In *Proceedings of the 2019 4th International Conference on Intelligent Transportation Engineering*, Singapore, 5–7 September 2019; pp. 190–195. [CrossRef]
43. Yao, B.; Zhong, Q.; Cui, H.; Chen, S.; Fu, C.; Gao, K.; Cui, S. LSTM-Based Vehicle Trajectory Prediction Using UAV Aerial Data. In *Proceedings of the KES-STIS International Symposium*; Springer: Singapore, 2023; pp. 13–21. [CrossRef]
44. Wang, J.; Liu, K.; Li, H. LSTM-Based Graph Attention Network for Vehicle Trajectory Prediction. *Comput. Netw.* **2024**, *248*, 110477. [CrossRef]
45. Wen, F.; Li, M.; Wang, R. Social Transformer: A Pedestrian Trajectory Prediction Method Based on Social Feature Processing Using Transformer. In *Proceedings of the 2022 International Joint Conference on Neural Networks (IJCNN)*, Padua, Italy, 18–23 July 2022; pp. 1–7. [CrossRef]
46. Ngiam, J.; Caine, B.; Vasudevan, V.; Zhang, Z.; Chiang, H.T.L.; Ling, J.; Roelofs, R.; Bewley, A.; Liu, C.; Venugopal, A.; et al. Scene Transformer: A Unified Architecture for Predicting Multiple Agent Trajectories. *arXiv* **2021**, arXiv:2106.08417. [CrossRef]
47. Bai, S.; Kolter, J.Z.; Koltun, V. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv* **2018**, arXiv:1803.01271. [CrossRef]
48. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30, pp. 5998–6008.
49. Liu, J.; Shen, C.; O'Donncha, F.; Song, Y.; Zhi, W.; Beck, H.E.; Bindas, T.; Kraabel, N.; Lawson, K. From RNNs to Transformers: Benchmarking Deep Learning Architectures for Hydrologic Prediction. *Hydrol. Earth Syst. Sci.* **2025**, *29*, 6811–6828. [CrossRef]
50. Guo, T.; Jiang, N.; Li, B.; Zhu, X.; Wang, Y.; Du, W. UAV Navigation in High Dynamic Environments: A Deep Reinforcement Learning Approach. *Chin. J. Aeronaut.* **2021**, *34*, 479–489. [CrossRef]
51. Microsoft AirSim. AirSim APIs: Multirotor State, Vehicle Control, Ground-Truth Kinematics, and Wind API. Official AirSim Documentation. 2026. Available online: <https://github.com/microsoft/AirSim/blob/main/docs/apis.md> (accessed on 26 August 2026).
52. Shah, S.; Dey, D.; Lovett, C.; Kapoor, A. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. In *Field and Service Robotics*; Springer: Cham, Switzerland, 2018; pp. 621–635. [CrossRef]
53. Microsoft AirSim. Sensors in AirSim: Default IMU Configuration and Sensor Model Parameters. Official AirSim Documentation. 2026. Available online: <https://github.com/microsoft/AirSim/blob/main/docs/sensors.md> (accessed on 26 August 2026).
54. PX4 Autopilot. Companion Computers. PX4 User Guide. 2026. Available online: https://docs.px4.io/main/en/companion_computer/ (accessed on 26 August 2026).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.